



HAL
open science

Can Homomorphic Cryptography ensure Privacy?

Antoine Guellier

► **To cite this version:**

Antoine Guellier. Can Homomorphic Cryptography ensure Privacy?. [Research Report] RR-8568, IRISA; Supélec Rennes, équipe Cidre; Inria. 2014, pp.109. hal-01052509v2

HAL Id: hal-01052509

<https://inria.hal.science/hal-01052509v2>

Submitted on 21 Oct 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial - ShareAlike 4.0 International License



Can Homomorphic Cryptography ensure Privacy?

Antoine Guellier

**RESEARCH
REPORT**

N° 8568

July 2014

Project-Team CIDRE



Can Homomorphic Cryptography ensure Privacy?

Antoine Guellier^{*†}

Project-Team CIDRE

Research Report n° 8568 — July 2014 — 109 pages

Abstract: The advent of information technology, the dramatic increase of computational and storage capacities along with the development of worldwide communications promise very personalized, well designed and convenient services. However, although privacy is an important right in our societies, these services often neglect and abuse their consumers' privacy, by collecting sensible data notably. This work focuses on the protection of privacy in information systems. For this, it is known that traditional cryptography (*e.g.* encryption or signature schemes) is necessary, for instance to ensure confidentiality, but it is not sufficient. To enable usability and privacy at the same time, stronger tools are required. Homomorphic cryptography is a possible and promising candidate for this purpose. This technology allows manipulating encrypted data and performing logical operations on it without actually accessing the data in the clear. In particular, homomorphic cryptography is envisioned as the perfect technology for secure computation (or storage) delegation in the cloud. This work investigates how homomorphic cryptography can enable privacy, by first giving a deep insight of the field of privacy in computer science (systems of interest, tools, main goals, ...) and then presenting homomorphic cryptography and more specifically (fully) homomorphic encryption, aggregating the work done in this branch of cryptography in the last 30 years. At last, this work gives clues to answer the main question *can homomorphic cryptography ensure privacy?*, and interesting leads currently investigated or yet to be considered.

Key-words: privacy, anonymity, anonymous, PETs, homomorphic, cryptography, fully homomorphic cryptography

* antoine.guellier@supelec.fr

† SUPELEC/Inria/CNRS/Université de Rennes 1/IRISA (UMR 6074)

**RESEARCH CENTRE
RENNES – BRETAGNE ATLANTIQUE**

Campus universitaire de Beaulieu
35042 Rennes Cedex

La cryptographie homomorphe peut-elle assurer la vie privée ?

Résumé : L'avènement des technologies de l'information, l'augmentation des capacités de calcul et de stockage des appareils ainsi que le développement des communications promettent des services personnalisés, faciles d'utilisation et utiles au plus grand nombre. Cependant, et bien que la vie privée soit un droit fondamental de nos sociétés, ces services négligent souvent la vie privée de leurs consommateurs et en abusent, notamment à travers la collecte intensive de données sensibles. Ces travaux s'intéressent à la protection de la vie privée dans les systèmes d'information. Pour cela, il est admis qu'il est nécessaire de faire appel à la cryptographie traditionnelle (*e.g.* schémas de chiffrement ou de signature), par exemple pour assurer la confidentialité, mais cela ne suffit pas. Afin de réconcilier utilité et vie privée, des outils plus puissants sont nécessaires. La cryptographie homomorphe est un candidat prometteur en vue d'atteindre ce but. Cette technologie permet de manipuler et d'effectuer des opérations logiques sur des données chiffrées sans avoir accès aux données sous-jacentes. En particulier, la cryptographie homomorphe est toute désignée pour permettre la délégation au *cloud* du stockage et du calcul sur des données sensibles. Ces travaux étudient comment la cryptographie homomorphe peut être utilisée pour assurer le respect de la vie privée, d'abord en donnant une vue approfondie du champ de recherche que constitue la protection de la vie privée, puis en présentant la cryptographie homomorphe et plus spécifiquement le chiffrement homomorphe complet en faisant une synthèse des travaux effectués dans cette branche de la cryptographie depuis les 30 dernières années. Enfin, ce travail donne des éléments de réponse à la question *la cryptographie homomorphe peut-elle assurer la vie privée ?*, ainsi que des axes de recherche en cours et d'autres à considérer.

Mots-clés : vie privée, anonymat, PETs, homomorphe, homomorphisme, cryptographie, chiffrement homomorphe

Contents

Introduction	5
1 Privacy	9
1.1 Proposed definitions	9
1.1.1 Several meanings	9
1.1.2 Multiple dimensions	11
1.1.3 Trial for a synthetic view of privacy: Approach \times Sector	12
1.2 Different approaches to privacy	14
1.2.1 The legal approach	14
1.2.2 The policy approach	15
1.2.3 The technological approach	16
1.2.4 The economics approach	17
1.3 Different sectors of privacy in computer science	18
1.3.1 Identity management and authentication	19
1.3.2 Personal information and data management	20
1.3.3 Anonymous communications and traffic analysis	26
1.3.4 Cryptographic constructions and protocols	29
1.4 Measuring privacy in computer science	30
1.4.1 Information theory	31
1.4.2 k-privacy	32
1.4.3 Differential privacy	33
1.5 Discussion	34
1.5.1 The Approach \times Sector view	34
1.5.2 Trust management	35
1.5.3 Privacy in public opinion	36
1.5.4 Do we need privacy ?	36
2 Homomorphic Encryption	39
2.1 General points	39
2.1.1 Definition	39
2.1.2 Historical overview	40
2.1.3 Mathematical tools and notations	42
2.1.4 Categorization of homomorphic schemes	42
2.1.5 Main theoretical and practical issues	43
2.2 Partially homomorphic schemes	44
2.2.1 Goldwasser-Micali	45
2.2.2 ElGamal	46

2.2.3	Benaloh, Naccache-Stern and Okamoto-Uchiyama	48
2.2.4	Paillier and derivatives	50
2.2.5	Implementations and practical considerations	52
2.2.6	Somewhat homomorphic schemes	53
2.3	Fully homomorphic schemes	55
2.3.1	Overview of FHE	55
2.3.2	Three different types of FHE schemes	58
2.3.3	Schemes over ideal lattices	59
2.3.4	Schemes over the integers	60
2.3.5	Schemes based on the (R-)LWE problem	62
2.3.6	Implementations and practical considerations	68
2.3.7	Future of FHE	71
2.4	Discussion	72
2.4.1	Security of homomorphic encryption	72
2.4.2	Partially or Fully homomorphic encryption ?	74
2.4.3	Evaluating a program on encrypted data: in practice	74
2.4.4	Advanced notions: other homomorphic constructions	75
2.4.5	Link with Functional Encryption, Garbled Circuits, MPC and Searchable Encryption	77
	Conclusion and perspectives	81
	What can homomorphic cryptography do for privacy?	81
	HE-based privacy in the literature	82
	What's left to do?	84
	Concluding remarks	86
	Bibliography	87
	List of Figures	107
	List of Tables	108

Introduction

Context In modern societies, the notion of privacy and the question of its protection has been growing along with the development of individual liberties. Several events in the 2nd millennium can attest of this momentum: the Age of Enlightenment and its writings, the French, British and American revolutions/declarations of independence, the Declaration of the Rights of Man and of the Citizen, and very recently, the Arab Spring in Tunisia, Egypt, Libya, Syria and other countries of the Arab world. All these events have in common to increase the importance the individual in society. Still today, achieving adequate balance between individual liberties and life in community is a dominating issue in the public place organization.

As described in the 1948 *Universal Declaration of Human Right of the United Nations*, privacy is a *right* of every individual. The first appearance of privacy as a right dates from the late XIXth century, shortly after the invention and development of photography. According to Warren and Brandeis, it is the continuation of the right of property [WB90].

Because it stems from social considerations, the perimeter of privacy depends on cultures, habits, local history and individual sensitivities. Delimiting privacy is far from trivial, especially because this notion is in constant evolution in the public opinion. As the EDWIGE (in France) or PRISM (mainly in USA) scandals in recent history attest, delimitation of privacy is performed in an ad-hoc manner, *i.e.* by fails and retries. Note that these “scandals” also show an opposition between national security and privacy. Once defined, privacy is neither easy to enforce, even with legal, technological and economical resources. Furthermore, gaps between reality and the proposed legal or technological theorization are always possible.

Issues The advent of information technologies in the second part of the XXth century vastly modified and extended the shape of privacy. In 1890, Warren and Brandeis [WB90] were already concerned that the growing easiness to take pictures would put everyone under constant surveillance and make true the prediction “what is whispered in the closet shall be proclaimed from the house-tops”. We are now far further from this truth, mostly due to the development of Internet along with computational and storage capacities. We are now able to automatically and systematically collect, store, transform, share and duplicate information at will. Furthermore, information can be stored for decades, centuries, or even hundred of centuries. While this is a great advance for information processing, and may help in decision making or facilitate the duty of remembrance, it turns out that one of the most valuable type of information are the individuals’ personal ones. This is where individuals privacy is endangered.

As the public rejoices in the new possibilities offered by technology advances, it is also concerned by the privacy loss incurred (when aware of it). Unfortunately, it is easy to see that the most known and used internet systems (e-commerce, social networks, web search engines, ...) are not privacy-friendly. We could even say they benefit from the privacy losses of their clients, to a certain extent. It falls down to the information technologist and the legislator to inform individuals on the dangers they are exposed to, and to provide protection of their privacy in

information system. Although we will only consider the information technologist and not the legislator in this document, these two actors have to work in pair and in parallel to enforce the protections.

Note that there is an important difference between privacy and information security. Security is based on 3 pillars: confidentiality, integrity and authenticity of data. The last two are orthogonal to privacy (although not incompatible), and confidentiality, as it is now widely known, is not enough to ensure privacy. In other words, even though confidentiality is necessary for privacy, it is not sufficient on its own. For instance, in network communications, even if the *payload* (*i.e.* the message) is encrypted, an entity observing the network can easily see who communicates with whom. That is a problem when, for instance, an individual accesses a sensible website, such as aa.org or wikileaks.org.

Related works The same technology advances that endanger privacy can actually be put in profit of privacy-preserving solutions. Indeed, many works in information security focus on designing or using advanced tools in order to adapt existing functionalities (*e.g.* social networks, internet communications, ...) to a privacy-preserving setting.

In this sense, cryptographic tools play a very large role in many privacy-preserving system propositions. Encryption and signature schemes are already widely used to ensure confidentiality, integrity and authentication, but can also be used for the larger task of protecting privacy. Trust management is another important component: many solution involve a trusted third party playing the role of a proxy between a client and a service provider, decorrelating the client's queries and data from its actual identity.

However, in many contexts, such as storage or computation delegation, basic cryptography proves itself insufficient. We have to resort to more powerful primitives. Homomorphic cryptography is a powerful, relatively dawning paradigm that might fill the gap. It enables manipulation and computation on encrypted data, without the need to access the underlying data. In other words, a possibly untrusted third party can be securely entrusted with the task of computing complex processes on sensible data. Homomorphic cryptography is, in particular, perfectly suited for computation delegation to the cloud, but is also relevant in many other applications and resolves many information security issues (although sometimes in an inefficient manner).

Approach and Objectives Our main and ultimate goal is to protect the privacy of individuals in information systems using homomorphic cryptography. This means proposing new solution for particular systems where privacy needs to be (re-)enforced and where homomorphic cryptography is a convenient tool. This can also mean drawing a general framework for "homomorphic cryptography based privacy", *i.e.* a generic privacy solution for many systems.

We intend to focus on systems where privacy is crucial, *i.e.* where the individuals *need* to use a system but are refrained by privacy leaks leading to serious consequences. As an example, we consider that reporters inside a totalitarian, censoring regime do need a way to communicate with the outside completely privately and confidentially, safe from the risk of being exposed and condemned by their government. We oppose these kind of systems to those where the user could actually avoid using them, at a reasonable cost. This is the case of social network, where users decide on their own what information they disclose and could make the choice to keep all information secret. Of course, these systems are nowhere near perfect (in terms of privacy): among many things and in particular, users are ill-informed, and the service provider often require the users to give up their property rights on the information they disclose.

Our motivation to focus on the former kind of systems is based on the fact that privacy/efficiency trade-offs are always necessary, and because of the possibly large overhead that imply the use of techniques to protect privacy (and homomorphic cryptography in particular), we believe our

solution(s) will be more easily be accepted and used by individuals if the need of privacy is high or critical.

Organization of the document The rest of this document is organized as follows. In chapter 1, we detail the multiple facets of numerical privacy. We propose a synthetic view of privacy solutions in information systems, and review the fields of computer science where privacy is relevant. In chapter 2, we move on to the presentation of homomorphic cryptography. More specifically, we focus on homomorphic *encryption* schemes: we categorize them according to their homomorphic capacities, describe each class, and review the schemes they are composed of. Both chapters are concluded by discussions and openings. In particular, we question the need for privacy and we present other homomorphic tools than encryption (*e.g.* signatures). In our conclusion, we re-unite privacy and homomorphic cryptography: we give clues as of the way to use the latter in order to protect the former.

Chapter 1

Privacy

This first chapter draws a view of the notion of privacy in computer science in general before taking a closer look at privacy from a technical and technological point of view. Although it takes into account the social dimension of privacy, this chapter leaves aside all social reflexions and does not form a general, complete definition of privacy as understood in a very broad sense: we mainly focus on privacy in the digital world and *numerical privacy* in the rest of this document. This chapter notably reviews the different approaches to privacy and sectors in which privacy is considered desirable or necessary, along with the technical tools provided to the computer scientist willing to take privacy in consideration inside information systems. The last section discusses the facts and results introduced in the chapter and explains the *why* of privacy.

1.1 Proposed definitions

The term “privacy” has been used to refer to multiple notions and concepts, even within the field of computer science alone, and has not always been clearly defined by those using it. We first give attention to the general meanings and acceptations of the term, focusing on *numerical privacy*, *i.e.* privacy in the digital world. After drawing several overlapping but different definitions for this notion, we give a practical way to consider privacy in a composed but synthetic view.

1.1.1 Several meanings

In a first approach, the notion of privacy can be defined aside from any technical considerations: its meaning is intuitive, known to anyone and appears in every day life. Indeed, each individual is most likely to be willing to keep some information about himself or his (past) actions private to himself or a set of adaptively chosen persons. One may want to hide information from journalists, co-workers or the general public, conceal the contents of a communication, or even the very fact that he communicated with a specific entity. For some, privacy can also mean spending time alone or with chosen company to dedicate to certain tasks such as (self-)reflexion or hobbies. Others mean by privacy the possibility to take actions without having to justify them. The concept, even though it seems at first glance intuitive and well understandable, has many facets when considering the plethora of *contexts* in which privacy is crucial, along with the *means* employed to ensure it. In particular, in the digital and ubiquitous computing world, where individuals do not have access or do not have the proper knowledge to understand complex systems, the concept of privacy is not trivial and comprises many dimensions. Several authors tried to give a

definition for privacy in general, which are often partial or far from the above naive and intuitive definition.

Already in 1967, Allan Westin was worried about privacy in the emerging computer society, when electronic devices were becoming more and more accessible. In its seminal book, *Privacy and Freedom* [Wes67], he lays the first elements of reflexion on the conflict between surveillance and privacy: he forms the idea that massive data collection and development of surveillance devices may harm individuals' privacy. To his mind, although privacy was (and is) an essential and necessary notion for the individuals' autonomy and liberty, it had never been well defined in social theory, and most studies on the subject were vague and confused. His proposed definition reduces privacy to a right and is centered on *information flow control* :

“Privacy is the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others.”

This very general definition has been widely used as a starting point, but as we can see it does not encompass all the examples listed above. Another approach, by Sara Baase [Baa02], defines 3 guarantees that the notion of privacy should imply: (i) freedom from intrusion, (ii) ability to control information dissemination, and (iii) freedom from surveillance, *i.e.* from being tracked, followed, watched. This definition is more complete as it does not *totally* reduce to information flow management. However, we can see points (ii) and (iii) as one and the same (*i.e.* freedom from surveillance is, in definitive, the ability to avoid unwanted information disclosure), and they are more or less equivalent to Westin's definition. The new element here is the freedom from intrusion in the sense of being “left alone” when desired. The definition of privacy is thus augmented compared to Westin's but also more complex. Both definitions are however very abstract and uneasy to work with.

In the field of computer science, privacy needs a practical definition that can be expressed as a formal goal. Following Pfitzmann *et al.* [PK01], privacy is defined by *anonymity*, the state for an individual of being not identifiable within a set of individuals (the anonymity set), and/or the property of *unlinkability* between an individual and its actions or among the actions of a same individual. They are notions specific to computer science, and we can see they differ from Westin and Baase's definitions (although they are not conflicting) as they are much more concrete. The danger using such specific terms is to forget a large portion of what “privacy” means: Pfitzmann's terminology might not embrace all the meanings of privacy.

As we can see, definitions of privacy are either very abstract and unusable, or very specific and possibly incomplete. One of the most relevant and complete definition of privacy, to our knowledge, is depicted in the taxonomy of Daniel Solove [Sol06], which is also a reflexion on the social meaning of privacy:

“Privacy is the relief from a range of kinds of social friction, [...] it is protection from a cluster of related activities that impinge upon people in related ways.”

“Privacy is too complicated a concept to be boiled down to a single essence. Attempts to find such an essence often end up being too broad and vague, with little usefulness in addressing concrete issues.”

Solove's point of view is, unlike ours, mainly social, and his taxonomy lists the possible privacy harms an individual might suffer in society. He clears out 4 main categories that encompass many (all ?) social meanings of privacy: *information collection*, *information processing*, *information dissemination* and *invasion*. This categorization, depicted in Fig. 1.1, is still driven by information flow control, but in the details it takes into account the social context (a context

that might be very different from one country/culture to another) to create a more complete map of (social) privacy, close to what anyone can experiment by himself. In this document, Solove also insists on the fact that privacy is a complicated concept that refers “to a wide and disparate group of related things”, and underlines the fact that using such a broad term must be done with caution in order to avoid confusion and distraction from real issues. Indeed, privacy is very often confused and reduced with a *fraction* of what the word refers to. In many occasions in computer science, for instance, authors of works on the anonymization of databases containing sensible information (*e.g.* medical databases) use the general word to name their field of work, “privacy”, instead of using the more specific field name, *e.g.* “database anonymization”.

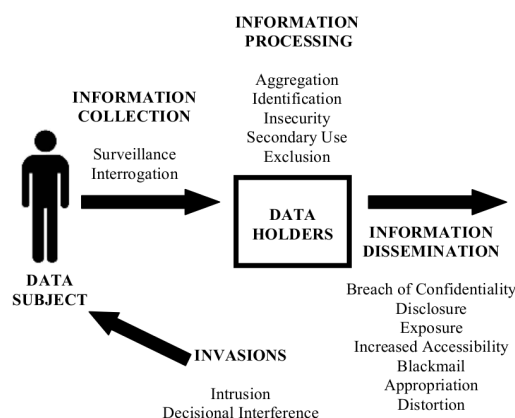


Figure 1.1: Solove’s taxonomy of privacy torts from a social point a view

Solove puts in light the composed nature of privacy, and the need for a careful approach of the field. If privacy is summed up in one phrase, the better it can achieve is to give a very “high level” and abstract definition, which will always be subject to interpretation and largely differ in meaning depending on the reader. On the contrary, a practical and formal definition easily reduces privacy to a fraction of what it refers to. We intend to proceed analogously to Solove when describing the field of *numerical* privacy and trying to obtain a synthetic and complete view of it. We need a way to explicit the many facets of privacy in computer science in order to avoid simplifications. In the subsequent sections we try to sketch a more systematic and concrete way to embrace the notion of privacy.

1.1.2 Multiple dimensions

As we can see, the notion of privacy is composed, we can put forward several definitions for the word, and it is easy to leave aside and “forget” a whole part of the field it represents. To be convinced of this, one simply needs to read from different sources: although most of the literature in privacy use the same word and follow the same abstract goal, the meaning of the term will largely differ when looking closely at the implicit hypothesis made.

We begin by giving here elements that can be used to characterize privacy-enhancing systems or “privacy solutions”, or more generally all works in numerical privacy. The motivation for this list is to have a better view on *how* and *by what* privacy works differs. We do not mean to clear out a new definition of privacy, as we saw the notion is too complex to be reduced to a few phrases, but rather to draw a synthetic and concrete view of it. More generally, the below list gives the different dimensions of privacy, aims to be sufficient to fully characterize a large portion

of works in this field, and will be used in the following section to divide the field of privacy in several classes.

Approach We claim that the most important and the first characteristic of a work in privacy is the chosen approach, that is, the general angle under which we consider privacy. In other words, the approach is the first *implicit* choice we make when designing a privacy solution, and will be omnipresent in every following choices. For instance, when defining privacy as a right and trying to ensure it by laws, one uses a *legal* approach, whereas managing numerical data and designing information flow control system takes place in a *technological* approach. More concrete examples are given in the next section.

Sector The second most important characteristic of a privacy system is the sector it addresses, *i.e.* what “real-life” privacy problem it considers and tries to solve. Examples of sectors include geolocation, authentication, social networks, or data mining.

Privacy Goals Given the approach and the sector, privacy solutions will then differ by their goals in term of privacy. This characteristic put in light *what* privacy properties the solution will provide, and possibly details *how much* privacy is guaranteed (*i.e.* gives a quantification for the privacy properties).

Tools Another important feature to characterize and classify a privacy work is the tools it uses to achieve the given privacy goals. Tools largely depend on the chosen approach and sector: one may use cryptographic and mathematical theories when working in computer science, as in the legal approach it would not make sense.

Assumptions and Models Although these are not always explicit in the literature, all solutions make assumptions on the context or on the parties, and use more or less formal models. This is a characteristic in the sense that when searching for privacy solution for a given problem, one must be cautious of the assumptions made by the authors of the system in order to avoid incompatibilities between the authors’ intention and the real usage.

Efficiency When choosing a privacy solution, and when all other characteristics are equal, the most efficient will often be the most suitable. Thus efficiency is an important characteristic. Note that here efficiency does not necessarily mean fast computability, and we use this term with a broader sense. As the evaluation of the efficiency largely depends on the chosen approach (*e.g.* efficiency relates to easy applicability in the legal approach), what is meant by this term is to be defined when designing the privacy solution.

This list could go deeper in the details, for example taking into account the number and nature of the entities interacting in the system and what protection each of them is guaranteed, but we claim that with the above-listed elements, privacy solutions and privacy-enhancing systems are sufficiently characterized. We go even further, claiming that with only the first two items of the list, we have enough information to obtain a detailed and synthetic view of the term “privacy”. The next section develops this idea.

1.1.3 Trial for a synthetic view of privacy: Approach \times Sector

From the list above we have at our disposal (almost) all the information needed to characterize a privacy solution. We could use those criteria, possibly with additional ones in special cases, to categorize and create a detailed taxonomy of privacy solutions. However, this is not our intention

here. We neither intend to deliver a definition for the word “privacy”, as many have already been proposed and are sufficient for a high level view of privacy. Instead the aim of this section is to clear out a way to embrace the field of privacy in one unified view that would be practical and usable, as a roadmap for categorizing privacy works. This categorization is important to obtain a clear view of the field of privacy, to be able to efficiently choose a privacy solution for a specific application, and lastly to avoid comparing privacy systems that are not comparable (*e.g.* comparing a technology for anonymous internet communications and a law enforcing the notion of *consent* is absurd).

In our criteria list, we already pointed out that some criteria are correlated and that relations between them exist. One of the most obvious relation is the dependence between the *approach* and the used *tools*: using a technological tool in a legal approach does not makes sense, just like law theory does not have its place in a technological solution, although there are links between the two and they must be considered altogether¹. The definition and quantification of *efficiency* also depends on the chosen approach: an efficient law will be an easy to apply and easy to enforce one, but a efficient technical solution will be a fast computable one (for instance). Another example is the choice of the *privacy goals*, that are largely correlated with the chosen sector: if the sector is a privacy-preserving geolocation, the goal may be to hide the user’s location among a group of individuals, which does not makes sense as a goal in private biometric authentication.

In the light of previous paragraphs, we argue that some criteria are more important than others, and a small subset of criteria determines the others. We would like to find a *minimal* number of criteria that are sufficient to determine all the others, and to create classes from this minimal set. Solving this problem would require to acquire knowledge of all works on privacy and characterize them precisely and without mistake. As we do not have this knowledge at hand and because the time necessary to obtain it is way too large, the best we can do is to draw an approximate answer from a non-negligible (hopefully) representative subset of privacy works. Thus we pick few criteria that *seem* to determine others according to our knowledge, which is mainly driven by computer security considerations. The set we propose to consider is { *approach*, *sector* }.

First, to justify the number of criteria in our selection, we argue that to perform a trade-off between ease-of-use and accuracy of our categorization, it is necessary to select only a few criteria (*e.g.* 2 or 3). Indeed, consider for instance using some criterion $crit_1$ to categorize all privacy works in the set Ω . If $Nval_{crit_1}$ is the number of possible values for the criterion $crit_1$, then the space is divided in $Nval_{crit_1}$ classes of size $\approx |\Omega|/Nval_{crit_1}$ ². And if $Nval_{crit_1}$ is small we obtain a few large classes, which allows a simple and fast prune when searching for a privacy solution for a specific application, or when “inserting” a new privacy work in the categorization. Then, to continue the example, if we use a second criterion $crit_2$, we obtain $Nval_{crit_1} \cdot Nval_{crit_2}$ classes of size $\approx |\Omega|/(Nval_{crit_1} \cdot Nval_{crit_2})$: the “pruning” is more effective, however the number of classes grows and so does the complexity of the categorization. With more criteria, we obtain better pruning but more classes and a more complex categorization. We prefer using a moderate number of (large) classes, as it eases the categorizations of privacy works each in its right place, without error. Indeed, many classes means many details to investigate in each privacy work, thus many factors of errors.

Then, the choice of the specific criteria is very important and mainly depends on what information is needed: when searching for *any* efficient privacy solution, of course, a categorization

¹Even though a law may recommend the use of a certain technology in its text, *e.g.* some cryptographic scheme, it will not actually *use* it and it does not look into the details of the scheme. Same goes for the opposite case: a technological approach will not consider using EU directives or governmental institutions to achieve its goals.

²It is true if we suppose that the privacy works are uniformly distributed among the $Nval_{crit_1}$ values of $crit_1$.

with the *efficiency* criterion is most suited. However, this example is not likely to be realistic. Rather, one will often or always have an *approach* and a *sector* in mind when searching for or designing a privacy solution. This is a first element that designates these two criteria over the others. Also, as another element in favor of this choice, we showed (but did not prove) earlier in this section that the couple (*approach*, *sector*) almost completely determines the other criteria of our list. Then, depending on the need, one can for instance append as third criterion the *tools*, when the desired tool for a particular application is chosen beforehand. Furthermore, for the sake of consistency and coherence, we would like that privacy works within a same class share similar characteristics so as to be able to compare them to each other. Precisely, most of the privacy systems within the same (*approach*, *sector*) class are comparable without introducing absurd comparisons. On the contrary, when considering systems from different approaches or from different sectors, there is a great risk of incompatibilities preventing comparisons. Lastly, we will see in section 1.2 that the number of approaches is limited, *i.e.* $N_{val_{approach}}$ is very small), yielding a small number of large classes. The number of sectors being moderately large, we achieve a trade-off between number of classes and accuracy. This explains our specific choice for these criteria.

Using the couple (*approach*, *sector*) we can divide privacy works in wide classes. Although those still contain a lot of diversity, it is a first sorting that allows to immediately have a synthetic view of the field of privacy. This result is of course not exact and questionable (see section 1.5.1 for a discussion on this matter), but we find it appropriate for an efficient categorization. It can be considered as a trade-off between a complete and accurate but very complex and heavy categorization that puts every privacy system in its own class, and a partial but efficient one (as for the exactness, both categorizations may be flawed independently).

The next two sections give a better insight of our proposed view of privacy by listing the possible approaches and existing sectors.

1.2 Different approaches to privacy

By *approach* to privacy, we mean the highest level angle under which privacy is considered, or more exactly under which the field of *numerical privacy* is considered. When designing a privacy solution, this aspect immediately implicitly chosen. In some cases the approach need not to be made explicit and is well understood, but in many occasions, authors use the word “privacy” with a specific approach in mind, assuming the reader will be in the same state of mind, which is not necessarily the case. Thus clearing out the different approaches is crucial, and in order to avoid confusion, one should always specify which approach is considered when speaking about “privacy”.

Note that enumeration in this section is inevitably tainted by the computer science oriented background of the authors and may not reflect the true nature of each approach.

1.2.1 The legal approach

Already mentioned several times, the *legal approach* is the easiest to picture. In this context, privacy is considered as a right and legal practitioners intend to ensure individuals’ or entities’ privacy using legal means. This can take the form of laws created by governments of course, but many other tools are widely used, such as *directives* (*e.g.* EU directives 95/46/CE [PtC95] and 2002/58/EC [PtC02]), federal laws in the US (*e.g.* Privacy [oE95] HIPA [UC96] Acts), international agreements (OCDE guidelines [fECOD13] or UN resolutions [Nat90]), and governmental institutions (*e.g.* *Federal Trade Commission* in the US and its *Fair Information Practice Principles* [Com00], or the *CNIL* in France).

We give here a very brief overview of the legal approach of privacy in the sector of computer science and information processing, as it is the one we are interested in. We leave aside all discussions about private property and “non-numerical privacy”. In modern society, the fact that privacy is a fundamental right of every individual is widely admitted. It is written in the *Universal Declaration of Human Right* of the United Nations of 1948, the right to privacy has been present in most national legislations since the XVIIIth century [Pio09], and it was adapted along the years to the evolution of information processing. Very broadly speaking, in the numerical privacy sector, modern laws mainly focus on the notion of personal information (*e.g.* the term used in US law is “Personally identifiable information”), and do not include (so far, and to the best of our knowledge) more complex privacy notions such as hiding the fact that *Alice communicated with Bob on date d* or *who are Alice’s friends* for instance. Generally, laws define what is and what is not considered as personal information and try to prevent or reprimand disclosure of individuals’ personal information [Ohm09]. Other points of interest in legal considerations include [Pio09]: the *information* of service users (*e.g.* user needs to be informed on what and how data is collected), the *consent* of the user (*e.g.* explicitly ask the user for some permission before performing an action), and the right of the user to *access and rectify* its information. They also regulate the *duration* of data holding by the service provider, the *communication* of these information to third parties, and the *motive* for collecting and manipulating specific data. All those enumerated points can be summed up in two main principles: *sovereignty* of the user over its data, and *minimization* of the collection of data. However, these laws are always mitigated by others allowing the disclosure of such information for judicial or “national security” reasons.

It is important to note that the blazing fast evolution of technology during the past decade has left the legislation behind: laws are not always adapted to the reality of information processing. Notably, many flaws allow information broker to collect and sell particular data without real legal framework. Indeed, some data are (rightfully) considered by the law as personal to the individual because they allow to directly identify him but recent works in privacy showed that almost any information can be used to indirectly identify an individual, such as the history of one’s web searches [CCP10] or movie ratings [NS08], and laws need time to take these results into account.

For a more complete history of the privacy laws in the United States, the reader might refer to a reflexion about law on personal information and *database anonymization* by Paul Ohm [Ohm09]. Concerning the European Union and more specifically France, see section 1.1.2 of Guillaume Piolle’s thesis [Pio09].

1.2.2 The policy approach

This second approach is quite close to the previous one and both are often presented as one. Actually, in a sense the policy approach is the continuation of the legal approach: it formalizes legal (or more generally, human-readable) statements and transform them into machine-understandable requirements. The goal in this approach is to ensure that some properties, expressed via policy statements, are observed by systems, and optionally to build tools that take human expressed properties as input and formalize them in applicable properties for information systems. This approach is thus also close to a technical approach in the sense that it uses computer technologies to enforce policies. However, in what we call the *policy approach*, pure technological solutions may be *used*, but not *invented*. Actually, when a framework for describing privacy policies is designed, how to enforce the policy with technological tools is not always considered, but comes at the end of the reflexion, sometimes as an optional step, to transform policies from simple “information” on how to handle data, into verifiable system properties. Although the policy approach is in between two other approaches, it is considered here on its own because this step

is necessary for building a bridge between law and technology. Moreover, the complexity of designing and composing policies being great enough, it deserves a special category.

When defining a policy or a “law to policy translator”, several challenges arise, mainly from the fact that the requirements and privacy properties aren’t expressed by computer experts but by legal ones. Indeed, those requirements must be transformed into machine-understandable properties without losing their original meaning, which is far from trivial. Some research focus specifically on that issue [ABSB⁺11, PD11], using deontic logic to express privacy requirements. Another common issue is the management of conflicting privacy rules or requirements [HAJ11] within a set of policies applied to a system. For instance, in a social networks where each user would be given the possibility to express privacy rules for its own data, when Alice publishes a picture depicting Bob and Charlie, how does the system handles the publication when Alice wants to make the picture public, Bob wants to disclose it only to his close friends, and Charlie demands this specific picture to be hidden.

In practice, in the particular sector of web applications, expression of privacy policies is done with the now widely used P3P (Platform for Privacy Preferences Project) standard from the World Wide Web Consortium [Con06]. This standard allows web site to specify their data processing and collection policies in XML, via a *P3P policy* accessible through a simple URL, so that it can be read and interpreted by web browsers. The policy is then presented to the user or even automatically compared to its privacy preferences. Another well know tool available in the policy approach and for various sectors are the *sticky policies* [KS02], which are integrated in pieces of information as meta data and specify the allowed uses of the information.

However, none of the two technologies ensure the policy is respected: P3P standard does not check the enforcement of the policy server-side, and sticky policies are just information on how the data *should* be used. In both cases, there is a need for a mean to enforce the observation of policies. The basic solution is to suppose that the client *trusts* the server to do what as it says it will do, and a more complex is to perform an audit by trusted experts such as government institutions. In all cases, the means to enforce the policy lie in the *technological approach*.

1.2.3 The technological approach

The angle to approach privacy we are focusing on in this document is the technological one, where privacy researchers try to enforce privacy via pure technological means, often with cryptographic, mathematical or hardware-based tools. We do not extensively describe this approach here, as the next sections and this whole chapter are dedicated to this purpose. Section 1.3 describes existing sectors and general tools of this approach.

However, we detail here two ideas belonging to the technological approach. First, describe what seems to be the *modus operandi* of every privacy work in this approach (it can possibly be adapted to other approaches). Indeed, it seem that most of the works in privacy are constructed in 4 main steps:

1. Choose a pre-existing system that seems to “need” privacy, and study it;
2. Investigate and decide what should be “private” or not;
3. Modelize and formalize these requirements into properties of the system;
4. Enforce these properties by technological means (e.g. adapt the infrastructure or the protocole, use cryptography or a trusted third party, ...)

We will roughly follow the same procedure in our constructions.

Secondly, we would like to point out here two “sub-approaches” that specify a bit more *where*, *how*, and *to what purpose* technology is used. Note that the separation between the two sub-

approaches is not always clearly defined, and they can be seen as a high level characteristic of (technological) privacy solutions.

Service-centric approach

In this case, privacy technologies are used to enable privacy in existing services, with as less impact on the service provider as possible. For example, designing privacy-preserving targeted advertising or geolocation-based services are within this scope. This approach often takes an existing service, and slightly modifies its implementation to take clients' privacy into account, without substantially modifying the functionality or questioning it. In this case, modifications are often needed on client *and* service side.

User-centric approach

On the contrary, in this case the user's privacy is the first and primary goal. No specific service is considered, and the aim is to fully protect the user and give him control over his "privacy". The only assumptions are that the user do not wish to give away (any) information or more generally compromise his privacy, and is even willing to refuse some services if their privacy terms do not match his requirements. If possible, in this approach, only the client-side is modified and only the user needs to take action, so that his protection only depends on his choices and the technology used, and not on the services accessed. The philosophy of this approach is to allow the user to have "unconditional privacy", *i.e.* to provide the user means to protect his privacy independently of the entity it is in contact with (*e.g.* a service, a system, ...). In other words, "unconditional privacy" refers to the ability of the user to protect its privacy even if no measures have been taken service-side. Achieving this kind of privacy is not always easy, and achieving it with only client-side modifications is almost impossible (there is often the need to make adjustments to the environment, if not the service). Of course, a trivial solution is for the user avoid any interaction with any entity, but we reject this solution as we want the user to still interact with the world and perform meaningful actions.

1.2.4 The economics approach

Finally, we argue there exists a fourth approach to privacy, the one called *economics*. As the name suggests, this approach is actually mainly business-oriented and uses market laws to characterise privacy "transactions". Its starting point is the emerging business model where users do not directly pay for a service, but the service provider collects and analyses their personal information. At first, it seems like a nice solution for the user as the cost of the service is reduced, and for the organisation as it still creates wealth. Society would also benefit from it (in the contemporary economic paradigm), under the form of increased growth and wealth. In the economics approach, works start from these observations, try to keep in mind that users might not actually want that business model if their privacy is endangered, and investigate to what extent the market is efficient to ensure user's privacy. Indeed, the main assumption here is that users manage their privacy the same way they would manage their physical goods and money: individuals choose to trade (or not) private information or a fragment of their privacy against advantages of other nature. Following the definition of Acquisti *et al.* [AG05]:

"According to that view, individuals are forward-looking, utility-maximizing Bayesian updaters who are fully informed or base their decisions on probabilities coming from known random distributions."

The “*utility-maximizing*” term means that the user *intends* to maximize its total benefits, defined via criteria left to his choice. For instance, if, in the opinion of some individual, privacy prevails over (small) financial advantages, he should choose to refuse a fidelity card in a shopping center, a card that often mentions name and address of the bearer and tracks his purchases, but also proposes reductions and special offers on shopping items. “*Forward-looking*” implies that individuals try to maximize their “utility” in the medium or long term. The notion “*Bayesian updater*” models an individual that is able to make rational decisions based on *a priori* beliefs, and to improve his beliefs at each new event (here, an event is a transaction). In practice, this means an individual makes rational decisions on how to handle a specific privacy transaction based on his expectations of the outcome of the transaction and his past experiences. The last part of the definition supposes the individuals are fully informed, in particular on the consequences on the transaction for all parties, or at least have access to knowledge common to all parties.

