Karlsruhe Institute of Technology

Cryptography Based on Correlated Data: Foundations and Practice

zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

der Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Rafael Baião Dowsley

aus Brasília, Brasilien

Tag der mündlichen Prüfung: 15. Juli 2016

Erster Gutachter: Prof. Dr. Jörn Müller-Quade

Zweiter Gutachter: Prof. Anderson C. A. Nascimento, PhD

Acknowledgements

I am grateful to many persons that have influenced and supported me in my journey to finish this thesis.

First of all, I would like to greatly thank my advisor Prof. Dr. Jörn Müller-Quade for always supporting me, for vastly inspiring me with all his enthusiasm and fascination for the fields of cryptography and security, and for giving me total freedom to pursue my research interests.

I am also in great debt with Prof. Dr. Anderson C. A. Nascimento for accepting to co-referee this thesis, for all the excellent guidance during my Bachelor's and Master's studies as well as afterwards, and for initially arousing my interest for the scientific research in the field of cryptography.

I owe a great amount of insights and knowledge to my brilliant co-authors, with whom it was a great pleasure to collaborate during the last couple of years. I would also like to express my gratitude to all my colleagues at KIT for the good times I enjoyed here. Special thanks to Brandon Broadnax for helping me with the German version of the abstract.

Last but undoubtedly not the least, I would like to thank my family for their unconditional support and love during all the times.

Abstract

Correlated data can be very useful in cryptography. For instance, if a uniformly random key is available to Alice and Bob, it can be used as an one-time pad to transmit a message with perfect security. With more elaborate forms of correlated data, the parties can achieve even more complex cryptographic tasks, such as secure multiparty computation. This thesis explores (from both a theoretical and a practical point of view) the topic of cryptography based on correlated data.

The first part considers physical assumptions that can be used to obtain simple forms of correlated data suitable for cryptographic purposes. We aim at constructing two important cryptographic primitives, namely commitments and oblivious transfer, and investigate the question of their existence as well as the theoretical limits of how efficiently the underlying resources can be used to construct them. For example, the existence of noisy channels between the parties allows unconditionally secure realizations of both primitives. As it turns out that noisy channels are valuable resources for cryptography, it becomes important to understand the optimal way in which these noisy channels can be used for implementing cryptographic tasks. Therefore, commitment and oblivious transfer capacities have been studied in the literature, which capture respectively the optimal way in which commitment and oblivious transfer can be realized using noisy channels. These capacities are cryptographic equivalents of Shannon's definition of channel capacity for the task of transmitting messages reliably over noisy channels. In the thesis we will further investigate the commitment and oblivious transfer capacity of some important channels. Another example is the so-called bounded storage model, in which it is assumed that the parties have limited storage capacity (an assumption orthogonal to the restrictions on computational power that are normally made in cryptography based on complexity theory). In this model there is a public random source available to the parties during an initial transmission phase, but since the parties only have bounded storage they can only store parts of this random source and therefore they end up with correlated data that can be used subsequently to implement cryptographic primitives. It is known that both commitment and oblivious transfer can be implemented in the bounded storage model without errors. We present the first secure protocols for commitment and oblivious transfer in the more realistic bounded storage model with errors, in which the public random sources available to the parties are not exactly the same, but instead are only required to have a small

Hamming distance (between themselves).

The second part of this thesis focuses on the practical side. We investigate the so-called trusted initializer model, in which there exists a trusted party that predistributes correlated data to the protocol participants during a setup phase, but does not take part in the rest of the protocol execution and in particular does not learn the parties' inputs. This allows for more structured forms of correlated data between the parties. In this model it is possible to obtain very efficient solutions that achieve information-theoretical security. We address the crucial question of obtaining highly practical and parallelizable protocols for the secure computation of some important machine learning tasks.

Zusammenfassung

Korrelierten Daten können sehr nützlich sein in der Kryptographie. Wenn sich beispielsweise Alice und Bob einen gleichverteilt zufälligen Schlüssel teilen, dann können sie das One-Time-Pad-Verfahren verwenden, um eine Nachricht mit perfekter Sicherheit zu übertragen. Mit komplexeren Formen von korrelierten Daten können die Parteien noch komplexere kryptographische Aufgaben lösen, wie zum Beispiel sichere Mehrparteienberechnung. Diese Arbeit untersucht sowohl aus theoretischer als auch aus praktischer Sicht das Thema der Kryptographie auf der Basis von korrelierten Daten.

Der erste Teil dieser Dissertation betrachtet physikalische Annahmen, die verwendet werden können, um einfache Formen von korrelierten Daten, die geeignet für kryptographische Zwecke sind, zu erhalten. Wir befassen uns mit zwei wichtige kryptographische Primitive, nämlich Commitments und Oblivious-Transfer, und untersuchen die Frage ihrer Existenz sowie die theoretischen Grenzen, wie effizient die zugrunde liegenden Ressourcen verwendet werden können, um sie zu konstruieren. Beispielsweise ermöglicht die Existenz verrauschte Kanäle zwischen den Parteien beide Primitive sicher zu realisieren ohne zusätzliche Annahme treffen zu müssen. Es stellt sich heraus, dass verrauschte Kanäle wertvolle Ressourcen für Kryptographie sind, und damit wird es wichtig, die optimale Art und Weise, wie diese verrauschten Kanäle für die Implementierung kryptographischer Aufgaben verwendet werden können, zu verstehen. Deshalb wurden Commitment- und Oblivious-Transfer-Kapazitäten in der Literatur untersucht. Sie erfassen die optimale Art und Weise, wie verrauschte Kanäle verwendet werden können, um Commitments beziehungsweise Oblivious-Transfer zu realisieren. Diese Kapazitäten sind kryptographische Äquivalente von Shannons' Kanalkapazität für die Aufgabe der zuverlässigen Übertragung von Nachrichten über verrauschte Kanäle. In dieser Arbeit untersuchen wir die Commitment- und Oblivious-Transfer-Kapazitäten von einigen wichtigen Kanälen. Ein weiteres Beispiel ist das so genannte Bounded-Storage-Modell, in dem angenommen wird, dass die Speicherkapazität der Parteien begrenzt ist (diese Annahme ist orthogonal zu der Beschränkung von Rechenleistung, die in der Kryptographie auf Basis der Komplexitätstheorie normalerweise gemacht wird). In diesem Modell gibt es eine öffentliche Zufallsquelle, die für die Parteien während einer anfänglichen Übertragungsphase zur Verfügung steht. Da aber die Parteien nur begrenzte Speicherkapazität haben, können sie nur Teile dieser Zufallsquelle speichern. Damit haben sie am Ende korrelierte Daten, die später verwendet werden können, um kryptographische Primitive zu implementieren. Es ist bekannt, dass sowohl Commitments als auch Oblivious-Transfer im Bounded-Storage-Modell ohne Fehler implementiert werden können. Wir präsentieren hier die ersten sicheren Protokolle für Commitments und Oblivious Transfer im realistischeren Bounded-Storage-Modell mit Fehlern, in dem die Zufallsquellen nicht gleich sind, sondern nur eine kleine Hamming-Distanz haben müssen.