Privacy solutions in the economics approach may propose ways to correct the aforementioned business model or the individuals’ behavior model, mitigate users’ privacy loss and try to achieve balance between individuals and data holders, notably via law and technology. Typical questions raised in this approach are for instance “*Who owns the data: the user or the collector ?*”, “*Have the user consented, and do we need explicit consent ?*”, “*Do we really need regulation through legal means, or is the market efficient to achieve balance in the case of privacy ?*”.

One of the main points of interest is the validation (and correction if necessary) of the “*Bayesian updaters*” model for individuals described above. Several works question the model and suspect it does not match reality [AG05, GA07, CS07]. Notably because of information asymmetry between users and service providers, but also because individuals are rarely aware of privacy risks and technological possibilities to prevent them. Other factors [AG05] include the humans’ *bounded rationality* which limits his ability to acquire and use information, and “psychological deviations from rationality” inherent to human nature (that for instance imply time-inconsistent decisions where a lesser immediate good is chosen over a greater good further in time). Some reflexions also speak of a “market failure” in the context of privacy [Acq12], that is to say they wonder if the market laws are applicable directly, and if the market model is valid in this context. On this matter, Acquisti gives some reasons for “hope” and other reasons for “concern” [Acq12]. Briefly, reasons to be concerned by the ability of the market to regulate privacy transactions are: the unprecedented easy access for third parties to a large number of individual’s personal life aspects (often without the person knowing it), partly because of the development of communication technologies; and the fact that giving more control to the users as it is often recommended by privacy advocates would be a false solution because users are unaware of the value of their information (it even sometimes leads to less privacy [WCS+13]). Reasons to hope are that human seems to naturally need publicity *and* privacy, and this tendency should appear in the long term; research on behavior in economics of privacy put in light psychological bias, but can actually help erasing those bias by informing individuals about underlying psychological processes and contradictions; and finally hope lies in technological advancements and “*privacy enhancing technologies*”.

1.3 Different sectors of privacy in computer science

After exposing the main angles under which it is possible to consider numerical privacy, we continue by enumerating the possible sectors, in order to complete our *Approach* \times *Sector* view. However, we will see that giving an exhaustive list of sectors in numerical privacy is pointless, as one can envision privacy in all aspects of computer science and all systems are subject to ameliorations in term of privacy. However, there are some that are more relevant than others,

and more significant. We try, in this section, to lay out a succinct hierarchy and list of sectors in which privacy is relevant and those that research already took hold of.

For a more complete list of existing privacy issues in society and the digital age, see the list from the Privacy Rights Clearinghouse [Cle13].

1.3.1 Identity management and authentication

Authentication is necessary in many systems where there is a personalized user space or where actions of the user must be imputable to the individual owning the identity. In numerical privacy, we are interested in allowing a user to be anonymous *and* authenticate at the same time. Indeed, in many contexts the *true* identity of a user does not need to be known and systems merely need to distinguish him uniquely in the set of individuals using the service. Sometimes, an even weaker form of authentication is sufficient, for instance when the user and the service only need to recognize each other for a temporary session and the service does not need to “remember” the session. In this case the user can use a different identity (or pseudonym) for each session, and he can use the service without the service provider being able to *link* the different sessions to him. On the contrary, the service provider might specifically want to impute malicious actions to a user in order to ban him from the service. In this case we speak of *accountability* of the user toward its actions. As an extreme example, we can imagine a user undertaking criminal actions while being anonymous: accountability is necessary in order to find the individual responsible for the actions and to prevent anonymity abuse. Although they seem like orthogonal requirements, accountability and anonymity can be guaranteed at the same time, *e.g.* via revocable anonymity [KWF06].

All the cases informally described above can be expressed with the notion of *linkability*: linkability between a user and its actions, and between actions of a same user in a system. The notion (*un*)*linkability* is crucial, and it is a widely accepted term used to define and quantify privacy properties. It has been defined by Pfitzmann *et al.* [PK01], along with the terms *anonymity*, *pseudonymity* and *unobservability*, all of which are relevant to formalize privacy properties:

Anonymity is the state of being not identifiable within a set of individuals or users, the anonymity set.

Pseudonymity is the use of pseudonyms as identities. Pseudonymity can be seen as anonymity with accountability: users do not use their real identifier to perform actions, but a pseudonym that may have a known connection with a real identifier. There are several degrees of pseudonymity (see below).

Unlinkability of two items (*e.g.* users, actions, messages, events, ...) in a system means that within this system, these items are no more and no less related after a “run” of the system than they are related with respect to the *a priori* knowledge of an observer. Note that anonymity is the unlinkability between an action and a specific identity, and stronger unlinkability leads to stronger anonymity.

Unobservability is the property of a system to produce unobservable events, *i.e.* events either undistinguishable from any other event, or events undetectable by an observer.

Unobservability being a very strong notion, difficult to achieve because often implying very inefficient constructions in practice, we do not consider it in this section. As anonymity and pseudonymity can be defined in terms of unlinkability, we mainly focus on the unlinkability. Thus we can restate the main objective of privacy-enhanced identity management in terms of unlinkability: the goal is to reconcile authentication, accountability *and* unlinkability.

We distinguish here two main kinds of unlinkability: unlinkability between the user and its actions, which is equivalent to anonymity, and unlinkability between actions of the same user (without knowing *which* user performed them). In many occasions, the two kinds of unlinkability are necessary because from a “sufficient” number of actions known to be related to a specific user, it is possible to infer the identity of the user. The main known method to ensure both types of unlinkability is using different virtual identities, *i.e.* pseudonyms. Depending on the necessary level of unlinkability, a user is provided with different types and numbers of pseudonyms [PK01]. The knowledge of the linking between a pseudonym and the identity of the user using it determines the degree of unlinkability between a user and its actions: a public linking leads to total linkability, a linking known by some specific entities (such as trusted third parties) can lead to revocable anonymity, and a linking known to the user only gives total unlinkability. Then, how and how many times a pseudonym is used determines the linkability between several actions performed by the same user: providing one unique pseudonym for one user leads to total linkability, providing as many one-time pseudonyms as the number of action of a user leads to total unlinkability. In between are 3 alternative solutions: one pseudonym for each *role* the user assumes in the system, one pseudonym for each entity the user is in contact with, or one pseudonym per *role and* entity. The favored approach is to give knowledge of the linking between pseudonym(s) and identity of the user to some trusted third parties so as to enable revocable anonymity. As for the number of pseudonyms and their usage, the choice depends on the available resources and the privacy goals.

It is important to note that the notions of unlinkability, pseudonymity and unobservability are relevant in many other contexts than identity management. In fact, Pfitzmann *et al.* [PK01] proposed this terminology for network communications (see section 1.3.3). In the rest of the chapter, unless specified otherwise, we refer to these notions as defined above.

Example: Biometric authentication We can illustrate identity management systems with the case of biometric authentication. Biometric data such as DNA or iris pattern are very sensitive data, but they are also very reliable for strong authentication as they are often very hard to counterfeit or steal. Thus, we would like to find a privacy-preserving way to perform biometric authentication. In this setting, typically, a user interacts with a secure device which has access to stored biometric profiles, and has the capacity to capture the biometric profile of the user. Authentication succeeds if the captured profile matches a profile in the database. Systems have been designed so that the biometric profile of individuals stored in a database are comparable with captured profiles, but the owner of the database can not link a stored profile to an individual’s identity (*e.g.* name and address). Other useful properties include the unlinkability between several authentications of the same user. Some solutions proposed applying distortions the profile using a one way function that will preserve the similarities between the stored and captured profiles [RCB01], others use fuzzy commitments and error-correcting codes [ZKVB11].

1.3.2 Personal information and data management

As it appears in the previous sections and in this whole chapter, the first, largest and most known sector is the one dealing with personal information management. Most of the works in (numerical) privacy consider this sector, and it is indeed an important one because of its broad sense. Indeed, personal data are everywhere and when interacting with systems, individuals inevitably leave numerical traces, either under a digital form (*e.g.* IP address or email) or “real world” information such as age, country, habits or name of friends. In many occasions, numerical privacy is (but should not be) reduced to personal data management, because of the predominance of the sector.

Even though it is possible to outline multiple sub-sectors of personal information management, there exists general principles that apply to the whole sector, principles that were partly exposed in section 1.2.1, as they are extracted from social and legal reflexions.

Sovereignty of the individual over its personal data: those data belong to him, and he should be able to access, rectify them and to accurately control how, when and who holds and processes them. Among other requirements, this principle implies the *right to be forgotten*, *i.e.* the possibility for the individual to erase all or a part of its numerical traces. When the data is on a distant server not controlled by the user, or in ubiquitous computing, this principle is very difficult to enforce.

Minimization of personal data: the individual should disclose only the information *necessary* to fulfill a given goal. For instance, for an individual to prove that he is adult he does not need to give his age or date of birth, but the only information he should disclose is “*my age is above 18*” (or 21).

Consent of the individual on its personal information collection, processing and dissemination. An *explicit* consent is preferred over an implicit, opt-out one.

Information and transparency: the individual should be able to accurately know *what* is done with his information.

Security against unauthorized access, modification or use is the responsibility of the data holder (and not the data owner, the individual).

Those principles should be enforced by technical means, and are often the starting points of new systems’ privacy goals of the “personal information management” sector. In the following sub-sections, we list some of its most representative sub-sectors, along with short descriptions.

Statistical databases and Big Data

This sub-sector has been given so much attention it could be a sector on its own. However, we argue it is only a small part of numerical privacy. By statistical databases, we mean databases containing demographic information and/or names of many individuals, along with sensible information, such as political opinion, religious orientation or health details. The typical model supposes one entry per individual in the database, composed of several fields for demographical and identification information, and other fields storing the sensible information. The typical example is a hospital database, where people are fully identified and a diagnostic is appended in a last field. These database are called “statistical” because of their main use: computing statistics on populations, and inferring relations between demographic parameters and some properties or events (*e.g.* correlating a specific location and an epidemic). “Big Data” refers to the management of very large statistical databases: as information storage and collection becomes easier every year, we are now in presence of extremely large databases and challenges arise in terms of processing, search, sharing and visualisation. We are not interested in these challenges, and we will consider Big Data and mere statistical databases as the same concept. We are only interested in the process of “curating” those objects in order to ensure the privacy of individuals in the database. Basically, this consists in perturbing or removing a (significant) part of the information contained in the database so that identifying individuals in the database is impossible. The challenge is to do so while leaving enough information in order for statisticians to be able to extract relevant knowledge from the database.

The extensive study of database curation dates from the 80s [Kim86] and can be explained by the need for companies, organisations and institutions to publish or share the data they collect on individuals. The collection is (as for now) legal, but the publication isn’t if the data can

be linked to the individuals it relates to. The publication or sharing of data is often motivated by the need to extract statistics and tendencies from them. It can be for public interests (*e.g.* analysing medical databases can predict epidemics) or for private profits (*e.g.* analyse implicit feedback from service users in order to improve the service). There are also companies that transform their data collection into profit by selling them, for instance to advertisers, for whom this information is crucial to offer better, targeted advertising.

We can distinguish two types of database curation: until beginning of 2000s was the era of database *anonymization* and *sanitization*, an approach that has been shown flawed, and since 2006, the technique called *differential privacy* has been the leading solution. To understand the “failure” of database *anonymization*, the reader may refer to Paul Ohm’s 2009 article [Ohm09] which describes very accurately anonymization techniques and why this approach is now considered obsolete. Here, we outline very briefly the ideas of the two techniques, and explain why *differential privacy* is favored over *anonymization*.

The basic idea behind database anonymization, or database sanitization is to remove from the database the information that could identify the individuals it contains. Several ways of doing so have been proposed, such as suppression of some database entries, randomization and perturbation, or query limitation (in nature or number). But the most known and widely used technique is called *k-anonymization*, invented by Sweeney in 2002 [Swe02]. The basic idea behind *k-anonymity*, besides removing individuals’ names, is to generalize the values of some well chosen attributes, called *quasi-identifiers*. Each entry of the database is processed so that for every individual contained in the database, there are at least $k - 1$ other individuals that share exactly the same values of quasi-identifiers. Before modifying the data, the database owner must choose the appropriate set of quasi-identifiers (*e.g.* demographic attributes such as age or ZIP code). These attributes should be those that can indirectly identify individuals. The database attributes are thus partitioned in 3 categories: the name of individuals (*i.e.* information that directly identify individuals), the quasi-identifiers, and the rest of the attributes. Those last attributes are called *sensible* information and may correspond for instance to the diagnostic in a medical database. Then the database owner removes the names of the individuals, and generalizes the quasi-identifiers in order to obtain a database partitioned in classes, each of them containing k entries sharing the same, generalized, quasi-identifiers before publishing it. Figure 1.2 shows a database ensuring the *k-anonymity* property. As it turns out, *k-anonymity* is not sufficient to avoid re-identification (*i.e.* linking a particular entry of the database to an identifier, thus to an individual). Two complementary variants have been proposed to cope with the flaws of *k-anonymity*, *l-diversity* [MKG07] (ensures a minimum diversity of the sensible attributes in each class of k entries) and *t-closeness* [LLV07] (ensures that the distribution of sensible attributes in each class is close to the global database distribution), but presented the same shortcomings in definitive. See Paul Ohm’s reflexion for more information [Ohm09].

In 2006, Dwork proved what was beginning to be admitted by the whole community: the assumption of the existence of quasi-identifiers was erroneous, in the sense that finding the correct set of attributes to play the role of quasi-identifier is not possible. In fact, several re-identifications, or *inference attacks*, such as Sweeney’s works [Swe00, Swe05] and the Netflix prize competition [NS08], led researchers to believe that *any* information is a quasi-identifier. This fact is particularly obvious with the Netflix inference attack [NS08], where, with only benign data such as users’ movie ratings, researchers were able to re-identify individuals. The key of these attacks is the *side-knowledge* of the attacker, which, combined with the anonymized dataset, leads to re-identification: for instance, combining voters list or public records with a medical database [Swe00, Swe05] or Netflix database with IMDB’s. The issue raised by Dwork [Dwo06] is that it is not possible to predict the attacker’s side-knowledge, thus it is impossible to chose the proper quasi-identifiers. To be more accurate, Dwork’s proof shows it is not possible, with the

	ZIP Code	Age	Disease
1	476**	2*	Heart Disease
2	476**	2*	Heart Disease
3	476**	2*	Heart Disease
4	4790*	≥ 40	Flu
5	4790*	≥ 40	Heart Disease
6	4790*	≥ 40	Cancer
7	476**	3*	Heart Disease
8	476**	3*	Cancer
9	476**	3*	Cancer

Figure 1.2: A 3-anonymous medical database
 Quasi-identifiers: {ZIP Code, Age}, Sensible = {Disease} [LLV07]

“anonymize and publish” approach described previously, to obtain anonymity *and* utility, where utility is a quantifiable parameter specifying *the quantity of useful information extractable from the database*. Indeed, utility is maximal and anonymity minimal when the database is released as is, and conversely if the database is completely emptied before release. The two notions thus seem antagonist.

But in 2006, Dwork proposed a new technique that reconciled anonymity and utility, *differential privacy* [Dwo06, Dwo08, DMNS06], which has been adapted to other context than statistical databases after its creation [AGK12]. In few words, the concept of differential privacy requires that, for a privacy-preserving mechanism \mathcal{M}_Q with Q the query to the database issued by the analyst, the probability distributions of \mathcal{M}_Q 's output on database DB_1 and DB_2 are statistically close, where DB_2 is DB_1 stripped from 1 entry only. Dwork instantiates such a mechanisms by setting $\mathcal{M}_Q(DB) = Q(DB) + \Delta(\text{Sensitivity}(Q))$: the mechanisms gives the exact answer for the query $Q(DB)$, masked using a Laplacian noise Δ calibrated following the *sensitivity* of the query Q^3 computed on the database. As the mechanism is tailored to a specific query which need to be known prior to the computation of the output, this mechanism is said *interactive* as the analyst must provide the query and wait for the answer. The key aspect of differential privacy which makes the difference with sanitization techniques, is that the utility is know *prior* to the database transformation, *i.e.* the query Q is known before the release of the answer, contrary to the “anonymize and publish” paradigm. When the utility is known, it is easier to robustly anonymize the database in a way preserving the desired utility, but *only* this particular utility, specified via the query Q . More details are provided in section 1.4.3.

Online

Online privacy also give rise to a lot of interest in numerical privacy research, as web applications are multiplying and growing in importance. Basically, online privacy works try to enable anonymous *online* service access on internet. Those services include:

Targeted advertising where the goal is to offer tailored adverts to the user, without the advertiser knowing *who* (which IP address) requested a specific ad, thus preventing profiling from the advertiser.

e-Commerce where the user might want to prevent the merchant from knowing its true identity.

³The sensitivity of Q is $\max_{DB_1, DB_2} |Q(DB_1) - Q(DB_2)|$, where DB_1 and DB_2 differ only in one entry.

Web searches and history as it has been shown that web searches reveal a lot of information on users [CCP10].

Anonymous service access can easily be implemented using anonymous proxies that strip all identifying information from the service request issued by the individual [DAM06], but this simple solution only moves the trust from the service provider to the proxy owner.

Other points of interest in online privacy include avoiding **web tracking** of users by websites, especially third party website that try to follow every movements of users in order to infer its habits and/or identity. Also, **(online) social networks** are a problematic topic, as their very definition is an infringement to the sovereignty and minimization principles (see section 1.3.2). Goals of researchers in this context is to re-enforce the control of the user over its data, and to investigate re-identification through social graph reconstruction [ZG09].

Authentication, authorizations and anonymous service access

We complete here section 1.3.1 on identity management with some considerations on authorizations and authentication. Indeed, in this section we left aside cases where the system or service provider does not need to associate any identity at all to the user in order to deliver the service, but it only needs to check some specific properties of the individual such as his age, his affiliation to a certain group, or other specific properties.

In this context, *anonymous credentials* [DAM06] play a large role: those constructions, typically based on zero-knowledge proofs [Gol01, Chapter 4] or group signature [CVH91], allow an individual to prove specific attributes that are certified by an authority recognized by both the service provider and the individual. Only the authority can make the link between an authorization and an individual. After a short protocol, the service provider is assured that the individual fulfils the necessary requirements, but does not know *who* he is. The requirements can take the form of attribute testing (*e.g.* is age above some threshold ?) or authorizations (*e.g.* a signed, anonymous piece of paper granting access some building).

IBM developed the IDEMIX (Identity Mixer) system [CMS10], implementing anonymous credentials and blind verification principles. The implementation involves extensive use of zero-knowledge proofs and group signatures and ensures unlinkability between an individual's actions. If necessary, anonymity is revocable: the issuer of the credentials (*i.e.* the trusted third party) can make the link between a specific use of the identity mixer and an individual.

Anonymous credentials are also used in a proposal for an *privacy-preserving digital identity card* [DG12]. Basically, this card is meant to communicate with a secure device allowed to ask one specific binary question, that can be a boolean expression composed of several questions, such as “*are you over 18 AND are you French ?*”. The device must possess and show a credential issued by the certification authority (*e.g.* the government) for each question it is allowed to ask. Anonymous credentials are then used by the card to answer the questions, and certify those answers. Other challenges in designing a privacy-preserving identity card are notably ensuring the non-transferability (impossibility to lend one's card), correctness (impossibility for the individual to cheat/lie) and unforgeability (impossibility to counterfeit a card) properties. The use of tamper-proof smartcards and biometric authentication embedded in the card helps achieving these goals.

Geolocation

Many services based on location are emerging since the advent of mobile phones and GPS devices, and require the user to share its geographic position with the service provider and/or third parties. The social network *Four Square* is a famous example, but many more systems are geolocated:

a city bus card records users' entry and exit points in and out the network, cellular phones are always (approximately) geolocated using relay antennas, ip addresses give away information on the region of the computer using it, and of course GPS systems communicate very precise location information. In general, geolocation mechanisms always associates a spatial location with a specific device (*e.g.* GPS, card, phone), often itself linked to one individual.

However, location and mobility patterns are very sensible data and should be considered similarly to personal data, as several studies show [GkdPC10, GP09]. In fact, from mobility traces, it is possible to infer *points of interest* such as working place (most stable place in the trace during the day) and home (the last stop of the day), to predict the individual's moves, and even approximate his habits, social relations and income (based on the places visited). Even if the mobility traces are anonymous, studies show it is possible to de-anonymize them using side-information such as social graph [SH12, MYR13].

Solutions are needed to protect individuals from disclosure of their location, while still enabling the use of location-based services. Some solutions are adapted from other contexts such as k-anonymity [GG03] (called *spatial cloaking*), mix-zones [BS03] (inspired from Chaum's mix-nets 1.3.3), and perturbation techniques such as randomization [ARZ99], insertion of dummies, or suppression or swapping of parts of the traces.

Cloud

The emerging concept called *cloud computing* refers to a powerful computational resource composed of a large distributed network of machines easily accessible via a unified, virtualized interface. It encompasses several services: *infrastructure*, *platform* and *software as a service*, respectively denoted *IaaS*, *PaS* and *SaaS*. Basically, a cloud provider grants computing resources to clients (expressed in terms of software, CPU frequency, storage size, ...), and manages all details concerning hardware, computing power, installation, administration, etc. This is especially convenient for companies willing to easily obtain computing resources without having to administrate a complex and expensive installation. The cloud provider makes extensive use of virtual machines and virtualization in order to abstract from hardware considerations. The challenge is then to design a smart, dynamic, *on demand* allocation of computing resources depending on the clients needs. This is the role of the *hypervisor*, a software that manages the virtual machines. Large cloud providers often control several *pools* of hypervisors.

In the cloud, resources are shared among several clients, and a resource belonging to Alice may pass onto Bob the minute after. This raises concerns in terms of security and privacy for individuals or companies hosting their data in the cloud. In particular, the data owner must be assured of the proper use of its data, of the non-transferability of the data's ownership and of adequate access control management in the cloud. Privacy researches in this sector try to enable the use of the cloud's large computing power, while hiding the actual data manipulated from the cloud provider. Zhang *et al.* proposed [ZYZ⁺12] a taxonomy of key privacy issues in cloud computing and lay the foundations for future researches in cloud security. In particular, they divide researches on the client from those on the cloud provider side. On the provider side, they also subdivide points of interest following the levels of abstraction in the cloud: the platform, virtual machine and hardware levels.

In higher levels of abstraction, *i.e.* at the application level, privacy solutions mainly involve cryptographic primitives. However, these solutions can not rely on cryptography alone to achieve privacy. Indeed, the impossibility result of van Dijk *et al.* [VDJ10] states that software alone can not guarantee privacy in the context of several clients mixing their data in an *off-line* multiparty computation [Gol04] setting (*e.g.* for similarity computations of profiles in social networks). They prove that others tools are necessary, and propose the use of trusted hardware to circum-

vent these shortcomings. Another possibility is to ask for clients involved in the computation to cooperate at some point in the protocol, but this requires dropping the *off-line* property of the protocol. Proposals for privacy-preserving cloud applications involve multiparty computation and homomorphic cryptography [LATV12, AJLA⁺12] (see also Chapter 2). In those solutions, computations performed by the cloud can be made *off-line*, but clients need to be online and cooperate at the decryption phase. Thus they do not allow complete *off-line* multiparty computation.

Misc.

There exist many more sub-sectors within the “personal data management” sector, however we do not provide a complete list here. Other points of interest include:

Mobiles phones Leakage of personal information through mobile OSs and applications, often without the user’s consent, poses serious threats: mobile phones are very personal devices, aware of many personal information and details and it is able to monitor a large portion of its bearer’s actions. The Inria project “Mobilities” [ABC⁺13] investigates these issues.

Radio Frequency Identification RFID tags are small wireless devices used to identify objects or persons. They can notably be used for physical tracking and localization. For instance, stores and malls may track their customers in order to construct profiles and clear out buying habits. Tracking is also possible city-wise, to monitor road traffic and vehicles. It is a major threat to individual’s privacy, as RFID tags can identify *and* locate individuals. The security and privacy of this technology is investigated [Jue06], as the practical threats and attacks are still unclear.

CCTV and video surveillance Along the same lines, surveillance via video cameras poses serious threats as the cost of these devices drops and automated facial recognition technologies progress.

Internet of things With the dropping cost of electronic devices, the emergence of IPv6 (and its very large addressing space) and the easiness of internet access, it is reasonable to assert that in a few years, all objects surrounding human beings will embed computational power and have internet access. In particular, the domestic devices in a house will form an “ambient intelligence” that should facilitate everyday life. These devices, however, form a privileged access inside one person’s sanctuary of privacy, his home. In this context, Oleshchuk [Ole09] proposes solutions based on secure multiparty computation to protect the data collected by these devices.

Digital Rights Management The very design of DRMs is an infringement to privacy, as the goal is to trace ownership of digital items in case of fraud or misuse. DRMs and broadcast encryption infrastructures can lead to consumers tracing and profiling by content distributors (*e.g.* by monitoring who watches which movie). Researches try to reconcile “traitor” (or misbehavior) tracing, while ensuring anonymity, possibly using revocable anonymity [CKSJ03, GPY⁺04].

1.3.3 Anonymous communications and traffic analysis

Although close to anonymous service access or identity management, traffic analysis prevention actually forms a well separated sector. It comprises notions absent in both sectors, notably hiding communication patterns and who communicates with who. Indeed, aside from confidentiality,

integrity and authenticity of the content of the exchanged messages, which are easily enforceable with encryption and signature, the communication link between two parties (or more) itself can also need to be confidential. This problem was introduced by David Chaum in 1981 [Cha81] in the context of electronic mail, under the name “traffic analysis information” and described as “the problem of keeping confidential who converses with whom, and when they converse”.

This work is motivated by the fact that traffic analysis poses an important threat against privacy, since the simple observation of source and destination identities (*e.g.* IP addresses) can reveal sensible information. For instance, observing an IP packet incoming from an identified individual towards a sensitive web page (*e.g.* www.aa.org) reveals private information on the individual. Or, more serious, a totalitarian government could easily eavesdrop outgoing communications from the country and detect dissidents willing to engage in illegal communications with external parties.

Definition

In this context, the notions of unlinkability, unobservability and pseudonymity (see section 1.3.1) are also relevant, as they were actually invented for this context [PK01]. Here, we are interested in the (un)linkability between network nodes (*e.g.* source and destination) and between network nodes and messages. More precisely, we consider the notions listed below. In the following definitions, according to the definition of *unlinkability*, by “impossible to know” we mean that the two items (*e.g.* a node and a message) are no more and no less related compared to the situation before the event (*e.g.* message sending).

Source/Message unlinkability: given a message and the identity of a potential source, it is impossible to know if the source is the actual issuer of the message;

Destination/Message unlinkability: similar to Source/message unlinkability;

Source/Destination unlinkability: given two nodes, it is impossible to know whether they communicate together or not (this notion is composed source/message *and* destination/message unlinkability for the same message⁴);

Relay/Message unlinkability: given a message and a potential relay node in the network, it is impossible to know if the message passed through the relay node;

Node anonymity: all or a subset of nodes in the network stay anonymous, *i.e.* their identities are never disclosed;

Message unobservability: it is impossible for some observer to know whether or not a message is issued, received or relayed in the network. Note: this property typically implies very inefficient solutions and is not considered in the rest of the section.

These definitions implicitly suppose the existence of an “observer” or and “attacker”, a malicious entity willing to learn as much as possible about the network, network nodes and communication patterns for its own profit. In a military network deployment, this entity can be the opponent army for instance. It is crucial to define the capacities of this observer or attacker in order to evaluate the above properties: a system might provide source/message unlinkability against an attacker *X*, but not against a more powerful attacker *Y*. The main models for network attackers are: *active* or *passive*, *global* or *local*, and *external* or *internal* to the network. In some cases, the location or role of the attacker in the network is also important (*e.g.* the attacker can be on

⁴For source/destination unlinkability to be verified, source/message and message/Destination unlinkability must both be verified.

a message's route or not, impacting its capacities). For more information, [GBP13, Section 3.2] provide a detailed definition of these models.

There exist interactions between the different privacy properties. For instance, if a system provides source unlinkability, but no destination unlinkability, when Alice sends a message to multiple recipients, who all answer back to Alice, an observer will notice a sudden increase in the amount of messages received by Alice and can infer Alice was the sender of the original messages. In other words, source unlinkability without destination unlinkability might result in neither source or destination unlinkability. Thus, the consistency and compatibility of a system's privacy properties must be thoroughly analyzed. Also, it is important to note that, even though *all* the above properties are enforced in the system, confidentiality and integrity of application-level messages are not implied. Adequate encryption and signature techniques must be applied if those are necessary. Conversely, encryption and signature do not provide anonymous communications. Also, it is important to note that providing anonymous communications and confidentiality/integrity is not enough to obtain anonymous access to a service: the two sectors are correlated but not equivalent, and if the application-level messages sent to the service provider contains identifying information, the anonymity provided by the network is vain (see section 1.3.2). However, confidentiality, integrity and more generally the contents of application-level message are not considered in anonymous communication solutions: in this sector, we focus on routing and traffic analysis.

MIX-nets

In its seminal work on traffic analysis and untraceable communications [Cha81], Chaum presented a protocol using particular routers, which he called MIXes. These MIXes have been the building blocks for many of the solutions proposed to provide anonymous communications. Basically, a MIX router hides the link between incoming and outgoing messages, using techniques tampering with time, ordering and appearance of messages. An adversary observing a specific MIX router may try to re-link messages in two main ways. First, following the assumption that an incoming message and an outgoing message have a higher probability to be linked when the time elapsed between the incoming message reception and the outgoing message emission is shorter. To prevent this kind of attack, two types of MIXes exist: *continuous MIXes* introduce a random delay between reception and forwarding (the protection they provide is efficient when traffic is dense, else, timing attacks are still feasible). The alternative is using *Pool MIXes* that function in batches: it waits for n messages to be received before forwarding them, in an order different from the order of arrival. Secondly, using size and/or appearance of message, attacker can very easily trace traffic. MIXes thus use padding techniques and fixed-size packets, along with decryption/re-encryption techniques (using pre-distributed, publicly known keys for instance). For more information on MIXes and MIXing techniques, one may refer to Diaz and Preneel's taxonomy [DP04]. A possible implementation of MIX router is depicted in Figure 1.3.

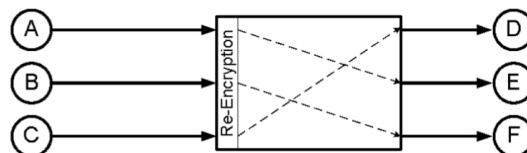


Figure 1.3: A simple MIX, using re-ordering and re-encryption

In a network, several MIX routers are often used in chain, so that unlinkability properties

are ensured even if all routers are compromised but one. Indeed, when there is only one MIX on the route, this MIX knows the sender and the receiver, thus neither the source/message, destination/message nor the source/destination unlinkability properties are observed. When two MIXes are used one mix can make the source-message link, the other can link message and destination, but none of the two MIXes can break source/destination unlinkability. The risk is that they collude to pool their knowledge and try to re-link source, destination and messages together. Therefore, the more MIXes, the better the unlinkability, but the slower the communication. In standard solutions, the first MIX can link a message and its source, and the last MIX can link the message and its destination [DMS04], but this can be avoided (*e.g.* via anonymous authentication between the source and the first MIX) [ZLLF06]. Moreover, note that an external observer can always trivially detect message emission (or reception) by a particular node, and therefore link a message to a sender (or a destination). This inherent characteristic of network communications is particularly uneasy to circumvent. The main way to get around it is for the network to ensure *unobservability*. For this, Chaum proposes the introduction of dummy messages, which is very costly in terms of bandwidth.

In order to distribute trust among many nodes, solutions like Tarzan [FM02] and Morphmix [RP02] propose to build a peer to peer network where any node can be a MIX router. Those are called MIX networks. Generally, routes and MIX routers are chosen randomly and adaptively, either by the source or dynamically by the MIXes en route.

High-latency/Low-latency networks

When implementing a MIX router, choices and trade-offs have to be made between delay and unlinkability (or more generally, privacy): while continuous MIXes introduce less delay, anonymity is low when traffic load is low, as pool MIXes may introduce large delay, but ensure high anonymity. Note that the chosen cryptographic primitives can induce more or less delay as well (*e.g.* asymmetric encryption is generally slower than symmetric encryption). In the literature, MIXing solutions are divided between high-latency and low-latency ones. The first category are dedicated to latency-insensitive applications with little interaction such as email and are able to resist a global eavesdropper. Low-latency MIXes support more dynamic applications such as web browsing, or SSH and TCP sessions, but are more vulnerable.

As an example, Mixminion [DDM03] is a high-latency network assuming the loss of efficiency in order to obtain strong anonymity properties. On the contrary, the basic Onion Routing protocol [GRS99] is designed for low-latency networks, and in the presence of low traffic, MIXes must continue forwarding traffic, even if gives a significant advantage to the attacker. The second generation of Onion Routing (Tor) [DMS04] goes even further in to reduce latency, as it no longer uses MIXing techniques. Indeed, the authors consider that the benefits in terms of privacy are too low (and not formally proved) and that the overhead they imply too large to justify their use. In practice, Tor no longer re-order message in batches, but still adds padding after decryption of a layer. The result is an efficient low-latency network, however vulnerable to a global attacker.

1.3.4 Cryptographic constructions and protocols

The last sector we put forward deals with cryptographic schemes and protocols. Although mentioning cryptography as a *sector* instead of a *tool* can be somewhat surprising, we argue cryptography is *both*. Indeed, as we can see, cryptography is extensively employed to enable confidentiality, anonymity and privacy in the technological approach. However, cryptographic constructions themselves can also benefit from privacy properties. One may take as a starting point an existing cryptographic construction as a public key scheme, and try to define, design

and incorporate properties that preserve “anonymity” within the construction. What is meant by “anonymity” in this case is to be defined depending on the cryptographic system considered, and the unlinkability-based definitions from sections 1.3.1 and 1.3.3 are also relevant here.

A simple example consists in modifying a public key scheme such that it ensures “*key privacy*”, a property preventing an adversary from distinguishing, given a ciphertext and a set of public keys, which key was used to encrypt the message. Indeed, messages intended to Alice are encrypted with Alice’s public key (which is often publicly known), and if, given a ciphertext, it is possible to reveal the key used for encryption, the link between the message and Alice is trivial. It is proven that key privacy is possible in several schemes such as El Gamal or Cramer-Shoup, but not for RSA for instance [BBDP01].

Key exchange and cryptographic authentication are also systems that can integrate anonymity properties [VYT05, GOR14]. As outlined in section 1.3.1 authentication and anonymity seem orthogonal objectives, but the cryptographic tools used in authentication can incorporate privacy properties. The solution is to authenticate users, not as individuals, but as legitimate members of a trusted group. For instance, using group signature, a user proves he has the knowledge of a secret common to a known group, and thus authenticates himself *as member of the group*, but it is impossible to uncover the signer’s identity from the signature. Another way of doing so is to use a certifying trusted third party: this entity delivers certificates that the user can show to prove himself to other entities (who can verify the certificate publicly). The drawback of this approach is that the third party knows the link between anonymous certificates and the users.

In functional encryption, where secret keys’ special construction allows decryption *and* computation of a certain function f at the same time, Boneh *et al.* [BRS13a] propose the notion of *functional privacy*. Typically, in basic functional encryption, an entity is given a secret key sk_f that can be used to compute $Dec(sk_f, Enc(pk, m)) = f(m)$, and security requirements ask that nothing except $f(m)$ leaks about the underlying message. However, nothing is said concerning f , and the computed function can be known and/or learned. Boneh and al. go further, and require that f stays secret, so that only $f(m)$ is computed without revealing neither f nor m . This requirement is useful in a context of computation delegation: for instance, a user might need to delegate email sorting between “urgent” and “other” mails to a proxy, without revealing the contents of its mails nor the “urgent” selection criteria. In practice, authors give a solution for identity-based encryption (a simpler variant of functional encryption where f is a simple predicate checking the identity embedded in a ciphertext) based on the assumption that identities that correspond to a given secret key are sampled from distributions with a certain amount of min-entropy. This result does not provide a strong functional privacy as one would expect, but this work lays the foundations for future developments.

1.4 Measuring privacy in computer science

In order to be able to compare privacy solutions, given a sector in the technological approach, there is a need for a *privacy metric*. In other words, it is necessary to *measure* and quantify privacy precisely. An impartial, objective measure is the only way to obtain certainty of the privacy properties ensured by a system, and to unambiguously compare privacy solutions.

Several ways of measuring privacy have been proposed and used. Some metrics are formally defined, others are more intuitive and easy to use but less accurate. What they measure needs to be specified, as they are generic tools. Depending on what is considered private or not and on the endorsed vision of privacy, the measures must focus on different objects. Indeed, although the metrics are well defined as theories, the intended objects of measurement are not included in their definitions. As a result, there is no precise framework for the measurement of privacy.

This often results in incomparable solutions and measurements, which is exactly the opposite of privacy metrics' primary goal.

However, privacy metrics are necessary, and while this field needs formalization and is quite recent, efforts in this direction can only be beneficial. In this section, we sketch a few privacy measurement tools and illustrate their use in privacy works. Note that, although we listed cryptography as a privacy sector, we do not present measurements of privacy properties in cryptographic schemes: in this particular context, proofs of privacy properties are analogous to those of traditional security properties, and this field of information security uses its own, very strict and formal, mathematical tools.

1.4.1 Information theory

The main privacy measurement tool, the most well defined and the most used is *information theory*. Invented by Shanon in 1949 [Sha49] to be applied to signal processing measurements, this method of information quantification has been widely used in computer science.

The key component of information theory is *entropy*, which measures the quantity of information (often in bits) needed to represent an object. For instance, as there are nearly $2^{33} \approx 7$ billions human beings on earth, 33 bits of information are needed to represent a person. Another way to see entropy is as a quantifier of the uncertainty of a random variable. In our example, if the random variable X is defined by a uniform law and takes as value 1 human being on earth, the entropy of X is 33. Remark that when the entropy is high, the value of the random variable is very uncertain, *i.e.* it can take many values, each with the approximatively same probability (X is close to uniform). When it is very low, the random variable is almost determined, in the sense that there is 1 value with high probability, and a few other very improbable outcomes. The exact formula used to measure the entropy of a random variable X is:

$$H(X) = - \sum_{a|Pr[x=a]>0} Pr[X = a].\log_2(Pr[X = a])$$

This formula computes the expected value of the logarithm of the probabilities, and thus the average number of bits needed to represent X . Indeed, as X can be described by $-\log_2(Pr[X = a])$ bits with probability $Pr[X = a]$ ⁵, the average number of bits to represent X is $\mathbb{E}[-\log_2(Pr[X = a])]$. We reformulate the above explanation with the value $H(X)$: when $H(X) \approx 0$, uncertainty of X is very low and ≈ 0 bits are needed to represent X , thus we can consider X is determined. On the contrary, if $H(X)$ is high, the outcome of X is very uncertain and a lot of information is needed to figure out the outcome of X .

A useful derivative of entropy is the *conditional entropy* $H(X|Y)$ which computes the uncertainty of X provided Y is known:

$$H(X|Y) = - \sum_{a,b|Pr[X=a|Y=b]>0} Pr[X = a \wedge Y = b].\log_2(Pr[X = a|Y = b])$$

This formula quantifies the average quantity of information needed to know the outcome of an event X when the outcome of the event Y is known. For instance, it is useful in cryptography to express a *perfect* cryptosystem: the phrase “knowledge of the ciphertext does not change uncertainty of the plaintext, and knowledge of the plaintext does not change uncertainty of the ciphertext” can be expressed as $H(P) = H(P|C)$ and $H(C) = H(C|P)$, where $H(P)$ is the entropy of the plaintext and $H(C)$ the entropy of the corresponding ciphertext [vTJ11].

⁵The idea is that the lower the probability that X takes value a , the more bits needed to describe a .

Use of entropy in privacy measurements In computer security, a very widely admitted way to verify the security of a system is to suppose the existence of an adversary, give him capacities and limits, and observe the harm that he can do to the system. More or less powerful models of adversary can be imagined according to possibilities given to the adversary and the strength of the assumptions made on the system and the environment. In general, models try to mimic real situations, and to over-estimate the adversary's capacities so that security is over-evaluated (rather than optimistically evaluated). When measuring privacy, we proceed following the same paradigm. We suppose the existence of an adversary who observes the system and the events produced within it, and tries to guess the outcome of an event. For instance, an adversary observing network communications can try to determine who sent a specific message. The knowledge of the adversary is then modeled by a random variable X which distribution probability gives the guesses of the adversary. In the latter example, X may take as value any identity within the network. To measure privacy guaranteed by the system, entropies of all event which outcome needs to be kept secret are computed. Note that the precise list of these events of interest depends on the privacy goals of the system, and may be hard to draw. Notably, there is no way to be certain that *all* sensible events have been considered.

Starting from this basic protocol for measuring privacy with entropy, several works enhanced the measuring by pointing and resolving flaws of the above measuring technique [GTD⁺08, THV04]. In the sector of anonymous communication and traffic analysis, Tóth *et al.* [THV04] notice that even though entropy gives the average guaranteed privacy, high entropy does not always mean good privacy. For instance, the authors consider hiding the identity of a source sending a message: when an adversary studies the outcome of X , where X is the identity of the source, and knows that $Pr[X = ID_0] = 0.5$ and $\forall i \in [1...100] Pr[X = ID_i] = 0.005$, entropy $H(X)$ is the same as uniform distribution over 20 possibilities. However, the adversary will most certainly infer that ID_0 sent the message because this value has much higher probability than any other. The authors thus introduce the notion of *local anonymity* that additionally requires an upper bound on all probabilities. In the presented approach, however, the assumed model does not always reflect real situations or is not trivially usable in practice because of incompatible assumptions between the measurement framework and the communication protocol. In particular, in the presented measures of privacy in anonymous communications, the use of MIX routers in the protocol is assumed, which is not the case in Tor for instance, and in many privacy-preserving communication protocol.

In definitive, information theory is a well defined, formal and powerful tool, but applying and using it must be done with great care. First one must point out *what* needs to be measured, but the most challenging is drawing the probability distribution correctly, *i.e.* in a manner adequately describing reality. The most effective way of doing so is, for now, using simplifying models to describe the environment.

1.4.2 k-privacy

A much more simple but less accurate metric can be derived from the technique of k-anonymity that has been presented in section 1.3.2, which deals with statistical databases. It is indeed a technique transforming a database by forming classes of k undistinguishable entries, but more generally, k-anonymity denotes the partition of the universe of entities in classes of size $\geq k$ where all members are equally probable candidates related to a given event. Therefore, the parameter k determines the size of the anonymity set(s) and measures privacy in the sense that a greater k gives better anonymity. In practice, the universe of entities is simply divided in two: one group of size k containing the possible candidates (the anonymity set), and the rest of the universe. Note that k-anonymity can be seen as a very simple form of information theory: in a sense, the

size k of the anonymity set measures the uncertainty of a random variable X associated to an identity. Within the anonymity set, X is a uniform distribution over k values, and probability associated to values outside the anonymity set is 0.

However, this quantification suffers from several shortcomings. The assumption that all k candidates are equally probable may be flawed, there is no framework or formalisation of k -privacy, and as there is no systematic way to apply it, forgetting or adding false candidates in the anonymity set is always possible. Also, k -privacy is only suitable in contexts where events are issued (or related) to entities, and the aim is to hide who issued which event. Note that information theory is more general as it models the knowledge of the adversary in a context-agnostic manner. Though very informal, this metric is useful as a pre-analysis of privacy properties. Instead of directly using information theory, one will prefer employing the simpler tool of k -privacy. It is easy then to quantify the probability of each candidate within the set of k entities in order to compute the entropy of the random variable answering the question “who is the originator of *this* action” for instance.

Example using k -privacy In the sector of privacy-preserving location-based systems, it is possible to define privacy requirements in terms of k -privacy: a subject is “ k -anonymous with respect to location information, if and only if the location information presented is indistinguishable from the location information of at least $k - 1$ other subjects” [GG03]. In other words, k -privacy in location-based systems is achieved for a given user when at least $k - 1$ other users were in the approximately same place at the approximately same time. The system designed by Gruteser and Grunwald [GG03] takes as parameter a minimum bound on the anonymity set, k_{min} , and ensures it by decreasing the accuracy of spatial information as much as necessary in order to always have k users in the same region at any time. This anonymity comes at the cost of decreased utility: if a user employs very deteriorated location information to enquire a location-based service (*e.g.* a listing the restaurants in the neighborhood), the answer will contain a lot of useless data, and the user might not actually obtain the information he was looking for. As in many sectors of privacy, a trade-off between utility and privacy is necessary.

1.4.3 Differential privacy

Differential privacy has also been presented in section 1.3.2, and, as k -privacy, it can also be seen as a metric. In its basic meaning and roughly speaking, differential privacy ensures that the removal or addition of a single database item does not substantially affect the outcome of any analysis [Dwo08]. However, in a broader sense, a differentially private mechanism \mathcal{M} on data space \mathcal{D} guarantees that the amount of information that is leaked when executing \mathcal{M} on the sensible input data $x \in \mathcal{D}$ is bounded and divided equally among all items in the data. The fundamental equation expressing the above definition for a ϵ -differentially private randomized mechanism $\mathcal{M} : \mathcal{D} \rightarrow \mathcal{D}'$ is:

$$\forall x, x' \in \mathcal{D} \forall t \in \mathcal{D}', \frac{Pr[\mathcal{M}(x) = t]}{Pr[\mathcal{M}(x') = t]} \leq e^\epsilon, \text{ with } x \subset x' \text{ and } x' \text{ contains 1 more item only}$$

Probability is taken over the coin tosses of \mathcal{M} . A mechanism \mathcal{M} satisfying this definition ensures that by the presence or absence of any item in the dataset, no outputs would become more or less likely. Thus \mathcal{M} protects every item in the dataset independently in a very strong sense, as the above equation is a statistical property of \mathcal{M} , independent from the computational power and knowledge of the adversary.

This equation is not a privacy protection *per se*, but a *goal* and a constraint a mechanism must observe. We mentioned in section 1.3.2 the use of Laplacian noise in order to achieve

differential privacy. However, this is only *one* way of satisfying the above equation. When constructing a privacy-preserving mechanism, one may want to achieve differential privacy, so as to obtain strong proven privacy properties, and it is possible to do so with theoretically any system (actually, if \mathcal{M} is a completely random function, it satisfies differential privacy). In a short survey [Dwo08], Dwork gives examples of differentially private mechanisms.

In the previous equation, the quantification of privacy is given by the ϵ parameter. It can either be fixed to a value arbitrary close to 0 (but then utility of the output is very low), and up to 2 or 3 in some cases, or it can be expressed as a function of the other system parameters. The value of ϵ depends on the acceptable information leakage and the sensitivity of the data. The advantage of this metric is the compositionality property that comes with differentially private mechanism: given a dataset x , if a user makes a k queries on D , the i^{th} query being ϵ_i -differentially private, then the overall mechanism is $\sum \epsilon_i$ -differentially private. This property is useful to bound the sensible information a user of the mechanism can learn with at most k queries. If necessary, the mechanism \mathcal{M} can refuse to answer to a client who has consumed all its “privacy budget”, *i.e.* when the client learned $\sum_{i=0}^k \epsilon_i \geq \Delta$ quantity of information. A proposition for a generalized metric based on ϵ -differential privacy has been recently proposed [CABP13] and applied to sectors different from statistical databases.

Example using differential privacy BLIP (BLoom then FLip) [AGK12] is a technique invented by Alaggar *et al.* to compute profile similarities, for instance in social networks. The profile is represented by a Bloom filter [Blo70], and in order to protect the items contained in the profile, the authors propose a simple bit flipping mechanism that they prove to be differentially private. Each bit is simply independently flipped with a probability $p < 0.5$, and as a result, items are ϵ -differentially private, with relation $p = 1/(1+e^{\epsilon/k})$, where k is the number of hash functions of the Bloom filter. The utility is best when $p \approx 0$, but the authors show that for their application (similarity computations), utility is non-trivial even for “small” values for the parameter ϵ . As a result, each item is independently protected against computationally unbounded adversary, the information leaked by a bit-flipped bloom filter does not exceed a certain threshold ϵ , and the authors show that reconstructing a user’s profile is hard.

1.5 Discussion

This overview of problematics inherent to privacy protection gives rise to various reflexions on existing solutions and subsequent works that need to be addressed. The proposed view of privacy will also be discussed.

1.5.1 The Approach \times Sector view

This chapter puts forward a categorization of the existing privacy works in research, following a Approach \times Sector matrix. We argue this categorization provides a pertinent and useful synthetic view, and consists in a trade-off between a very complex but complete and exact categorization, and a disorderly inconsistent listing.

As said earlier, the choice of approach and sector as the only two criteria is motivated by their predominance and the possibility to easily compare solutions within the same sector and approach. Of course, many particular privacy works will be problematic and will not fit in one particular class. As an example, biometric authentication relates to the “identity management and authentication” sector, but is also relevant in the “protection of personal data” sector,

because biometric data are extremely personal. Conversely, there will also be “holes” in the Approach \times Sector matrix. Indeed, some sectors are approach-specific, for instance the sector of cryptography is only relevant in the technological approach.

A particular issue with the proposed categorization is that it does not provide an easily retrievable list of general privacy-enhancing tools. Actually, depending on what we are interested in, a new categorization is necessary for each purpose. However, note that the approach criteria is always implicit (*i.e.* when using the word “privacy”, one often has an approach in mind) and will most probably always be the first criteria of a categorization. The choice we made is using it together with the sector criteria, which is indeed discutable. Here, our goal was to obtain a synthetic view of the field of numerical privacy using a minimal number of criterion. Choosing sector above tools, privacy goals or models is justified by the relative precedence of sectors on others dimensions, and above all by the need we expressed.

Another point worth questioning is the completeness of the approaches and sectors lists in sections 1.2 and 1.3, as well as the list of criteria in section 1.1.2. They reflect modern privacy considerations and the author’s sensibilities. Although one might have other criteria in mind, the current list in section 1.1.2 contains the main important ones, and adding more criterion could lead to more, useless complexity. Concerning the list of sectors, as mentioned in introduction of section 1.3, all information systems are subject to privacy research. Thus we merely sketched the main fields in computer science where privacy is an active research theme. As for the list of approaches, although persons from different profession or sensibility might propose different ways to consider privacy, these 4 are the only one concerning *numerical privacy*, to our knowledge.

In a few words, privacy may be conceived under many angles, others than those proposed here. Because of the complexity of the field, a complete, exact and efficient categorization is hard to achieve. Although our categorization is approximative, to our knowledge, this initiative is the first intending to provide such a systematic categorization.

1.5.2 Trust management

In light of existing propositions, we note that trust management is a common issue in privacy, and more generally in information system security. A common goal for all privacy solutions is to ensure the user protection of his privacy when using numerical services and resources, by the mean of diverse technological solutions (we only consider the technological approach here). However, technological solutions are often deployed by the service/resource provider himself, who is precisely the entity that would benefit from a privacy violation. Therefore, default trust of the user should be minimal in the service provider, and instead should rely on other objects.

Typically, to resolve trust issues, there exist two main ways: to invoke a “trusted third party” (*e.g.* a governmental agency) in which the user is willing to place his trust, or to *distribute* the trust. The first proposition only “moves” trust from one entity to another, and in practice takes the form of certification authorities, audit of systems by experts and/or legal enforcements. Distribution of trust divides the user’s trust in multiple entities and relies on the assumption that a majority of entities will be honest. This latter approach avoids the unique point of failure, but it is vulnerable to collusions. In practice, trust is often distributed among the other users of a given system, with the assumption that there are many other users.

We envision a third option to trust management: when a technology is formally proven secure (as well as its implementation), and its deployment depends on the user’s initiative, then the user can place his trust in the technology, and not in entities controlled by human administrators. By user’s initiative, we mean that the user himself can decide alone to use a given technology which ensures that the service provider can not harm his privacy. Of course, the service provider must comply with the technology, but deployment and implementation do not depend on him: if the

user, and solely the user, follow the proper protocol, the service provider can not tamper with his privacy. This is a very strong requirement, and trust is at the lowest in this case (*i.e.* user does not need to trust any entity) provided technology is reliable.

1.5.3 Privacy in public opinion

As, fundamentally, privacy is a social concern, we could have described a “social approach” to privacy in section 1.2. However we argue that the social approach is actually implicit in each other approach and can not form an approach on its own in a categorization of privacy works or privacy solutions. There are however purely social studies dealing with numerical privacy that we did not mention in this chapter. Those studies need to be taken into account in order to design systems that will be used and have a non-negligible utility in society.

To sum up results in this field, a quotation from a survey by Taylor [Tay03] is sufficient:

Most people are “privacy pragmatists” who, while concerned about privacy, will sometimes trade it off for other benefits.

Indeed, Taylor’s survey draws three categories of persons: “privacy fundamentalists” who care about privacy (26 %), “privacy unconcerned” who explicitly do not value their privacy (10 %), and “privacy pragmatists”, who *claim* to care about their privacy, but are willing to trade it against short term benefits (64 %). As explained in section 1.2.4, this is mainly due to misinformation, bounded rationality and human psychology limitations.

Another experimentation, by Grossklags and Acquisti [GA07], investigates the “willingness to pay” for protecting personal information and “willingness to sell” the same information. Although biased because of a small tested population, who does not represent well of the global population (47 persons, almost all students), and the fact that interviewees were aware of the general principle of the study (thus were already willing to take the risk to disclose personal information), this study shows the overwhelming willingness to sell over willingness to pay, or in other words, monetary benefits are preferred over privacy in a large majority of cases.

Several studies attest the misinformation and misunderstanding of the general public regarding consequences of private information disclosure [WNK⁺11, SCK⁺13, WCS⁺13]. Individuals often accept disclosing a large amount of unnecessary information in order to access a service that is convenient for a given purpose, well designed or very used among their social circle. However, they either do not know that their data is collected, the value of those data for information brokers or the consequences and serious harms this collection can do to them, or they simply “do not care” and declare having “nothing to hide”.

1.5.4 Do we need privacy ?

The question immediately emerging from the observations made in the previous section is: “Do we actually need privacy?”. Indeed, privacy is a social concern, but paradoxically the public opinion, in practice, do not want or need privacy. We leave aside special contexts as surveillance in totalitarian regime where common opinion agrees on the fact that privacy is inexistent and people’s right are flouted ; or specific cases involving crimes or socially unacceptable acts where privacy should on the contrary be avoided. We briefly review common arguments against privacy and show their weaknesses, and eventually point two reflexions on the meaning and implications of (numerical) privacy in modern societies.

The most common argument is the “*nothing to hide*” one: supposedly, if an individual has done nothing “wrong”, *i.e.* illegal or socially unacceptable actions, he should not bother being watched and under surveillance, and should not need to keep information about himself secret.

This argument is very close to the “*surveillance is necessary for the nation’s security*” one, extensively employed since the 9/11 events in the United States, and can be refuted in the same manner. A first, easy answer is simply that actually everyone has something to hide, would it only be the moments and discussions with his/her partner. As put forth by David Flaherty [Fla99], no one can “withstand even a few minutes’ questioning about intimate aspects of their lives”. For a more complete analysis of the “nothing to hide” argument, see Daniel Solove’s paper on this matter [Sol07], who argues that this argument narrows privacy to information concealing and surveillance, whereas the problems raised by data collection are numerous and privacy has multiple facets (see section 1.1). Among other issues, Solove mentions the “secondary use” of the data: at first, data is collected for a given purpose, but the data may easily be used for an unrelated goal, without the consent of the user. Thus the data collected by government agencies, at first gathered for public good, may travel across several platforms and be in definitive publicly published.

Other typical arguments include phrases like “*no one cares about privacy anymore*” or “*privacy is dead at the digital age*”. They are more observations than real arguments. Privacy is dead only if all research and social efforts are abandoned, which is far from the case at this moment. The “secondary use” and misinformation issues can also be invoked here to mitigate the first phrase: if individuals were better informed on the use of their data and the true consequences of data collections/analysis/disclosures, they may reconsider their judgement.

On the privacy advocates side we can point the reader toward two reflexions. First, in 2009, Danah Boyd presented a talk on privacy in the context of Big Data [Boy10]. Although she recognizes the potentially great utility of analysing very large datasets (*e.g.* analysing DNA to help cure diseases), she questions the ethics of doing so. Actually, the author calls for a more ethic use of big data and personal data in general. She tries to mitigate the enthusiasm and haste of information brokers who very quickly took hold of the concept of Big Data.

Finally, a recent MIT technology review by Evgeny Morozov, entitled “The Real Privacy Problem” [Mor13], offers a deep understanding of privacy and the social meaning of the lack thereof. In this reflexion, the author considers privacy as a *mean* (as opposed to a goal in itself) toward democracy, arguing that arising automated decision processes, in particular via analysis of large datasets, are replacing societal debates, reflexions and discussions and lead to unjustifiable and unjustified decisions. This is in accordance with Boyd’s opinion. But according to Morozov, in order to grant privacy its right place in the public debate, and to efficiently protect privacy in the long term, legal and technological means are insufficient. There is a true need for a political debate and an ethic vision of privacy, and the solution can only come from a public realization. For this, Morozov argues that individuals need to “sabotage the system and provoke more questions”, in particular by refusing to use services tracking users.

Chapter conclusion Privacy, even restricted to the *numerical* context and to the technological approach, is a very vast field of research. As put forward in section 1.3, many or all information systems can benefit from privacy properties. In other words, privacy researches can focus on any existing system: the choice is open. Of course, some systems or sectors are more relevant than others, in the sense that protecting privacy is more important. For instance, protecting privacy in medical information systems is more important than in digital rights management because privacy leaks in the former lead to more serious consequences. The aim is to clear out pertinent sectors where privacy is crucial.

In this chapter, we offered a possible view of numerical privacy that we consider sufficient to organize the knowledge in this field of research. We also presented the main tools to measure privacy, letting us hope for a way to quantify the privacy of newly designed solutions. All these elements provide the necessary tools and knowledge to improve upon existing privacy works or initiate new directions of research in privacy, *e.g.* by formalizing privacy and privacy metrics, or by pointing to new sectors where privacy is yet to be considered.

We now describe in the next chapter *how*, or more precisely *with what tool* privacy can be enforced. We turn toward cryptography, which is the one of main tools in privacy solutions, and in this case we are interested in *homomorphic cryptography*. The great possibilities offered by this cryptographic paradigm let us hope for meaningful applications to the protection of privacy.

Chapter 2

Homomorphic Encryption

As we are interested in protection of *numerical* privacy in all its forms, we expect the need for powerful confidentiality-enhancing tools. Here we consider the use of a rather new, (theoretically) extremely powerful cryptographic tool to achieve privacy requirements in various systems: homomorphic encryption. This particular type of encryption allows to process data in the encrypted domain, *i.e.* it allows any party in possession of the corresponding public key to perform logical operation on encryption of messages, yielding logical operations on the underlying messages. In particular, homomorphic encryption allows to outsource computations on sensible data to a (possibly untrusted) third party while preventing it from learning the actual contents of the data, which is specifically the expressed need in the context of private cloud-assisted computing for instance. For more than 30 years now, this type of encryption has been extensively studied, and in 2009 a major breakthrough by Craig Gentry gave rise to the first *fully homomorphic encryption* (FHE) scheme, a scheme capable of evaluating *any* function on encrypted data.

This chapter presents the concept and history of homomorphic encryption, details several fully and non-fully homomorphic encryption schemes along with their implementations and their practical use. This chapter is solely focused on describing homomorphic encryption, while its use to enable privacy is left as discussion in the conclusion (section 2.4.5).

2.1 General points

This first section introduces the notion of homomorphic cryptography and provides the reader with the necessary notions and tools to approach this chapter with the appropriate knowledge. We first define homomorphic encryption, and then give an abstract view of this particular field of cryptography by drawing a short history and categorizing schemes in 3 main classes according to their homomorphic properties. At last, we give a good insight of the main pending issues, both theoretical and practical, in homomorphic cryptography research.

2.1.1 Definition

In a traditional public key encryption (PKE), schemes are typically composed of three *probabilistic polynomial-time* (PPT) algorithms¹ primitives:

Key generation $(pk, sk) \leftarrow \text{KeyGen}(1^\lambda, \text{params})$: According to the parameters params , outputs a public encryption key pk , and a secret decryption key sk , guaranteeing a

¹Probabilistic algorithms running in *polynomial time* in the size of their inputs.

security of λ bits, *i.e.* $\geq 2^\lambda$ operations are needed to break the scheme. Typically, $\lambda = 80, 128, 256$ or more. Note that λ specified *via* a value of λ bits, but we merely use its *length* as parameter for security. Sometimes, **KeyGen** is split in two primitives: **Setup** inputs 1^λ and outputs *params*, and **KeyGen** only uses *params* as input.

Encryption $c \leftarrow \text{Enc}(pk, m)$: Outputs an encryption c of message m under public key pk .

Decryption $m \leftarrow \text{Dec}(sk, c)$: Using sk , decrypts the ciphertext c to recover the message m . If sk is not the key corresponding to the encryption key pk , or if an error occurs, outputs \perp .

Basically, a homomorphic encryption scheme possesses a fourth PPT primitive, which captures its capacity to evaluate functions on encrypted data:

Evaluation $C \leftarrow \text{Eval}(pk, f, c_1, \dots, c_n)$: with $c_i \leftarrow \text{Enc}(pk, m_i)$, computes $f(m_1, \dots, m_n)$ in the encrypted domain, and outputs an encryption of the result under pk such that

$$\text{Dec}(sk, C) = f(m_1, \dots, m_n) \quad (2.1)$$

The set \mathcal{F} of functions $f \in \mathcal{F}$ a scheme can accept as input for **Eval** while guaranteeing correct decryption determines its homomorphic capacities. This set is defined by the properties of the scheme and can be expressed through various computation model (see section 2.1.4). Equation 2.1 formalizes the *correctness* of the homomorphic scheme's operations, and in some case is expressed in a probabilistic manner: a scheme is said correct if equation 2.1 holds with very high probability.

Example of homomorphic encryption: RSA Unpadded In its original design, RSA is multiplicatively homomorphic, that is to say it is possible to compute $\text{Enc}(pk, m_1 \times m_2)$ from the encryptions of m_1 and m_2 : with $pk = (N, e)$, we have

$$\text{Enc}(pk, m_1) = m_1^e \pmod N \quad \text{Enc}(pk, m_2) = m_2^e \pmod N$$

$$\begin{aligned} \text{Enc}(pk, m_1) \times \text{Enc}(pk, m_2) &= m_1^e \times m_2^e \pmod N \\ &= (m_1 \times m_2)^e \pmod N \\ &= \text{Enc}(pk, m_1 \times m_2) \\ &= \text{Eval}(pk, [(x, y) \mapsto x \times y], \text{Enc}(pk, m_1), \text{Enc}(pk, m_2)) \end{aligned}$$

2.1.2 Historical overview

The search for an homomorphic encryption scheme goes back to the year 1978, shortly after the invention of RSA: the accidental malleability of RSA led Rivest, Adleman and Dertouzos [RAD78] to form the idea of *computing on encrypted data*. The existence of such a mechanism was unknown at the time and considered an open question. A first answer was (informally) given in the form “Yes, but there must be interaction between the owner of the secret key sk and the *evaluator*”, and led to the creation of Secure Multi-party Computation [Gol04], a possible way to compute with encrypted data. See section 2.4.5 for a brief description of this technology and its link with our concerns here.

The question of the existence of a *non-interactive* mechanism was reformulated and somewhat formalized by Feigenbaum and Merritt [FM91] in 1991, who proposed the notion of *algebraic* homomorphic cryptographic schemes that could securely evaluate $\text{Enc}(pk, m_1 + m_2)$ and $\text{Enc}(pk, m_1 \times m_2)$ solely from pk , $\text{Enc}(pk, m_1)$ and $\text{Enc}(pk, m_2)$, without interaction between any party,

and without knowledge of the secret key. Such a homomorphic scheme enables arbitrary computations on encrypted data, *i.e.* its associated set of evaluable functions \mathcal{F} contains all functions that are computable in the clear. Indeed, if the message space is $\{0, 1\}$, addition of messages corresponds to a logical XOR, and multiplication to a logical AND. As the set of gates $\{XOR, AND\}$ is functionally complete, any circuit and thus any function can be computed with such a scheme.

However, the first homomorphic schemes that appeared allowed only addition *or* multiplication but not both. The first known, if we leave aside RSA which is insecure in its unpadded version, is the scheme of Goldwasser and Micali [GM82] in 1982. It allowed homomorphic addition of messages in $\{0, 1\}$, which corresponds to a homomorphic XOR on bits. Goldwasser-Micali is also the first probabilistic semantically-secure scheme, and at the time, the size of a ciphertext encrypting 1 bit was very large, *e.g.* ≥ 1024 bits. Many other schemes allowing *one* operation were proposed, among them the most known are ElGamal [ElG85] in 1985 and Paillier [Pai99] in 1999 (see section 2.2).

Early 2000s began to appear schemes that allowed more than one operation [BGN05]. For instance, MGH [MGH10] can perform many additions and “several” multiplications. However, these schemes are particularly inefficient and can not handle a reasonable number of multiplications (*e.g.* 100), because of the *exponential ciphertext size blowup* phenomenon which seriously limits the schemes: after L multiplications, the size of a resulting ciphertext is C^L for some (possibly constant) C , and becomes too large to be manipulated. Because of the difficulties encountered while trying to enable homomorphic addition *and* multiplication in a same scheme, some works relied on simpler computation models different from Feigenbaum and Merritt’s such as boolean circuits [SY99] or branching programs [IP07]. Note that schemes of this type all share this exponential ciphertext blowup characteristic and are inherently limited in their homomorphic operations: we will call these schemes *somewhat* homomorphic.

It is only recently, in 2009, that was invented the first *fully* homomorphic encryption (FHE) in the sense of Feigenbaum and Merritt, *i.e.* allowing both unlimited additions and multiplications. Before this date, neither theoretical or practical solution existed to evaluate *any* function on encrypted data. Gentry’s thesis [Gen09a] brought the first theoretical result for doing so, basing security on hard problems in *ideal lattices*. His construction was designed to encrypt *bit* messages only, and to evaluate boolean circuits. Two elements were decisive for the apparition of FHE: first, Gentry proposed a powerful somewhat homomorphic scheme capable of evaluating many additions and a few multiplications, but with constant ciphertext size. In other words, Gentry solved the exponential ciphertext size growth. To be more accurate, Gentry *moved* the problem: the ciphertext from his schemes did not grow in size but contained “noise” that would grow exponentially *inside* the ciphertext, such that when the noise exceeds a certain threshold, decryption can not be performed anymore (or decryption outputs erroneous values). The key idea to obtain a FHE scheme from this somewhat homomorphic encryption scheme is then to “refresh” ciphertexts when noise becomes too large by *homomorphically evaluating the decryption primitive on ciphertexts*, thus resetting its noise to a minimal level. This step is called *bootstrapping*, and is still today the only known method to obtain pure fully homomorphic encryption (see section 2.3.1 for details).

Since 2009, FHE has been an extensively studied subject, especially since 2011 when Brakerski and Vaikuntanathan [BV11b] noted that FHE could be based on the *Learning With Errors* (LWE) problem, a much simpler tool compared to ideal lattices. The result from Gentry’s thesis was almost purely theoretical: as expressed by Smart in his talk at CRYPTO’12 [GHS12b], in 2009 FHE was “totally and utterly impractical”, in 2012 it became “totally impractical” and was already about to become only “impractical”. Today, FHE is still impractical compared to efficient public key schemes such as RSA or elliptic curve-based schemes, but are (almost) practical for

use-cases involving a reasonable number of multiplication [AMFF⁺13].

2.1.3 Mathematical tools and notations

Before starting the main matter of this chapter, we define the main mathematical notions and notations we use.

First, all logarithms, unless specified otherwise, are logarithms base 2. We use Landau complexity notations in $O(\cdot)$, $o(\cdot)$, $\Omega(\cdot)$, $\omega(\cdot)$. The notation $\tilde{O}(n)$ refers to $O(n)$ along with logarithmic factors disregarded because of their asymptotic smaller values compared to n (for instance). We also sometimes use the notation $poly(n)$ to denote a polynomial complexity in n and $polylog(n)$ to denote a polynomial complexity in $\log n$. By $|N|$ we denote the bit-size of the number N , equal to $\log_2 N$ (we make use of both notations alternatively).

Vectors are denoted by bold font letters \mathbf{a} , matrices by upper case bold font letters \mathbf{A} and its transpose \mathbf{A}^T . The term $\langle \mathbf{a}, \mathbf{b} \rangle$ denotes the scalar product of vector \mathbf{a} and \mathbf{b} . The reduction of x modulo some number n , $x \bmod n$, is sometimes denoted $[x]_n$ for compactness. When applied to vectors or matrix, *e.g.* $[\mathbf{a}]_n$, it is simply defined as the reduction modulo n of the vector or matrix coefficients.

Cryptography

We use the notions of security from the *Foundations of Cryptography* (Vol. 2) [Gol04]: IND-CPA, IND-CCA1 and IND-CCA2. When using other security notions, they will be explained. We sometimes mention elliptic curve cryptography and pairing-based cryptography [HMOV04]. In the document, λ always denotes the security parameters of cryptographic schemes, unless specified otherwise.

In order to characterize homomorphic schemes, we make use of *Nick's complexity classes*, denoted NC . In particular, NC^i refers to circuits of polynomial *size* (number of gates) and *depth* (number of gates of the longest path from inputs to output) in $O(\log^i(\text{size of input}))$ built from fan-in 2 logical gates. The schemes' capacities can also be expressed in terms of multivariate polynomials, which most general form is

$$P(x_1, \dots, x_k) = \sum_{i=1}^M a_i \left(\prod_{j=1}^k x_j^{d_{i,j}} \right)$$

With M the number of monomials, k the number of variables, $d_i = \sum_{j=1..k} d_{i,j}$ the degree and a_i the coefficient of the i^{th} monomial, and $d = \max_{i=1..M} d_i$ the degree of the polynomial. The link between the polynomial and circuit models is direct: evaluating a polynomial of degree d requires a circuit of depth $O(\log d)$ [Vai12, MGH10].

2.1.4 Categorization of homomorphic schemes

As observed in previous sections, a homomorphic encryption scheme is mainly defined by its computational capacities over encrypted data, *i.e.* the set \mathcal{F} of functions it can evaluate. The functions in this set can be expressed through various computation models: polynomials of a specific degree, *Nick's classes* or more generally boolean circuits. The simplest way of characterizing the capacities of a scheme is in terms of simple algebraic operations, as Feigenbaum and Merritt propose. Typically, the two desirable operations are multiplication and addition of messages in the encrypted domain. Indeed, as explained earlier, these two operations lead to functional completeness if plaintexts are bits.

We propose the following categorization of homomorphic schemes. For each category, we detail its computational capacities in terms of algebraic operations (in nature and number), as well as in terms of polynomials² and/or circuits if relevant.

Partially homomorphic encryption (PHE)

Allowing only 1 operation, *i.e.* addition or multiplication but not both, in a limited or unlimited number.

This corresponds to the evaluation of multivariate polynomials of degree $d = 1$ in the case of unlimited additions, or multivariate monomials of arbitrary degree in the case of unlimited multiplication.

Somewhat homomorphic encryption (SHE)

Allowing 2 operations, one of which in a very limited number. In general, multiplications are limited, and additions are unlimited (or considered as such, because they are negligible compared to multiplications' cost).

This corresponds to evaluation of multivariate polynomials of degree $d = L$ if L multiplications are possible in the scheme.

In some cases, it is relevant to characterize the scheme in terms of boolean circuits of bounded size and depth, using Nick's classes for instance.

Fully homomorphic encryption (FHE)

Allowing both operations in an unlimited manner.

This corresponds to multivariate polynomials of arbitrary degree.

There are two other elements we did not take into account in this categorization by homomorphic capacities: the category called *Leveled* homomorphic encryption which is the main matter of section 2.3.1, and the *homomorphic scalar multiplication* property. The scalar multiplication is an operation that takes place at the frontier between the encrypted and clear domains: one operand m_2 is in clear, and the other m_1 is encrypted by $\text{Enc}(pk, m_1)$. Schemes allowing scalar multiplication respect the equation: $\text{Enc}(pk, m_1) \odot m_2 = \text{Enc}(pk, m_1 \cdot m_2)$ (for \odot an abstract operator). In other words, they allow multiplication of an encrypted value with a clear value.

In the rest of this chapter, we rely on this categorization and provide one section each for partially and fully homomorphic encryptions (respectively, sections 2.2 and 2.3). However, because of their small number, their inefficiency and their relatively limited capacities, somewhat homomorphic schemes are not given a section on their own. We merely mention them in section 2.2.6.

2.1.5 Main theoretical and practical issues

In order to offer the reader a better understanding of problematics encountered when designing a homomorphic scheme, we provide in this section a brief description of horizontal issues in homomorphic encryption.

In traditional encryption schemes, we have to perform trade-off between security and efficiency: the scheme must be secure in order to adequately hide information, and efficient so that its primitives are easily computable by the appropriate parties. Indeed, in a perfectly efficient scheme we have $\text{Enc}(pk, m) = m$, and to communicate m to Bob, Alice simply sends it in clear. On the contrary, a secure but inefficient solution (to the point it is not actually working) is for Alice to keep the message m secret and not disclosing it. To this security-efficiency duo, homomorphic encryption adds a third element to take into account: the *homomorphic capacities*. Indeed, in somewhat homomorphic encryption, the homomorphic capacities of the schemes

²For simplicity, we will consider all polynomials coefficients a_i as equal to 1 in the following paragraphs.

could be made arbitrary large, but it comes at the cost of exponential or worse running times of cryptographic primitives, and notably Eval. All three components have to be taken into account in the same time, and trade-offs have to be performed.

The above abstract remarks do not reveal much on the actual points of issues of homomorphic schemes. To be more precise, the common issues in homomorphic encryption include:

Inefficiency or Insecurity Obviously, as per the security-efficiency-homomorphism trade-off, in order to enable homomorphism, it is often necessary to give up some security and/or efficiency. As security can hardly be compromised, most homomorphic schemes trade some efficiency against homomorphic capacities. This is especially true for SHE and FHE.

Public key size A common issue with traditional cryptography: reducing the public key size. This is especially a problem in SHE and FHE, where some schemes have extremely large public keys, up to several GBs.

Ciphertext expansion The ciphertext expansion is defined as the ratio of the ciphertexts size on the messages size (in bits). During many years cryptographers tried to minimize this ratio in order to save space and bandwidth. Starting from the Goldwasser-Micali cryptosystem [GM82] in 1982, with a ciphertext expansion of roughly $|N|$ bits (with N similar to a RSA modulus, of *e.g.* 1024 bits), a ratio arbitrary close to 1 has been achieved by Damgård-Jurik [DJ01] only in 2001. This result comes at the cost of degraded running time performances, and increased messages and ciphertexts sizes (paradoxically). However, many SHE and FHE schemes created afterwards show even worse ciphertext expansions than the Goldwasser-Micali scheme.

Ciphertext size growth Independently from the ciphertext expansion, the ciphertext size growth is defined as the growth of ciphertexts' size when they are applied homomorphic operation. In other words, the ciphertext size growth is the ratio between $|\text{Eval}(pk, f, c_1, \dots, c_2)|$ and $|c_i|$. This problem mainly rises in SHE schemes, where it is uneasy to enable multiplication on top of addition, and schemes use redundant structures to store information (see section 2.2.6).

Ciphertext noise growth Equivalent of the ciphertext expansion problem of SHE in the FHE context. In a sense, this issue is replaced by the *noise growth* issue (see section 2.3).

Computation model The goal of homomorphic encryption is to compute on encrypted data. In this sense, we naturally think of computations in the encrypted domain in the *random-access machine* (RAM) model, *i.e.* if possible, we would like to execute RAM programs directly in the encrypted domain. But as we already showed, in practice, much simpler but less powerful models are used. In PHE we mainly use the simple algebraic model because of the limited capacities of the scheme. In SHE and FHE schemes capable of evaluating more complex processes, we use *circuits*. It is known that any RAM program can be turned into a circuit, yet this comes at a great cost. We discuss the this point in more details in section 2.4.3.