Der zweite Teil der Arbeit konzentriert sich auf die praktische Seite. Wir untersuchen das so genannte Trusted-Initializer-Modell, in dem es eine vertrauenswürdige Partei gibt, die korrelierte Daten an die Protokollteilnehmer während einer Aufbauphase verteilt, aber nicht an dem Rest der Protokollausführung teilnimmt und insbesondere nicht die Eingabe der Parteien lernt. Dies ermöglicht strukturiertere Formen von korrelierten Daten zwischen den Parteien. In diesem Modell ist es möglich, sehr effiziente Lösungen zu erzielen, die informationstheoretische Sicherheit erreichen. Wir konzentrieren uns auf die Frage, praktische und parallelisierbare Protokolle für die sichere Berechnung einiger wichtiger Aufgaben aus dem Bereich des maschinellen Lernens zu konzipieren.

Contents

1. Introduction

Correlated data can be very useful for cryptographic purposes. Imagine for instance that a random key is chosen and given to Alice and Bob. This key can then be used as a one-time pad in order to transmit messages with unconditional security. When more elaborated forms of correlated data are available to the parties, even more complex cryptographic tasks can be achieved, such as secure computation without revealing the private inputs. For example: suppose that Alice has input $x \in \mathcal{X}$ and Bob input $y \in \mathcal{Y}$ and they want to compute the function $f(x, y)$ without leaking any additional information about x or y. Lets consider the table $\mathbf{T}$ that contains all possible outputs of f: the columns are indexed by $x \in \mathcal{X}$ and the rows by $y \in \mathcal{Y}$, and each element has the respective output $f(x, y)$. Let $\mathbf{R}$ be a permuted version of $\mathbf{T}$ in which first the columns are permuted according to a permutation known by Alice and then the rows are permuted according to a permutation known by Bob. If Alice and Bob could get random shares $\mathbf{R}_A$ and $\mathbf{R}_B$ such that $\mathbf{R} = \mathbf{R}_A + \mathbf{R}_B$, securely computing $f(x, y)$ would be quite easy: Alice selects which column should be used, Bob which row, and then they simply sum their shares of that element in order to obtain the result. Although this one-time table technique does not scale well with the inputs' size, it illustrates how useful correlated data can be for cryptography. In this thesis we will explore the topic of cryptography based on correlated data both from a theoretical as well as a practical point of view. Our focus is on unconditional security (also known as information-theoretical security), in which the security guarantees should hold even against computationally unbounded adversaries; this in contrast with computational security, in which the adversaries are restricted to be probabilistic polynomial time Turing machines and computational hardness assumptions are necessary.

First we study physical assumptions that can be used to obtain simple forms of correlated data suitable for cryptographic purposes: we consider the scenario where noisy channels are available between the parties as well as the Bounded Storage Model, in which the memory of the parties are bounded. We target at two quintessential cryptographic primitives, namely commitment and oblivious transfer, and investigate the question of their existence as well as the theoretical limits of how efficiently the underlying resources can be used to achieve these primitives.

The second part of the thesis assumes that more structured forms of correlated data are available by using the commodity based model. The crucial question of obtaining highly practical and parallelizable protocols for the secure computation of some important problems such as machine learning training and classification is addressed.

This chapter presents an overview about the topics that will be discussed during the thesis. A comparison with the most relevant works will be present in each individual chapter. The outline of this chapter is as follows: we will first discuss the commitment and oblivious transfer primitives. Then we explain the physical assumptions we considered: noisy channels and the Bounded Storage Model. Finally, we present some considerations about practical secure computation as well as specific cases of secure machine learning.

Commitment Schemes

Blum [Blu83] introduced commitment schemes. They are one of the most essential primitives in modern cryptography and are widely used in applications such as contract signing [EGL85], identification protocols [FS87], zero-knowledge proofs [GMW91, Gol01, BCC88], and more generally in two-party and multi-party computation protocols [GMW87, CDv88, CCD88]. Intuitively, commitment protocols have a role in the digital world that is similar to that of sealed envelopes in first-price sealed-bid auctions. In such auctions all the bidders first place their bids in sealed envelopes which in the end are opened to determine the winner and the price. The sealed envelopes have a dual role in that type of auction: on one hand, they should keep the secrecy of the bids during the bidding process; on the other hand, they should stop the winner from changing the final price. A commitment scheme is a two-phase protocol between two mutually distrustful parties, Alice and Bob. First they execute the commitment phase, in which Alice chooses a message m and commits to it. At any time afterwards, Alice can decide to execute the opening phase in order to reveal m to Bob. Similarly to the sealed envelopes, a commitment scheme needs to meet two security properties: hiding, which guarantees that Bob cannot learn any information about m before the opening phase; and binding, which guarantees that Alice cannot change the committed value m without Bob detecting it.

In the context of computational security, commitment protocols can be designed based on generic assumptions such as the existence of pseudorandom generators [Nao91] or more efficiently based on the hardness of various specific computational problems [Eve81, Blu83, Ped92]. On the other hand, if unconditional security is desired and no setup or physical assumption is made, then commitment is impossible to obtain. This work investigates solutions based on the existence of noisy channels between the parties as well as in the Bounded Storage Model.

Oblivious Transfer

Oblivious transfer (OT) is another two-party primitive that is fundamental for two-party and multi-party computation. Alice has as input two strings s_0, s_1 and Bob a choice bit c, and Bob learns the string s_c. The protocol is secure for Alice if Bob cannot learn any information about s_{1-c}, and it is secure for Bob if Alice cannot learn the choice bit c. The usefulness of OT comes from the fact that it breaks the symmetry of (correlated) information knowledge between the parties, i.e., Alice and Bob get data which is correlated, but not the same. Indeed OT, despite its very simplistic appearance, is a very powerful primitive and is complete for two-party and multi-party computation [GMW87, Kil88, CvT95, IPS08], i.e., given any secure implementation of OT it is possible to obtain, without any additional assumption, secure two-party and multi-party computation protocols to evaluate any polynomial time computable function.

Given its power, it is no surprise that OT has been the subject of much research by cryptographers. In the context of computational security it can be obtained from dense trapdoor permutations [Hai04] or assuming the hardness of many specific computational problems [Rab81, BM90, Kal05, PVW08, DvdGMN08, DvdGMQN12, DDN14]. Like commitment schemes, unconditionally secure OT cannot be obtained if no setup or physical assumption is made. Nonetheless, it is possible to obtain unconditionally secure OT protocols if either there are noisy channels between the parties or the memory of the parties are bounded. Both scenarios will be studied in this work.

Cryptography based on Noisy Channels

The cryptographic usefulness of noisy channels was first noticed by Wyner [Wyn75], who proposed a scheme for exchanging a secret-key in the presence of an eavesdropper who receives the transmitted symbols over a degraded channel with respect to the legitimate receiver's channel. Csiszár and Körner [CK78] extended the possibility result to the class of general (non-degraded) broadcast channels. Maurer [Mau93] later pointed out that public communication can improve the participants' ability to generate a secret. In the case of commitment and oblivious transfer protocols, the first schemes based on noisy channels were developed by Crépeau and Kilian [CK88]. The efficiency of these solutions were largely improved by Crépeau [Cré97] and the topic was further studied both from the theoretical as well as the efficient protocol designing points of view by many subsequent works [DKS99, KM01, SW02, WNI03, CMW05, IMN06, AC07, NW08, PDMN11].