2.2 Partially homomorphic schemes

The main idea of this section is to introduce the main existing partially homomorphic schemes, and detail their capacities, performances and security. The aim is not to draw an exhaustive

listing, but rather to give an overview of the field and what can be achieved with such schemes. This section is partially inspired by a 2007 survey on homomorphic cryptography [FG07].

The 4 first subsections present the 4 main PHE schemes, along with their derivatives. For each scheme, we briefly describe its functioning and the involved mathematics. We then provide details on its homomorphic capacities, security and efficiency. Non cryptographic-oriented readers may skip the technical parts. All the schemes are summarized in Table 2.1, page 52. The 5th subsection considers PHE schemes under a practical point of view, and the last one briefly presents a few SHE schemes.

2.2.1 Goldwasser-Micali

Created in 1982 by Shafi Goldwasser and Silvio Micali [GM82], the GM cryptosystem is the first semantically secure public key encryption scheme, and thereby the first semantically secure homomorphic scheme.

To explain this scheme, we need to define the notions of quadratic residue and the Legendre and Jacobi symbols. Very briefly, we say that an integer a is a *quadratic residue modulo N* if there exist an integer b such that $b^2 \equiv a \pmod{N}$, *i.e.* such that a is congruent to a square modulo N . If N is prime, a is a quadratic residue modulo N if and only if $a^{(N-1)/2} \equiv 1 \pmod{N}$, which is easily computable [GM82]. The *Legendre symbol* of an integer a with respect to prime number p , noted $\left(\frac{a}{p}\right)$, is equal to: 0 when $a \equiv 0 \pmod{p}$, +1 when a is a quadratic residue modulo p , and -1 when it is not. The *Jacobi symbol*³ is defined with the Legendre symbol and its notation is similar: for any integers a and N with $N = p_1 p_2$ the product of two primes, the Jacobi symbol of a is defined as $\left(\frac{a}{N}\right) = \left(\frac{a}{p_1}\right) \left(\frac{a}{p_2}\right)$. Note that computing the Jacobi symbol $\left(\frac{a}{N}\right)$ is considered easy in general (even when the factorization of N is not known) [GM82].

The security of the GM scheme is based on the quadratic residuosity problem. The idea is to use the fact that in the special case when $\left(\frac{a}{N}\right) = +1$, it is difficult to know if a is a quadratic residue modulo N or not. Indeed, by definition, for a to be a quadratic residue modulo N , it must be a quadratic residue modulo p_1 and p_2 [GM82]. Yet, to have $\left(\frac{a}{N}\right) = +1$, we can have $\left(\frac{a}{p_1}\right) = \left(\frac{a}{p_2}\right) = +1$ or -1 . In the first case, a is indeed a quadratic residue modulo N , in the other it is not. Knowing only $\left(\frac{a}{N}\right) = +1$, and without the factorization of N , it is believed to be hard to distinguish the two cases. Note that with the factorization of N the problem is easy, but factoring is known to be hard for large prime factors. We can now detail the functioning of the scheme:

Key generation $(pk, sk) \leftarrow \text{KeyGen}(1^\lambda)$: Pick two large prime numbers p_1 and p_2 , compute $N = p_1 p_2 \in \mathbb{Z}$. Pick $x \in \mathbb{Z}_N^*$ such that $\left(\frac{x}{N}\right) = +1$ and x is *not* a quadratic residue modulo N . Output $((N, x), (p_1, p_2))$.

Encryption $c \leftarrow \text{Enc}(pk, m)$: For $m \in \{0, 1\}$, pick a random $r \in \mathbb{Z}_N^*$ and output $x^m r^2 \pmod{N}$. Note that in any case $\left(\frac{c}{N}\right) = +1$, but c is a quadratic residue modulo N only if $m = 0$.

Decryption $m \leftarrow \text{Dec}(sk, c)$: Test if c is a quadratic residue modulo N using the factorization of N and the fact that $\left(\frac{c}{p}\right) = +1 \Leftrightarrow c^{(p-1)/2} \equiv 1 \pmod{p}$ for any prime p . If c is a quadratic residue modulo N , output 0, else output 1.

³For simplicity, we present a restricted definition of the Jacobi symbol.

The GM scheme is homomorphic for the XOR operation:

$$\begin{aligned} \text{Enc}(pk, b_1) \times \text{Enc}(pk, b_2) &= x^{b_1+b_2} \cdot r_1^2 \cdot r_2^2 \pmod N \\ &\in \text{Enc}(pk, b_1 \oplus b_2) \end{aligned}$$

Ciphertexts can be combined this way in an unlimited manner. The homomorphic operation is indeed a XOR, and not a OR, because when $b_1 = b_2 = 1$, the resulting ciphertexts is $x^2 r_1^2 r_2^2 \pmod N$ which is a quadratic residue modulo N , thus it decrypts to 0. GM is also scalar homomorphic

$$\begin{aligned} \text{Enc}(pk, b_1)^{b_2} &= x^{b_1 \cdot b_2} \cdot r^{2 \cdot b_2} \pmod N \\ &\in \text{Enc}(pk, b_1 \cdot b_2) \end{aligned}$$

Note however that if $b_2 = 0$, the resulting ciphertext is equal to 1 and leaks information on b_2 . This can be thwarted using *re-randomization*: a ciphertext encrypting some message can be turned into another randomly distributed ciphertext that *looks like* a totally unrelated ciphertext, but that encrypts the same message. For this, given $c = x^m r^2 \pmod N$, compute $c \cdot r'^2 = x^m (r r')^2 \pmod N$ for a random number r' . Re-randomizing requires knowing the public key used to encrypt the ciphertext. This feature is useful in many contexts, for example to avoid the tracing of messages in networks [Rap06].

The cryptosystem is semantically secure (*i.e.* IND-CPA secure) under the assumption that the quadratic residuosity problem modulo a composite number N is intractable, and on the assumption that factorizing such a N is hard. To obtain $\lambda = 80$ bits of security, it is widely believed that with $|N| = 1024$ is sufficient. For $\lambda = 128$ bits, $|N|$ should be 2048, etc.

Considering efficiency, note that a message composed of l bits expands to $l \cdot |N|$. Ciphertext expansion is thus $l \cdot |N| / l = |N|$ bits, which is very large. The complexity of these primitives largely depends on $|N|$ as well: encryption asks for a square and one multiplication and can be computed in time $\tilde{O}(|N|)$, and decryption requires 2 exponentiations, computable in time $\tilde{O}(|N|^2)$ [GM84]. Thus, a trade-off security-efficiency appears: as $|N|$ increases, security increases, but efficiency decreases.

Finally, note that according to Gjøsteen [Gjø05], the security of GM can also be based on the *decisional subgroup membership problem*. This problem is defined over a group \mathbb{H} (the ciphertext space) and two non-trivial subgroups \mathbb{H}_1 (the encoded message space) and \mathbb{H}_2 (the set of the “cloaking” elements of the encryption) such that $\mathbb{H} = \mathbb{H}_1 \times \mathbb{H}_2$ and $\mathbb{H}_1 \cap \mathbb{H}_2 = \{1\}$. It consists in deciding if, given an element $x \in \mathbb{H}$, it also belongs to \mathbb{H}_1 or not. For an appropriate choice of groups, the problem is believed to be hard. GM is actually a special case of the subgroup membership problem, where \mathbb{H} is the subgroup of \mathbb{Z}_N^* whose elements have Jacobi symbol $+1$, and \mathbb{H}_1 is the subgroup of \mathbb{H} of quadratic residue modulo N . In the same article, Gjøsteen actually notes that many homomorphic cyptosystems’ security is based on the subgroup membership problem.

2.2.2 ElGamal

Defined three years later by Taher ElGamal [ElG85], this cryptosystem is one of the most used in practice, after RSA. Note however that it is not always used for its homomorphic property, but merely as a standard, efficient, public key scheme. It is defined as follows:

Key generation $(pk, sk) \leftarrow \text{KeyGen}(1^\lambda)$: Pick an adequate (see below) cyclic group \mathbb{G} subgroup of \mathbb{Z}_p^* , of large prime order q (such that $q|(p-1)$) with generator g . Pick at random $x \in \mathbb{Z}_q$ and compute $h = g^x \in \mathbb{G}$. Output $((p, q, g, h), x)$.

Encryption $c \leftarrow \text{Enc}(pk, m)$: For $m \in \mathbb{Z}_q$, pick a random $r \in \mathbb{Z}_q$. Encode m into an element m' of \mathbb{G} . Output the pair $(g^r, m'.h^r) \in \mathbb{G}^2$ (values are modulo p).

Decryption $m \leftarrow \text{Dec}(sk, c)$: Let $c = (c_1, c_2)$. Compute

$$\frac{c_2}{c_1^x} = m' \frac{h^r}{(g^r)^x} = m' \frac{g^{xr}}{g^{xr}} = m' \pmod{p}$$

Output m , the decoding of m' in \mathbb{Z}_q .

Before exposing the homomorphic properties of ElGamal, we detail the scheme's security and explain why we need to encode messages. First, bear in mind that it would be convenient if the message space was $\mathcal{M} = \{0, \dots, n\}$ for some n , as it is often the case in public key cryptography. ElGamal proposed initially [EIG85], to set \mathcal{M} and \mathbb{G} to be \mathbb{Z}_p^* for a large prime p : as $\mathcal{M} = \mathbb{G}$, we can directly encrypt integers from a set close to our goal, *i.e.* $\{1, \dots, p\}$. However, in this setting, the scheme is insecure [CMPP06]. Indeed the security of ElGamal relies on the Decision Diffie-Hellman (DDH) assumption [TY98] which states that given g a generator of some commutative group $\mathbb{G} = \langle g \rangle$, along with three values $g^a, g^b, h \in \mathbb{G}$ for some $a, b \in \mathbb{Z}$, it is impossible to know if $h = g^{ab}$ or not. For this assumption to hold, \mathbb{G} must be a group of prime order q . This is not the case when $\mathbb{G} = \mathbb{Z}_p^*$ because the order of \mathbb{G} is $(p-1)$, which is not a prime number when p is prime. Hence, we need to change our choice for \mathbb{G} . There are several types of groups in which the DH assumption holds, but typically \mathbb{G} is chosen as a small subset of \mathbb{F}_p^* of prime order q , with p a large prime, and q that divides $(p-1)$ (*i.e.* \mathbb{G} is a Schnorr group). Typically, $|p| = 1024$ bits and $|q| = 160$ bits. However, if we want to keep the message space close to $\{0, \dots, n\}$ for some n , it is necessary to *encode* messages into \mathbb{G} , because the scheme operates inside this structure. Also, this encoding must be efficiently invertible so that at the decryption phase, the correct message is decoded. Thus, we set the message space to be \mathbb{Z}_q . With these settings and parameters, ElGamal, as described above, is proved IND-CPA secure under the DDH assumption.

Finally, as GM, ElGamal's security alternatively relies on a specific case of the subgroup membership problem, where the ciphertext space is $\mathbb{H} = \mathbb{G} \times \mathbb{G}$, \mathbb{H}_1 is generated by (g, g^x) with $x \in \mathbb{Z}_q^*$, and \mathbb{H}_2 is generated by $(1, g)$ [Gjø05].

We now consider homomorphism. Let's first suppose that the message space is actually \mathbb{G} . For $i \in \{1, 2\}$, $m'_i \in \mathbb{G}$ and $c_i = \text{Enc}(pk, m_i) = (c_{i,0}, c_{i,1}) = (g^{r_i}, m'_i.h^{r_i})$, we have:

$$\begin{aligned} c_1 \times c_2 &= (c_{1,0} \times c_{2,0}, c_{1,1} \times c_{2,1}) \\ &= (g^{r_1+r_2}, m'_1.m'_2.h^{r_1+r_2}) \\ &\in \text{Enc}(pk, m'_1.m'_2) \end{aligned}$$

ElGamal also supports scalar multiplication: if $(c_1, c_2) = \text{Enc}(pk, m'_1)$, then $(c_1, c_2 \times m'_2)$ encrypts $m'_1.m'_2$; homomorphic scalar exponentiation: $(c_1^{m_2}, c_2^{m_2})$ encrypts $m'_1^{m_2}$, for any $m_2 \in \mathbb{Z}_q$; and re-randomization: $(c_1.g^{r'}, c_2.y^{r'})$, with $r' \in \mathbb{Z}_q$ a random number, is a randomly distributed ciphertext for the same message as $c = (c_1, c_2)$.

Then, if the encoding from the message space $\mathcal{M} = \mathbb{Z}_q$ to \mathbb{G} is an isomorphism, the resulting scheme is homomorphic for these properties stated above, with a message space as desired. For this, we must have $\text{Encode}(m_1). \text{Encode}(m_2) = \text{Encode}(m_1.m_2)$. However, as noted in [CMPP06], the existence of such an efficient encoding is unclear, and it is most likely that encoding will degrade homomorphism. A solution is to directly work in $\mathcal{M} = \mathbb{G}$, which is suitable in some contexts but not in general, or to use the *encoding-free* version of ElGamal from Chevalier-Mames *et al.* [CMPP06]. In this work, the authors slightly modify the scheme to avoid encoding and still ensure IND-CPA security. This scheme's performances are similar to the original ElGamal,

but security relies on the *Decisional Class Diffie-Hellman* problem, which has not been proven equivalent to the DDH problem. Thus, the IND-CPA security of the scheme is conjectured, but not proven (yet).

Another alternative to avoid encoding consists in using the hash-based ElGamal, in which hash messages before encryption. We disregard this solution, as it implies losing all homomorphic properties. Finally, one can also use the *additive* variant of ElGamal, where the encryption of a message $m \in \mathbb{Z}_q$ is $(g^r, g^m h^r)$ and no encoding is needed. Multiplying two ciphertexts gives $(g^{r_1+r_2}, g^{m_1+m_2} h^{r_1+r_2}) \in \text{Enc}(m_1 + m_2 \bmod q)$, and decryption requires computing a discrete logarithm, which is computationally impossible in general, but feasible for small message values.

Generally speaking, ElGamal is among the most efficient PHE schemes. We consider the multiplicative variant, without taking encoding in account. As the messages are represented by 1 group element, and the ciphertexts by 2 group elements, the ciphertext expansion is 2, which is much better than GM and quasi-optimal⁴. Encryption requires 2 exponentiations and 1 multiplication and its complexity is $\tilde{O}(|q||p|)$, as decryption consists in 1 exponentiation and 1 division, computable in time $\tilde{O}(|q||p|)$.

2.2.3 Benaloh, Naccache-Stern and Okamoto-Uchiyama

Benaloh's cryptosystem can be seen as an improved generalization of GM to higher residues [Ben94]. Designed in 1994, the main goal of the scheme was to create a *dense* probabilistic cryptosystem, *i.e.* a scheme with ciphertext expansion as small as possible. We detail the scheme as described by Fousse *et al.* [FLA11] who provide a corrected version of Benaloh⁵:

Key generation $(pk, sk) \leftarrow \text{KeyGen}(1^\lambda)$: Choose a block size $l \in \mathbb{Z}$, and two large primes p and q , such that:

- l divides $(p - 1)$
- $\gcd(l, (p - 1)/l) = 1$
- $\gcd(l, q - 1) = 1$

Set $N = pq$. Pick $y \in \mathbb{Z}_N^*$ such that:

- \forall prime factor r of l , $y^{(p-1)(q-1)/r} \neq 1 \pmod N$

Output $((y, l, N), (p, q))$.

Encryption $c \leftarrow \text{Enc}(pk, m)$: For $m \in \mathbb{Z}_l$, pick a random $r \in \mathbb{Z}_N^*$. Output $y^m r^l \pmod N \in \mathbb{Z}_N^*$.

Decryption $m \leftarrow \text{Dec}(sk, c)$: Notice that, because we have $r^{(p-1)(q-1)} = 1 \pmod N$ by Euler's theorem,

$$\begin{aligned} c^{(p-1)(q-1)/l} &= y^{m(p-1)(q-1)/l} r^{l(p-1)(q-1)/l} \\ &= y^{m(p-1)(q-1)/l} \pmod N \end{aligned}$$

Because, by construction, $y^{(p-1)(q-1)/l} \neq 1 \pmod N$, if $c^{(p-1)(q-1)/l} = 1 \pmod N$, output $m = 0$.

Else, if l is small enough, perform an exhaustive search, *i.e.* $\forall m' \in \mathbb{Z}_l$, test if $y^{-m'} c$ is an encryption of 0. If yes, output $m = m'$.

⁴An expansion factor in $]1, 2]$ is possible, but 2 is the smallest possible integer factor.

⁵[FLA11] proves the original scheme incorrect in the sense that decryption is ambiguous.

If l is of moderate size, we can use the baby-step giant-step algorithm to speed up decryption.

If l is larger but has many small prime factors, we can rely on index calculus techniques [FLA11].

Benaloh is additively homomorphic and supports homomorphic scalar multiplication:

$$\begin{aligned} \text{Enc}(pk, m_1) \times \text{Enc}(pk, m_2) &= y^{m_1+m_2} (r_1 r_2)^l \pmod{N} \\ &\in \text{Enc}(pk, m_1 + m_2 \pmod{l}) \end{aligned}$$

$$\begin{aligned} \text{Enc}(pk, m_1)^{m_2} &= y^{m_1 \times m_2} (r^{m_2})^l \pmod{N} \\ &\in \text{Enc}(pk, m_1 \times m_2 \pmod{l}) \end{aligned}$$

This scheme is also re-randomizable: for a random number $r' \in \mathbb{Z}_N^*$, $c' = c \cdot r'^l \pmod{N}$ is a randomly distributed ciphertext for the same message that c encrypts.

Security of the private key is based on factorization, so p and q should be large enough so that $|N| \geq 1024$. The scheme is IND-CPA secure, based on the *higher residuosity problem*, which is similar to the quadratic residue problem: the assumption is that given z , l and N , it is computationally impossible to decide if z is a l^{th} residue modulo N , *i.e.* it is hard to find x such that $x^l \equiv z \pmod{N}$. Once more, according to Gjøsteen [Gjø05], this scheme's security can be based on the subgroup membership problem where $\mathbb{H} = \mathbb{Z}_N^*$, \mathbb{H}_1 is the cyclic subgroup of order l of \mathbb{H} generated by y , and \mathbb{H}_2 is the set of invertible l^{th} powers in \mathbb{H} .

We can see that there is a balance between security, ciphertext expansion and efficiency. On one hand, the ciphertext expansion is $|N|/|l|$, which decreases when l increases. On the other hand, encryption (consisting in 2 exponentiations) runs in time $\tilde{O}(|l||N|)$ and decryption in time $\tilde{O}(|l||N|^2)$ for the exhaustive search method, $\tilde{O}(\sqrt{l})$ for the baby-step giant-step method, and the index-calculus method is in-between. However, in this last method, the parameter l need to be a large smooth factor of $(p-1)$. As l is part of the public key, this provides a hint towards the factorization of N , and endangers the security of the secret key. In definitive, complexity increases but ciphertext expansion decreases with l .

Naccache-Stern [NS98] (1998) is a derivative of Benaloh and improves upon the basic scheme. The main differences lie in the properties of the values p , q and l and in the definition of the decryption process, which uses the Chinese Remainder theorem. This leads to a less complex decryption, in time $\tilde{O}(|N|^3)$ (using pre-computations). Thus l can be set higher than in Benaloh, yielding a better ciphertext expansion for similar performances. Security is based on the same assumptions (the higher residuosity problem or alternatively the subgroup membership problem) and homomorphic properties are the same as Benaloh.

Okamoto-Uchiyama [OU98] (1998) can be seen as a improvement of Benaloh. But it is actually very close to Paillier's scheme and lays its foundations The main difference with previous works is that N is defined as p^2q , for p and q primes of k bits. Messages m are taken modulo p and encrypted by $c = g^{Nr+m} \pmod{N}$ with $g \in \mathbb{Z}_n^*$ and r a random element of \mathbb{Z}_n . The general ideas behind encryption and decryption are the same in this scheme and in Paillier's (but the parameters and requirements are different). Thus, we defer their description to the section on Paillier's scheme. Ciphertext expansion in Okamoto-Uchiyama is exactly 3: messages are modulo p (thus k bits) and ciphertext are modulo N (thus $3k$ bits). Security is based on the factorization of N .

2.2.4 Paillier and derivatives

Designed in 1999 by Pascal Paillier, this scheme is the most known, and maybe the most efficient partially homomorphic encryption scheme. Although Paillier interestingly comes back to a standard RSA modulus $N = pq$, it is the direct continuation of the work of Okamoto and Uchiyama. In the following description, $\lambda(N)$ denotes the Carmichael function (in our case, $\lambda(N) = \text{lcm}(p-1, q-1)$), and $L_N(\cdot)$ denotes the function defined over the set $\mathcal{S}_N = \{x < N^2 \mid x \equiv 1 \pmod N\}$ as $L_N(x) = (x-1)/N$.

Key generation $(pk, sk) \leftarrow \text{KeyGen}(1^\lambda)$: Pick two large primes p and q and compute $N = pq$. Randomly select $g \in \mathbb{Z}_{N^2}^*$ such that $\text{gcd}(L_N(g^{\lambda(N)} \pmod{N^2}), N) = 1$. Output $((N, g), (p, q))$. Alternatively, the secret key can be $\lambda(N)$.

Encryption $c \leftarrow \text{Enc}(pk, m)$: For $m \in \mathbb{Z}_N$, pick a random $r \in \mathbb{Z}_N^*$. Output $g^m r^N \pmod{N^2} \in \mathbb{Z}_{N^2}^*$.

Decryption $m \leftarrow \text{Dec}(sk, c)$: Output

$$\frac{L_N(c^{\lambda(N)} \pmod{N^2})}{L_N(g^{\lambda(N)} \pmod{N^2})} \pmod N$$

Note that $L_N(g^{\lambda(N)} \pmod{N^2})$ can indeed be inverted, as in KeyGen we verified that its gcd with N is 1.

We explain how the decryption works in a few paragraphs. First, note that the scheme is additively homomorphic and scalar multiplicatively homomorphic:

$$\begin{aligned} \text{Enc}(pk, m_1) \times \text{Enc}(pk, m_2) &= g^{m_1+m_2} (r_1 \cdot r_2)^N \pmod{N^2} \\ &\in \text{Enc}(pk, m_1 + m_2 \pmod N) \end{aligned}$$

$$\begin{aligned} \text{Enc}(pk, m_1)^{m_2} &= g^{m_1 \times m_2} (r_1^{m_2})^N \pmod{N^2} \\ &\in \text{Enc}(pk, m_1 \times m_2 \pmod N) \end{aligned}$$

As the other schemes, Paillier is re-randomizable. For a random number $r' \in \mathbb{Z}_N^*$, $c \cdot r'^N \pmod{N^2}$ looks unrelated to c but encrypts the same message.

The security of Paillier is based on the *composite residuosity class problem*, which states that deciding whether a number z is a N^{th} residue modulo N^2 , *i.e.* deciding if there exists $y \in \mathbb{Z}_{N^2}^*$ such that $y^N = z \pmod{N^2}$, is intractable. Alternatively, we can base security of Paillier on the decisional subgroup membership, where $\mathbb{H} = \mathbb{Z}_{N^2}^*$ is the ciphertext space, \mathbb{H}_1 is the subgroup generated by g , and \mathbb{H}_2 is the subgroup of invertible N^{th} powers modulo N^2 .

Paillier, as ElGamal, has a quasi-optimal ciphertext expansion equal to 2. As for performances, encryption requires 2 exponentiations computable in time $\tilde{O}(|N|^2)$. The most computationally expensive task in decryption is the exponentiation $c^{\lambda(N)} \pmod{N^2}$, computable in time $\tilde{O}(|N|^2)$. The value $L_N(g^{\lambda(N)} \pmod{N^2})^{-1} \pmod N$ can be pre-computed once and for all at the key generation. Those asymptotic complexities are comparable to previous works, but the scheme is sensibly better in practice.

Now, to understand how the decryption works, we first describe the case when the generator g is $1+N$, which is the typical choice in most implementations (it is easy to check that $1+N$ is a suitable candidate for g in the KeyGen procedure). In this case because of Carmichael's theorem

stating that for all x , $x^{N \cdot \lambda(N)} = 1 \pmod{N^2}$, we have for any $c = (1 + N)^m r^N \pmod{N^2}$:

$$\begin{aligned} c^{\lambda(N)} &= (1 + N)^{m\lambda(N)} r^{N\lambda(N)} \\ &= (1 + N)^{m\lambda(N)} \\ &= 1 + \binom{m\lambda(N)}{1} N + \binom{m\lambda(N)}{2} N^2 + \dots \\ &= 1 + m\lambda(N) \cdot N \pmod{N^2} \end{aligned}$$

Now, applying L_N on $c^{\lambda(N)} \pmod{N^2}$ yields $m\lambda(N)$. Computing $L_N((1 + N)^{\lambda(N)} \pmod{N^2})$ in the same way leads to $\lambda(N)$, therefore the decryption procedure outputs $\frac{m\lambda(N)}{\lambda(N)} = m \pmod{N}$.

In the general case, when $g \neq 1 + N$, the decryption procedure is also correct. Indeed, as all suitable g in the KeyGen procedure (including $1 + N$) are generators of the subgroup of $\mathbb{Z}_{N^2}^*$ we denoted \mathbb{H}_1 earlier, they are all interchangeable. Thus, for any ciphertext $c = g^m \cdot r^N \in \mathbb{Z}_{N^2}^*$ there is a unique pair (m', r') such that $c = (1 + N)^{m'} r'^N \pmod{N^2}$. And in particular, $\exists x, y \in \mathbb{Z}_N^*$ s.t. $g = (1 + N)^x y^N \pmod{N^2}$. Thus $c = g^m r^N = ((1 + N)^x)^m \cdot (y^m r)^N \pmod{N^2}$, and we have $\text{Dec}(sk, c) = x \cdot m \cdot \lambda(N) / (x \cdot \lambda(N)) = m \pmod{N}$.

Damgård-Jurik [DJ01] (2001) simplifies and generalizes Paillier. The main modification is the replacement of the message and ciphertext space: messages are taken in $\mathbb{Z}_{N^s}^*$, and ciphertexts lie in $\mathbb{Z}_{N^{s+1}}^*$. Also, the parameter g is simply set to $1 + N$, reducing the public key size without endangering security. We can see that, if $s = 1$ this yields Paillier's cryptosystem, and when s increases, the ciphertext expansion $\frac{s+1}{s}$ gets closer to 1. This is the main achievement of DJ: to offer an expansion arbitrary close to 1. However, the cost is a quadratic overhead in s : the encryption and decryption costs are respectively at least s^2 and s times Paillier's complexity. As a result, it is actually more efficient to use Paillier k times than DJ with $s = k$. The authors show that the security of their scheme is equivalent to Paillier's security. In the same paper, Damgård and Jurik describe a threshold version of Paillier, and detail an application to electronic voting.

Galbraith [Gal02] (2002) transposes Paillier and DJ in the elliptic curves setting. The scheme is however less efficient: expansion is equal to 3, and encryption and decryption are slower than Paillier's, but the exact comparison of the schemes is unclear [FG07]. The scheme works over the group defined by the elliptic curve $E(\mathbb{Z}_{N^2})$. In this context, we have as public parameters $N = pq$ and the description of the elliptic curves (*i.e.* its coefficients). The secret key is a value derived from the factorization of N , noted $d = \text{lcm}(\#E(\mathbb{F}_p), \#E(\mathbb{F}_q))$. The public key also contains a point Q , such that the order of Q is a divisor of d , *i.e.* such that $dQ = 0$ in $E(\mathbb{Z}_{N^2})$. To encrypt $m \in \mathbb{Z}_N$, choose a random $r \in \mathbb{Z}_N^*$ and output the point $S = rQ + (mN : 1 : 0)$. Homomorphic addition of messages is performed by point addition. Decryption is done by multiplying S with the secret value d : $dS = r(dQ) + (mdN : 1 : 0) = (mdN : 1 : 0)$. Knowing d and N , it is easy to recover m using the first coordinate of the result. In Rappe's thesis, a threshold version of Galbraith's cryptosystem can be found [Rap06].

This paragraph closes the presentation of PHE schemes. This list is not meant to be exhaustive, but merely presents the main PHE schemes and put in light their evolution. We could have also mentioned the scheme of Castagnos [Cas06], the Paillier-ElGamal amalgam [DJ03], or the Regev [Reg05] and LPR [LPR10] schemes. The last two schemes are actually the foundations of some recent FHE schemes and will be given more attention in section 2.3.5. All the

presented schemes are summed up in Table 2.1. We detail homomorphic capacities (except re-randomization, which is a common feature to all PHE schemes), ciphertext expansion, complexity of encryption and decryption, and parameters size for a 128-bit security.

Scheme	Op.	Exp.	Times		Security $\lambda = 128$	Misc.
			Enc	Dec		
RSA (77)	\times	1	$ N ^2$	$ N ^2$	$ N = 2048$	Deterministic. Insecure if homomorphic.
GM (82)	\oplus	$ N $	$ N $	$ N ^2$	$ N = 2048$	Bit-wise
ElGamal (85)	$\times, +, \odot, \otimes$	2	$ q p $	$ q p $	$ q = 320$ $ p = 2048$	Encoding required
Benaloh (94)	$+, \odot$	$ N / l $	$ N l $	$\geq \sqrt{l}$	$ N = 2048$	$l \geq 2$ arbitrary
N-S (98)	$+, \odot$	$ N / l $	$ N l $	$ N ^3$	$ N = 2048$	Improves Benaloh
O-U (98)	$+, \odot$	3	$ N ^2$	$ N ^3$	$ N = 2048$	Improves Benaloh
Paillier (99)	$+, \odot$	2	$ N ^2$	$ N ^2$	$ N = 2048$	Improves O-U
D-J (01)	$+, \odot$	$\frac{s+1}{s}$	$s^2 N ^2$	$s N ^2$	$ N = 2048$	Generalizes Paillier. $s \geq 1$ arbitrary.
Galbraith (02)	$+$	3	\geq D-J		$ N = 2048$ $ E(\mathbb{Z}_{N^2}) = 256$	D-J in elliptic curves

All **Times** are complexity given in \tilde{O} notation (except Galbraith).

The **Operation** denoted by \odot is the scalar multiplication, while \otimes is scalar exponentiation.

Note: these figures are estimations, and may hide substantial complexity factors.

Table 2.1: Comparison of PHE schemes

2.2.5 Implementations and practical considerations

In conclusion of this section on PHE schemes, we discuss the most practical PHE schemes, and present known implementations. In practice the main schemes used by the cryptographic community or in cryptographic libraries are either ElGamal, Paillier (and its generalization), and Okamoto-Uchiyama. To the best of our knowledge, other schemes' implementations (if they exist) are merely academic proof of concepts.

ElGamal is a serious alternative to RSA, and is actually used in many standards, for instance in GPG (Gnu Privacy Guard, an implementation of the PGP protocol). However, this scheme is not used in its homomorphic form. Instead, its hash-based variant is used, so as to provide IND-CCA2 security (a higher security than IND-CPA, desirable in general). Thus, ElGamal's usage seems to be restricted to traditional cryptography, and it is unclear why its homomorphic version has been disregarded. It may be partially explained by the necessary message encoding, that either makes the scheme inefficient or harms homomorphism. In the same order of idea,

the scheme of Okamoto and Uchiyama was used to design EPOC, a cryptosystem standardized by IEEE (IEEE P1363, submitted in 2000), but here again the scheme is in a IND-CCA2, non-homomorphic variant, thus of little interest to us.

In definitive, only Paillier and Damgård-Jurik are actually used for their homomorphic properties. The main applications of Paillier include electronic voting [Ste08] and private information retrieval [Cha04]. There exists many implementation of Paillier, in Java⁶, Python⁷ or in C^{8,9}. However, according to the authors themselves, these implementations should not be used to perform sensible computations, as their code have not been studied and secured. Therefore, in practice, even Paillier and its derivatives are still academic objects, and no standard implementation has been issued as of today.

We still report running times for the C implementation of Paillier by John Bethencourt. Tests were run on a Intel i7 2.20GHz. Timings are reported in Table 2.2.

$ N $	KeyGen	Enc	Dec	Add	Sc. mult
512	8.7ms	1.4ms	0.3ms	< 0.1ms	< 0.1ms
1024	63.1ms	3.7ms	2.4ms	< 0.1ms	0.2ms
2048	649.5ms	19.5ms	17.1ms	< 0.1ms	1.2ms

Table 2.2: Implementation of Paillier (libpaillier)

In a parallel work, Jakobsen *et al.* [JMN10, Section 5.3] construct an efficient version of Paillier and measure their performances in terms of CPU cycles for the encryption primitive. In particular, for $|N| = 2048$, the number of CPU cycles for encryption is around 2^{25} , which approximately corresponds to a time of 16ms on a 2GHz single core CPU, and is in accordance with our own measurements. Note that the measured running times are above what could be expected. This is mainly due to the simple and unoptimized implementations used. Also, surprisingly, encryption takes more time than decryption. This can be explained by the pre-computation of $\lambda(N)$ and $L_N(g^{\lambda(N)} \bmod N)^{-1}$ in KeyGen instead of Dec. Also, this last fact is in accordance with Damgård and Jurik's estimations [DJ01, Fig. 1], which show that the number of multiplications in Paillier's encryption is $|N|/2$ times more than in RSA's.

Although it would be desirable to also measure Paillier's performances in a simple use-case such as electronic voting, there is actually very few tests that were run, or at least, the results were not published. There exist an interactive demonstration of electronic voting with Paillier¹⁰, but it runs on Javascript (which has very poor performances), and only works for very small modulus size $|N|$ that do not represent real world settings. This demonstration is however useful to understand a concrete example of e-voting with Paillier.

At last, note that the practical applications of PHE schemes in general are numerous: electronic voting, secure multi-party computation, private information retrieval, oblivious transfer, (non-interactive) knowledge proofs, secret sharing, commitment schemes, e-auctions, ... Many of these applications can be found in Rappe's thesis [Rap06].

2.2.6 Somewhat homomorphic schemes

As mentioned before, SHE schemes are of little interest. Therefore we do not dedicate an section to them, but briefly present the most known schemes in the literature.

⁶The homomorphic encryption project

⁷<https://github.com/mikeivanov/paillier>

⁸libpaillier, from John Hopkins University

⁹PaillierGMP

¹⁰<http://security.hsr.ch/msevote/paillier>

The first construction allowing to compute on encrypted data in an *non-interactive* manner dates from 1999 [SYY99]. It was designed to evaluate NC^1 circuits (*i.e.* circuits of logarithmic depth and gates fan-in¹¹ 2): one party, Alice, possesses an input x , and the other, Bob, has a circuit \mathcal{C} . Both parties want to keep their input secrets, but should learn $i\mathcal{C}(x)$ at the end. Though it is not a SHE scheme *per se*, this construction allows computation of several OR and NOT gates. As the set of logical gates $\{NOT, OR\}$ provides functional completeness, we can theoretically evaluate any function. However, the limitation comes from the huge ciphertext size growth: for a depth d circuit, ciphertexts' size at output is $O(8^d)$. This is why the construction can only evaluate logarithmic depth circuits, *i.e.* in order to obtain polynomial time running times, one ought to limit the circuit's depth to $O(\log |inputs|)$.

A second construction have been proposed by Ishai and Paskin in 2007 [IP07]. The main difference with the previous construction is that this work relies on *branching programs* (BP) instead of circuits. Barrington proved that a branching programs include NC^1 circuits [Bar89], thus this construction is slightly better than the previous one. This work allows the evaluation of a branching program BP owned by Bob on Alice's secret input x . The idea can be summed up as follows: to encrypt x , Alice encrypts each of its bits by generating $|x|$ Oblivious Transfer requests, and sends them all at once to Bob. Bob recursively generate Oblivious Transfers answers in a bottom-up manner, and sends to Alice the answer associated to the branching program's initial node. Alice then recursively decodes the Oblivious Transfer answers and eventually recovers the branching program output corresponding to x . Ciphertexts size are polynomial in the BP length (*i.e.* the longest path in the branching program). Alice's workload is also polynomial in $length(BP)$, but Bob's workload depends polynomially on the *size* (*i.e.* number of nodes) of the program. However, as a NC^1 circuit of depth $O(\log n)$ is equivalent to a branching program of length $O(n)$, we did not make much progress and the scheme is limited to evaluating polynomial size branching programs (with gates fan-in possibly above 2), or equivalently, logarithmic depth NC^1 circuits (with gates fan-in equal to 2, by definition).

One of the most known SHE scheme, though not used in practice, is BGN (for Boneh-Goh-Nissim). It was designed in 2005 [BGN05], and improved by Gentry *et al.* in 2010 [GHV10]. The homomorphic properties of this scheme are quite odd: it allows unlimited additions and 1 multiplication. As the title of the original article suggests, this corresponds to evaluation of multi-variate polynomials of degree 2. Actually, BNG is simply an additive homomorphic scheme where ciphertexts lie in a multiplicative group in the manner of Paillier. To enable 1 multiplication, the idea of Boneh *et al.* is to use a bilinear application. The security is directly based on the subgroup membership problem. The main limitation of this scheme is the size of the plaintexts space: the scheme allows encryption of small values up to a bound T . Indeed, the decryption operation depends polynomially on T , limiting its value. There is no ciphertext size growth (contrarily to other SHE schemes), but ciphertext expansion is quite large: $|N|/|T|$ with $T \approx \log N$. In Gentry's improvement [GHV10], the homomorphic capacities stay the same, but the plaintext space is larger, the encryption/decryption complexities are lower, and the ciphertext expansion is also lower. The scheme actually uses matrices and is well suited to encrypt polynomials or large integers. This scheme is also the first SHE scheme which security is based on the LWE problem (see section 2.3.5).

We close our short SHE schemes list with the MGH cryptosystem [MGH10], the most recent but also the most powerful one. Indeed, it allows many additions and several multiplications, or equivalently it evaluates multivariate polynomials of degree d , which corresponds to circuits of depth $\log d$. The scheme is limited because of the exponential ciphertexts growth in d . However, as the growth is slower than Sander *et al.* [SYY99] and Ishai *et al.* [IP07], more multiplications are possible in practice. The main idea is to use a simple additive homomorphic encryption

¹¹The gate's number of input wire.

scheme (satisfying some reasonable properties), along with a tensorial product technique in order to enable multiplications. In a sense, multiplication of encrypted messages corresponds to the tensorial product of their ciphertexts. The resulting ciphertext then contains redundant data, and its size is quadratic (*i.e.* if $|c| = n$, $|c \otimes c| = n^2$). The workload at decryption depends exponentially on d (the degree of the evaluated polynomial). Their construction is actually a general template to transform any additive scheme into a SHE scheme by *chaining* it multiple times with itself. The authors argue that their template fits very well with the BNG transformation of Gentry *et al.*, allowing unlimited additions and $2d$ multiplications for the same price.

To sum up, SHE schemes all allow many additions and either a very limited number of multiplications (with no ciphertext size growth), either several multiplications but at the cost of an exponential ciphertext growth. Moreover, none of these schemes is efficient, and their security may be somewhat hazardous. Therefore, we disregard these schemes in the rest of the chapter.

2.3 Fully homomorphic schemes

As we have seen so far, cryptographers have been searching for a way to compute arbitrary functions on encrypted data for more than 30 years. Early 2000s, homomorphic encryption has made a lot of progress, and eventually, in 2009, the groundbreaking work of Gentry in his thesis [Gen09a] gave rise to the first (theoretically) fully homomorphic encryption scheme. This type of scheme is capable of evaluating any program that would be executable in the clear. This section is dedicated to the description of techniques and schemes derived from Gentry's construction, starting from 2009 until early 2014.

First, we describe the emergence of FHE schemes and the idea developed by Gentry. We then detail the main existing schemes, dividing them according to their underlying cryptographic concepts. At last, we consider FHE from a practical point of view.

2.3.1 Overview of FHE

Outbreak of FHE and Gentry's idea

Before Gentry's thesis, even theoretically, FHE was believed impossible to achieve. In 2009, it became a tangible possibility, although it was at the time extremely far from practical (and still today, to some extent). Two main ideas were crucial for the apparition of FHE: the suppression of the ciphertext size growth, and the *bootstrapping* technique.

The starting point of Gentry is a powerful somewhat homomorphic scheme capable of evaluating a small (but reasonable) number of additions and multiplications, *i.e.* a few AND and XOR logical gates. However, contrarily to other SHE schemes, the ciphertexts from Gentry's scheme no longer suffered size growth when they were applied homomorphic operations. In other words, Gentry solved the problem of the *compactness*: the size of the output of the Eval primitive no longer depended on the complexity of the evaluated function or the number of inputs. However, this gave rise to a new problem, inherent to Gentry's SHE scheme: the *noise growth*. For security reasons, ciphertexts in Gentry's scheme contained some randomness (as all semantically secure schemes), and this randomness was added and multiplied along with plaintexts during homomorphic operations. This is what Gentry called the *noise*. When the noise (*i.e.* the accumulated randomness) exceeded a certain threshold (typically, a modulo), decryption would output erroneous values, *i.e.* the scheme would no longer be correct. To sum up, Gentry's ciphertexts were

of fixed size, but in their inner structure, the noise was growing exponentially, in such a way that after a few operations, decryption was erroneous.

To circumvent this issue, the trick proposed by Gentry was to *refresh* ciphertext when the noise became too large. Indeed, as freshly encrypted ciphertext had very low noise level, Gentry proposed to decrypt and immediately re-encrypt noisy ciphertexts, so as to reset their noise. As a result, to evaluate a complex function on encrypted data, an evaluator would need to decrypt-then-re-encrypt data every time the noise reached a critical level. The issue was however that this seemed to imply giving away the secret decryption key for third parties to be able to compute on encrypted data, which is unacceptable. The workaround imagined by Gentry was to perform the decrypt-then-re-encrypt operation *in the encrypted domain*. This means *homomorphically decrypting* ciphertexts when their noise became too large, so as to reset it. Gentry called this idea *bootstrapping*, because it implies the SHE scheme to be able to evaluate its own decryption circuit. This decrypt-then-re-encrypt procedure is called **Recrypt**, and is the key to create a FHE scheme. Note that, in practice this procedure begins by re-encrypting and *then* decrypts, for obvious security reasons. We describe it here in an abstract manner.

Recrypt $c_{clear} \leftarrow \text{Recrypt}(pk, f_{\text{Dec}}, \text{Enc}(pk, sk), c_{noisy})$: Takes as input the public key, the function describing the scheme's own decryption primitive, the secret key sk encrypted under its corresponding public key, and a noisy ciphertext $c_{noisy} \in \text{Enc}(pk, m)$. Performs the following operations:

1. Generates $c' = \text{Enc}(pk, c_{noisy})$. c' can be seen as a *doubly encrypted* ciphertext.
2. Outputs $\text{Eval}(pk, f_{\text{Dec}}, \text{Enc}(pk, sk), c')$.

If the SHE scheme can correctly evaluate f_{Dec} , *i.e* if $f_{\text{Dec}} \in \mathcal{F}$, the result is a valid ciphertext of m encrypted under pk , with low noise. The **Recrypt** primitive is meant to be run by any third party that needs to compute on the encrypted data. The basic idea is for the evaluator, who only knows the public key and a circuit she wants to evaluate, to apply logical gates of the circuit one by one, and use the **Recrypt** primitive between each gate. Note that the **Recrypt** primitive takes as input the secret key encrypted under its own public key. This means that the value $\text{Enc}(pk, sk)$ must be public. This value is sometimes called *evaluation key*, and may be part or distinct from the public key. Publicizing the evaluation key may be harmful for the scheme's security. Actually, there is no proof (at the moment) that it does not. Gentry merely made the assumption that it was not harmful. This (strong) assumption is called *circular assumption*.

We now have all the elements to formulate Gentry's bootstrapping theorem, central in his work.

Theorem 1 - Bootstrapping [Gen09a] (Informal)

A SHE scheme capable of evaluating its own decryption primitive, plus at least one non-negligible operation, can be transformed into a FHE scheme under the circular security assumption.

We presented the main ideas that gave rise to the first FHE scheme. We sum them up in the following description of what is now called *Gentry's blueprint*, or Gentry's procedure to construct FHE schemes (we provide a simplified version of it). Note that this procedure uses a computational model based on circuits instead of programs or functions: because designing a FHE scheme is very complex, we resort to a very simple computational model.

Definition 1 - Gentry's blueprint

1. *Construct a SHE scheme capable of evaluating a few AND and XOR gates.*

2. Express the decryption primitive under the form of a circuit, and simplify (Gentry uses the term “squash”) the decryption circuit so that the SHE scheme is capable of evaluating its own decryption circuit plus an additional AND gate.
3. Make the circular security assumption, and use the bootstrapping theorem to turn the scheme into a FHE one.

This procedure is, still today, the only known way to achieve fully homomorphic encryption. Gentry details how to apply this blueprint to the scheme of van Dijk *et al.*, making the parallel between FHE and a jewelry store owner delegating tasks to untrusted workers [Gen10].

A more recent paradigm: Leveled FHE

Gentry’s initial procedure presents several strong shortcomings, and although it is the only way to achieve pure FHE, many modern schemes deviate from it. Therefore, they are no longer actually *fully* homomorphic schemes, but we will see that they offer consequent homomorphic capacities and far better performances. This section explains why and how recent FHE schemes deviate from Gentry’s blueprint.

The first shortcoming of Gentry’s blueprint is its (in)security. Indeed, the circular assumption has not been extensively studied yet, and when simplifying the decryption circuit, one may need to make additional strong assumptions. For instance, in his thesis, Gentry simplifies the decryption procedure by publicizing a hint on the secret key, so as to offload decryption work to KeyGen (see section 2.3.3). Secondly, the Recrypt primitive is extremely costly. In the first schemes, its complexity was $\Omega(\lambda^4)$, with λ the security parameter, and it was called after *each* circuit gate evaluation.

In view of these remarks, there are two possibilities: to directly address these shortcomings, or simply to disregard bootstrapping. Works following the first possibility allowed to reduce the asymptotic cost of bootstrapping to $\tilde{O}(\lambda \log \lambda)$ [ASP13, ASP14], to remove some assumptions [GH11a], and to reduce the number of calls to Recrypt during the homomorphic evaluation of a circuit (*i.e.* avoid using it systematically at each gate) [LP13]. However, even with these advances, bootstrapping remains very costly in practice. This is why the second solution, which propose *not* to use bootstrapping at all, actually prevails today. Although the resulting schemes are not strictly speaking fully homomorphic ones, they offer substantial capacities, and are far less costly. The first scheme without bootstrapping was introduced by Brakerski, Gentry and Vaikuntanathan in 2012 [BGV12], and represents a major milestone in the evolution of FHE: since then, this has been the favored approach by the community. The authors named this kind of FHE schemes *leveled fully homomorphic encryption* (LHE):

Definition 2 - Leveled Fully Homomorphic Encryption

A HE scheme is said leveled if it can evaluate circuits of polynomial size and depth at most L , with L a parameter of the scheme.

Understand that L can be set arbitrarily high, thus LHE schemes are also (theoretically) capable of evaluating any function on encrypted data. Of course, the complexity of the scheme increases with L . To better understand the notion of LHE, we detail its differences with SHE and FHE: *Somewhat vs. Leveled*. In SHE schemes, the homomorphic capacities are not clearly defined, *i.e.* we do not have precise bounds on the complexity of the functions they can evaluate. In most cases, the multiplicative depth of evaluable circuits is logarithmic (in size of the inputs). On the other hand, LHE schemes can evaluate circuits of *linear* multiplicative-depth, this depth can be exactly controlled with the parameter L , and correctness is guaranteed for circuits of depth lower or equal to L . At last, ciphertext size growth is often exponential in SHE, while it is null

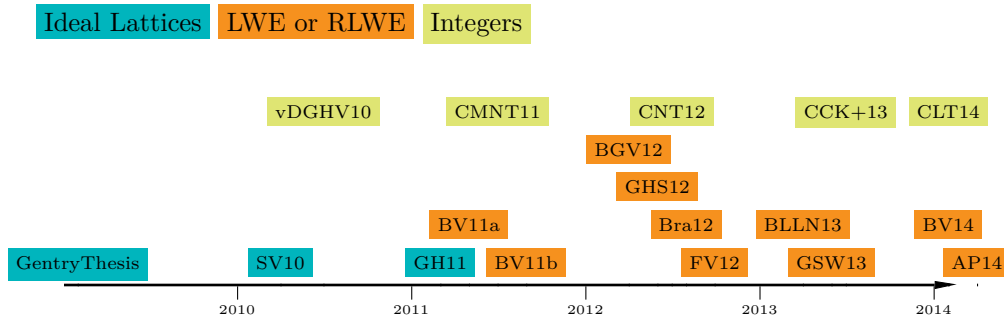


Figure 2.1: Listing of FHE schemes

in LHE.

Leveled vs. Fully. Complexity and overhead of LHE schemes depend on the requested depth L . In FHE, complexity only depends on the security parameter (*i.e.* it is independent of the evaluated function). In both FHE and LHE, ciphertext size growth is null. It may seem like FHE is better, because its cost is independent of the size of the evaluated circuit. But because of the great cost of the Recrypt operation, LHE is much more efficient in practice, at least up to a certain threshold L_{max} . Above this threshold, the circuit's depth becomes very large, and the cost of LHE becomes larger than FHE's. The value of this threshold is unclear and depends on the scheme considered.

In definitive, research in FHE can be divided in two main (complementary) directions: improving bootstrapping and pure FHE, or improving LHE. The two directions follow the same goal, *i.e.* to manage and minimize the ciphertext noise growth efficiently. Pure FHE manages noise *via* bootstrapping, but LHE has to find other ways to keep noise at a low level, at least until the depth L is reached. Thus, research in LHE focuses on inventing and improving techniques, such as *tensoring* [BGV12], *modulus switching* [BGV12], *key switching* (also called *re-linearization*) [BGV12, Bra12], *packing* (or *batching*) of many plaintexts in 1 ciphertext [SV10, BGV12], and *scale invariance* [Bra12] (which actually replace the costly technique of modulus switching).

In the rest of this section, the generic term FHE refers to both pure FHE and LHE.

2.3.2 Three different types of FHE schemes

We borrow the categorization from Cheon *et al.* [CCK⁺13] and divide schemes in 3 main classes, according to the concepts and cryptographic problems they relate to. First schemes, including Gentry's original one, were based on ideal lattices. After what, in 2011, Brakerski and Vaikuntanathan [BV11b] noticed that FHE could be based on much simpler problems, to wit *Learning With Errors* (LWE) or Ring-LWE (R-LWE). A year after, the number of new FHE schemes per year exploded, as shown by Figure 2.1, and remains high until today (note that several schemes are not represented in the timeline). Also, starting from 2010, a different branch of FHE schemes, based on plain integers, is investigated in parallel. At the rate of 1 new scheme every year approximately, this branch adapts the new techniques from the (R-)LWE branch to the integer setting. Although integer-based schemes are very inefficient, they provide different assumptions and concepts to base security on, which is crucial in cryptography. Figure 2.1 presents the main FHE schemes from 2009 until today, and Fig 2.2 shows the evolution of underlying cryptographic concepts used to enable FHE. As we can see, the (R-)LWE branch is by far the most productive one.

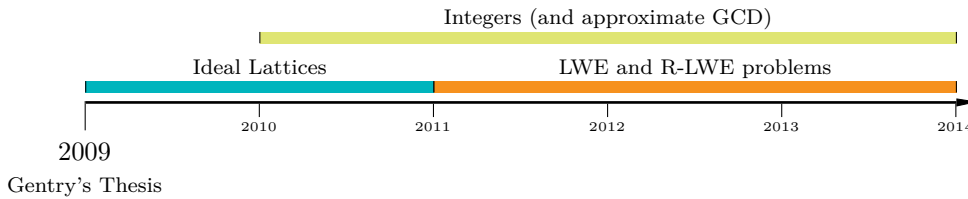


Figure 2.2: Evolution of underlying concepts of FHE schemes

Note that in the rest of this work, we present *public key* FHE schemes, but most schemes can be either symmetric or asymmetric. Actually, there exist a generic transformation from symmetric to asymmetric FHE [Rot11]. Also, bear in mind that, unless specified otherwise, all FHE schemes are meant to encrypt only bits, *i.e.* the message space is $\{0, 1\}$, and to evaluate circuits.

2.3.3 Schemes over ideal lattices

The first FHE schemes were based on *ideal lattices* [Gen09a, Gen09b, SV10, SS10, GH11b, GH11a], and involved complex mathematical and cryptographic notions. In a nutshell, a lattice \mathcal{L} is a discrete subgroup of \mathbb{R}^n , often taken on \mathbb{Z}^n . A lattice is generated by a *basis* $\mathbf{B} = (\mathbf{b}_1, \dots, \mathbf{b}_n) \in \mathbb{Z}^{n \times n}$ composed of n linearly independent vectors, *i.e.* $\mathcal{L} = \left\{ \sum_{i=1}^n a_i \mathbf{b}_i \mid a_i \in \mathbb{Z} \right\}$. An ideal of some ring R is defined as the set $I \subset R$ such that $\forall a \in I, \forall b \in R, a \times b \in I$, *i.e.* I *absorbs* multiplication by elements of R . An *ideal lattice* \mathcal{IL} of the lattice \mathcal{L} described above is thus an ideal of the ring $R = \mathbb{Z}[X]/f(X)$ (the set of polynomials over \mathbb{Z} modulo an irreducible polynomial $f(X) \in \mathbb{Z}[X]$). Indeed, the set $\mathbb{Z}[X]/f(X)$, with $f(X)$ a polynomial of degree n , is isomorphic to the set \mathbb{Z}^n , which is the support of \mathcal{L} .

Schemes based on these tools are in limited number, of very limited practical use, and none of them is LHE (*i.e.* they all use bootstrapping). Their theoretical complexity is huge, and in practice, running times of their primitives are far beyond reasonable. We still describe the characteristics of the scheme of Gentry as it appears in [Gen09b].

The structure of ideal lattices seemed well suited as a first candidate for FHE, because they naturally provide additive and multiplicative homomorphism. Also, schemes in ideal lattices typically have low complexity decryption functions, *e.g.* logarithmic in the size of the lattice [Gen09b], which is convenient to enable bootstrapping. Gentry's first scheme's security was based on the (decisional) *closest vector problem* (CVP) for ideal lattices, which consists, given a lattice \mathcal{L} , a vector space V and a vector $\mathbf{v} \in V$ not in the lattice, in finding $\mathbf{v}' \in \mathcal{L}$ the vector in \mathcal{L} closest to \mathbf{v} up to an approximation factor γ . That is, finding \mathbf{v}' such that $\|\mathbf{v}'\| \leq \gamma \cdot \|\mathbf{v}_{exact}\|$ with \mathbf{v}_{exact} the actual closest vector to \mathbf{v} in \mathcal{L} . The building block SHE scheme was, according to Gentry, slightly better than the SHE scheme BGN [BGN05], *i.e.* it allowed unlimited additions and more than one multiplication for comparable efficiency. However, the original decryption primitive consisting in a matrix-vector product, it could not be evaluated by the scheme as is. In order to simplify the decryption circuit and enable bootstrapping, the scheme was modified to publish a set of vectors at key generation. This set contained a secret subset which sum was roughly equal to the secret key. As a result, the simplified decryption circuit simply consisted in a sparse subset sum plus some other minor operations, and could be evaluated by the scheme. The hardness assumption invoked to prove the security of the modified scheme is the *sparse*

subset sum (SSS) problem, which states that it is impossible to distinguish a random set of elements from a set of elements for which the sum of a secret subset is equal to 0. The hardness assumptions of SSS and SVP are strong ones, and need to be further studied. Thus, although the security of the scheme was proved, it relies on strong assumptions. The per-gate overhead (*i.e.* the additional cost of homomorphic gate evaluation) of this scheme is roughly $\tilde{O}(\lambda^6)$ [SS10], a cost mainly explainable by the need for ciphertext refreshing at each gate.

Two main contributions tried to improve upon this scheme. Stehlé and Steinfeld obtained a per-gate overhead of $\tilde{O}(\lambda^{3.5})$ by providing a more precise analysis of the SSS problem yielding smaller parameters, and by allowing negligible but non-zero decryption error. The second work, by Gentry and Halevi [GH11a], marks the first deviation from the blueprint: it still uses bootstrapping, but it does not need to simplify the decryption circuit, and thus removes of the SSS assumption. To do so, they express the decryption function of their basic SHE scheme as a sequence composed of: a series of additions, then a series of multiplications, and another series of additions. The series of additions are performed by the SHE scheme, as the multiplications are done by switching to a multiplicative PHE scheme such as ElGamal. This involves homomorphically evaluating ElGamal's decryption function to switch back to the SHE. Eventually, they obtain a scheme as complex as the original one, they also use bootstrapping, but avoid the strong SSS assumption and achieve better security.

2.3.4 Schemes over the integers

As soon as 2010, van Dijk *et al.* decided to transpose Gentry's FHE scheme to plain integers [vdGHV10], a much less complex tool to work with compared to ideal lattices. Drafts of these schemes were previously proposed prior to 2009 as SHE candidates, but van Dijk *et al.* formalized it, improved it, and integrated it within Gentry's blueprint to create a FHE scheme. Along the years, more integers-based schemes were designed [CMNT11, CNT12, CCK⁺13, CLT14], mostly adapting the new techniques borrowed from the (R-)LWE to the integer settings, an operation which is often non-trivial. Some new techniques also emerged from integer-based schemes, such as the public key compression techniques [CNT12]. As the authors note, the main appeal of the integer approach is its conceptual simplicity.

Concerning efficiency, integer-based schemes perform equivalently or even worse than their ideal lattice counterparts. Similarly to Gentry's original scheme which security is based on approximation problems in ideal lattices, schemes on the integers are based on the approximate GCD problem. This problem asks to find p given many $x_i = p \cdot q_i + r_i \approx p \cdot q_i$ for q_i large integers, r_i small values (compared to q_i), and p an odd integer. Note that this assumption is still being studied and new attacks have recently been disclosed [CN12, CNT12].

As the 2010 scheme from van Dijk *et al.* is one of the simplest, we describe it here. More exactly, we describe the SHE scheme that constitutes the building block of the FHE one. Note: in the following, the value $x \bmod y$ lies in $(-y/2, y/2]$, and not in $[0, y)$.

Setup $params \leftarrow \text{Setup}(1^\lambda)$: Set $\rho = \omega(\log \lambda)$ the bit-length of the noise (high enough to protect against brute-force attack on the noise) ; $\eta \geq \rho \cdot \Theta(\lambda \cdot \log^2 \lambda)$ the bit-length of the secret key (high enough to support homomorphism and evaluate the simplified decryption circuit) ; $\gamma = \omega(\eta^2 \log \lambda)$ the bit-length of the integers in the public key (high enough to thwart attacks on the approximate GCD problem) ; and $\tau \geq \gamma + \omega(\log \lambda)$ the number of integers in the public key (high enough for the security proof to hold). Let $\mathcal{D}_{\gamma, \rho}(p)$ be the distribution outputting $x_i = pq_i + r_i$ for $q_i \in \{0, \dots, \lfloor 2^\gamma/p \rfloor\}$ and $r_i \in \{\lceil -2^\rho \rceil, \dots, \lfloor 2^\rho \rfloor\}$. Output $params = (\rho, \eta, \gamma, \tau, \mathcal{D}_{\gamma, \rho}(p))$.

Key generation $(pk, sk) \leftarrow \text{KeyGen}(params)$: Pick a large η -bit prime $p \in \{2^{\eta-1}, \dots, 2^\eta - 1\}$. Sample $\tau + 1$ values x_i from $\mathcal{D}_{\gamma, \rho}(p)$ and relabel them so that x_0 is the largest.

Restart unless x_0 is odd and $(x_0 \bmod p)$ is even. Output $((x_0, \dots, x_\tau), p)$.

Encryption $c \leftarrow \text{Enc}(pk, m)$: For $m \in \{0, 1\}$, pick a random $r \in \{-2^{2\rho} + 1, \dots, 2^{2\rho} - 1\}$, and a random subset $S \subseteq \{1, \dots, \tau\}$. Output $c = (m + 2r + 2 \sum_{i \in S} x_i) \bmod x_0$.

Decryption $m \leftarrow \text{Dec}(sk, c)$: Output

$$\begin{aligned} & \left[\left[c \right]_p \right]_2 \\ &= \left[\left[m + 2r + 2 \sum_{i \in S} x_i \bmod x_0 \right]_p \right]_2 \\ &= \left[\left[m + 2r + 2 \sum_{i \in S} x_i - (p \cdot q_0 + r_0) \cdot \lfloor c/x_0 \rfloor \right]_p \right]_2 \\ &= \left[\left[m + 2r + 2(p \sum_{i \in S} q_i + \sum_{i \in S} r_i) - (p \cdot q_0 + r_0) \cdot \lfloor c/x_0 \rfloor \right]_p \right]_2 \\ &= \left[\left[m + 2r' + pq' \right]_p \right]_2 \\ &= m \text{ if } r' < p/4 \end{aligned}$$

Noting that ciphertexts have the form $\text{Enc}(pk, m) = m + 2r' + pq'$, we can see that

$$\begin{aligned} c_1 + c_2 &= m_1 + m_2 + 2(r'_1 + r'_2) + p(q'_1 + q'_2) \\ &\in \text{Enc}(pk, m_1 + m_2 \bmod 2) \end{aligned}$$

The addition of two ciphertexts gives an encryption of the addition of their underlying plaintexts, with noise bit-length $2\rho + 1$ (assuming c_1 and c_2 are freshly encrypted ciphertext with noise bit-length roughly equal to $|r'| = 2\rho$)¹². In the same way, we have

$$\begin{aligned} c_1 \times c_2 &= m_1 \cdot m_2 + 2(r'_1 \cdot m_2 + r'_2 \cdot m_1 + 2r'_1 \cdot r'_2) \\ &\quad + p(q'_1 m_2 + q'_2 m_1 + pq'_1 q'_2 + 2r'_2 q'_1 + 2r'_1 q'_2) \\ &= m_1 \cdot m_2 + 2r'' + pq'' \\ &\in \text{Enc}(pk, m_1 \cdot m_2 \bmod 2) \end{aligned}$$

If c_1 and c_2 are fresh ciphertexts, the resulting ciphertext has a noise bit-length roughly $2 \times 2\rho + 2$. After l multiplications, the noise bit-length is approximately $2^l \cdot 2\rho$: the noise grows exponentially with the number of multiplications applied. As for the decryption to be correct, the noise must remain smaller than the half of the modulo p , *i.e.* $2^l \cdot 2\rho < |p/4| \leq \eta - 2$, the scheme roughly allows $l = \log(\eta/\rho)$ multiplications. In other words, the bit size of p is exponential in l , which imposes a strong limitation of l to small values. Note that this scheme is *not* leveled homomorphic, and the aim is only to be able to evaluate the decryption circuit (and one additional operation).

For the proposed parameters [vDGHV10] $\rho = \lambda$, $\eta \approx \lambda^2$, $\gamma \approx \lambda^5$ and $\tau = \gamma + \lambda$, the public key size is above $\tilde{O}(\lambda^{10})$ (because the public key is composed of $(\tau + 1)$ γ -bit values). This results in 10^6 TB for $\lambda = 80$, which is above any practical system. Approximately $l = \log(\eta/\rho) = \log \lambda$ successive multiplications are possible, which is barely enough to evaluate the decryption circuit. To simplify the decryption circuit, which in its original form asks for a computation of $\lfloor c/p \rfloor$, the authors augment the public and secret keys so that the decryption can be computed with only additions and subtractions. Finally, note that the ciphertext expansion is approximately $|x_0|/1 = \gamma \approx \lambda^5$, which is enormous.

¹²This is an approximation. See the original paper for a detailed analysis of the noise growth.

Several works improve upon this first integer-based scheme. In 2011 Coron *et al.* [CMNT11] reduce the public key size from $\tilde{O}(\lambda^{10})$ to $\tilde{O}(\lambda^7)$, by encrypting using a quadratic (instead of linear) form in the x_i values, thus yielding a scheme asymptotically comparable to ideal lattice-based schemes. In 2012, Coron *et al.* [CNT12] again reduce the public key size, to $\tilde{O}(\lambda^5)$ using higher degree in the x_i values, and import the modulus switching technique from [BGV12] to the integer setting, yielding a LHE scheme on the integers (the first one). From 2012 and after, all integer-based schemes are LHE ones. A year later, Cheon *et al.* [CCK⁺13] import the ciphertext batching/packing technique from [SV10, BGV12] using the Chinese Remainder Theorem. With this technique, one ciphertext can now encrypt $\tilde{O}(\lambda^2)$ plaintexts at a time, and the ciphertext expansion is lowered from $\tilde{O}(\lambda^5)$ in the original scheme to $\tilde{O}(\lambda^3)$. Finally, in 2014, Coron *et al.* [CLT14] integrate the scale invariant property of Brakerski [Bra12], allowing $|p| = \eta$ to be *linear* in the parameter L of the LHE scheme, instead of exponential. This means that exponentially more multiplications are possible with the same p compared to the 2010 scheme.

2.3.5 Schemes based on the (R-)LWE problem

The third type of FHE schemes are the ones based on the LWE and Ring-LWE problems [BV11a, BV11b, BGV12, GHS12a, Bra12, FV12, BLLN13, GSW13, BV14, ASP14]. The (R-)LWE branch of FHE is the most active one, and these schemes are the most efficient ones to date. Except for the two 2011 schemes, all the schemes from this branch are LHE ones, but can be made fully homomorphic using bootstrapping. In general, leveled schemes are quite efficient, but using bootstrapping lowers their efficiency to a level comparable to ideal lattice-based schemes. In this section, we will present the LWE problem and Brakerski's scheme [Bra12]. Then, to put in light the security-efficiency-homomorphism trade-off in modern fully HE schemes, we briefly show how to derive concrete parameters for Brakerski's scheme. Finally, we give the differences between LWE and Ring-LWE, and sump up recent advances in FHE.

The Learning With Error problem

The *Learning With Errors* (LWE) problem was put forward in 2005 by Regev [Reg05]. It comprises 3 parameters: the dimension n , the modulus q , and the error factor α . The last two often depend on n , which is the main parameter.

In the rest of the section, we denote $\mathcal{U}(S)$ the uniform distribution over some set S . \mathcal{N}_s denotes a discrete Gaussian distribution on \mathbb{Z} centered in 0 and of parameter s . For more information on discrete Gaussians, see the description of Regev [Reg05, Sec. 2, §2]. By $x \leftarrow \mathcal{D}$ we denote the sampling of an element from distribution \mathcal{D} .

Definition 3 - The $\text{LWE}_{n,q,\alpha}$ problem

Let $n > 1$ and $q > 2$ be two integers, and $\alpha \in]0, 1[$. Define $\mathcal{D}_{n,q,\alpha}^{\text{LWE}}(\mathbf{s})$ for a vector $\mathbf{s} \in \mathbb{Z}_q^n$ as the distribution over \mathbb{Z}_q^{n+1} obtained as follows:

$$\mathbf{a} \leftarrow \mathcal{U}(\mathbb{Z}_q^n), e \leftarrow \mathcal{N}_{\alpha q}, \text{ output } (\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^{n+1}$$

The decisional version asks to distinguish between $\mathcal{D}_{n,q,\alpha}^{\text{LWE}}(\mathbf{s})$ and $\mathcal{U}(\mathbb{Z}_q^{n+1})$.

The computational version asks, given polynomially many samples of $\mathcal{D}_{n,q,\alpha}^{\text{LWE}}(\mathbf{s})$, to find \mathbf{s} .

Both versions are intractable for the right choice of parameters, and were proven equivalent [Reg05]. We do not describe the ring-based variant of this problem, because it is very similar except that it works with the ring $R = \mathbb{Z}[X]/f(X)$ for $f(X)$ an irreducible polynomial of degree d , and that \mathbf{a} and \mathbf{s} are no longer vectors but polynomials. We will see later what this implies for FHE schemes.

For the security of LWE to hold, n , q and α need to respect several constraints. We defer the description of these constraints to section 2.3.5.

Brakerski's scheme

Brakerski's 2012 scheme [Bra12] is one of the simplest LWE-based LHE schemes to date, and we use it as a representative example of recent FHE scheme. To be more precise, the following scheme is a description of Regev's encryption scheme [Reg05], which is the starting point of Brakerski.

Setup $params \leftarrow \text{Setup}(1^\lambda)$: Output the LWE parameters (n, q, α) accordingly to λ (see section 2.3.5).

Key generation $(pk, sk) \leftarrow \text{KeyGen}(params)$: Sample $\mathbf{s} \leftarrow \mathcal{U}(\mathbb{Z}_q^n)$. Let $N = (n + 1)\lceil \log q \rceil$. Sample a matrix $\mathbf{A} \leftarrow \mathcal{U}(\mathbb{Z}_q^{N \times n})$ and a vector $\mathbf{e} \leftarrow \mathcal{N}_{\alpha q}^N$. Compute $\mathbf{b} = [\mathbf{A} \cdot \mathbf{s} + \mathbf{e}]_q \in \mathbb{Z}_q^N$. Output $((\mathbf{A}, \mathbf{b}), \mathbf{s})$.

Note that the public key is an instance of $\text{LWE}_{n,q,\alpha}$ for the vector $\mathbf{s} = sk$.

Encryption $c \leftarrow \text{Enc}(pk, m)$: For $m \in \{0, 1\}$, pick a random $\mathbf{r} \in \{0, 1\}^N$ and output

$$\mathbf{c} = (\mathbf{r}^T \mathbf{A}, \mathbf{r}^T \mathbf{b} + \lfloor \frac{q}{2} \rfloor m) \in \mathbb{Z}_q^{n+1}$$

Note that \mathbf{c} is again the re-randomized version of the LWE instance linking pk and sk . By the *Leftover Hash Lemma* [ILL89], this re-randomized version is equivalent to a normal LWE instance.

Decryption $m \leftarrow \text{Dec}(sk, c)$: Let $\mathbf{c} = (\mathbf{u}, v)$. Compute

$$\begin{aligned} v - \langle \mathbf{u}, \mathbf{s} \rangle &= \mathbf{r}^T (\mathbf{A} \mathbf{s} + \mathbf{e}) + \lfloor \frac{q}{2} \rfloor m - \langle \mathbf{r}^T \mathbf{A}, \mathbf{s} \rangle \\ &= \lfloor \frac{q}{2} \rfloor m + \mathbf{r}^T \mathbf{e} \end{aligned}$$

If the result is closer to 0, output 0, if the result is closer to $q/2$, output 1.

Note that for decryption to work, it is *necessary* to have $\mathbf{r}^T \mathbf{e} < q/4$, else, decryption may output erroneous values. More exactly, we must have $\mathbf{r}^T \mathbf{e} < \lfloor \frac{q}{2} \rfloor / 2$, but we simplify the notation. It is easy to see that the scheme is additively homomorphic:

$$\begin{aligned} c_1 + c_2 &= (\mathbf{u}_1 + \mathbf{u}_2, v_1 + v_2) \\ &= ((\mathbf{r}_1^T + \mathbf{r}_2^T) \mathbf{A}, (\mathbf{r}_1^T + \mathbf{r}_2^T) \mathbf{b} + \lfloor \frac{q}{2} \rfloor (m_1 + m_2)) \\ &\in \text{Enc}(pk, m_1 + m_2 \pmod 2) \end{aligned}$$

However, decryption works *only if* $(\mathbf{r}_1^T + \mathbf{r}_2^T) \mathbf{e} \leq q/4$. This limits the number of additions that can be performed on a same ciphertexts.

Multiplication is far less trivial, as the Regev scheme is not originally multiplicatively homomorphic. Because ciphertexts are vectors, the multiplication of two ciphertexts yields a matrix or a scalar number. In the latter case, the ciphertext can no longer be decrypted. In the other case, the size of a ciphertext is squared at multiplication, which is in contradiction with the compactness property of FHE schemes (see section 2.3.1).

To enable multiplication and give rise to a LHE scheme with allowed multiplicative depth L , Brakerski modifies the Regev scheme as described below. The operation \otimes denotes the vector or matrix tensorial product.

- Encryption and decryption primitives do not need substantial changes;
- In KeyGen, Brakerski generates:
 - $L + 1$ secret keys, *i.e.* $\mathbf{s}_i \in \mathbb{Z}_q^n$ for $i \in \{0, \dots, L\}$, set $sk_i = \mathbf{s}_i$ and the decryption secret key to $sk = s_L$;
 - L matrices $\mathbf{P}_{i-1:i} \in \mathbb{Z}_q^{N^2 \lceil \log q \rceil \times (n+1)}$ (recall that $N = (n+1) \lceil \log q \rceil$), called *re-linearization matrices*. The matrix $\mathbf{P}_{i-1:i}$ can be seen as an application transforming a ciphertext matrix resulting from a multiplication and encrypted under sk_{i-1} , into a ciphertext vector for the same message, encrypted under sk_i .
- To multiply two ciphertexts $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{Z}_q^{n+1}$:
 1. Compute the vectors $\mathbf{c}_i^{(1)} = \text{PowersOfTwo}(c_i) = [(\mathbf{c}, 2\mathbf{c}, 4\mathbf{c}, \dots, 2^{\lceil \log q \rceil - 1} \mathbf{c})]_q \in \mathbb{Z}_q^N$.
 2. Compute the *tensor product* of $\mathbf{c}_1^{(1)} \otimes \mathbf{c}_2^{(1)}$.
This yields a matrix of size $N \times N$, which is flattened in a vector $\mathbf{c}^{(2)}$ of size N^2 .
At this point, $\mathbf{c}^{(2)}$ contains in some way the encryption of $m_1.m_2$ under a tensor secret key $sk_i^2 = \mathbf{s}_i \otimes \mathbf{s}_i$: indeed, the secret key hidden in the ciphertext was multiplied along with the messages¹³.
The ciphertext $\mathbf{c}^{(2)}$ also contains a lot of noise because randomness inside \mathbf{c}_1 and \mathbf{c}_2 (\mathbf{r}, \mathbf{A} and \mathbf{e}) was also multiplied. In particular, an undesirable factor q appears in the noise, and largely participate in the noise increase.

The goals of the next steps are threefold: (1) to reduce the noise, (2) to come back to a linear secret key, and (3) to obtain a ciphertext vector in \mathbb{Z}_q^{n+1} .

3. Compute $\mathbf{c}^{(3)} = \left\lfloor \frac{2}{q} \cdot \mathbf{c}^{(2)} \right\rfloor$.

This step is crucial to cancel the factor q in $\mathbf{c}^{(2)}$ and to reduce the noise in the ciphertext. This is the main contribution of Brakerski: it allows *scale invariance*.

4. Compute $\mathbf{c}^{(4)} = \text{BitDecomp}(\mathbf{c}^{(3)}) = (\mathbf{w}_0, \dots, \mathbf{w}_{\lceil \log q \rceil - 1}) \in \mathbb{Z}_q^{N^2 \lceil \log q \rceil}$
Such that $\mathbf{c}^{(3)} = \sum_i 2^i \mathbf{w}_i \pmod q$.

5. Finally, compute and output $[\mathbf{P}_{i-1:i}^T \cdot \mathbf{c}^{(4)}]_q \in \mathbb{Z}_q^{n+1}$.

We eventually come back to a vector of size $n+1$, because $\mathbf{P}_{i-1:i}$ is of size $(N^2 \lceil \log q \rceil) \times (n+1)$ and $\mathbf{c}^{(4)}$ is of size $N^2 \lceil \log q \rceil$.

And thanks to the structure of $\mathbf{P}_{i-1:i}$ the ciphertexts is now encrypted under $sk_i = \mathbf{s}_i$, without losing information.