Commitment and OT Capacities

Given the importance of the commitment and OT primitives and the exceptional value of noisy channels for cryptographic purposes, researchers started to investigate the questions of which channels can be used to implement these primitives and what is the optimal rate in which they can be used to implement these primitives. Commitment and oblivious transfer capacities were defined and are the cryptographic equivalents for these primitives of the Shannon capacity for information transmission. In the case of commitment capacity, which was first defined by Winter et al. [WNI03], this amounts to determining the optimal ratio between the length of the committed values and the number of uses of the noisy channel. Winter et al. [WNI03] characterized the commitment capacity of discrete memoryless channels. Afterwards, Nascimento et al. [NBSI08] determined the commitment capacity of Gaussian channels. In the case of oblivious transfer capacity, which was first proposed by Nascimento and Winter [NW08], this amounts to determining the optimal ratio between the length of the strings in the OT protocol and the number of uses of the noisy channel. Nascimento and Winter [NW08] identified some noise resources that have strictly positive OT capacity. Imai et al. [IMN06] obtained the OT capacity of erasure channels against honest-but-curious adversaries (i.e., adversaries which always follow the protocol instructions but try to learn additional information) and a lower bound on its OT capacity against malicious adversaries (which can arbitrarily deviate from the protocol). Ahlswede and Csiszár [AC07, AC13] showed new bounds for the OT capacity of Generalized Erasure Channels (GEC) against honest-but-curious adversaries, which were partially extended to the malicious case by Pinto et al. [PDMN11]. We should also mention that the question of determining the optimal way of using noisy channels was also studied for other cryptographic tasks, for instance in the vast literature on secrecy capacity [Wyn75, CK78, LYCH78, OW85, Mau93, AC93, CN04, PB05, BR06, LYT10, LPS07, GLEG08, CN08, AFJK09, BMK09, EU11, OH11].

In Chapter 3 we extend to the case of malicious adversaries the remaining bounds of Ahlswede and Csiszár [AC07, AC13] on the OT capacity of Generalized Erasure Channels and in Chapter 4 we determine the commitment capacity of Unfair Noisy Channels (UNC).

Bounded Storage Model

In this work we also consider the Bounded Storage Model (BSM) [Mau92]. In this model, the storage capacity of the (dishonest) participants is bounded, instead of the usual bound on the computational power that is used in cryptography based on complexity theory. It also assumed that the parties have access to a public random string during an initial transmission phase. This string can be obtained from a natural source, from a trusted third party, or, in some cases even generated by one of the parties. One appealing feature of the BSM is that the security is unconditional and holds even if the parties get infinite storage capacity after the transmission phase. Another interesting property is that in the BSM no additional assumption needs to be made; this is in strong contrast with the case of bounds on the computational power, in which case computational hardness assumptions are also necessary.

Cachin and Maurer [CM97] proposed key agreement protocols in the Bounded Storage Model both with and without a small pre-shared key between the parties. If the public random source has size ℓ and the pre-shared key has size $O(\log \ell)$, then it can be used to select bits from the source. If there is no pre-shared key is available, then the parties need $O(\sqrt{\ell})$ samples from the source, thus making the protocol less practical. Dziembowski and Maurer [DM08] later proved that this last bound is optimal, i.e., any key agreement by public discussion protocol requires $O(\sqrt{\ell})$ samples from the parties.

Cachin et al. [CCM98] presented the first OT protocol in the BSM. Ding [Din01] and Hong et al. [HCR02] presented improvements to that protocol in a slightly different model. Finally, Ding et al. [DHRS04] obtained the first constant-round OT protocol. In the case of commitment, Shikata and Yamanaka [SY11] and independently Alves [Alv10] studied the problem of commitment in the BSM and provided solutions that were based on the work of Ding et al. [DHRS04].

A weakness of the BSM is that it assumes that the random source can be reliably broadcasted to all parties without transmission errors, which is hard to realize in practice. In this work we considered a more realistic variant, the so called Bounded Storage Model with Errors, in which errors can be introduced in the public random source in arbitrary positions; it is only assumed that the error frequency is not too large. This model captures both the situation in which the source is partially controlled by an adversary as well as errors due to noise in the channel. Ding [Din05] previously studied this model and obtained secret key agreement protocols. In Chapter 5 we present the first commitment and OT protocols in this model.

Practical Secure Computation

Secure computation is a very important topic in modern cryptography and deals with the problem of two or more mutually distrustful parties that want to make computations over their data without leaking any information other than the specified output of the desired functionality. Despite the apparent complexity of the problem, general solutions meeting different security notions were proposed decades ago for both the two-party case as well as the multi-party, for example, [Yao82, GMW87, CCD88, Kil88, CvT95]. There are solutions evaluating either boolean or arithmetic circuits, and achieving either computational or unconditional security.

The ongoing research effort into obtaining more efficient secure computation follows approaches such as optimizing Yao's garbled circuits technique [Yao82] for two-party computation [FJN$^+$13, Lin13, LR14], evaluating RAM programs [KSS13, WHC$^+$14, AHMR15], optimizing OT-based protocols [NNOB12, SZ13, ALSZ15] and constructing protocols in the preprocessing model [DPSZ12, DKL$^+$13, DSZ15]. Despite all the progress on the performance of general multiparty computation protocols, when compared to multi-party computation protocols that are tailored to one specific functionality, the general protocols still pay a high price in terms of efficiency in order to achieve generality. These protocols typically require the function to be represented by an arithmetic circuit [BDOZ11, DPSZ12, DKL$^+$13] or a boolean circuit [NNOB12, FJN$^+$13, Lin13, LR14], and the circuits need to be evaluated gate by gate; thus introducing an overhead proportional to the number of gates. In addition, protocols that are secure against malicious adversaries have an extra overhead due to the mechanisms that are employed in order to verify that the parties are following the protocol instructions: such as message authentication codes [BDOZ11, DPSZ12, DKL$^+$13] and circuit validation techniques [FJN$^+$13, Lin13, LR14]; the protocols attaining the best performance [SZ13, DSZ15] are only secure against honest-but-curious adversaries.

Given the improvements that can be reached by using tailor-made protocols, this approach was pursued for many important functionalities, such as scalar products [GLLM05], means [KLML05], statistics [BSMD10], equality [DFK$^+$06], comparison [DFK$^+$06] and exponentiations [DFK$^+$06]. In this work we focus on the design of highly practical and parallelizable protocols for particular functionalities. All protocols obtained in the second part of this thesis achieve unconditional security and work in the commodity-based model, in which a trusted initializer (TI) distributes correlated randomness for the parties during a setup phase but does not engage in the protocol afterwards. In our protocols both the TI and the parties only have to perform simple operations over a finite field; such operations are simple enough to be executed in resource constrained environments (e.g. embedded computers) while being embarrassingly parallelizable for simultaneous execution of several protocol instances in large scale environments (e.g. big data applications). We particularly focus on protocols for secure machine learning.

Privacy-Preserving Learning

Traditional machine learning methods usually require all the training dataset to be directly available to the learning algorithm. However, increasingly often the training data that is useful for deriving a model is distributed among multiple parties that cannot or will not share their data due to economic reasons or privacy legislation. Therefore privacy-preserving learning algorithms, which allow the parties to learn a model without leaking any additional information about the training dataset, are growing in importance. In this thesis we deal with the problem of designing a privacy-preserving protocol for performing linear regression over a dataset that is distributed over multiple parties. We provide security definitions, a protocol, and security proofs. Our solution, which is presented in Chapter 6, is information-theoretically secure and works in the commodity-based model.

There were many attempts in the literature at obtaining secure linear regression protocols over distributed databases, but most of the works do not even aim at obtaining the level of privacy usually required by modern cryptographic protocols (such as Karr et al. [KLSR05] and Du et al.[DHC04], see also [SKLR04, KLSR09]). Hall et al. [HFN11] proposed a protocol aiming at a strong notion of security. They used the framework of secure two-party protocols and simulation based definitions of security from Goldreich [Gol04]. We would like to point out that as some of their protocols rely on function approximations, rather than exact computations, they should have considered the framework of Feigenbaum et al. [FIM$^+$01, FIM$^+$06]. The truncation protocol also has a small (correctable) problem as explained in [CDNN15]. Nikolaenko et al. [NWI$^+$13] proposed a solution based on homomorphic encryption and garbled circuits for a different scenario in which the multiple parties encrypt the training data and upload the ciphertexts to a third party. This party computes the regression model with the help from a semi-honest Crypto Service Provider that performs the heavy cryptographic operations. The Crypto Service Provider is assumed to not collude with other parties and actively engages in the protocol during its execution. This contrasts with our solution, in which the trusted initializer does not engage in the protocol execution after the setup phase. Our online phase is far faster than the previous protocols. If a trusted initializer is not available or desirable, the parties can run an offline phase, which is only computationally secure, in order to generate the required correlated data. Even in this case, our total time is still smaller due to the fact that our solution is extremely parallelizable.

Privacy-Preserving Classifiers

Machine learning classifiers have a great potential for improving our daily lives and can be used in many scenarios: by healthcare providers to diagnose patients, by search engines and recommendation platforms to produce more accurate results, by wearable devices for making personal health recommendations, by websites to decide the contents to be shown to each particular user, and so on. But if this classification is done in the clear, either the user Alice has to reveal her data or the model owner Bob has to reveal his model, and both options are not satisfactory as both Alice's data as well as Bob's model can contain sensitive information. In this thesis we will deal with the problem of performing the classification in a such way that Bob learns nothing about Alice's data, and Alice learns as little as possible about Bob's model.

In the past solutions were given for a weaker security model in which Alice learns the classification model [BLN13] or for very specific classifiers with limited applications [AB06, AB07, EFG$^+$09, SSW10, BFK$^+$09, BFL$^+$09, BFL$^+$11]. General privacy-preserving classifiers were proposed just recently by Bost et al. [BPTG15, BPTG14] for the case of hyperplane-based classifiers, Naive Bayes and decision trees and by Wu et al. [WFNL15] for decision trees and random forests. Both works relied on the Paillier encryption scheme as well as other computationally secure building blocks. In Chapter 7 we present more efficient protocols for evaluating these classifiers which enjoy unconditional security.

Outline

Chapter 2 presents our notation and the background knowledge and lemmas that are used in the subsequent chapters. In Chapter 3 we establish lower bounds on the oblivious transfer capacity against malicious adversaries of generalized erasure channels with low erasure probability. In Chapter 4 we determine the commitment capacity of Unfair Noisy Channels. Chapter 5 introduces the first commitment and oblivious transfer protocols in the Bounded Storage Model with Errors. Chapter 6 presents our protocol for privacy-preserving learning using linear regression. In Chapter 7 we present some privacy-preserving classifiers. Finally, Chapter 8 contains our final remarks.

2. Preliminaries

In this chapter we present our notation and model as well as supporting theorems, lemmas and definitions that are used in the subsequent chapters.

2.1 Notation

Calligraphic letters are used for denoting domains of random variables and other sets, upper case letters for random variables and lower case letters for realizations of the random variables. The cardinality of a set $\mathcal{X}$ is written as $|\mathcal{X}|$, the set $\{1, \ldots, \ell\}$ as $[\ell]$ and the set of all subsets $\mathcal{S} \subseteq [\ell]$ with $|\mathcal{S}| = t$ as $\binom{[\ell]}{t}$. For a tuple $X^\ell = (X_1, X_2, \ldots, X_\ell)$ and a tuple $\mathcal{R}$ of non-repeated elements of $[\ell]$, $X^\mathcal{R}$ is the restriction of X^ℓ to the positions specified by $\mathcal{R}$. We denote by $\mathbb{Z}_q$ the ring of order q and by $\mathbb{Z}_q^\ell$ the space of all ℓ-tuples of elements of $\mathbb{Z}_q$. $\mathbb{Z}_q^{\ell_1 \times \ell_2}$ represents the space of all $\ell_1 \times \ell_2$ matrices with elements belonging to $\mathbb{Z}_q$. Similar notation $\mathbb{F}_q$, $\mathbb{F}_q^\ell$ and $\mathbb{F}_q^{\ell_1 \times \ell_2}$ is used for a finite field if the additional mathematical properties are emphasized.

For a random variable X over $\mathcal{X}$, $P_X : \mathcal{X} \to [0,1]$ with $\sum_{x \in \mathcal{X}} P_X(x) = 1$ will denote its probability distribution. For a joint probability distribution $P_{XY} : \mathcal{X} \times \mathcal{Y} \to [0,1]$, $P_X(x) := \sum_{y \in \mathcal{Y}} P_{XY}(x,y)$ denotes the marginal probability distribution and $P_{X|Y}(x|y) := P_{XY}(x,y)/P_Y(y)$ the conditional probability distribution if $P_Y(y) \neq 0$. The statistical distance between two probability distributions is defined as follows:

Definition 2.1 (Statistical distance) *The* statistical distance $\|P_X - P_Y\|$ *between two probability distributions P_X and P_Y with alphabet $\mathcal{X}$ is*

$$\|P_X - P_Y\| = \max_{\mathcal{S} \subseteq \mathcal{X}} \left| \sum_{x \in \mathcal{S}} P_X(x) - P_Y(x) \right| .$$

We say P_X and P_Y are ε-close if $\|P_X - P_Y\| \leq \varepsilon$.

A function $\varepsilon(\cdot)$ is negligible in the security parameter n if it is asymptotically smaller than the inverse of any fixed polynomial in n. Two sequences $X(n)$ and $Y(n)$ of random variables are said to be *statistically close*, denoted by $X \overset{s}{\approx} Y$, if there exists a negligible function $\varepsilon(\cdot)$ such that for every $n \in \mathbb{N}$, $\|P_{X(n)} - P_{Y(n)}\| \leq \varepsilon(n)$. They are said to be *computationally indistinguishable*, denoted by $X \overset{c}{\approx} Y$, if there exists a negligible function $\varepsilon(\cdot)$ such that for every $n \in \mathbb{N}$ and for every non-uniform probabilistic polynomial time distinguisher D it holds that

$$|\Pr[\,D(X(n)) = 1\,] - \Pr[\,D(Y(n)) = 1\,]| \leq \varepsilon(n).$$

A sequence $E(n)$ of events happens with overwhelming probability in the security parameter n if

$$\Pr[\,E(n)\,] \geq 1 - \varepsilon(n),$$

where $\varepsilon(\cdot)$ is a negligible function of n.

$x \overset{\$}{\leftarrow} \mathcal{X}$ denotes choosing an element x uniformly at random over $\mathcal{X}$ and U_r a vector uniformly chosen from $\{0,1\}^r$. $y \overset{\$}{\leftarrow} F(x)$ denotes the act of running the probabilistic algorithm F with input x and obtaining the output y. If the randomness needs to be made explicit, we use the notation $F(x; r)$ where r is the randomness. $y \leftarrow F(x)$ is similarly used for deterministic algorithms.