This last step is called *re-linearization* (in the sense that ciphertexts are transformed back into a vector) or *key-switching* (in the sense that ciphertexts are now encrypted under s_i instead of s_{i-1}). It is the crucial procedure making the difference with the MGH scheme [MGH10]: MGH is actually the first scheme to use the tensor product for HE, but did not find the way to avoid the ciphertext size expansion phenomenon. It is the most expensive step of the whole scheme. Therefore, multiplication is the most expensive operation.

¹³Equivalently, we can see $\mathbf{c}^{(2)}$ as an encryption under a double public key, because sk is hidden in pk via a LWE instance.

At the end of the procedure, for freshly encrypted ciphertexts \mathbf{c}_1 and \mathbf{c}_2 with bounded noise $|\mathbf{e}_1| = |\mathbf{e}_2| \leq B \approx \alpha q^{14}$ the noise becomes $B \cdot \text{poly}(n)$. Generalizing this for L multiplications, the noise will be $B \cdot \text{poly}(n)^L$. As we can see, multiplication increases noise much more than addition (this is usually the case in FHE). Then, to be able to perform L multiplications, we must have $B \cdot \text{poly}(n)^L < q/4$. A detailed analysis of the noise growth in Brakerski's scheme [Bra12] leads to the relation $O(n \log q)^{L+O(1)} \leq q/B$, that the scheme should satisfy to allow L multiplications.

In practice, the public key is extremely large, because it contains L re-linearization matrices $\mathbf{P}_{i-1:i}$, each of size $N^2 \lceil \log q \rceil \times (n+1) \approx (n \lceil \log q \rceil)^3$. And the ciphertext expansion is $(n+1) \cdot |q|/1 \approx n|q|$, which is also huge. In a general manner, the overall complexity of the scheme increases when n or q increase.

Deriving the parameters: the security-efficiency-homomorphism trade-off

On one hand, we know that q/B must be high enough to allow the desired L multiplications: we can derive lower bounds on the LWE parameters. On the other hand, n and q must not be too large to make the scheme practical: we can derive asymptotic upper bounds. There is a third family of constraints on the values of n , q and α : those dictated by the security requirements of LWE.

Indeed, as for any cryptographic problem, the parameters of LWE must be set according to the best known attack against it. In our case, the best known attacks against LWE are algorithms resolving approximation problems in (standard) lattices such as the *shortest vector problem* denoted GapSVP_γ (in its decisional variant). It is interesting to note that FHE schemes tried to emancipate from lattices using the LWE problem, but their security still relies on problems similar to Gentry's original scheme. In the computational problem associated to GapSVP_γ , the goal is to find a short vector in a given lattice, up to an approximation factor γ . This problem is believed hard for small values of γ . Two different cryptographic proofs (*reductions*) show that if an algorithm exists for breaking $\text{LWE}_{n,q,\alpha}$ in polynomial time, there exists an algorithm to break GapSVP_γ in polynomial time for an approximation factor $\gamma = \tilde{O}(n/\alpha)$. This means that $\text{LWE}_{n,q,\alpha}$ is at least as hard as $\text{GapSVP}_{\tilde{O}(n/\alpha)}$. The first reduction is quantum [Reg05] (*i.e.* uses a quantum algorithm), and the second is classical [BLP⁺13] (*i.e.* uses a classical algorithm). We will only consider the classical proof, which imposes more constraints on the parameters. The best practical algorithm against GapSVP_γ is BKZ2.0 [CN11], a lattice basis reduction algorithm running in time $2^{\tilde{\Omega}(n/\log \gamma)}$. Therefore, to obtain a security of λ bits, we must have $2^{\tilde{\Omega}(n/\log \gamma)} \geq 2^\lambda$. The following list sums up the constraints from the classical reduction, along with the accumulated relations from previous paragraphs. Recall that $B \approx \alpha q$, by definition.

1. $q/B \geq O(n \log q)^{L+O(1)}$ (for homomorphism),
2. $n, \log q \leq \text{poly}(\lambda)$ (for efficiency),
3. $q \geq 2^{n/2}$ (for reduction to GapSVP),
4. $\alpha q \geq \sqrt{n}$ (for reduction to GapSVP),
5. $\gamma \approx \tilde{O}(n/\alpha)$ (for reduction to GapSVP),
6. $\gamma \leq O(n^{O(\log n)})$: γ should be at the maximum sub-exponential in n for GapSVP_γ . But here, Brakerski considers *quasi-polynomial* approximation factors. Ideally it should be in $O(n^c)$ for a small c , *e.g.* 2. (for security),
7. $2^{\tilde{\Omega}(n/\log \gamma)} \geq 2^\lambda$ (for security against BKZ2.0).

¹⁴More formally, we say that the distribution $\mathcal{N}_{\alpha q}$ of the error vector \mathbf{e} is supported on $[-B, B]$ with very high probability.

There are several links between these constraints, and some of them are opposed. In particular, by constraints 1 and 3, q should be greater than $B.n^{2L}$ (if we simplify greatly). However, according to constraints 5 and 6, $n/\alpha \leq n^{\log n}$, so we need to have $q \leq B.n^{\log n-1}$. This means that, for a constant initial noise $B = 8$ for instance (which is the proposed value in [LN14]):

$$8n^{2L} \leq q \leq 8n^{\log n-1}$$

This inequality can be satisfied only when $2L+1 \leq \log n$. Therefore, n must be *exponential* in L : to be able to perform several multiplications, n must be set very high. Yet, constraint 2 requires n to be polynomial in λ . Thus, L is limited to values in $O(\log(\text{poly}(\lambda)))$. More generally, we can see that augmenting q/B leads to better homomorphism because it leaves room for the noise within ciphertext to grow, but lowers security and/or performances.

Finally, constraint 7, which expresses the LWE parameters in function of the desired bit-security λ , dictates how concrete parameters values should be chosen. However, as of today, the running time of the BKZ2.0 algorithms are still unclear, and further studies are necessary to disclose what is hidden in the Ω notation in exponent. In the literature, n is often set between 10^3 and 10^5 , but the exact resulting security is unclear.

The reader may find a similar (far more thorough and accurate) analysis for two Ring-LWE-based schemes in [LN14]. The authors follow the same abstract procedure as in this section: analyse noise growth and deduce lower bounds, analyse security and deduce lower bounds, and finally deduce parameters values. The results highlight that, given a reasonable value for L , it is impossible for security and correctness to hold for “small” values of n (as it would be desirable for efficiency). For instance, for $L = 50$, Fig. 2.3 shows the maximum and minimum bit-size of q in function of n for security and correctness to hold (respectively). These curves were estimated based on the 5 points provided by Lepoint *et al.* [LN14, Table 3]. As we can see, before $n \approx 34000$, it is impossible to have $L = 50$.

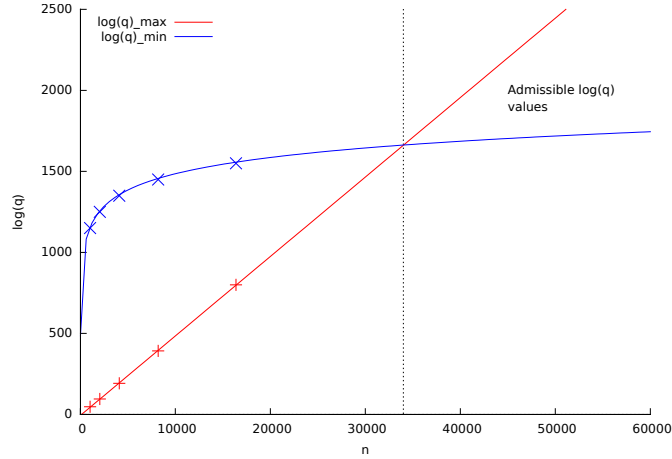


Figure 2.3: Maximal (resp. minimal) bit-size of q in function of n for the security (resp. correctness) of the BLLN scheme to hold (for $L = 50$)

Differences between LWE and Ring-LWE-based schemes

This section describes the differences between Ring-LWE and LWE constructions, and explain why the former leads to more efficient schemes. For a full description of Ring-LWE and Ring-LWE

based schemes, the reader may refer to the works of Fan *et al.* [FV12] or Bos *et al.* [BLLN13].

Firstly, we have seen that LWE is resistant against quantum *and* classical algorithms. Ring-LWE is only proven quantum resistant. Its classical security is conjectured, but still need to be proved. Also, Ring-LWE reduces to $GapSVP_\gamma$, but in *ideal* lattices, while LWE reduces to $GapSVP_\gamma$ in classical lattices.

Secondly, LWE works with vectors and matrices, as Ring-LWE works with polynomials. It is known that a polynomial can be represented as the vector of its coefficients (and conversely), but the great difference is that R-LWE-based schemes can use the natural and efficient arithmetics of polynomial rings to combine ciphertexts, which is not the case with vectors. This is why the public key and ciphertexts in Ring-LWE are only a couple of ring elements, whereas in LWE, they are large matrices or vectors. Note however, that Ring-LWE still need the costly *key-switching* operation because multiplying ciphertexts also gives rise to “doubly encrypted” ciphertexts.

To sum up LWE is (for now) more secure, but has an inherent quadratic overhead compared to Ring-LWE.

Recent evolutions

In 2012, at the moment Brakerski, Vaikuntanathan and Gentry [BGV12] introduced leveled homomorphic encryption, the approximation factor of $GapSVP_\gamma$ was sub-exponential (which is the maximum allowed), and security was based on *quantum* $GapSVP_\gamma$ (less restrictive on n , q and α than the classical version). The great achievement of this work is to manage the noise efficiently (using *modulus switching*) so that after L multiplications the noise would be $(B.poly(n))^L$ instead of $(B.poly(n))^{2^L}$ in previous works. In other words, for the same parameters and performances, exponentially more multiplications were possible.

Later in the same year, Brakerski [Bra12] managed to reduce the approximation factor to quasi-polynomial while basing security on classical $GapSVP_\gamma$ at the same time. Brakerski was able to achieve such improvements mainly thanks to its *scale invariance* property, which do not consider the noise *magnitude*, but the noise *ratio*, defined as q/B . In practice, Brakerski’s idea (put very simply) consists in dividing ciphertexts by a factor q after each multiplication (see step 3 of Brakerski’s multiplication procedure, section 2.3.5).

The schemes from [BGV12] and [Bra12] were initially described for LWE, but can be easily ported to R-LWE for better performances. The schemes of Fan-Vercauteren [FV12] (2012) and Bos *et al.* [BLLN13] (2013), however, were directly designed on R-LWE. They follow the lines of Brakerski’s scheme, using scale invariance, and benefit from the great efficiency of R-LWE. As a result, they are among the most efficient schemes today. In particular, Bos *et al.* rely on a slightly modified variant of R-LWE, and exhibits better efficiency than Fan and Vercauteren’s scheme.

In parallel, in 2013, a new scheme introduced by Gentry *et al.* [GSW13] brought new techniques for FHE. In this scheme, ciphertexts are directly represented by matrices. Consequently, homomorphic addition and multiplication are respectively performed by natural matrix addition and multiplication. Decryption is based on techniques relying on *eigenvectors* and *eigenvalues*. The main improvement compared to previous scheme is that there is no longer the need for key-switching or re-linearization, which are the most costly operations in modern schemes (both in terms of space and computation). Also, the noise growth is again lowered: for L multiplications, it grows from B to $B.(n+1)^L$. The approximation factors are roughly the same as in Brakerski. Unfortunately, this scheme is based on LWE and does not transpose to R-LWE. As a result, in practice, it does not show significant improvements in terms of efficiency compared to R-LWE schemes.

Finally, a 2014 scheme from Brakerski and Vaikuntanathan [BV14] reached a major stage in

FHE: they made LWE-based FHE as secure as traditional lattice-based public key encryption. That is to say, they achieved LHE with approximation factors in $O(n^2)$. Their construction improves upon the above 2013 scheme of Gentry *et al.* [GSW13], and shows a very low noise growth: after L multiplications, the noise magnitude is only $B.L.poly(n)$. This is the first scheme with *linear* noise growth in L , allowing (again) exponentially more multiplication compared to previous schemes. This almost optimal result allows to set polynomial factors in $O(n^{2+\epsilon})$, with ϵ as close to 0 as necessary. Their scheme is however very inefficient. Indeed, the parameter n depends exponentially on $1/\epsilon$, which means that efficiency degrades exponentially when the approximation factor get closer to $O(n^2)$. Also, the authors make several assumptions on the circuit evaluated in order to optimize homomorphic operations, and thus restrict the functions evaluable by the scheme.

2.3.6 Implementations and practical considerations

This section gives details on the performances and practical efficiency of FHE and LHE schemes, and provides a list of known FHE implementations (for all types of schemes).

Schemes on ideal lattices

Two ideal lattice-based schemes were implemented: one by Smart and Vercauteren [SV10], the other by Gentry and Halevi [GH11b]. As the Smart-Vercauteren implementation only includes the basic SHE scheme, we do not consider it here. The second implementation realizes all the scheme, including the bootstrapping part, and make use of many minor optimizations in order to make the scheme work in practice. We report performances in Table 2.3, for a claimed security of $\lambda \approx 80$ bits¹⁵. Even though the implementation gave up some security in order to gain some performances, these numbers are far from practical. The goal of the authors was merely to show that FHE was possible.

Security	$ pk $	KeyGen	Enc	Dec	Recrypt
$\lambda = 80$ $n = 512$	17MB	2.5s	0.19s	< 0.01s	6s
$\lambda = 80$ $n = 32768$	2.3GB	2.2h	3min	0.66s	31min

n is the lattice dimension.

Table 2.3: Implementation of an ideal lattice scheme [GH11b] (2011)

Schemes on the integers

Every year or so, a new integer-based scheme is published and a new implementation is provided. We report in Table 2.4 running times and public key sizes for the 2014 LHE scheme which includes all the improvements from section 2.3.4. The platform used was an Intel Xeon at 2.9GHz. Note that for $\lambda = 80$, figures are worse than those of ideal lattices, but this is explained by two factors: in the ideal lattice implementation, security is actually below 80 bits, and the platform is more powerful. Indeed, for $\lambda = 62$, the 2014 integer scheme performs better than the ideal lattice one.

¹⁵The actual security is below 80 bits, as the authors do not make n , the lattice dimension, depend on λ

Security	$ pk $	KeyGen	Enc	Dec	Recrypt
$\lambda = 42$	3.2MB	0.5s	$< 0.1s$	$< 0.1s$	0.1s
$\lambda = 62$	704MB	5min	3s	0.2s	2.8s
$\lambda = 80$	100GB	213h	5min	24s	4.6min

Table 2.4: Implementation of an integer-based scheme [CLT14] (2014)

Schemes on (R-)LWE

We report in Table 2.5 running times for two Ring-LWE based schemes: FV [FV12] and BLLN¹⁶ [BLLN13]. These results are extracted from a work by Lepoint and Naehrig [LN14] who compare the capacities and efficiencies of the two schemes. The tests were performed on a Intel Core i7-2600 at 3.4 GHz. We can see that, apart from the key generation procedure, BLLN performs better than FV, especially when multiplying ciphertexts.

Scheme	Security	KeyGen	Enc	Dec	Mult.
FV	$\lambda = 80$	0.2s	34ms	65ms	148ms
BLLN	$\lambda = 80$	3.4s	16ms	15ms	49ms

Table 2.5: Implementation of two Ring-LWE-based schemes [LN14] (2014)

In addition to these implementations, the authors propose a detailed analysis of the scheme and deduce possible values for the parameter q for several values of the parameters n and L . From these results, it is straightforward to compute the ciphertext and keys sizes, and to evaluate the number of multiplications allowed L (by interpolation from [LN14, Table 3]). Indeed, in FV, ciphertexts consist in 2 ring elements, secret keys are 1 ring element, and public keys comprise $(1 + \log_{2^{32}} q)$ ring elements. A ring element itself is a degree n polynomial with $|q|$ -bits coefficients, and thus consists in a $n \times |q|$ -bit value. Table 2.6 shows the ciphertexts and keys sizes of the FV scheme for a selection of three settings, where the claimed security is $\lambda = 80$.

Params	L	$ ciphertext $	$ sk $	$ pk $
$n = 4096$ $ q = 127$	4	127KB	63.5KB	315.5KB
$n = 8192$ $ q = 300$	10	600KB	300KB	3MB
$n = 16384$ $ q = 700$	23	2.8MB	1.4MB	31.3MB

Table 2.6: Ciphertexts and key sizes in the FV scheme

Although far above traditional public key encryption schemes, figures in Tables 2.5 and 2.6 are very reasonable, compared to all previous implementations we presented. This can be explained by the efficiency of R-LWE-based schemes, and by the precise analysis of Lepoint *et al.* However, remind that messages are bits, thus the ciphertext expansion is at least $127KB/1 \approx 13.10^3$, which is even higher than the Goldwasser-Micali scheme 30 years ago.

¹⁶Also denoted YASHE for “Yet Another Somewhat Homomorphic Encryption” scheme.

Homomorphic evaluations of block ciphers

In some works, in addition to implementing a scheme, the authors also test the performances of the scheme in a use-case. Often, the test consists in evaluating a block cipher on encrypted data. Initially, the motivation was to show that quite complex computation could be performed with FHE, and that FHE was possible. More recently, being able to evaluate block ciphers is motivated by the impossibility to send FHE encrypted data over the network, for example to the cloud. Indeed, as noted by Lepoint *et al.* [LN14], the ciphertext expansion of FHE schemes is so large that 4MB of data becomes 75TB of ciphertext once encrypted. The proposed solution for a user to be able to send encrypted data to the cloud it is to first encrypt the data using a block cipher such as AES. In a first phase, the user sends its data encrypted under a secret AES key K , along with $\text{Enc}(pk, K)$ (where pk is the user's FHE public key). The cloud can then encrypt a second time the data using the client's public key pk , and homomorphically evaluate the decryption procedure of the block cipher in order to obtain data solely encrypted under the FHE public key pk . This solution trades communication complexity against computation load, but the cloud has supposedly huge computational power.

The first evaluation of a block cipher using FHE was performed by Gentry *et al.* in 2012 [GHS12b], using the R-LWE version of [BGV12] (typically denoted BGV) to evaluate AES. This idea later inspired Coron *et al.* who performed the same exercise for integer-based scheme [CLT14]. And in 2014 Lepoint *et al.* evaluated the SIMON block cipher [BSS+13], a slightly simpler block cipher, with the same implementations of the FV and BLLN schemes as in section 2.3.6. Results are summed up in Table 2.7. The platforms used to run the tests are respectively: the BlueCrystal Phase 2 super computer from Bristol University, an Inter Xeon E5-2690 (2.9GHz), and an Intel Core i7-2600 (3.4GHz).

Note that all works first transform the block cipher into a circuit evaluable by their scheme. Also, they all use *leveled* homomorphic encryption schemes and the ciphertext batching technique. When using batching, each ciphertext contains several *slots*, where each slot can contain 1 plaintext, and homomorphic operations are performed slot-wise. Said otherwise, 1 ciphertext can contain a certain number l of plaintexts, and operations on the plaintexts inside a given ciphertext are performed in parallel in a SIMD (*Single-Instruction-Multiple-Data*) manner. The authors use this technique to speed up the evaluation of the block cipher. As a result, the relative time per AES (or SIMON) block is reasonable. However, bear in mind that the overall time to evaluate the block cipher on all the blocks is important (*e.g.* 65h with BGV), and that the evaluator only obtains the result after this long period of time (the results arrive all together in one batch). Finally, bear in mind that the claimed security of FHE schemes is approximative.

Table 2.7 shows that the time per block of recent schemes are very reasonable, if latency is not an issue. In particular, the BLLN scheme is almost efficient, for $\lambda = 80$ and the SIMON block cipher. This gives hope for practical applications in the next years, and may efficiently resolve the problem of communicating homomorphically encrypted data to the cloud (but not to regular entities). Furthermore, Table 2.7 shows the great progress made in two years in R-LWE-based LHE, and the inefficiency of integer-based schemes even in 2014.

List of known implementations

Finally, we list the existing FHE implementation in Table 2.8. Note that many are not public, most of them are only at the stage of academic tool, and none is actually secure (their code has not been audited). Also, to manipulate large number, most of them use the GMP or Flint libraries. Keep in mind that this table is not a comparison of FHE schemes implementation, as it would be vain to try and compare them. Trying to compare running times of schemes based on different concepts, with different security levels (not always well defined), and whose

Year	Cipher	Scheme	Based on	λ	# slots	Evaluation time	Time per block
2012	AES	BGV [BGV12]	R-LWE	128	54	34h	37min
2012	AES	BGV [BGV12]	R-LWE	128	720	65h	5min
2014	AES	CLT [CLT14]	Integers	72	569	3h35min	23s
2014	AES	CLT [CLT14]	Integers	80	1875	102h	3min15s
2014	SIMON	FV [FV12]	R-LWE	80	1800	51min	1.70s
2014	SIMON	FV [FV12]	R-LWE	128	2048	3h27min	6.06s
2014	SIMON	BLLN [BLLN13]	R-LWE	80	1800	17min	0.57s
2014	SIMON	BLLN [BLLN13]	R-LWE	128	2048	1h08min	2.04s

For every work, λ is the security of both the block cipher and the FHE scheme.

Table 2.7: Homomorphic evaluations of block ciphers

implementation is run on different platform would not let us learn anything we do not already know: qualitatively, R-LWE schemes perform better than LWE schemes, which in turn are better than integer-based schemes (and ideal lattices come last).

Scheme	Based on	L/FHE	Origin	Public	Remarks
vDGHV10	Integers	Fully	[vDGHV10]	No	Updates: [CMNT11, CNT12, CCK+13, CLT14]
Gen09	Ideal \mathcal{L}	Fully	[GH11b]	No	
vDGHV09	Integers	Fully	Stephen Crane	Yes	C++, Not maintained
SV11	Ideal \mathcal{L}	Fully	hcrypt [BPS12]	Yes	Encrypted VM, C++ and Java
BGV12	R-LWE	Leveled	[GHS12b]	No	Eval. AES
BGV12	R-/LWE	Leveled	[AMFF+13]	No	
BGV12	R-LWE	Leveled	Shai Halevi <i>et al.</i>	Yes	C++, Still maintained
FV12 BLLN13	R-LWE	Leveled	[LN14]	Yes	Most up-to-date

Column 2 says whether the implementation is based on ideal lattices (“Ideal \mathcal{L} ”), integers, R-LWE, LWE or both (“R-/LWE”).

Column 3 says whether the leveled or fully homomorphic version was implemented.

Table 2.8: Main FHE/LHE implementations

2.3.7 Future of FHE

Along the section, we have seen the progress made by the FHE technology. Although running times of LHE schemes are becoming reasonable, they are still above RSA 2048 by a factor 100, and the huge ciphertext expansion of LHE scheme is prohibitive.

In addition to the efficiency (and security) issues, cryptographers have identified several other problems worth considering.

- Can bootstrapping be practical, and performed with natural operations? [ASP14]
- Is there an alternative to bootstrapping to achieve *pure* FHE?

- Can FHE be as secure *and* as efficient as standard lattice cryptography?
- Is it possible to construct an Identity-based LHE/FHE scheme? The answer is theoretically, yes [GSW13].
- Is it possible to construct an Attribute-based LHE/FHE scheme? Idem [GSW13].
- Is it possible to use FHE to design functional encryption schemes, or vice-versa? [GGH⁺13]
- Is it possible to use FHE to obfuscate programs, or vice-versa? [GGH⁺13]

Finally, we conclude this section by recalling that the state-of-the-art FHE is constantly changing, and that in only 5 years, the complexity of FHE has dropped dramatically, thanks to a very active community and a great interest of cryptographers in this topic. Although not practical as of today, we can conjecture that in a few years, FHE will be practical for simple use-cases such as email filtering, and for a computationally powerful entity such as the cloud.

2.4 Discussion

In the light of the elements presented in this chapter, we now have enough knowledge to start using homomorphic encryption in new systems designs. However, there are several points we did not mention, mainly for simplicity reasons. First of all, we did not compare the security of homomorphic encryption schemes and traditional ones: indeed, as HE ciphertexts are by definition alterable, security can seem to be compromised. Also, in this last section we review the basic pros and cons of PHE and FHE, the evaluation of programs on encrypted data with FHE, advanced homomorphic properties of other types of schemes than encryption, and the link between HE and other similar technologies to compute over encrypted data.

2.4.1 Security of homomorphic encryption

In traditional public-key encryption schemes, the highest and most desirable level of security is IND-CCA2, *i.e.* *indistinguishability under adaptive chosen ciphertext attack* [Gol04]. With a simple glance at the definition of the IND-CCA2 security (with decryption oracle access before *and after* the challenge step), it is easy to see that any homomorphic scheme can not attain this level of security. Indeed, after supplying m_0 and m_1 in the challenge part, and upon receiving $\text{Enc}(pk, m_b)$ for a random b , the adversary can simply ask for the decryption of $\text{Enc}(pk, m_1 - m_b)$ (resp. $\text{Enc}(pk, m_1/m_b)$ for multiplicative schemes) as she knows the encryptions $\text{Enc}(pk, m_1)$ and $\text{Enc}(pk, m_b)$. If the result is 0 (resp. 1), the adversary knows that $m_b = m_1$, else $m_b = m_2$, and wins the game with probability 1.

Therefore, because of the *malleable* nature of HE schemes (*i.e.* the possibility to apply operations on ciphertexts that imply meaningful transformations on underlying messages), in general, their maximum security level is IND-CPA [FG07]. Some schemes were proven IND-CCA1 secure [BSW12], but in the general case it is unclear. Actually, to the best of our knowledge, this question have not been studied yet. We can say however, that bootstrappable FHE schemes are never IND-CCA1 secure: as the encryption of the secret key is public, the adversary only needs to ask for its decryption to break the scheme.

This (in)security is not a point in favor of homomorphic encryption: using a HE scheme means that ciphertexts and their messages can be modified publicly by any entity, creating an open door for various attacks. However, note that this malleability is precisely what is sought in homomorphic encryption, and the loss of security and the homomorphic properties are bound:

we can not strictly have both IND-CCA2 security and homomorphism. Thus, in practice, one may need a second layer of traditional encryption to communicate ciphertexts over untrusted channels, or to authenticate the messages.

There are however recent works that try to reconcile homomorphism and security. For instance, Canneti *et al.* [CKN03] relax and modify the IND-CCA2 security game so as to forbid decryption oracle queries with ciphertexts that are related to the challenge messages m_1 and m_2 . This yields a new security definition denoted “Replayable CCA” (RCCA). In a similar way, An *et al.* [ADR02] define gCCA security for “benignly-malleable security”, where the decryption oracle do not answer to queries on ciphertexts satisfying a particular binary relation with the challenge ciphertext. These two notions were generalized by Prabhakaran *et al.* [PR08], with the notion of “homomorphic CCA” (HCCA) security. In their definition, only some operations on the ciphertexts are allowed, and the oracle does not answer queries on ciphertexts obtained from non-authorized operations. However these result only modify the security definition, but do not provide more security, *i.e.* homomorphic schemes’ security stays the same, but security definitions change so as to match their reality. Indeed, in gCCA and RCCA security, the “attack” described above where the adversary asks for the decryption of $\text{Enc}(pk, m_1 - m_b)$ is still possible, and HCCA security does not guarantee any higher security neither.

In order to actually reconcile homomorphism and IND-CCA2 security in the same scheme, Emura *et al.* [EHO⁺13] proposed the notion of *Keyed-Homomorphic Public-Key* encryption, where the knowledge of a *secret evaluation key* sk_{ev} allows to modify ciphertexts (but not to decrypt them). Without this key the scheme is IND-CCA2 secure, and when it is known, the scheme is only IND-CCA1 secure. With this definition, CCA2 security is guaranteed for all parties without knowledge of sk_{ev} , while homomorphism is still possible for privileged parties. This construction does not give rise to a strictly speaking IND-CCA2 secure homomorphic scheme, but it is as close as we can get, considering the opposition between the two notions. A realisation of a threshold keyed homomorphic encryption scheme has been proposed by Libert *et al.* [LPJY14], the first of this kind according to the authors.

Broken HE schemes

We conclude this paragraph on HE schemes security by a brief overview of broken HE schemes. Indeed, several attempts for efficient PHE or SHE schemes were proposed and dismantled shortly after their release. In particular, two PHE schemes by Domingo-Ferrer [DF96, DF02] were broken respectively by Cheon *et al.* in 2006 [CKN06] and Wagner *et al.* [Wag03], and a 2006 PHE scheme by Grigoriev and Ponomarenko [GP06] was broken the same year [CBW07]. By “broken” we mean that the attacks against the schemes showed that the claimed IND-CPA security was actually flawed. These attacks do not seem to use the homomorphic properties of the schemes, *i.e.* their cryptanalysis do not rely on their homomorphism, and therefore are of little interest to us.

There is however, to our knowledge, one cryptanalysis using the homomorphic properties of the scheme it attacks. Indeed, Brakerski [Bra13] showed that the powerful SHE scheme from Bogdanov and Lee [BL11] was insecure *due to its homomorphic capacities*. Even better, the author proved that homomorphism “becomes a liability” when the decryption function is weakly-learnable. A weakly learnable function is, very informally, a function whose output can be guessed with non-negligible but small probability, given many input-output samples. In the case of Bogdanov-Lee, the decryption function is a noisy scalar product. What Brakerski shows is that, if a scheme is “homomorphic enough” (in this case, if it can evaluate the majority function), and if its decryption function is “weakly learnable”, then it is insecure. Using the majority function, Brakerski transforms ciphertexts with noise, say, $1/10$ to ciphertexts with noise $1/(10n)$ for some

$n > 1$, thus yielding a collection of low-noise ciphertexts, making the decryption function easier to learn, to a point that the secret key can be uncovered. The Bogdanov-Lee scheme was based on the “*Learning parity with noise*” problem, very close to the LWE problem but for elements in $\{0, 1\}$, it was quite efficient, allowed several additions and multiplications and was considered as a FHE candidate. But due to this cryptanalysis, it was not retained.

2.4.2 Partially or Fully homomorphic encryption ?

There are several differences between PHE and FHE schemes. Firstly, the formers are almost as efficient as traditional public-key encryption (PKE) schemes but allow very restricted computations, while the others are computationally very expensive (although a lot of progress were made) but are theoretically extremely powerful. As for their security, they both claim to be IND-CPA, but FHE/LHE schemes work on less standard and less studied assumptions. This implies a possibly lower security for FHE, but ongoing studies on FHE schemes’ assumptions may re-establish equivalent security between PHE and FHE. Finally, we can say that, in general, PHE schemes provide algebraic homomorphic operations, such as natural addition over the integers, while FHE/LHE schemes are meant to encrypt and manipulate bits. This implies in particular that programs to be evaluated on encrypted data must before be transformed into a circuit (on this point, see paragraph 2.4.3).

As guidelines, we can say that the choice of using a PHE or FHE/LHE scheme completely depend on the use-case and the expressed needs. Indeed, before choosing a scheme, one should always clearly state what information needs to be kept secret from which party, and what are the operations that must be performed in the encrypted domain. Then, if there exist a PHE scheme powerful enough to perform these operations, use it preferably over FHE (for instance, addition and scalar multiplication are enough for several secure multi-party protocols). FHE/LHE schemes should be considered only if there is no other possibility. Typically, the FHE/LHE machinery is used only in very complex systems or delicate open problems, to exhibit a theoretical possibility result (a *proof-of-concept*).

Also, it is widely recommended to use LHE rather than FHE, at least up to a certain multiplicative depth. In many occasions, only a few multiplications are needed, and for $L = 1, 2$ or 3 , R-LWE-based schemes are relatively efficient when used on computationally powerful platforms.

2.4.3 Evaluating a program on encrypted data: in practice

FHE schemes can theoretically compute arbitrary functions on encrypted data. However, we did not detail the procedure to do so. Indeed, it is not possible to simply take a program in C and run it on encrypted data.

Suppose an evaluator has a RAM program¹⁷ $P(\cdot)$ that she wants to evaluate on input x in the encrypted domain. First, as FHE schemes work with bit messages, x needs to be encrypted bit-wise. We denote $\tilde{x} = \{\text{Enc}(pk, x[i])\}_i$ this collection of encryptions. Then, the evaluator can not give \tilde{x} as input to $P(\cdot)$: the program needs to be transformed into a boolean circuit. Moreover, in this circuit, gates must be replaced by their corresponding operations on ciphertexts (typically, addition and multiplication of ciphertexts). At the end, the circuit may output several encrypted bit, which need to be decrypted separately and re-assembled to give the final result.

The transformation of $P(\cdot)$ from a RAM program into a circuit implies a high overhead in terms of time and space, in addition to the overhead caused by homomorphic gate evaluations. Indeed, a RAM program running in time T yields a circuit of size (*i.e.* number of

¹⁷A program running on a *random-access machine*, the natural computation model for computer programs.

gates) $O(T^3 \log T)$ [GHRW14]. The intuition of this result is explained quite simply by Vaikuntanathan [Vai12]: imagine that given an encryption of an integer $n > 0$, we are asked to execute the RAM program “`while(n < 106) n++;`”. As we do not have access to the actual value of n , the only way known as of today is to flatten the program, *i.e.* to create 10^6 hard coded conditional statements. Where the RAM program takes 10^6 rounds in the worst case, the equivalent circuit will *always* need 10^6 rounds. In particular, the RAM program stops directly if $n = 10^6 + 1$, but the circuit needs go through 10^6 rounds still: the circuit represents the *worst case* execution of the program.

It is considered an open problem to design *input-specific* (as opposed to *worst-case*) program evaluation on encrypted data. With an input-specific procedure, the circuit from the above example would not always perform 10^6 rounds, and its average complexity would be lower. However, input-specific evaluation poses another threat: if the evaluation of the above program on encrypted data terminates early, this means n was slightly under 10^6 (or above it), and if it takes a long time to terminate, this means n was closer to 0. Therefore information on n is leaked, although, this value was precisely the one we wanted to keep secret in the first place (that is why we encrypted it). The source of difficulty in the program-circuit transformation is the presence of input-dependent loops or conditional statements in programs: the evaluation is highly inefficient, and/or information is leaked on the sensible data.

To mitigate the results from above and avoid circuit transformation, *Garbled RAM* can be used [LO14, GHRW14]. The main idea is to get rid of the costly circuit transformation step and directly evaluate RAM programs on encrypted (or “garbled”) data. However, Garbled RAM constructions are not meant to work with FHE, and the link between the two technologies is still unclear.

2.4.4 Advanced notions: other homomorphic constructions

This chapter presented the notion of homomorphic *encryption*, and encryption only. However, other cryptographic constructions can be said homomorphic.

Homomorphic Signatures Johnson *et al.* [JMSW02] put forward the notion of homomorphic *signature*, the analog of homomorphic encryption for signature schemes. A signature scheme is said homomorphic if it allows operations on signed messages, in the sense that the modified signature is a valid signature of the modified message. It is not trivial to understand why one would want such a property on signatures, which are precisely meant to attest the integrity and authenticity of a message. Initially motivated by applications to network coding, they are actually very useful to enable verifiable computation mechanisms. Verifiable computation is employed when some computation is delegated to an untrusted third party: we give the input and a homomorphic signature on it, the third party modifies the data and the signature accordingly, so that when the third party outputs the result, the authenticity of the data can still be universally verifiable.

Of course, a homomorphic signature scheme is not secure against existential forgeries (EUF-CMA), and a new security definition is required for these construction. Basically, a homomorphic signature scheme is secure with respect to the operations it can compute if an adversary can not forge signatures for messages independent from the message she previously queried to the signing oracle. In the seminal paper of Johnson *et al.* [JMSW02], the authors construct 3 signature schemes, respectively homomorphic for: set operations, integer addition, and for sanitization. Interestingly, Johnson *et al.* prove that an additive homomorphic signature scheme can not be secure.

Several constructions have then been designed, and in particular, Libert *et al.* recently proposed *structure preserving*¹⁸ homomorphic signatures [LPJY13]. There are other types of signatures supporting alterations, which we did not investigate: sanitizable signatures [ACMT05, CJL12], and content extraction signatures [SBZ02].

Malleable proofs Along the same lines, there exist proofs and commitment systems allowing modifications on what is proved/committed [AFG⁺10, CKLM12]. Motivations for these construction are more or less the same as homomorphic signatures, *i.e.* mainly, to enable verifiable computation with short and non-interactive proofs. Chase *et al.* [CKLM12] put forward the notion of *controlled* malleability for proofs, which refers to the limitation of the possible modifications third parties are allowed to perform. The authors also note that Groth-Sahai proofs are malleable, and show how simulation sound extractability and controlled malleability properties can be both satisfied in a same construction.

Targeted Malleability Using malleable proofs, Chase *et al.* [CKLM12] then propose the construction of *controlled malleable encryption* schemes: thanks to the controlled malleability property of their proofs, they actually attain the HCCA security level put forward by Prabhakaran *et al.* (and presented in section 2.4.1). This type of encryption has been formalized later by Boneh *et al.* [BSW12] and given the name of *targeted malleability*. Whereas Chase *et al.* use a IND-CPA secure encryption scheme along with malleable proofs, Boneh *et al.* use homomorphic schemes along with standard proofs. The main challenge in their construction is to avoid proofs size blowup. Indeed, the naive solution for targeted malleability is to ask to the entity modifying the ciphertexts to output a proof for each computation she performed that the new ciphertext was obtained applying an authorized function on the original one. However, this solution makes the proofs size grow at least linearly with the number of homomorphic operations. Here, the authors require the proof growth to be sub-linear. They achieve this goal using a *double encryption* paradigm (the same message is encrypted twice, under two different keys), and *succinct* (non-interactive) arguments that have the property to be smaller than their corresponding witness by a constant factor (say, 1/4). Then again, the notion of targeted malleability is useful in the setting of computation delegation, in order to ensure that only a certain set of allowed functions are computed on encrypted data.