We use additively secret sharings to perform computation over a ring $\mathbb{Z}_q$. For a value x that is randomly shared with parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$ using shares over a ring $\mathbb{Z}_q$, each party $\mathcal{P}_i$ gets an uniformly random $x_i \in \mathbb{Z}_q$ subject to the constraint that $x = \sum_{i=1}^{u} x_i$, where the operations are in the ring. Let $[\![x]\!]_q$ denote the resulting secret sharing and $[\![x]\!]_q \overset{\$}{\leftarrow} x$ the operation of creating and distributing the shares, which can be executed either by one of the parties $\mathcal{P}_i$ or by an external participant. In order to unify the treatment of the protocols with the case in which one input x is held by a single party $\mathcal{P}_i$, we write $[\![x]\!]_q \leftarrow x$ to denote the sharing in which $\mathcal{P}_i$ (which is always clear from the context) computes with the share x and the remaining parties with shares equal to zero. Given $[\![x]\!]_q$, $[\![y]\!]_q$ and a constant c, it is trivial for the parties to compute a secret sharing $[\![z]\!]_q$ corresponding to $z = x + y$, $z = x - y$, $z = cx$ or $z = x + c$. All these operations can be performed locally by the parties without any interaction by simply adding, subtracting or multiplying the shares respectively for the first three cases, and by having a pre-agreed party adding the constant in the last case. These operations will be denoted respectively by $[\![z]\!]_q \leftarrow [\![x]\!]_q + [\![y]\!]_q$, $[\![z]\!]_q \leftarrow [\![x]\!]_q - [\![y]\!]_q$, $[\![z]\!]_q \leftarrow c[\![x]\!]_q$ and $[\![z]\!]_q \leftarrow [\![x]\!]_q + c$. For a secret sharing $[\![x]\!]_q$, the parties can open the value x by revealing their shares x_i. Extending this for a vector $\mathbf{x}$ (respectively for a matrix $\mathbf{X}$), $[\![\mathbf{x}]\!]_q$ (respectively $[\![\mathbf{X}]\!]_q$) will denote the element-wise secret sharing and the notation for the operations will be similar to the one above.

Let $\log x$ denote the logarithm of x in base 2. The binary entropy function is denoted by h: for $0 \leq x \leq 1$, $h(x) = -x \log x - (1-x)\log(1-x)$. By convention, $0 \log 0 = 0$. $H(X)$ denotes the entropy of X and $I(X;Y)$ the mutual information between X and Y. If x and y are strings, $x \oplus y$ denotes their bitwise exclusive-or, $x \,\|\, y$ their concatenation. If additionally they have the same length, $\mathsf{HD}(x, y)$ denotes their Hamming distance, that is, the number of positions in which they differ.

2.2 Entropy Measures

The main entropy measure used in this work is the *min-entropy* as its conditional version captures the private randomness that can be extracted from a random variable X given a correlated random variable Y that an adversary knows.

Definition 2.2 (Min-entropy) *Let P_{XY} be a probability distribution over $\mathcal{X} \times \mathcal{Y}$. The* min-entropy of X*, denoted by $H_\infty(X)$, and the* conditional min-entropy of X given Y*, denoted by $H_\infty(X|Y)$, are respectively defined as*

$$\begin{aligned} H_\infty(X) &= \min_{x\in\mathcal{X}}(-\log P_X(x)) \\ H_\infty(X|Y) &= \min_{y\in\mathcal{Y}} \min_{x\in\mathcal{X}}(-\log P_{X|Y=y}(x)). \end{aligned}$$

X is called a κ-source *if $H_\infty(X) \geq \kappa$.*

The min-entropy has the problem of being sensitive to small changes in the probability distribution and for this reason its *smooth* version [RW05] will be used instead. Intuitively, the smooth min-entropy is the maximum min-entropy in the neighborhood of the probability distribution. The smooth min-entropy is defined as follows.

Definition 2.3 (Smooth min-entropy) *Let $\varepsilon > 0$ and P_{XY} be a probability distribution. The* ε-smooth min-entropy of X given Y *is defined by*

$$H_\infty^\varepsilon(X|Y) = \max_{X'Y':\|P_{X'Y'}-P_{XY}\|\leq\varepsilon} H_\infty(X'|Y')$$

Similarly, we also define the max-entropy and its smooth version.

Definition 2.4 ((Smooth) Max-entropy) *The max-entropy is defined as*

$$H_0(X) = \log|\{x \in X|P_X(x) > 0\}|$$

and its conditional version is given by

$$H_0(X|Y) = \max_y H_0(X|Y = y).$$

The smooth variants are defined as

$$H_0^\varepsilon(X) = \min_{X':\|P_{X'}-P_X\|\leq\varepsilon} H_0(X'),$$

$$H_0^\varepsilon(X|Y) = \min_{X'Y':\|P_{X'Y'}-P_{XY}\|\leq\varepsilon} H_0(X'|Y').$$

The notion of *min-entropy rate* and a few results regarding its preservation are also be used in this work.

Definition 2.5 (Min-entropy rate) *Let X be a random variable with an alphabet $\mathcal{X}$, Y be an arbitrary random variable, and $\varepsilon \geq 0$. The* min-entropy rate $R_\infty^\varepsilon(X|Y)$ *is defined as*

$$R_\infty^\varepsilon(X|Y) = \frac{H_\infty^\varepsilon(X|Y)}{\log|\mathcal{X}|}.$$

The following lemma says that a source with high min-entropy also has high min-entropy when conditioned on a correlated short string and is a restatement of a lemma in Ding et al. [DHRS04]. This lemma makes the Bounded Storage Model interesting as it implies that a memory-bounded adversary has limited information about the public random string.

Lemma 2.6 *Let $X \in \{0,1\}^\ell$ be such that $R^\varepsilon_\infty(X) \geq \alpha$ and Y be a random variable over $\{0,1\}^{\gamma\ell}$. Fix $\varepsilon' > 0$. Then*

$$R^{\varepsilon'+\sqrt{8\varepsilon}}_\infty(X|Y) \geq \alpha - \gamma - \frac{1+\log(1/\varepsilon')}{\ell}.$$

Proof: Let $\rho = \alpha - \gamma - \frac{1+\log(1/\varepsilon')}{\ell}$. By lemma 3.16 in Ding et al. [DHRS04] we have that if $R^\varepsilon_\infty(X) \geq \alpha$ then

$$\Pr_{y \overset{\$}{\leftarrow} Y}\left[R^{\sqrt{2\varepsilon}}_\infty(X|Y=y) \geq \rho\right] \geq 1 - \varepsilon' - \sqrt{2\varepsilon}.$$

To get the desired result, let $\mathcal{G} = \{y \in \mathcal{Y} | R^{\sqrt{2\varepsilon}}_\infty(X|Y=y) \geq \rho\}$ and P_{XY} be the joint probability distribution of X and Y. Let P'_{XY} be the distribution that is $\sqrt{2\varepsilon}$-close to P_{XY} and is such that $P'(X=x|Y=y) \leq 2^{-\rho\ell}$ for any $x \in \mathcal{X}, y \in \mathcal{G}$. Let P''_{XY} be obtained by letting $P''(X|Y=y) = P'(X|Y=y)$ for $y \in \mathcal{G}$ and defining $P''(X=x|Y=y) = 2^{-\ell}$ for any $x \in \mathcal{X}$, $y \notin \mathcal{G}$. As $\Pr[\mathcal{G}] \geq 1 - \varepsilon' - \sqrt{2\varepsilon}$, it holds that $\|P''_{XY} - P'_{XY}\| \leq \varepsilon' + \sqrt{2\varepsilon}$ and so $\|P''_{XY} - P_{XY}\| \leq \varepsilon' + 2\sqrt{2\varepsilon}$. Since $P''(X=x|Y=y) \leq 2^{-\rho\ell}$ for every $x \in \mathcal{X}, y \in \mathcal{Y}$, the lemma follows. ▌

The following smooth entropy analogues of the chain rule for conditional Shannon entropy are due to Renner and Wolf [RW05].

Lemma 2.7 ([RW05]) *Let $\varepsilon, \varepsilon', \varepsilon_1, \varepsilon_2 \geq 0$ and P_{XYZ} be a tripartite probability distribution. Then*

$$\begin{aligned} H^{\varepsilon+\varepsilon'}_\infty(XY|Z) - H^{\varepsilon'}_\infty(Y|Z) &\geq H^\varepsilon_\infty(X|YZ) \\ &\geq H^{\varepsilon_1}_\infty(XY|Z) - H^{\varepsilon_2}_0(Y|Z) - \log(1/(\varepsilon - \varepsilon_1 - \varepsilon_2)) \end{aligned}$$

and

$$\begin{aligned} H^{\varepsilon+\varepsilon'}_0(XY|Z) - H^{\varepsilon'}_0(Y|Z) &\leq H^\varepsilon_0(X|YZ) \\ &\leq H^{\varepsilon_1}_0(XY|Z) - H^{\varepsilon_2}_\infty(Y|Z) + \log(1/(\varepsilon - \varepsilon_1 - \varepsilon_2)). \end{aligned}$$

2.3 Averaging Samplers and Randomness Extractors

In the Bounded Storage Model, due to the assumption that it is infeasible to store the whole source string, it is not possible to apply an extractor to the complete string; the extractor needs to be locally computable [Vad04]. A typical approach for using the source is the sample-then-extract paradigm: first some positions of the source are sampled and then an extractor is applied on these positions. In this context, *averaging samplers* [BR94, CEG95, Zuc97] are a fundamental tool that intuitively produce samples such that, for any function, its average value taken over the sampled string is roughly the same as the average when taken over the original string.

Definition 2.8 (Averaging sampler) *A function* $\mathsf{Samp}\colon \{0,1\}^r \to [\ell]^t$ *is an* (μ, ν, ε)-averaging sampler *if for every function* $f\colon [\ell] \to [0,1]$ *with average* $\frac{\sum_{i=1}^{\ell} f(i)}{\ell} \geq \mu$ *it holds that*

$$\Pr_{\mathcal{S} \overset{\$}{\leftarrow} \mathsf{Samp}(U_r)}\left[\frac{1}{t}\sum_{i\in\mathcal{S}} f(i) \leq \mu - \nu\right] \leq \varepsilon. \tag{2.1}$$

Among the several useful properties enjoyed by averaging samplers, of particular importance for us is the fact that averaging samplers roughly preserve the *min-entropy rate.*

Lemma 2.9 ([Vad04]) *Let $X \in \{0,1\}^\ell$ be such that $R_\infty(X|Y) \geq \rho$. Let τ be such that $1 \geq \rho \geq 3\tau > 0$ and $\mathsf{Samp}\colon \{0,1\}^r \to [\ell]^t$ be an (μ, ν, ε)-averaging sampler with distinct samples for $\mu = (\rho - 2\tau)/\log(1/\tau)$ and $\nu = \tau/\log(1/\tau)$. Then for $\mathcal{S} \overset{\$}{\leftarrow} \mathsf{Samp}(U_r)$*

$$R_\infty^{\varepsilon'}(X^{\mathcal{S}}|\mathcal{S}, Y) \geq \rho - 3\tau,$$

where $\varepsilon' = \varepsilon + 2^{-\Omega(\tau\ell)}$.

For $t < \ell$, the uniform distribution over subsets of $[\ell]$ of size t is an averaging sampler, also called the (ℓ, t)-*random subset sampler.* The following lemma is just a restatement of Lemma 5.5 from Babai and Hayes [BH05]

Lemma 2.10 *Let $0 < t < \ell$. For any $\mu, \nu > 0$, the (ℓ, t)-random subset sampler is a $(\mu, \nu, e^{-t\nu^2/2})$-averaging sampler.*

A *strong extractor* [NZ96, DRS04, DORS08] is a function that takes as input a string with high min-entropy and outputs a string that is statistically close to an uniformly distributed string.

Definition 2.11 (Strong extractor) *A function $\mathsf{Ext}\colon \{0,1\}^\ell \times \{0,1\}^r \to \{0,1\}^m$ is a (κ, ε)-*strong extractor *if for every κ-source $X \in \{0,1\}^\ell$, and for random variables R and M uniformly distributed in the bit-strings of length r and m, respectively, we have*

$$\|P_{\mathsf{Ext}(X,R),R} - P_{M,R}\| \leq \varepsilon.$$

The following lemma from Zuckerman [Zuc97] specifies the parameters of an explicit strong extractor construction.

Lemma 2.12 ([Zuc97]) *Let $\beta, \psi > 0$ be arbitrary constants. For every $\ell \in \mathbb{N}$ and every $\varepsilon > e^{-\ell/2^{O(\log^* \ell)}}$, there is an explicit construction of a $(\beta\ell, \varepsilon)$-strong extractor $\mathsf{Ext}\colon \{0,1\}^\ell \times \{0,1\}^r \to \{0,1\}^m$ with $m = (1-\psi)\beta\ell$ and $r = O(\log \ell + \log(1/\varepsilon))$.*

Universal Hash Functions [CW79] are strong extractors and can extract the optimal number of nearly random bits [RTS00] according to the Leftover-Hash Lemma (similarly the Privacy-Amplification Lemma) [BBR88, ILL89, BBCM95, HILL99].

Definition 2.13 (t-universal hash functions) *A family of functions $G = \{g\colon \mathcal{H} \to \mathcal{L}\}$ is called a family of t-universal hash functions if for $g \overset{\$}{\leftarrow} G$ and for any $x_1, \ldots, x_t \in \mathcal{H}$, the induced distribution on $(g(x_1), \ldots, g(x_t))$ is uniform over $\mathcal{L}^t$.*

For any $\mathcal{H} = \{0,1\}^h$ and $\mathcal{L} = \{0,1\}^\ell$, there exists a t-universal family of hash functions for which the function description has size *poly*(h,t) bits, and the sampling and computing times are in *poly*(h,t).

Lemma 2.14 *Let $\mathcal{G}$ be a 2-universal class of functions $g : \{0,1\}^\ell \to \{0,1\}^m$. Then for G uniformly random in $\mathcal{G}$ and a κ-source $X \in \{0,1\}^\ell$ we have that*

$$\|P_{G(X),G}, P_{U_m,G}\| \leq \frac{1}{2}\sqrt{2^{-\kappa+m}}.$$

*In particular, it is a $(\kappa, \varepsilon)-$*strong extractor *when $m \leq \kappa - 2\log(\varepsilon^{-1}) + 2$.*

One possible method for obtaining universal hash functions is the multiplication by a random matrix of appropriate size.

The noise-resilient variant of strong extractors, the so called *fuzzy extractors* [DRS04], will also be used in this work. They enable any party with a string that is close enough in the Hamming distance metric to the original source to reproduce the extracted string.