Multi-key and threshold (F)HE Another interesting feature of homomorphic schemes are their threshold capacity. For instance, in PHE, the Damgård-Jurik scheme is an adaptation of Paillier to the threshold setting, and in FHE the BGV scheme has been similarly adapted by Asharov *et al.* [AJLA⁺12]. In a threshold encryption scheme, basically, several parties hold 1 secret key each but share the same public key: encryptions with the public can only be decrypted using all (or a subset of) the secret keys. In the FHE case, the same authors also note another interesting properties of (R-)LWE-based schemes: they are naturally *key-homomorphic*, *i.e.* addition of two key pairs yields a valid key pair. Threshold FHE allows server-aided multi-party computation: several parties provide inputs encrypted under the same public key to the server, who performs the computation and send back the output to all parties, who in turn cooperate together in a small decryption protocol using their respective secret keys. However, in this setting, the users have to interact to generate the public key and the secret keys, *i.e.* they have to be *online* before the computation. To thwart this issue, López-Alt *et al.* [LATV12] go

¹⁸Structure preserving signatures have the particularity that messages, signatures and public keys are all bilinear group elements [AFG⁺10], which makes them easy to compose with other cryptographic tools such as Groth-Sahai proofs.

even further and propose the notion of *multi-key* FHE: based on NTRU (a LWE-based scheme that can be turned into a FHE scheme), they construct a scheme capable of evaluating operations on data encrypted under several, totally unrelated keys. They also provide a generic construction to transform other FHE schemes into a multi-key FHE one. Unfortunately, in their solution the clients involved in the computation still have to cooperate at the decryption phase in order to get the result of the server’s computation. However, as it has been proven that completely *off-line* server-aided multi-party computation using *only* FHE is impossible [VDJ10], this solution is so far optimal in terms of communication and cooperation between parties. The threshold or multi-key properties may prove useful, as in many settings we need to “mix” data encrypted under different keys.

2.4.5 Link with Functional Encryption, Garbled Circuits, MPC and Searchable Encryption

Homomorphic cryptography links to many other cryptographic constructions that we did not mention in this document. We briefly present a selection of notions, explicit their link (or differences) with homomorphic cryptography, and point references for the interested reader.

Secure Multi-party Computation Secure multi-party computation (SMPC) [Yao82] is a setting where n parties want to compute a function f on the collection of their private inputs x_i . The constraint is that each party keeps its input x_i secret from all other parties during the protocol. In the end, all (or a subset) of parties should learn $f(x_1, \dots, x_n)$. Variants of SMPC also exist: in Verifiable SMPC all parties should be able to check that the computation performed was indeed the agreed function f , which is useful when only 1 party (the *evaluator*) performed the computation ; in Non-interactive SMPC, each party should send only 1 message in total during the protocol ; and in Private SMPC, it is required that f stays private to the evaluator (*i.e.* it can be considered as its private input). Verifiable SMPC is achievable thanks to zero-knowledge proofs for instance, and Private SMPC can use Garbled Circuits (see below).

Like HE, SMPC allows to compute on “hidden” inputs. Also, it is interesting to note that one of the (many) ways to achieve SMPC is to use FHE or PHE. Actually, the works of López-Alt *et al.* [LATV12] and Asharov *et al.* [AJLA⁺12] presented in section 2.4.4 are mainly focused on FHE-based SMPC. As for PHE schemes, there are many multi-party computation protocols that can be found in the literature [Rap06].

Garbled Circuits Garbled circuits (GC) are the first practical tools invented to compute on “encrypted data”. Proposed by Yao [Yao86]¹⁹, the main idea behind GCs was to enable secure two-party computations with function privacy. Yao proposed a two-party protocol between a *garbler* who creates the garbled circuit and encodes its inputs, and an *evaluator* who evaluates the garbled circuit on the garbler’s and her own inputs. The protocol ensures that each party keeps her input private. To garble a given circuit, the idea is to replace each of its gates with a garbled version of it, such that the truth table of the gate is preserved but unintelligible. The garbled circuit can then be evaluated by any party, but this party can not guess what she is actually evaluating. A garbled circuits, in Yao’s original construction, is usable only one time. Indeed, if the evaluator could evaluate the garbled circuit more than once on different inputs, she would learn information on the inner structure of the circuit, and in the worst case, completely un-garble it. Recent works [GKP⁺13] show how to construct secure *re-usable* garbled circuits (but the solution is quite inefficient).

¹⁹Yao actually first presented GC in an oral presentation associated to the paper [Yao86]. A writing description can be found in [Gol03, Section 5.3.2].

Although GC are inefficient (especially because of the circuit computational model), they are still studied today and considered as a possible candidate for computing over encrypted data. An interesting work from Kolesnikov [KSS09] show how to combine HE and garbled circuits to produce two-party computation protocols with better efficiency.

Garbled RAM As discussed earlier in this section, executing *random access machines* programs is much more efficient than transforming them into circuit and evaluating them. Consequently, a (relatively) new field or research aims to directly garble RAM programs [LO13, GHRW14, LO14]. As defined by Lu and Ostrovsky [LO13], garbled RAM consists in an alternative to securely evaluate programs. The idea is, instead of always evaluating *all* the paths of a program, *i.e.* instead of evaluating the worst-case execution of the program flattened into a circuit, to *prune off* dead paths in the execution tree. In other words, only the relevant path in the program execution are evaluated. This yields a great efficiency improvement when the program accepts large inputs. The main challenge is then to keep the path and the memory access patterns secret. To hide the access pattern, garbled RAMs use the Oblivious RAM computation model of Goldreich *et al.* [GO96], that exactly fulfills this goal. To hide the execution path, the authors actually use one garbled circuit for each atomic step of the execution. The trick is that, the garbled circuit at step i actually dynamically generates the garbled circuit for the step $i + 1$. As garbled circuits, garbled RAMs are meant to be used in a protocol between an evaluator and a garbler.

Garbled RAMs are still at early an development stage, and are quite inefficient for now. The authors in [GHRW14] claim a running time in $t \cdot \text{poly}(\lambda) \cdot \text{polylog}(n) \cdot |C_{CPU}|$, where t is the original running time of the program, λ is the security parameter, n is the size of the input, and $|C_{CPU}|$ is the size of the step-circuit. In other words, the overhead of garbled RAMs is polylogarithmic. Also re-usable garbled RAMs have not been investigated yet, but are an open problem. In definitive, garbled RAMs are still theoretical object, but may become a viable candidate for computing on encrypted data without having to flatten the whole program into a circuit.

Functional Encryption Functional encryption (FE) may be the best concurrent to FHE. As formalized by Boneh *et al.* [BSW11], functional encryption schemes comprise special secret keys that allow evaluation of a given function and decryption *at the same time*. A FE scheme is composed of the 4 traditional primitives, but slightly modified: **Setup** outputs the public key and a master secret key msk , **KeyGen** takes as input a function f and msk and outputs a secret key sk_f , **Enc** works as usual, but **Dec**, given sk_f and a ciphertext $\text{Enc}(pk, m)$, outputs $f(m)$. The function f robustly embedded in the secret key sk_f is applied *simultaneously* with the decryption procedure. The more functions f the scheme is able to evaluate, the more powerful it is. The security of such scheme is not trivial. Note that FE is a generalization of several encryption paradigm (including standard encryption). It can lead to attribute-based encryption (ABE), where the class of function is restricted to predicates on a user's attribute (*i.e.* **Dec** outputs m only if $P(\text{attributes}) = \text{true}$ for some predicate P , else \perp). In ABE, the secret key of the user contains its attributes and the predicate is fixed, or, alternatively, the secret key contains the predicate definition, and the attributes are embedded in the ciphertext. FE is also an abstraction of identity-based encryption (IBE) where the predicates are further restricted to identity checking (*i.e.* **Dec** outputs m only if $id_decryptor = ID$ for some ID , else \perp). In this case, the secret key simply contains its owner's identity, and the authorized decryptor's identity is contained in the ciphertext.

It is easy to see the resemblance between FE and HE. However, there are several important conceptual differences: evaluation on encrypted data with HE is made publicly using only the scheme's public key while in FE special secret keys must be delivered by the authority holding

msk. Also, evaluation with FE outputs plaintexts whereas after an FHE evaluation, one has to ask the owner of the secret key for decryption. Another difference is that (F)HE is actually more advanced than FE: it is more powerful and more efficient (because designing a secure FE scheme with is also efficient and somewhat powerful is very hard).

Proposed constructions of FE schemes [BRS13a, BRS13b] are very limited in the set of functions f it allows, and are quite inefficient. But it is interesting to note that FE for *any* function f can be enabled using FHE, at least in theory [GGH⁺13]. The idea is to (1) evaluate the function f with FHE, and (2) to input the result and non-interactive proofs that the computed function was indeed f to a secured functionality (*e.g.* a tamper-proof secure hardware token [DMMQN11]) so that (3) it checks the proofs and securely executes the FHE decryption function (the secret key can be hard-coded into the secure functionality). An alternative to the hardware token method is using obfuscation (see last paragraph of this section). Both constructions are very far from practical, and consist in a first step toward powerful FE.

Searchable Encryption Searchable encryption (SE) can yet be seen as another flavor of FE, where the functions f are of the form “Is the word w present in the document?”. However, it has a literature of its own thus we present it apart from FE. The basic, most general setting in SE is as follows. A database owner \mathcal{D} needs to delegate storage of her database to a computationally powerful server \mathcal{E} (*e.g.* the cloud). We suppose that the database contains sensible data, thus it is encrypted before it is handed to \mathcal{E} . Then, some clients \mathcal{C} , upon authorization from \mathcal{D} , can perform requests to the server on the database, with the constraint that \mathcal{E} does not learn the content of the queries but answers it appropriately, and without \mathcal{C} learning anything more than the information requested. This setting is the one described by Jarecki *et al.* [JJK⁺13], but then, many settings can be derived from this one: for instance, \mathcal{E} can be a mail server, \mathcal{D} the receiver of some mails, and the goal is for \mathcal{E} to test (non-interactively) tags on emails so as to forward them in the proper folder (*e.g.* it can act as a spam filter). In this sense \mathcal{E} will also act as client and use SE to filter mails. This example setting is the one put forward in [BC04].

We distinguish two kinds of SE schemes: public key (asymmetric) or private-key (symmetric). In the public key setting, anyone can publish data in the database (so there is no actual owner \mathcal{D}). Generally speaking symmetric SE is more efficient thanks to the efficiency of symmetric over asymmetric cryptography. Also, we can differentiate single-keyword SE, general boolean queries SE, and fully-qualified queries SE: the former allows the search for 1 keyword at a time, boolean queries SE allows queries consisting in any boolean formula, and the latter allows any SQL-like queries. Recently, two works proposed practical symmetric SE schemes for arbitrary boolean queries on relational databases or free text. The first [CJJ⁺13] mainly provide a protocol for conjunction queries, which is extensible to general boolean queries, and works for 1 client only, which is actually the owner of the database. The other [JJK⁺13] extends this work in the multi-client setting, where the database owner can delegate querying rights to other clients. The authors present their work as an *outsourced symmetric private information retrieval* protocol. All their constructions are practical, and the complexity of the scheme is substantially independent of the size of the database. A protocol supporting general SQL queries can be found in [PRZB11], but is not as efficient.

Misc. As we can see all the presented constructions (SMPC, GC, GRAM, FE, SE) are ways to “compute on encrypted data”. This notion can be extended to comprise even more constructions, such as obfuscation. Actually the notions of FE, FHE, garbling and obfuscation are closely related. In particular, an obfuscator for any program can be constructed using FHE plus an obfuscator for NC^1 circuits. And the resulting obfuscator in turn leads to functional encryption for any function f [GGH⁺13].

Chapter conclusion As the first chapter provided a detailed view of numerical privacy, this chapter consists in a survey of homomorphic encryption. The first section is a high level historical and technical overview of homomorphic encryption, while the following sections detail partially and fully homomorphic schemes, list them, and compare them when possible. We expect to only *use* homomorphic cryptography, and not to design new schemes, but argue that a deep understanding of this tool is necessary. Indeed, it ensures we use them in proper the way, *e.g.* it avoids degrading security by using them in context they are not supposed to. It also permits to take into account the efficiency-security-homomorphism trade-off and to be able to *compose* HE with other cryptographic primitives. At last, if some minor modifications to some scheme are necessary, for example to adapt it to some setting, a good understanding of the scheme allows to easily conclude if and how it is possible.

The field of (fully) homomorphic cryptography is thriving, and the state-of-the-art schemes are constantly changing. Also, as a general trend, we can see that the paradigm of homomorphic cryptography is being extended to other schemes than encryption, and this progression should continue during the next years. Generally speaking, the “computing on encrypted data” idea has too many applications in society for its development to stop suddenly.

Because our main goal in this document is to protect privacy, and because we assume that for this end, homomorphic *encryption* is the main necessary tool, we only briefly exposed other homomorphic constructions. Indeed, homomorphic signatures and proofs are mainly useful to handle malicious parties: they come on top of a privacy solution, to re-enforce its security and ensure it is well applied.

Conclusion and perspectives

This document presented in details privacy in computer science on one hand, and homomorphic cryptography on the other hand. This succinct concluding chapter gives insight on how to combine both, *i.e.* how to use homomorphic cryptography for the protection of privacy.

What can homomorphic cryptography do for privacy?

Our main goal, as stated in introduction, is to design privacy-enhancing systems using homomorphic cryptography and similar technologies. The natural question we need to tackle is “what can HE do to enable/enhance privacy in information systems?”. We give two different answers, a first naive one, and a less trivial one.

A first naive answer As noted in section 1.3, almost any information system can benefit privacy properties, although some systems are more pertinent than others, *i.e.* although it makes more sense to protect privacy in some systems than others. On the other hand, we have at our disposal the power of (fully) homomorphic encryption, that can theoretically evaluate any processing on data without actually accessing it.

Therefore, a first answer to the above question is “*everything*”: FHE can resolve every privacy problem. Indeed, intuitively and for any system, we can imagine encrypting all sensible data with a FHE scheme, and for any necessary processing by any party, this party uses the homomorphic properties of the scheme to produce the desired encrypted output. To decrypt, 1 round of communication with the owner of the data (which is also the owner of the secret key) is then necessary. By this mean, the owner of the data can also verify what computation were performed using verifiable computation techniques, and what output the evaluator is going to learn.

Actually, this fact has already been put forward by Barak and Brakerski in a couple of blog posts [BB12b, BB12a], in a slightly more formal manner. They say of FHE that it is the “swiss-army knife of cryptography”, a tool which resolves many cryptographic problems. As examples, they detail how to design two-party SMPC and zero-knowledge proofs systems from FHE. But there are many more applications, such as electronic voting, social networks, program obfuscation, ... They argue that FHE provides a *unified* and generic solution to many problems, but are aware that for each specific system, there exist more secure and efficient solutions tailored for it.

A more cautious answer As second, more moderate, non trivial and efficient answer to our question is possible. It consists in saying that a solution must be designed case by case in a tailored and “smart” way for each problem we encounter. It is the favored answer in the community, and actually, in the literature it is possible to find many use of PHE or FHE schemes

for specific applications. We detail some of these applications in the following section, where we make the difference between PHE and FHE applications.

HE-based privacy in the literature

In light of our second answer, we give several existing solution of homomorphic encryption based privacy. We will see that this approach is far not recent, and many systems have been considered already.

Using PHE schemes As the first PHE scheme appeared more than 30 years ago, its applications are many. Though not all are privacy-oriented, a non-negligible part is. The main, established application is secure electronic voting, where the users' vote must absolutely stay private. Generally speaking, constructing on Yao's concept of multi-party computation [Yao82], many works are devoted to designing efficient and secure multi-party protocols for various purposes such as integer comparison (known as the millionaires problem), PIR, Oblivious Transfer, Commitment, (Non-Interactive) Zero-knowledge Proofs, electronic auctions or secret sharing schemes. There are even proposition for system as complex as media fingerprinting for traitor tracing [PW97]. Many references of privacy-preserving PHE applications can be found in Rappe's thesis [Rap06].

To provide a minimum insight on how constructions with PHE work, we detail simple protocols: one for electronic voting, and one MIX-net protocol (see section 1.3.3), both extracted from [DJ03], which propose yet another extension of Paillier, building on the work of Damård-Jurik [DJ01]. Their scheme is a threshold variant of Paillier: for a given public key, the secret key can be split into several parts so that all owners of the parts must collaborate to decrypt a ciphertext. Alternatively, the public key can also be split in several parts. The authors also design simple ways to prove validity of encryption, *i.e.* to prove that a given ciphertext actually encrypts a valid message. We simplify the protocols in order to put in light the use of homomorphic properties.

Electronic voting using Threshold Paillier

Setting The scheme is designed for a set of voters $V = \{1, \dots, n\}$, 1 candidate and "yes/no" votes. It can be used l times for l candidates. No need for trusted authority.

Protocol

1. Each voter $i \in V$ publishes (*i.e.* broadcasts) n values $c_{ij} = \text{Enc}(pk_j, s_{ij})$ for $j \in [1, n]$, where pk_j is voter j 's public key and such that $\sum_{j \in V} s_{ij} = 0$.

2. Each voter $j \in V$ retrieves encryptions c_{ij} for all $i \in V$ and computes

$$c_j = \prod_{i \in V} c_{ij} = \text{Enc}(pk_j, \sum_{i \in V} s_{ij})$$

3. Each voter $j \in V$ computes $t_j = \text{Dec}(sk_j, c_j)$ (t_j is a "random looking" value).

4. Each voter $j \in V$ submits his vote $v_j \in \{0, 1\}$ by publishing $x_j = v_j + t_j$.

5. All voters can compute and agree on the result of the vote:

$$result = \sum_{j \in V} x_j = \sum_{j \in V} v_j + \sum_{j \in V} t_j = \sum_{j \in V} v_j + \sum_{i \in V} \sum_{j \in V} s_{ij} = \sum_{j \in V} v_j$$

Properties of the protocol Decentralized, dispute-free (honest voters all agree on the result), perfect ballot secrecy (even a majority of corrupted voters can not learn honest voters' votes).

Handling cheaters The complete protocol handles cheaters using several proofs of valid encryptions and NI-ZK proofs at step 1 and 4, and by discarding invalid values.

MIX-net using Threshold Paillier

This MIX-net protocol is not network oriented, and merely focuses on hiding the sender of a message. In other words, there is no routing procedure, and the model supposes a “bulletin board” where all users and server can read/write values publicly.

Setting Any number n of users and w MIX servers. Need for a trusted trust party at setup.

- Protocol*
1. The trusted third party generates (pk_i, sk_i) for each server i and securely distributes them. It also publishes on the board the public parameters of the scheme, g and N , and the product of the public keys, resulting in 1 valid Paillier public key, noted $pk = h = g^{sk_1 + \dots + sk_w}$.
 2. To send a message m a user posts its encryption $c = \text{Enc}(pk, m)$ on the board.
 3. Iteratively, each MIX server (one at a time) re-randomize the encryption on the board and re-writes the result. For example in basic Paillier, re-randomisation of a given $c = g^m r^N \pmod{N^2}$ is $c' = c \cdot r'^N \pmod{N^2}$ for a random r' . Note that the link between c and c' is hidden because each MIX server always processes a *batch* of ciphertexts, all of which are of the same size.
 4. When all MIXes finished the iterative re-randomization phase on c , they proceed to a threshold decryption protocol, and write the output on the board. As a result, the sender of a message can not be found.

Properties of the protocol Universally verifiability of MIXes outputs, strong correctness even against $(w - 1)/2$ malicious MIXes and $n - 2$ malicious users, order flexible (order of MIXes at step 3 is not important). Ciphertext and public key sizes are do not depend on the number of servers and the protocol is provably secure.

Handling malicious servers or users Proofs of valid encryptions and NI-ZK proofs are used at several steps

Using FHE schemes Although fully homomorphic encryption is much more recent and still in development, there are already many applications envisioned by cryptographers. Considering our second answer from the previous section, where we require solutions to be (relatively) efficient and tailored for the system they apply to, we will not use FHE as a simple magic tool for all problems. Instead, we will only use the heavy and powerful machinery of FHE for very complex systems or for hard open problems in computer science. Indeed, there are some systems or problems at the cutting edge of computer research, where finding even a theoretical solution is very complex. In such a setting, we can consider using FHE.

As with PHE schemes, the literature already contains privacy solutions using FHE, which roughly correspond to the ones presented in section 2.4: functional encryption for all functions [GKP+13, GGH+13], program obfuscation [GKP+13, GGH+13], private machine learning [GLN13], re-usable garbled circuits [GKP+13]...

There is also a setting we mentioned several times in this document where using FHE is relevant: storage and computation delegation in the cloud. FHE is indeed suited, because the

cloud is supposed to have a very large computing power, and because there is (as of today) no other known way of delegating computation to a third party without giving up access to the underlying data. Even garbled circuits and RAMs presented in section 2.4.5 are less pertinent, as their efficiency is comparable to FHE or worse, and most of those constructions yields a one-time executable program. Furthermore, garbled circuits/RAMs hide the computed function, which is not the main goal in this setting.

Note that using FHE in the cloud is not as simple as just encrypting and sending data: there are several difficulties, when looking into it. Indeed, if we consider the most general case, the requirements are quite challenging. Ideally, we would like every cloud client to simply encrypt her data with her public key, send it to the cloud along with a description of the functions it is allowed to compute on it, and then be able to adaptively ask for computations involving her data and possibly data from other users. Additionally, if some user u_1 asks for a computation involving data from herself, u_5, u_8 and u_{12} for instance, we would like that neither u_5, u_8 nor u_{12} need to be *online* (*i.e.* connected to the cloud) at this moment. As a simple use-case, social networks profile similarity computations are included in this setting. Now, in this quite simple description, there are already several issues. First, the basic step of encrypting and sending data to the cloud using a FHE scheme is as of today impracticable: a few megabytes of cleartext yields several terabytes of encrypted data [LN14]. The solution proposed to overcome this issue is to encrypt the data using a symmetric block cipher, encrypt the symmetric key with the FHE scheme, and send that to the cloud. As a result, the necessary bandwidth is dramatically reduced, and the server can obtain the data encrypted under the FHE scheme by re-encrypting what it received from the client with the FHE public key, and then *homomorphically* evaluating the decryption procedure of the block cipher, using the encrypted symmetric key. But the most serious limitation have been pointed out by Van Dijk *et al.* [VDJ10]: the authors actually prove that if the protocol described above is possible, then virtual black box obfuscation is possible. Yet, this is in contradiction with the impossibility result from Barak *et al.* [BGI⁺12] stating that this kind of obfuscation is, in general, impossible. Solutions to overcome this issue were already described in section 2.4.4: the idea is to require a minimum of interaction between the users, *i.e.* between u_1, u_5, u_8 and u_{12} in our example [AJLA⁺12, LATV12].

What’s left to do?

We have seen that there are many applications of HE to protect privacy in the literature. Yet, our goals is actually yo use HE for privacy. Therefore, it is reasonable (and necessary?) to ask “what more can be done in this field of research?”. Indeed, with a glance at the existing works, it may seem like there is “no room left” for other privacy solutions using homomorphic encryption. However, we believe it is not the case, may it be simply because FHE is a very recent technology and no practical privacy solution has ever been proposed yet. We go even further, arguing that the full potential of FHE may not have been discovered yet. And generally speaking, new privacy issues keep appearing every year, providing matter for new works in this field.

We give three general directions to design HE-based privacy solutions.

Direction 1: proceed « traditionally » The first, natural one is to proceed as presented in the previous section, and by following the procedure described in section 1.2.3: Chose one or several existing information system(s), study them, and use homomorphic cryptography to ensure privacy.

The main difficulty in this direction is to clear out pertinent and relevant systems where HE is (or at least *seems*) the suited tool to ensure privacy. In other words, the difficulty is to show

that HE is a or *the* good solution, possibly compared to state-of-the-art privacy solutions, for the chosen system(s). Indeed, as said earlier, HE and specially FHE are generic tools that can be used for privacy, but they produce generic and sometimes inefficient solutions. The chosen system(s) should fit well with HE, in the sense that a non-trivial and efficient way of using HE should be possible.

The envisioned systems are the following:

Computation delegation to the cloud As explained earlier, this setting is the main use-case for FHE.

Private network communications The literature in *censorship* [Win14], *i.e.* in strongly private anonymous communications, is thriving. In particular, the efficient Tor [DMS04] protocol is very popular. However, its efficiency comes at the cost of degraded privacy, and we believe that stronger privacy properties such as *deniability*, which makes it nearly impossible to even know if a user uses a privacy-preserving protocol or not, can and need to be ensured in this kind of system.

Authentication protocols Authentication and authenticated key-exchange protocols are the building blocks of many cryptographic protocol. Achieving authentication and privacy at the same time is very challenging, but may prove useful in many contexts such as seller-buyer interactions on Internet. Recently, a milestone was reached by Gambs *et al.* by achieving prover anonymity, deniability and security at the same time [GOR14].

Direction 2: Improving controlled homomorphic cryptography A second option is to improve the nascent field of controlled (or targeted) malleability for encryption, signature and/or proof schemes. This direction is very cryptographic-oriented and requires the knowledge and ability to compose complex cryptographic primitives while preserving the security proofs. The finality would be to create a generic (but non trivial) solution for privacy using controlled FHE: instead of “simply” encrypting the sensible data and allowing anyone to perform any computation on the data, one would precisely control what is done with it. By covering the gap between security and homomorphism, we believe homomorphic cryptography would become more easily accepted as privacy-preserving solution.

The basic idea would be to continue the works of Boneh *et al.* presented in section 2.4.4, while improving efficiency and ciphertext size growth. Ideally, we can imagine a controlled malleability system directly embedded into the gears of homomorphic schemes. That is to say, a scheme would, by itself and without need for proofs on the side, inherently limit the possible operations on ciphertexts. Also, it would be interesting to combine the keyed-homomorphism property from Emura *et al.* [EHO⁺13] with controlled malleability, so as to fully control *who* modifies the data and *what* are the modifications.

The main difficulty in this approach is, as said earlier, the acquisition of a strong knowledge in cryptography and cryptographic proofs. The task is also uneasy because there are already several highly skilled cryptographers working on this matter. At last, even if this the objective is fulfilled, there is a great chance that efficiency turn out to be even worse than plain FHE. This is a limitation for short term practical consideration, but improving efficiency may be, of course, part of further works.

Direction 3: a « HE-based privacy » framework Finally, we see a third possible, very abstract goal: designing a generic (non-trivial) framework for “homomorphic cryptography based privacy”, which would directly answer our ultimate goal. In other words, instead of proposing

very specific solutions for some systems, the idea would be to offer a solution for a substantial number of systems. The desired framework would need to be very generic, but different from the “encrypt-and-publish” one discussed earlier, and it should not be *void* (*i.e.* it should provide privacy technicians useful, non-trivial information). And of course, the framework should mainly use homomorphic cryptography. Finally, to instantiate the framework, one should need less work than for designing a tailored solution.

This very abstract description is a *desiderata*, but tells very little about how to design it. To obtain elements of answer, one need to first work on several systems, study them and enforce privacy within them. With the acquired experience, we can hope that some common elements between particular solutions will emerge. In other words, the goal would be to infer general properties from particular solutions. Then, one needs to see how homomorphic cryptography connects with these general properties, *i.e.* how homomorphic cryptography can help ensuring them.

This exercise might prove very difficult, and such a framework may actually not exist. Even though the practicality of the solutions derived from the framework might be limited (because generic solutions often imply limited efficiency), cryptography and privacy always benefit from theoretical positive result. The idea is that the framework would directly yield theoretical solution for many privacy problems, thus bringing an easy answer to the question “is ensuring privacy in *that context* possible?”. Therefore, we believe a HE-based privacy framework would be pertinent and desirable. As side contribution, it would bring a formalization of privacy with HE, which as of today lacks to the literature.

We could also say there is a fourth approach/direction, which is a combination of the others. Indeed, one way to proceed may be to begin with direction 1, deduce general properties, and join with direction 3 afterwards. Alternatively, the join can be made with direction 2, if we assume that the acquired experience from direction 1 can be put to profit of understanding homomorphic cryptography and the needs in term of controlled malleability. The limiting factor of this fourth composite approach is the time: studying one system and ensuring privacy in it is already time-consuming, and trying to repeat this for 2 or 3 other systems in order to tackle direction 2 or 3 afterwards is quite ambitious.

Concluding remarks

This document offers a deep understanding of privacy and presents homomorphic cryptography. Privacy is an emergent problematic on the rise. Recent actuality and the fact that the general public is now willing to use privacy-enhancing technologies such as Tor in censorship, anti-PRISM systems in the USA, or anonymizing proxy everywhere in the world, witnesses it. Also, the blazing fast evolution of (F)HE let us hope for practical solutions within the next decades. On the other hand, because the state-of-the-art (F)HE is instable, it is hard to conceive systems using current schemes: one needs to foresee and plan ahead in order to produce viable solutions in the long term.

To sum up the knowledge extracted from the two chapters, we can say that there are many possibilities to combine HE and protection of privacy: a very large portions of information systems can benefit from privacy properties, and FHE is a solution for many cryptographic or privacy issues. In other words, there are many ways to use (F)HE for privacy, but some choices are more pertinent than others. The goal is now to clear out the most relevant ones, and focus on the applications the most useful to the public.

Bibliography

- [ABC⁺13] Jagdish Prasad Achara, Franck Baudot, Claude Castelluccia, Geoffrey Delcroix, and Vincent Roca. Mobilitics: Analyzing privacy leaks in smartphones. *ERCIM News*, 2013(93), 2013.
Cited on page [26](#)
- [ABSB⁺11] Guillaume Aucher, Catherine Barreau-Saliou, Guido Boella, Annie Blandin-Obernesser, Sébastien Gambs, Guillaume Piolle, and Leendert Van Der Torre. The Coprelabri project : the logical approach to privacy. In *2e Atelier Protection de la Vie Privée (APVP 2011)*, Sorèze, France, June 2011. 6 pages.
Cited on page [16](#)
- [ACMT05] Giuseppe Ateniese, Daniel H. Chou, Breno Medeiros, and Gene Tsudik. Sanitizable signatures. In Sabrina De Capitani Vimercati, Paul Syverson, and Dieter Gollmann, editors, *Computer Security, ESORICS 2005*, volume 3679 of *LNCS*, pages 159–177. Springer Berlin Heidelberg, 2005.
Cited on page [76](#)
- [Acq12] Alessandro Acquisti. Privacy and market failures: Three reasons for concern, and three reasons for hope. *JTHTL*, 10(2):227–234, 2012.
Cited on page [18](#)
- [ADR02] JeeHea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In Lars R. Knudsen, editor, *Advances in Cryptology — EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 83–107. Springer Berlin Heidelberg, 2002.
Cited on page [73](#)
- [AFG⁺10] Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *LNCS*, pages 209–236. Springer Berlin Heidelberg, 2010.
Cited on page [76](#)
- [AG05] A. Acquisti and J. Grossklags. Privacy and rationality in individual decision making. *Security Privacy, IEEE*, 3(1):26–33, 2005.
Cited on pages [17](#) and [18](#)
- [AGK12] Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. Blip: Non-interactive differentially-private similarity computation on bloom filters. In Andréa W. Richa and Christian Scheidele, editors, *Stabilization, Safety, and Security*

- of Distributed Systems*, volume 7596 of *LNCS*, pages 202–216. Springer Berlin Heidelberg, 2012.
Cited on pages 23 and 34
- [AJLA⁺12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold fhe. In *Proceedings of the 31st Annual international conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT’12, pages 483–501, Cambridge, UK, 2012. Springer-Verlag.
Cited on pages 26, 76, 77, and 84
- [AMFF⁺13] C. Aguilar-Melchor, S. Fau, C. Fontaine, G. Gogniat, and R. Sirdey. Recent advances in homomorphic encryption: A possible future for signal processing in the encrypted domain. *Signal Processing Magazine, IEEE*, 30(2):108–117, 2013.
Cited on pages 42 and 71
- [ARZ99] Marc P. Armstrong, Gerard Rushton, and Dale L. Zimmerman. Geographically masking health data to preserve confidentiality. *Statistics in Medicine*, 18(5):497–525, 1999.
Cited on page 25
- [ASP13] Jacob Alperin-Sheriff and Chris Peikert. Practical bootstrapping in quasilinear time. In Ran Canetti and JuanA. Garay, editors, *Advances in Cryptology – CRYPTO 2013*, volume 8042 of *LNCS*, pages 1–20. Springer Berlin Heidelberg, 2013.
Cited on page 57
- [ASP14] Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. Cryptology ePrint Archive, Report 2014/094, 2014. <http://eprint.iacr.org/>.
Cited on pages 57, 62, and 71
- [Baa02] Sara Baase. *A Gift of Fire: Social, legal, and ethical issues for computers and the Internet*. Prentice Hall, 2002.
Cited on page 10
- [Bar89] David A. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in {NC1}. *Journal of Computer and System Sciences*, 38(1):150 – 164, 1989.
Cited on page 54
- [BB12a] Boaz Barak and Zvika Brakerski. Building the swiss army knife. windows on theory. Blog <http://windowsontheory.org/2012/05/02/building-the-swiss-army-knife/>, 2012.
Cited on page 81
- [BB12b] Boaz Barak and Zvika Brakerski. The swiss army knife. windows on theory. Blog <http://windowsontheory.org/2012/05/01/the-swiss-army-knife-of-cryptography/>, 2012.
Cited on page 81
- [BBDP01] Mihir Bellare, Alexandra Boldyreva, Anand Desai, and David Pointcheval. Key-privacy in public-key encryption. In Colin Boyd, editor, *Advances in Cryptology*

- *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 566–582. Springer Berlin Heidelberg, 2001.
Cited on page 30
- [BC04] Dan Boneh and Giovanni Di Crescenzo. Public key encryption with keyword search. In *Proceedings of Eurocrypt 2004, LNCS 3027*, pages 506–522, 2004.
Cited on page 79
- [Ben94] Josh Benaloh. Dense probabilistic encryption. In *In Proceedings of the Workshop on Selected Areas of Cryptography*, pages 120–128, 1994.
Cited on page 48
- [BGI⁺12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, May 2012.
Cited on page 84
- [BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-dnf formulas on ciphertexts. In *Proceedings of the Second international conference on Theory of Cryptography, TCC’05*, pages 325–341, Berlin, Heidelberg, 2005. Springer-Verlag.
Cited on pages 41, 54, and 59
- [BGV12] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *ITCS 2012*, pages 309–325, Cambridge, Massachusetts, 2012. ACM.
Cited on pages 57, 58, 62, 67, 70, and 71
- [BL11] Andrej Bogdanov and Chin Ho Lee. Homomorphic encryption from codes. Cryptology ePrint Archive, Report 2011/622, 2011. <http://eprint.iacr.org/>.
Cited on page 73
- [BLLN13] JoppeW. Bos, Kristin Lauter, Jake Loftus, and Michael Naehrig. Improved security for a ring-based fully homomorphic encryption scheme. In Martijn Stam, editor, *Cryptography and Coding*, volume 8308 of *LNCS*, pages 45–64. Springer Berlin Heidelberg, 2013.
Cited on pages 62, 67, 69, and 71
- [Blo70] Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, jul 1970.
Cited on page 34
- [BLP⁺13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing, STOC ’13*, pages 575–584, New York, NY, USA, 2013. ACM.
Cited on page 65
- [Boy10] Danah Boyd. Privacy and publicity in the context of big data. WWW. Raleigh, North Carolina, April 2010.
Cited on page 37
- [BPS12] M. Brenner, H. Perl, and M. Smith. How practical is homomorphically encrypted program execution? an implementation and performance evaluation. In *Trust*,

Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on, pages 375–382, 2012.

Cited on page 71

[Bra12] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *LNCS*, pages 868–886. Springer Berlin Heidelberg, 2012.

Cited on pages 58, 62, 63, 65, and 67

[Bra13] Zvika Brakerski. When homomorphism becomes a liability. In Amit Sahai, editor, *Theory of Cryptography*, volume 7785 of *LNCS*, pages 143–161. Springer Berlin Heidelberg, 2013.

Cited on page 73

[BRS13a] Dan Boneh, Ananth Raghunathan, and Gil Segev. Function-private identity-based encryption: Hiding the function in functional encryption. In *CRYPTO*, pages 461–478. Springer, 2013.

Cited on pages 30 and 79

[BRS13b] Dan Boneh, Ananth Raghunathan, and Gil Segev. Function-private subspace-membership encryption and its applications. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013*, volume 8269 of *LNCS*, pages 255–275. Springer Berlin Heidelberg, 2013.

Cited on page 79

[BS03] A.R. Beresford and F. Stajano. Location privacy in pervasive computing. *Pervasive Computing, IEEE*, 2(1):46–55, Jan 2003.

Cited on page 25

[BSS⁺13] Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The simon and speck families of lightweight block ciphers. Cryptology ePrint Archive, Report 2013/404, 2013. <http://eprint.iacr.org/>.

Cited on page 70

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *Theory of Cryptography*, volume 6597 of *LNCS*, pages 253–273. Springer Berlin Heidelberg, 2011.

Cited on page 78

[BSW12] Dan Boneh, Gil Segev, and Brent Waters. Targeted malleability: Homomorphic encryption for restricted computations. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, pages 350–366, New York, NY, USA, 2012. ACM.

Cited on pages 72 and 76

[BV11a] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:109, 2011.