Definition 2.15 (Fuzzy extractor) *A pair of functions* $\mathsf{Ext}\colon \{0,1\}^\ell \times \{0,1\}^r \to \{0,1\}^m \times \{0,1\}^q$, $\mathsf{Rec}\colon \{0,1\}^\ell \times \{0,1\}^r \times \{0,1\}^q \to \{0,1\}^m$ *is an* $(\kappa, \varepsilon, \delta, \beta)$-fuzzy extractor *if:*

– (Security) *For every* κ*-source* $X \in \{0,1\}^\ell$ *and for random variables* R *and* M *uniformly distributed in the bit-strings of length* r *and* m*, let* $(Y, Q) \leftarrow \mathsf{Ext}(X, R)$*. Then* $\|P_{YRQ} - P_{MRQ}\| \le \varepsilon$*.*

– (Recovery) *For every* $x, x' \in \{0,1\}^\ell$ *such that* $\mathsf{HD}(x, x') \le \delta\ell$*, let* $r \overset{\$}{\leftarrow} U_r$*,* $(y, q) \leftarrow \mathsf{Ext}(x, r)$*. Then it should hold that* $\Pr[\,\mathsf{Rec}(x', r, q) = y\,] \ge 1 - \beta$*.*

Fuzzy extractors are a special case of *one-way key-agreement schemes* [HR05, KR09] and ultimately equivalent to performing information reconciliation followed by privacy amplification [RW04]. Due to the restriction to close strings with respect to the Hamming distance, syndrome-based fuzzy extractors can be used, as summarized in Ding's lemma [Din05].

Lemma 2.16 ([Din05]) *Let* $1 \ge \beta, \psi > 0$ *and* $1/4 > \delta > 0$ *be arbitrary constants. There is a constant* σ*, depending on* δ*, such that for every sufficiently large* $\ell \in \mathbb{N}$*, and every* $\varepsilon > e^{-\ell/2^{O(\log^* \ell)}}$*, there exists an explicit construction of a* $(\beta\ell, \varepsilon, \delta, 0)$*-fuzzy extractor* $(\mathsf{Ext}, \mathsf{Rec})$*, where* Ext *is of the form* $\mathsf{Ext}\colon \{0,1\}^\ell \times \{0,1\}^r \to \{0,1\}^m \times \{0,1\}^q$ *with*

$$\begin{aligned} m &= (1-\psi)\beta\ell, \\ r &= O\left(\log \ell + \log \varepsilon^{-1}\right), \\ q &\le \frac{1-\sigma}{(1-\psi)\beta} m. \end{aligned}$$

Remark 2.17 *The code used to correct the errors has code size* ℓ *with rate* σ *and can correct* $\delta\ell$ *errors. According to the Gilbert-Varshamov bound, for a given* υ *with* $0 < \upsilon < 1/2$ *and* $0 \le \xi \le 1 - h(\upsilon)$*, there exists a random linear code with minimum distance* $\upsilon\ell$ *and* $\sigma \ge 1 - h(\upsilon) - \xi$*; however this construction has no efficient decoding. One alternative solution is to use the concatenated solution in Theorem 4 of Guruswami and Indyk [GI02], which achieves the Zyablov bound and provides a code with linear-time encoding and decoding that, for a given* $0 < \sigma < 1$ *and* $\xi > 0$*, can correct* $\delta\ell$ *errors, where*

$$\delta \ge \max_{\sigma < \widetilde{\sigma} < 1} \frac{(1 - \widetilde{\sigma} - \xi) y}{2}$$

and y *is the unique number in* $[0, 1/2]$ *with* $h(y) = 1 - \sigma/\widetilde{\sigma}$*.*

2.4 Typical Sequences

In this section we define the concept of typical sequences largely following the book of Csiszár and Körner [CK82].

Definition 2.18 *For a probability distribution P_X on $\mathcal{X}$ and $\varepsilon > 0$ the* ε-typical sequences *form the set*

$$\mathcal{T}^{\ell}_{P_X,\varepsilon} = \{x^{\ell} \in \mathcal{X}^{\ell} : \forall x \in \mathcal{X} \ |N(x|x^{\ell}) - \ell P_X(x)| \leq \varepsilon\ell \text{ and } P_X(x) = 0 \Rightarrow N(x|x^{\ell}) = 0\},$$

with the number $N(x|x^{\ell})$ denoting the number of symbols x in the string x^{ℓ}.

The *type* of x^{ℓ} is the probability distribution $P_{x^{\ell}}(x) = \frac{1}{\ell}N(x|x^{\ell})$. Then, $x^{\ell} \in \mathcal{T}^{\ell}_{P_X,\varepsilon} \Rightarrow |P_{x^{\ell}}(x) - P_X(x)| \leq \varepsilon, \forall x \in \mathcal{X}$.

Properties 2.19

1. $P_X^{\otimes \ell}(\mathcal{T}^{\ell}_{P_X,\varepsilon}) \geq 1 - 2|\mathcal{X}| \exp(-\ell\varepsilon^2/2)$.
2. $|\mathcal{T}^{\ell}_{P_X,\varepsilon}| \leq \exp(\ell H(P_X) + \ell\varepsilon D)$.
3. $|\mathcal{T}^{\ell}_{P_X,\varepsilon}| \geq (1 - 2|\mathcal{X}| \exp(-\ell\varepsilon^2/2)) \exp(\ell H(P_X) - \ell\varepsilon D)$,

with the constant $D = \sum_{x:P_X(x)\neq 0} - \log P_X(x)$.

Extending the concept to the conditional ε-typical sequences:

Definition 2.20 *Consider a channel $W : \mathcal{X} \to \mathcal{Y}$ and an input string $x \in \mathcal{X}^{\ell}$. For $\varepsilon > 0$, the* conditional ε-typical sequences *form the set*

$$\begin{aligned} \mathcal{T}^{\ell}_{W,\varepsilon}(x^{\ell}) &= \{y^{\ell} : \forall x \in \mathcal{X}, y \in \mathcal{Y} \ |N(xy|x^{\ell}y^{\ell}) - \ell W(y|x)P_{x^{\ell}}(x)| \leq \varepsilon\ell \\ &\quad \text{and } W(y|x) = 0 \Rightarrow N(xy|x^{\ell}y^{\ell}) = 0\} \\ &= \prod_x \mathcal{T}^{\mathcal{I}_x}_{W_x,\varepsilon P_{x^{\ell}}(x)^{-1}}, \end{aligned}$$

where $\mathcal{I}_x$ are the sets of positions in the string x^{ℓ} where $x_k = x$.

Properties 2.21

1. $W^{\ell}_{x^{\ell}}(\mathcal{T}^{\ell}_{W,\varepsilon}) \geq 1 - 2|\mathcal{X}||\mathcal{Y}| \exp(-\ell\varepsilon^2/2)$.
2. $|\mathcal{T}^{\ell}_{W,\varepsilon}| \leq \exp(\ell H(W|P_{x^{\ell}}) + \ell\varepsilon E)$.
3. $|\mathcal{T}^{\ell}_{W,\varepsilon}| \geq (1 - 2|\mathcal{X}||\mathcal{Y}| \exp(-\ell\varepsilon^2/2)) \cdot \exp(-\ell H(W|P_{x^{\ell}}) - \ell\varepsilon E)$,

with the constant $E = max_x \sum_{y:W(y)\neq 0} - \log W_x(y)$ and the conditional entropy $H(W|P_X) = \sum_x P_X(x)H(W_x)$. See [CK82] for more details.