Cited on page 62

[BV11b] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In Phillip Rogaway, editor,

- Advances in Cryptology – CRYPTO 2011*, volume 6841 of *LNCS*, pages 505–524. Springer Berlin Heidelberg, 2011.
Cited on pages 41, 58, and 62
- [BV14] Zvika Brakerski and Vinod Vaikuntanathan. Lattice-based fhe as secure as pke. In *Proceedings of the 5th Conference on Innovations in Theoretical Computer Science, ITCS '14*, pages 1–12, New York, NY, USA, 2014. ACM.
Cited on pages 62 and 67
- [CABP13] Konstantinos ChatzikoKolakis, MiguelE. Andrés, NicolásEmilio Bordenabe, and Catuscia Palamidessi. Broadening the scope of differential privacy using metrics. In Emiliano Cristofaro and Matthew Wright, editors, *Privacy Enhancing Technologies*, volume 7981 of *LNCS*, pages 82–102. Springer Berlin Heidelberg, 2013.
Cited on page 34
- [Cas06] Guilhem Castagnos. *Quelques schémas de cryptographie asymétrique probabiliste*. PhD thesis, Université de Limoges, 2006.
Cited on page 51
- [CBW07] Su-Jeong Choi, Simon R. Blackburn, and Peter R. Wild. Cryptanalysis of a homomorphic public-key cryptosystem over a finite group. *Journal of Mathematical Cryptology*, 1:351–358, December 2007. <http://eprint.iacr.org/>.
Cited on page 73
- [CCK⁺13] JungHee Cheon, Jean-Sébastien Coron, Jinsu Kim, MoonSung Lee, Tancrede Lepoint, Mehdi Tibouchi, and Aaram Yun. Batch fully homomorphic encryption over the integers. In Thomas Johansson and PhongQ. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 315–335. Springer Berlin Heidelberg, 2013.
Cited on pages 58, 60, 62, and 71
- [CCP10] Claude Castelluccia, Emiliano De Cristofaro, and Daniele Perito. Private information disclosure from web searches. In *Privacy Enhancing Technologies*, pages 38–55, 2010.
Cited on pages 15 and 24
- [Cha81] David L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–90, February 1981.
Cited on pages 27 and 28
- [Cha04] Yan-Cheng Chang. Single database private information retrieval with logarithmic communication. In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *Information Security and Privacy*, volume 3108 of *LNCS*, pages 50–61. Springer Berlin Heidelberg, 2004.
Cited on page 53
- [CJJ⁺13] David Cash, Stanislaw Jarecki, Charanjit S. Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. Highly-scalable searchable symmetric encryption with support for boolean queries. In *CRYPTO (1)*, pages 353–373, 2013.
Cited on page 79
- [CJL12] Sébastien Canard, Amandine Jambert, and Roch Lescuyer. Sanitizable signatures with several signers and sanitizers. In Aikaterini Mitrokotsa and Serge Vaudenay,

- editors, *Progress in Cryptology - AFRICACRYPT 2012*, volume 7374 of *LNCS*, pages 35–52. Springer Berlin Heidelberg, 2012.
Cited on page 76
- [CKLM12] Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Sarah Meiklejohn. Malleable proof systems and applications. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 281–300. Springer Berlin Heidelberg, 2012.
Cited on page 76
- [CKN03] Ran Canetti, Hugo Krawczyk, and Jesper B. Nielsen. Relaxing chosen-ciphertext security. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *LNCS*, pages 565–582. Springer Berlin Heidelberg, 2003.
Cited on page 73
- [CKN06] Jung Hee Cheon, Woo-Hwan Kim, and Hyun Soo Nam. Known-plaintext cryptanalysis of the domingo-ferrer algebraic privacy homomorphism scheme. *Inf. Process. Lett.*, 97(3):118–123, February 2006.
Cited on page 73
- [CKSJ03] C. Conrado, Frank Kamperman, G.-J. Schrijen, and W. Jonker. Privacy in an identity-based drm system. In *Database and Expert Systems Applications, 2003. Proceedings. 14th International Workshop on*, pages 389–395, Sept 2003.
Cited on page 26
- [Cle13] Privacy Rights Clearinghouse. Privacy today: A review of current issues. <https://www.privacyrights.org/ar/Privacy-IssuesList.htm>, May 2013.
Cited on page 19
- [CLT14] Jean-Sébastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Scale-invariant fully homomorphic encryption over the integers. In Hugo Krawczyk, editor, *Public-Key Cryptography – PKC 2014*, volume 8383 of *LNCS*, pages 311–328. Springer Berlin Heidelberg, 2014.
Cited on pages 60, 62, 69, 70, 71, and 109
- [CMNT11] Jean-Sébastien Coron, Avradip Mandal, David Naccache, and Mehdi Tibouchi. Fully homomorphic encryption over the integers with shorter public keys. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *LNCS*, pages 487–504. Springer Berlin Heidelberg, 2011.
Cited on pages 60, 62, and 71
- [CMPP06] Benoît Chevallier-Mames, Pascal Paillier, and David Pointcheval. Encoding-free elgamal encryption without random oracles. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography - PKC 2006*, volume 3958 of *LNCS*, pages 91–104. Springer Berlin Heidelberg, 2006.
Cited on page 47
- [CMS10] Jan Camenisch, Sebastian Mödersheim, and Dieter Sommer. A formal model of identity mixer. In Stefan Kowalewski and Marco Roveri, editors, *Formal Methods for Industrial Critical Systems*, volume 6371 of *LNCS*, pages 198–214. Springer Berlin Heidelberg, 2010.
Cited on page 24

- [CN11] Yuanmi Chen and PhongQ. Nguyen. Bkz 2.0: Better lattice security estimates. In DongHoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 1–20. Springer Berlin Heidelberg, 2011.
Cited on page 65
- [CN12] Yuanmi Chen and PhongQ. Nguyen. Faster algorithms for approximate common divisors: Breaking fully-homomorphic-encryption challenges over the integers. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 502–519. Springer Berlin Heidelberg, 2012.
Cited on page 60
- [CNT12] Jean-Sébastien Coron, David Naccache, and Mehdi Tibouchi. Public key compression and modulus switching for fully homomorphic encryption over the integers. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 446–464. Springer Berlin Heidelberg, 2012.
Cited on pages 60, 62, and 71
- [Com00] US Federal Trade Commission. Privacy online: Fair information practices in the electronic marketplace: A federal trade commission report to congress. <http://www.ftc.gov/reports/privacy-online-fair-information-practices-electronic-marketplace-federal-trade-commission>, May 2000.
Cited on page 14
- [Con06] World Wide Web Consortium. Platform for privacy preferences specification 1.1. <http://www.w3.org/P3P/>, 2006.
Cited on page 16
- [CS07] Gregory J. Conti and Edward Sobiesk. An honest man has nothing to fear: user perceptions on web-based information disclosure. In *SOUPS*, pages 112–121, 2007.
Cited on page 18
- [CVH91] David Chaum and Eugène Van Heyst. Group signatures. In *Proceedings of the 10th Annual International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT’91, pages 257–265, Berlin, Heidelberg, 1991. Springer-Verlag.
Cited on page 24
- [DAM06] Yves Deswarte and Carlos Aguilar Melchor. Current and future privacy enhancing technologies for the internet. *Annales Des Télécommunications*, 61(3-4):399–417, 2006.
Cited on page 24
- [DDM03] George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a type iii anonymous remailer protocol. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, SP ’03, pages 2–, Washington, DC, USA, 2003. IEEE Computer Society.
Cited on page 29

- [DF96] Josep Domingo-Ferrer. A new privacy homomorphism and applications. *Inf. Process. Lett.*, 60(5):277–282, December 1996.
Cited on page 73
- [DF02] Josep Domingo-Ferrer. A provably secure additive and multiplicative privacy homomorphism. In *Proceedings of the 5th International Conference on Information Security, ISC '02*, pages 471–483, London, UK, UK, 2002. Springer-Verlag.
Cited on page 73
- [DG12] Yves Deswarte and Sébastien Gambs. The challenges raised by the privacy-preserving identity card. In David Naccache, editor, *Cryptography and Security: From Theory to Applications*, volume 6805 of *LNCS*, pages 383–404. Springer Berlin Heidelberg, 2012.
Cited on page 24
- [DJ01] Ivan Damgård and Mads Jurik. A generalisation, a simplification and some applications of paillier’s probabilistic public-key system. In Kwangjo Kim, editor, *Public Key Cryptography*, volume 1992 of *LNCS*, pages 119–136. Springer Berlin Heidelberg, 2001.
Cited on pages 44, 51, 53, and 82
- [DJ03] Ivan Damgård and Mads Jurik. A length-flexible threshold cryptosystem with applications. In Rei Safavi-Naini and Jennifer Seberry, editors, *Information Security and Privacy*, volume 2727 of *LNCS*, pages 350–364. Springer Berlin Heidelberg, 2003.
Cited on pages 51 and 82
- [DMMQN11] Nico Döttling, Thilo Mie, Jörn Müller-Quade, and Tobias Nilges. Basing obfuscation on simple tamper-proof hardware assumptions. *Cryptology ePrint Archive*, Report 2011/675, 2011. <http://eprint.iacr.org/>.
Cited on page 79
- [DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography, TCC'06*, pages 265–284, Berlin, Heidelberg, 2006. Springer-Verlag.
Cited on page 23
- [DMS04] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th Conference on USENIX Security Symposium*, volume 13 of *SSYM'04*, pages 21–21, Berkeley, CA, USA, August 2004. USENIX Association.
Cited on pages 29 and 85
- [DP04] Claudia Diaz and Bart Preneel. Taxonomy of mixes and dummy traffic. In Yves Deswarte, Frédéric Cuppens, Sushil Jajodia, and Lingyu Wang, editors, *Information Security Management, Education and Privacy*, volume 148 of *IFIP International Federation for Information Processing*, pages 217–232. Springer US, 2004.
Cited on page 28
- [Dwo06] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming*, vol-

- ume 4052 of *LNCS*, pages 1–12. Springer Berlin Heidelberg, 2006.
Cited on pages 22 and 23
- [Dwo08] Cynthia Dwork. Differential privacy: A survey of results. In Manindra Agrawal, Dingzhu Du, Zhenhua Duan, and Angsheng Li, editors, *Theory and Applications of Models of Computation*, volume 4978 of *LNCS*, pages 1–19. Springer Berlin Heidelberg, 2008.
Cited on pages 23, 33, and 34
- [EHO⁺13] Keita Emura, Goichiro Hanaoka, Go Ohtake, Takahiro Matsuda, and Shota Yamada. Chosen ciphertext secure keyed-homomorphic public-key encryption. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *Public-Key Cryptography – PKC 2013*, volume 7778 of *LNCS*, pages 32–50. Springer Berlin Heidelberg, 2013.
Cited on pages 73 and 85
- [ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In GeorgeRobert Blakley and David Chaum, editors, *Advances in Cryptology*, volume 196 of *LNCS*, pages 10–18. Springer Berlin Heidelberg, 1985.
Cited on pages 41, 46, and 47
- [fECOD13] The Organization for Economic Co-Operation and Development. The 2013 oecd privacy guidelines, 2013.
Cited on page 14
- [FG07] Caroline Fontaine and Fabien Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP J. Inf. Secur.*, 2007:15:1–15:15, January 2007.
Cited on pages 45, 51, and 72
- [Fla99] David H. Flaherty. *Visions of Privacy: Policy Choices for the Digital Age*, chapter Visions of Privacy: Past, Present, and Future. Studies in comparative political economy and public policy. University of Toronto Press, 1999.
Cited on page 37
- [FLA11] Laurent Fousse, Pascal Lafourcade, and Mohamed Alnuaimi. Benaloh’s dense probabilistic encryption revisited. In Abderrahmane Nitaj and David Pointcheval, editors, *Progress in Cryptology – AFRICACRYPT 2011*, volume 6737 of *LNCS*, pages 348–362. Springer Berlin Heidelberg, 2011.
Cited on pages 48 and 49
- [FM91] Joan Feigenbaum and Michael Merritt. *Distributed Computing and Cryptography*, volume 2 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, chapter Open Questions, Talk Abstracts, and Summary of Discussions, pages 1–45. The American Mathematical Society, 1991.
Cited on page 40
- [FM02] Michael J. Freedman and Robert Morris. Tarzan: A peer-to-peer anonymizing network layer. In *Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS ’02*, pages 193–206, New York, NY, USA, 2002. ACM.
Cited on page 29
- [FV12] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. <http://eprint.iacr.org/>.
Cited on pages 62, 67, 69, and 71

- [GA07] Jens Grossklags and Alessandro Acquisti. When 25 cents is too much: An experiment on willingness-to-sell and willingness-to-protect personal information. In *WEIS*, 2007.
Cited on pages 18 and 36
- [Gal02] Steven D. Galbraith. Elliptic curve paillier schemes. *Journal of Cryptology*, 15(2):129–138, 2002.
Cited on page 51
- [GBP13] Antoine Guellier, Christophe Bidan, and Nicolas Prigent. Homomorphic cryptography based anonymous routing. Master’s thesis, Université de Rennes 1, Supélec (CIDRE), July 2013.
Cited on page 28
- [Gen09a] Craig Gentry. *A Fully Homomorphic Encryption Scheme*. PhD thesis, Stanford University, 2009.
Cited on pages 41, 55, 56, and 59
- [Gen09b] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC ’09 Proceedings of the 41st annual ACM symposium on Theory of computing*, STOC ’09, pages 169–178, Bethesda, MD, USA, 2009. ACM.
Cited on page 59
- [Gen10] Craig Gentry. Computing arbitrary functions of encrypted data. *Communications of the ACM*, 53(3):97–105, 2010.
Cited on page 57
- [GG03] Marco Gruteser and Dirk Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*, MobiSys ’03, pages 31–42, New York, NY, USA, 2003. ACM.
Cited on pages 25 and 33
- [GGH⁺13] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 40–49, Oct 2013.
Cited on pages 72, 79, and 83
- [GH11a] Craig Gentry and Shai Halevi. Fully homomorphic encryption without squashing using depth-3 arithmetic circuits. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS ’11, pages 107–109, Washington, DC, USA, 2011. IEEE Computer Society.
Cited on pages 57, 59, and 60
- [GH11b] Craig Gentry and Shai Halevi. Implementing gentry’s fully-homomorphic encryption scheme. In *Proceedings of the 30th Annual international conference on Theory and applications of cryptographic techniques: advances in cryptology*, EURO-CRYPT’11, pages 129–148, Berlin, Heidelberg, 2011. Springer-Verlag.
Cited on pages 59, 68, 71, and 109
- [GHRW14] Craig Gentry, Shai Halevi, Mariana Raykova, and Daniel Wichs. Garbled ram revisited, part i. Cryptology ePrint Archive, Report 2014/082, 2014. <http://>

eprint.iacr.org/.

Cited on pages 75 and 78

- [GHS12a] Craig Gentry, Shai Halevi, and Nigel P. Smart. Fully homomorphic encryption with polylog overhead. In *Proceedings of the 31st Annual international conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT'12, pages 465–482, Cambridge, UK, 2012. Springer-Verlag.
Cited on page 62
- [GHS12b] Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the aes circuit. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *LNCS*, pages 850–867. Springer Berlin Heidelberg, 2012.
Cited on pages 41, 70, and 71
- [GHV10] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. A simple bgn-type cryptosystem from lwe. In *EUROCRYPT*, pages 506–522, 2010.
Cited on page 54
- [Gjø05] Kristian Gjøsteen. Homomorphic cryptosystems based on subgroup membership problems. In Ed Dawson and Serge Vaudenay, editors, *Progress in Cryptology – Mycrypt 2005*, volume 3715 of *LNCS*, pages 314–327. Springer Berlin Heidelberg, 2005.
Cited on pages 46, 47, and 49
- [GKdPC10] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. Show me how you move and i will tell you who you are. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS*, SPRINGL '10, pages 34–41, New York, NY, USA, 2010. ACM.
Cited on page 25
- [GKP⁺13] Shafi Goldwasser, Yael Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, STOC '13, pages 555–564, New York, NY, USA, 2013. ACM.
Cited on pages 77 and 83
- [GLN13] Thore Graepel, Kristin Lauter, and Michael Naehrig. Ml confidential: Machine learning on encrypted data. In Taekyoung Kwon, Mun-Kyu Lee, and Daesung Kwon, editors, *Information Security and Cryptology – ICISC 2012*, volume 7839 of *LNCS*, pages 1–21. Springer Berlin Heidelberg, 2013.
Cited on page 83
- [GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, STOC '82, pages 365–377, New York, NY, USA, 1982. ACM.
Cited on pages 41, 44, and 45
- [GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270 – 299, 1984.
Cited on page 46

- [GO96] Oded Goldreich and Rafail Ostrovsky. Software protection and simulation on oblivious rams. *J. ACM*, 43(3):431–473, May 1996.
Cited on page 78
- [Gol01] Oded Goldreich. *The Foundations of Cryptography - Basic Tools*, volume 1. Cambridge University Press, June 2001.
Cited on page 24
- [Gol03] Oded Goldreich. Cryptography and cryptographic protocols. *Distributed Computing*, 16(2-3):177–199, 2003.
Cited on page 77
- [Gol04] Oded Goldreich. *The Foundations of Cryptography - Basic Applications*, volume 2. Cambridge University Press, May 2004.
Cited on pages 25, 40, 42, and 72
- [GOR14] Sebastien Gambs, Cristina Onete, and Jean-Marc Robert. Prover anonymous and deniable distance-bounding authentication. Cryptology ePrint Archive, Report 2014/114, 2014. <http://eprint.iacr.org/>.
Cited on pages 30 and 85
- [GP06] Dima Grigoriev and Ilia Ponomarenko. Homomorphic public-key cryptosystems and encrypting boolean circuits. *Applicable Algebra in Engineering, Communication and Computing*, 17(3-4):239–255, 2006.
Cited on page 73
- [GP09] Philippe Golle and Kurt Partridge. On the anonymity of home/work location pairs. In Hideyuki Tokuda, Michael Beigl, Adrian Friday, A.J. Bernheim Brush, and Yoshito Tobe, editors, *Pervasive Computing*, volume 5538 of *LNCS*, pages 390–397. Springer Berlin Heidelberg, 2009.
Cited on page 25
- [GPY⁺04] Bok-Min Goi, Raphael C.-W. Phan, Yanjiang Yang, Feng Bao, Robert H. Deng, and M.U. Siddiqi. Cryptanalysis of two anonymous buyer-seller watermarking protocols and an improvement for true anonymity. In Markus Jakobsson, Moti Yung, and Jianying Zhou, editors, *Applied Cryptography and Network Security*, volume 3089 of *LNCS*, pages 369–382. Springer Berlin Heidelberg, 2004.
Cited on page 26
- [GRS99] David Goldschlag, Michael Reed, and Paul Syverson. Onion routing. *Commun. ACM*, 42(2):39–41, February 1999.
Cited on page 29
- [GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013*, volume 8042 of *LNCS*, pages 75–92. Springer Berlin Heidelberg, 2013.
Cited on pages 62, 67, 68, and 72
- [GTD⁺08] Benedikt Gierlichs, Carmela Troncoso, Claudia Diaz, Bart Preneel, and Ingrid Verbauwhede. Revisiting a combinatorial approach toward measuring anonymity. In *Proceedings of the 7th ACM workshop on Privacy in the electronic society, WPES '08*, pages 111–116, New York, NY, USA, 2008. ACM.
Cited on page 32

- [HAJ11] Hongxin Hu, Gail-Joon Ahn, and Jan Jorgensen. Detecting and resolving privacy conflicts for collaborative data sharing in online social networks. In *Proceedings of the 27th Annual Computer Security Applications Conference, ACSAC '11*, pages 103–112, New York, NY, USA, 2011. ACM.
Cited on page [16](#)
- [HMV04] Darrel Hankerson, Alfred J. Menezes, and Scott Vanstone. *Guide to Elliptic Curve Cryptography*. Springer Professional Computing. Springer, 2004.
Cited on page [42](#)
- [ILL89] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing, STOC '89*, pages 12–24, New York, NY, USA, 1989. ACM.
Cited on page [63](#)
- [IP07] Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In Salil P. Vadhan, editor, *Theory of Cryptography*, volume 4392 of *LNCS*, pages 575–594. Springer Berlin Heidelberg, 2007.
Cited on pages [41](#) and [54](#)
- [JJK⁺13] Stanislaw Jarecki, Charanjit S. Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. Outsourced symmetric private information retrieval. In *ACM Conference on Computer and Communications Security*, pages 875–888, 2013.
Cited on page [79](#)
- [JMN10] Thomas P. Jakobsen, Marc X. Makkes, and Janus Dam Nielsen. Efficient implementation of the orlandi protocol. In *ACNS*, pages 255–272, 2010.
Cited on page [53](#)
- [JMSW02] Robert Johnson, David Molnar, Dawn Song, and David Wagner. Homomorphic signature schemes. In Bart Preneel, editor, *Topics in Cryptology*, volume 2271 of *LNCS*, pages 244–262. Springer Berlin Heidelberg, 2002.
Cited on page [75](#)
- [Jue06] A. Juels. Rfid security and privacy: a research survey. *Selected Areas in Communications, IEEE Journal on*, 24(2):381–394, Feb 2006.
Cited on page [26](#)
- [Kim86] Jay Kims. A method for limiting disclosure in microdata based on random noise and transformation. In *Proc. of the Survey Research Methods Section, ASA*, pages 370–374, 1986.
Cited on page [21](#)
- [KS02] G. Karjoth and M. Schunter. A privacy policy model for enterprises. In *Computer Security Foundations Workshop, 2002. Proceedings. 15th IEEE*, pages 271–281, 2002.
Cited on page [16](#)
- [KSS09] Vladimir Kolesnikov, A. R. Sadeghi, and T. Schneider. How to combine homomorphic encryption and garbled circuits. *Signal Processing in the Encrypted Domain*, 2009:100, 2009.
Cited on page [78](#)

- [KWF06] Stefan Köpsell, Rolf Wendolsky, and Hannes Federrath. Revocable anonymity. In Günter Müller, editor, *Emerging Trends in Information and Communication Security*, volume 3995 of *LNCS*, pages 206–220. Springer Berlin Heidelberg, 2006.
Cited on page 19
- [LATV12] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multi-party computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the 44th symposium on Theory of Computing*, STOC '12, pages 1219–1234, New York, New York, USA, 2012. ACM.
Cited on pages 26, 76, 77, and 84
- [LLV07] Ninghui Li, Tiancheng Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 106–115, 2007.
Cited on pages 22 and 23
- [LN14] Tancrede Lepoint and Michael Naehrig. A comparison of the homomorphic encryption schemes fv and yashe. In David Pointcheval and Damien Vergnaud, editors, *Progress in Cryptology – AFRICACRYPT 2014*, volume 8469 of *LNCS*, pages 318–335. Springer International Publishing, 2014.
Cited on pages 66, 69, 70, 71, 84, and 109
- [LO13] Steve Lu and Rafail Ostrovsky. How to garble ram programs? In Thomas Johansson and PhongQ. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 719–734. Springer Berlin Heidelberg, 2013.
Cited on page 78
- [LO14] Steve Lu and Rafail Ostrovsky. Garbled ram revisited, part ii. Cryptology ePrint Archive, Report 2014/083, 2014. <http://eprint.iacr.org/>.
Cited on pages 75 and 78
- [LP13] Tancrede Lepoint and Pascal Paillier. On the minimal number of bootstrappings in homomorphic circuits. In AndrewA. Adams, Michael Brenner, and Matthew Smith, editors, *Financial Cryptography and Data Security*, volume 7862 of *LNCS*, pages 189–200. Springer Berlin Heidelberg, 2013.
Cited on page 57
- [LPJY13] Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Linearly homomorphic structure-preserving signatures and their applications. In Ran Canetti and JuanA. Garay, editors, *Advances in Cryptology – CRYPTO 2013*, volume 8043 of *LNCS*, pages 289–307. Springer Berlin Heidelberg, 2013.
Cited on page 76
- [LPJY14] Benoît Libert, Thomas Peters, Marc Joye, and Moti Yung. Non-malleability from malleability: Simulation-sound quasi-adaptive nizk proofs and cca2-secure encryption from homomorphic signatures. In PhongQ. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology – EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 514–532. Springer Berlin Heidelberg, 2014.
Cited on page 73
- [LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology –*

- EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer Berlin Heidelberg, 2010.
Cited on page [51](#)
- [MGH10] Carlos Aguilar Melchor, Philippe Gaborit, and Javier Herranz. Additively homomorphic encryption with d-operand multiplications. In *Proceedings of the 30th annual conference on Advances in cryptology, CRYPTO'10*, pages 138–154, Berlin, Heidelberg, 2010. Springer-Verlag.
Cited on pages [41](#), [42](#), [54](#), and [64](#)
- [MKGV07] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkatasubramanian. L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), March 2007.
Cited on page [22](#)
- [Mor13] Evgeny Morozov. The real privacy problem. Technical report, MIT, October 2013.
Cited on page [37](#)
- [MYR13] C.Y.T. Ma, D.K.Y. Yau, N.K. Yip, and N.S.V. Rao. Privacy vulnerability of published anonymous mobility traces. *Networking, IEEE/ACM Transactions on*, 21(3):720–733, June 2013.
Cited on page [25](#)
- [Nat90] United Nations. Guidelines for the regulation of computerized data file. Resolution A/RES/45/95 of the General Assembly of United Nations, December 1990.
Cited on page [14](#)
- [NS98] David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. In *Proceedings of the 5th ACM conference on Computer and communications security, CCS '98*, pages 59–66, New York, NY, USA, 1998. ACM.
Cited on page [49](#)
- [NS08] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *IEEE Symposium on Security and Privacy*, pages 111–125, 2008.
Cited on pages [15](#) and [22](#)
- [oE95] US Department of Education. Family educational rights and privacy, 20 u.s.c. § 1232g(a)(5)(a) (2006). Federal Register / Vol. 60, No. 10, January 1995.
Cited on page [14](#)
- [Ohm09] Paul Ohm. Broken promises of privacy: Responding to the surprising failure of anonymization. *UCLA Law Review*, Vol. 57, p. 1701, 2010, 2009.
Cited on pages [15](#) and [22](#)
- [Ole09] V. Oleshchuk. Internet of things and privacy preserving technologies. In *Wireless Communication, Vehicular Technology, Information Theory and Aerospace Electronic Systems Technology, 2009. Wireless VITAE 2009. 1st International Conference on*, pages 336–340, 2009.
Cited on page [26](#)
- [OU98] Tatsuaki Okamoto and Shigenori Uchiyama. A new public-key cryptosystem as secure as factoring. In Kaisa Nyberg, editor, *Advances in Cryptology — EUROCRYPT'98*, volume 1403 of *LNCS*, pages 308–318. Springer Berlin Heidelberg,

1998.
Cited on page [49](#)
- [Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the 17th international conference on Theory and application of cryptographic techniques*, EUROCRYPT'99, pages 223–238, Berlin, Heidelberg, 1999. Springer-Verlag.
Cited on page [41](#)
- [PD11] Guillaume Piolle and Yves Demazeau. Representing privacy regulations with deontico-temporal operators. *Web Intelligence and Agent Systems: an International Journal (WIAS)*, 9(3):209–226, July 2011.
Cited on page [16](#)
- [Pio09] Guillaume Piolle. *Agents utilisateurs pour la protection des données personnelles : modélisation logique et outils informatiques deswar*. PhD thesis, Université Joseph Fourier, juin 2009.
Cited on page [15](#)
- [PK01] Andreas Pfitzmann and Marit Köhntopp. Anonymity, unobservability, and pseudonymity — a proposal for terminology. In Hannes Federrath, editor, *Designing Privacy Enhancing Technologies*, volume 2009 of *LNCS*, pages 1–9. Springer Berlin Heidelberg, 2001.
Cited on pages [10](#), [19](#), [20](#), and [27](#)
- [PR08] Manoj Prabhakaran and Mike Rosulek. Homomorphic encryption with cca security. In Luca Aceto, Ivan Damgård, LeslieAnn Goldberg, MagnúsM. Halldórsson, Anna Ingólfssdóttir, and Igor Walukiewicz, editors, *Automata, Languages and Programming*, volume 5126 of *LNCS*, pages 667–678. Springer Berlin Heidelberg, 2008.
Cited on page [73](#)
- [PRZB11] Raluca Ada Popa, Catherine M. S. Redfield, Nikolai Zeldovich, and Hari Balakrishnan. Cryptdb: Protecting confidentiality with encrypted query processing. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, SOSP '11, pages 85–100, New York, NY, USA, 2011. ACM.
Cited on page [79](#)
- [PtC95] The European Parliament and the Council. Directive 1995/46/ec of the european parliament and of the council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data. European Union, editor, Official Journal of the European Communities, October 1995.
Cited on page [14](#)
- [PtC02] The European Parliament and the Council. Directive 2002/58/ec of the european parliament and of the council of 12 july 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector. European Union, editor, Official Journal of the European Communities, July 2002.
Cited on page [14](#)
- [PW97] Birgit Pfitzmann and Michael Waidner. Anonymous fingerprinting. In Walter Fumy, editor, *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *LNCS*,

- pages 88–102. Springer Berlin Heidelberg, 1997.
Cited on page 82
- [RAD78] R.L. Rivest, L. Adleman, and M.L. Dertouzos. On data banks and privacy homomorphisms. In *Foundations on Secure Computation*, Academia Press, pages 169–179, 1978.
Cited on page 40
- [Rap06] Doerte K. Rappe. *Homomorphic Cryptosystems and their Applications*. PhD thesis, University of Dortmund, Germany, 2006. <http://eprint.iacr.org/>.
Cited on pages 46, 51, 53, 77, and 82
- [RCB01] N.K. Ratha, J.H. Connell, and R.M. Bolle. Enhancing security and privacy in biometrics-based authentication systems. *IBM Systems Journal*, 40(3):614–634, 2001.
Cited on page 20
- [Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the Thirty-seventh Annual ACM Symposium on Theory of Computing*, STOC '05, pages 84–93, New York, NY, USA, 2005. ACM.
Cited on pages 51, 62, 63, and 65
- [Rot11] Ron Rothblum. Homomorphic encryption: From private-key to public-key. In Yuval Ishai, editor, *Theory of Cryptography*, volume 6597 of *LNCS*, pages 219–234. Springer Berlin Heidelberg, 2011.
Cited on page 59
- [RP02] Marc Rennhard and Bernhard Plattner. Introducing morphmix: Peer-to-peer based anonymous internet usage with collusion detection. In *Proceedings of the 2002 ACM Workshop on Privacy in the Electronic Society*, WPES '02, pages 91–102, New York, NY, USA, 2002. ACM.
Cited on page 29
- [SBZ02] Ron Steinfeld, Laurence Bull, and Yuliang Zheng. Content extraction signatures. In Kwangjo Kim, editor, *Information Security and Cryptology — ICISC 2001*, volume 2288 of *LNCS*, pages 285–304. Springer Berlin Heidelberg, 2002.
Cited on page 76
- [SCK⁺13] Manya Sleeper, Justin Cranshaw, Patrick Gage Kelley, Blase Ur, Alessandro Acquisti, Lorrie Faith Cranor, and Norman Sadeh. "i read my twitter the next morning and was astonished": A conversational perspective on twitter regrets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 3277–3286, New York, NY, USA, 2013. ACM.
Cited on page 36
- [SH12] Mudhakar Srivatsa and Mike Hicks. Deanonymizing mobility traces: Using social network as a side-channel. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12, pages 628–637, New York, NY, USA, 2012. ACM.
Cited on page 25
- [Sha49] C. E. Shannon. Communication theory of secrecy systems*. *Bell System Technical Journal*, 28(4):656–715, 1949.
Cited on page 31

- [Sol06] Daniel J. Solove. A taxonomy of privacy. *University of Pennsylvania Law Review*, Vol. 154, No. 3:447, January 2006.
Cited on page 10
- [Sol07] Daniel J. Solove. ‘i’ve got nothing to hide’ and other misunderstandings of privacy. *San Diego Law Review*, Vol. 44:745, 2007.
Cited on page 37
- [SS10] Damien Stehlé and Ron Steinfeld. Faster fully homomorphic encryption. In Masayuki Abe, editor, *Advances in Cryptology - ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 377–394. Springer Berlin Heidelberg, 2010.
Cited on pages 59 and 60
- [Ste08] Andreas Steffen. E-voting simulator based on the paillier cryptosystem. <http://security.hsr.ch/msevote/paillier>, 2008.
Cited on page 53
- [SV10] N. P. Smart and F. Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *Proceedings of the 13th international conference on Practice and Theory in Public Key Cryptography, PKC’10*, pages 420–443, Paris, France, 2010.
Cited on pages 58, 59, 62, and 68
- [Swe00] L. Sweeney. Uniqueness of simple demographics in the U.S. population. Technical report, Carnegie Mellon University, School of Computer Science, Data Privacy Laboratory, Pittsburgh, USA, 2000.
Cited on page 22
- [Swe02] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.
Cited on page 22
- [Swe05] Latanya Sweeney. Privacy-enhanced linking. *SIGKDD Explor. Newsl.*, 7(2):72–75, December 2005.
Cited on page 22
- [SYY99] Tomas Sander, Adam Young, and Moti Yung. Non-interactive cryptocomputing for nc1. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science, FOCS ’99*, page 554, Washington, DC, USA, 1999. IEEE Computer Society.
Cited on pages 41 and 54
- [Tay03] Humphrey Taylor. Most people are “privacy pragmatists” who, while concerned about privacy, will sometimes trade it off for other benefits. Technical report, Harris Interactive, 2003.
Cited on page 36
- [THV04] Gergely Tóth, Zoltán Hornák, and Ferenc Vajda. Measuring anonymity revisited. In Sanna Liimatainen and Teemupekka Virtanen, editors, *Proceedings of the Ninth Nordic Workshop on Secure IT Systems*, pages 85–90, November 2004.
Cited on page 32

- [TY98] Yiannis Tsiounis and Moti Yung. On the security of elgamal based encryption. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography*, volume 1431 of *LNCS*, pages 117–134. Springer Berlin Heidelberg, 1998.
Cited on page [47](#)
- [UC96] 104th US Congress. Health insurance portability and accountability act of 1996. U.S. Government Printing Office, August 1996.
Cited on page [14](#)
- [Vai12] Vinod Vaikuntanathan. How to compute on encrypted data. In Steven Galbraith and Mridul Nandi, editors, *Progress in Cryptology - INDOCRYPT 2012*, volume 7668 of *LNCS*, pages 1–15. Springer Berlin Heidelberg, 2012.
Cited on pages [42](#) and [75](#)
- [vDGHV10] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *LNCS*. Springer Berlin Heidelberg, gilbert, henri edition, 2010.
Cited on pages [60](#), [61](#), and [71](#)
- [VDJ10] Marten Van Dijk and Ari Juels. On the impossibility of cryptography alone for privacy-preserving cloud computing. In *Proceedings of the 5th USENIX Conference on Hot Topics in Security*, HotSec’10, pages 1–8, Berkeley, CA, USA, 2010. USENIX Association.
Cited on pages [25](#), [77](#), and [84](#)
- [vTJ11] Henk C. A. van Tilborg and Sushil Jajodia, editors. *Encyclopedia of Cryptography and Security, 2nd Ed.* Springer, 2011.
Cited on page [31](#)
- [VYT05] DuongQuang Viet, Akihiro Yamamura, and Hidema Tanaka. Anonymous password-based authenticated key exchange. In Subhamoy Maitra, C.E. Veni Madhavan, and Ramarathnam Venkatesan, editors, *Progress in Cryptology - INDOCRYPT 2005*, volume 3797 of *LNCS*, pages 244–257. Springer Berlin Heidelberg, 2005.
Cited on page [30](#)
- [Wag03] David Wagner. Cryptanalysis of an algebraic privacy homomorphism. In Colin Boyd and Wenbo Mao, editors, *Information Security*, volume 2851 of *LNCS*, pages 234–239. Springer Berlin Heidelberg, 2003.
Cited on page [73](#)
- [WB90] Samuel D. Warren and Louis D. Brandeis. The right of privacy. *Harvard Law Review*, 4:193–195, 1890.
Cited on page [5](#)
- [WCS+13] Shomir Wilson, Justin Cranshaw, Norman Sadeh, Alessandro Acquisti, Lorie Faith Cranor, Jay Springfield, Sae Young Jeong, and Arun Balasubramanian. Privacy manipulation and acclimation in a location sharing application. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp ’13, pages 549–558, New York, NY, USA, 2013. ACM.
Cited on pages [18](#) and [36](#)

- [Wes67] Alan F. Westin. *Privacy and Freedom*. New York: Atheneum, 1967.
Cited on page 10
- [Win14] Philipp Winter. Censorship bibliography. <http://www.cs.kau.se/philwint/censorbib/>, 2014.
Cited on page 85
- [WNK⁺11] Yang Wang, Gregory Norcie, Saranga Komanduri, Alessandro Acquisti, Pedro Giovanni Leon, and Lorrie Faith Cranor. "i regretted the minute i pressed share": A qualitative study of regrets on facebook. In *Proceedings of the Seventh Symposium on Usable Privacy and Security*, SOUPS '11, pages 10:1–10:16, New York, NY, USA, 2011. ACM.
Cited on page 36
- [Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *FOCS*, pages 160–164, 1982.
Cited on pages 77 and 82
- [Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 162–167, 1986.
Cited on page 77
- [ZG09] Elena Zheleva and Lise Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *WWW*, pages 531–540, 2009.
Cited on page 24
- [ZKVB11] Xuebing Zhou, A. Kuijper, R. Veldhuis, and C. Busch. Quantifying privacy and security of biometric fuzzy commitment. In *Biometrics (IJCB), 2011 International Joint Conference on*, pages 1–8, Oct 2011.
Cited on page 20
- [ZLLF06] Yanchao Zhang, Wei Liu, Wenjing Lou, and Yuguang Fang. Mask: anonymous on-demand routing in mobile ad hoc networks. *Wireless Communications, IEEE Transactions on*, 5(9):2376–2385, September 2006.
Cited on page 29
- [ZYZ⁺12] Gaofeng Zhang, Yun Yang, Xuyun Zhang, Chang Liu, and Jinjun Chen. Key research issues for privacy protection and preservation in cloud computing. In *Cloud and Green Computing (CGC), 2012 Second International Conference on*, pages 47–54, 2012.
Cited on page 25

List of Figures

1.1	Solove's taxonomy of privacy torts from a social point a view	11
1.2	A 3-anonymous medical database	23
1.3	A simple MIX, using re-ordering and re-encryption	28
2.1	Listing of FHE schemes	58
2.2	Evolution of underlying concepts of FHE schemes	59
2.3	Minimal and maximum bit-sizes for parameter q	66

List of Tables

2.1	Comparison of PHE schemes	52
2.2	Implementation of Paillier (libpaillier)	53
2.3	Implementation of an ideal lattice scheme [GH11b] (2011)	68
2.4	Implementation of an integer-based scheme [CLT14] (2014)	69
2.5	Implementation of two Ring-LWE-based schemes [LN14] (2014)	69
2.6	Ciphertexts and key sizes in the FV scheme	69
2.7	Homomorphic evaluations of block ciphers	71
2.8	Main FHE/LHE implementations	71



**RESEARCH CENTRE
RENNES – BRETAGNE ATLANTIQUE**

Campus universitaire de Beaulieu
35042 Rennes Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399