It is a well know fact that if x^{ℓ} and y^{ℓ} are conditional ε-typical according the Definition 2.20, then

$$|\mathcal{T}^{\ell}_{W,\varepsilon}| \leq 2^{\ell(H(Y|X)+\varepsilon)}.$$

The following lemma and its prove come from Dowsley and Nascimento [DN14].

Lemma 2.22 *Let $W : \mathcal{X} \to \mathcal{Y}$ be a discrete memoryless channel and $x^\ell \in \mathcal{X}^\ell$, $y^\ell \in \mathcal{Y}^\ell$ be the input and output strings of this channel. Let $\mathcal{A}$ be a random subset of $[\ell]$ such that $|\mathcal{A}| = \delta\ell$, $0 < \delta \leq 1$. Let $x^\mathcal{A}$ and $y^\mathcal{A}$ be the restrictions of x^ℓ and y^ℓ to the positions in the set $\mathcal{A}$. If x^ℓ and y^ℓ are conditional ε-typical, then $x^\mathcal{A}$ and $y^\mathcal{A}$ are conditional 2ε-typical for any $\varepsilon > 0$ and ℓ large enough.*

Proof: By hypothesis x^ℓ and y^ℓ are conditional ε-typical, so for every symbols x and y we have that

$$\left|N(xy|x^\ell y^\ell) - \ell P_{x^\ell}(x)W(y|x)\right| \leq \varepsilon n,$$

for a large enough ℓ.

Given the conditional ε-typical strings x^ℓ and y^ℓ, the probability of selecting one pair with the specific values x and y for the substrings $x^\mathcal{A}$ and $y^\mathcal{A}$ is $\frac{N(xy|x^\ell y^\ell)}{\ell}$. We have that

$$P_{x^\ell}(x)W(y|x) - \varepsilon \leq \frac{N(xy|x^\ell y^\ell)}{\ell} \leq P_{x^\ell}(x)W(y|x) + \varepsilon.$$

Therefore, by the Chernoff bound [Che52], for ℓ large enough with overwhelming probability the number of pairs of x and y in the substrings $x^\mathcal{A}$ and $y^\mathcal{A}$, $N(xy|x^\mathcal{A}y^\mathcal{A})$, is limited by

$$\delta\ell\left(P_{x^\ell}(x)W(y|x) - \varepsilon - \varepsilon'\right) \leq N(xy|x^\mathcal{A}y^\mathcal{A}) \leq \delta\ell\left(P_{x^\ell}(x)W(y|x) + \varepsilon + \varepsilon'\right),$$

for any $\varepsilon' > 0$. Making $\varepsilon' = \varepsilon$ we have that the substrings $x^\mathcal{A}$ and $y^\mathcal{A}$ are conditional 2ε-typical. ▮

2.5 Commitment Protocols

A commitment protocol is a family of two-party protocols indexed by the security parameter n. Each protocol proceeds in two phases, commitment and opening, that are executed between a committer Alice and a verifier Bob. In the commitment phase, Alice commits to a message m, but without leaking any information about m to Bob. Later on, Alice can execute the opening phase at any time she wishes in order to disclose the message m to Bob. The security guarantee for Alice is that nothing about m should be learned by Bob in the commitment phase, while the security guarantee for Bob is that Alice should not be able to change the committed message after the commitment phase. We proceed with a more detailed description of the definitions and our model.

Both parties have access to local randomness and there is a bidirectional noiseless channel between them. Depending on the specific scenario being analyzed in next chapters, Alice and Bob will also have access to additional resources in the commitment phase. All the messages generated by Alice and Bob are well-defined random variables, depending on the value that Alice wants to commit to, m, and the local randomness of the parties. For the sake of notation simplicity we will not explicitly mention the dependence on the security parameter n.

Commitment Phase: Alice wants to commit to an input string $m \in \mathcal{M}$ which is a realization of a random variable M. The parties interact using the available resources. Let $\mathsf{trans}^{\mathsf{CP}}(m)$ denote all the communication in this phase and $\mathsf{view}^{\mathsf{CP}}_{\mathsf{Bob}}(m)$

Bob's view at the end of this phase.

Opening Phase: Alice sends Bob the string $\widetilde{m}$ she claims she committed to and the parties can then exchange messages in several rounds. Let $\mathsf{trans}^{\mathsf{OP}}(\widetilde{m})$ denote all the communication in this phase. In the end Bob performs a test

$$\mathsf{test}\left(\mathsf{view}^{\mathsf{CP}}_{\mathsf{Bob}}(m), \mathsf{trans}^{\mathsf{OP}}(\widetilde{m})\right)$$

that outputs 1 if Bob accepts Alice's commitment and 0 otherwise.

Security: A commitment protocol is called $(\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}, \lambda_{\mathsf{B}})$-secure if it satisfies the following properties:

1. λ_{C}-correct: if Alice and Bob are honest, then for every possible m

$$\Pr\left[\text{no aborts and } \mathsf{test}\left(\mathsf{view}^{\mathsf{CP}}_{\mathsf{Bob}}(m), \mathsf{trans}^{\mathsf{OP}}(m)\right) = 1\right] \geq 1 - \lambda_{\mathsf{C}}.$$

2. λ_{H}-hiding: if Alice is honest then

$$I\left(M; \mathsf{view}^{\mathsf{CP}}_{\mathsf{Bob}}(M) | \mathsf{view}^{\mathsf{BC}}_{\mathsf{Bob}}\right) \leq \lambda_{\mathsf{H}},$$

where $\mathsf{view}^{\mathsf{BC}}_{\mathsf{Bob}}$ denotes Bob's view before the start of the commitment phase.

3. λ_{B}-binding: if Bob is honest, then there are no m and $\widetilde{m} \neq \widehat{m}$ such that

$$\Pr\left[\mathsf{test}\left(\mathsf{view}^{\mathsf{CP}}_{\mathsf{Bob}}(m), \mathsf{trans}^{\mathsf{OP}}(\widetilde{m})\right) = 1\right] \geq \lambda_{\mathsf{B}}$$

and

$$\Pr\left[\mathsf{test}\left(\mathsf{view}^{\mathsf{CP}}_{\mathsf{Bob}}(m), \mathsf{trans}^{\mathsf{OP}}(\widehat{m})\right) = 1\right] \geq \lambda_{\mathsf{B}}.$$

2.6 Oblivious Transfer

In an oblivious transfer protocol Alice inputs two string $s_0, s_1 \in \mathcal{M}$ and has no output; while Bob inputs a choice bit c and outputs s_c. As in the case of commitment, it is assumed that both parties have access to local randomness, that there is a bidirectional noiseless channel between them, and depending on the scenario there will be additional resources available to them. The security parameter is n but will be omitted from the notation for the sake of simplicity. Let $\mathsf{view}^{\mathsf{OT}}_{\mathsf{Alice}}(s_0, s_1; c)$ denote the view of an Alice that uses strategy Alice and interacts with an honest Bob. Similarly, let $\mathsf{view}^{\mathsf{OT}}_{\mathsf{Bob}}(s_0, s_1; c)$ denote the view of a Bob that uses strategy Bob and interacts with an honest Alice.

Intuitively, the protocol is secure for Bob if $\mathsf{view}^{\mathsf{OT}}_{\mathsf{Alice}}(s_0, s_1; C)$ and C are independent; and it is secure for Alice if Bob obtains no information about S_{1-C}. However, since we want to give a general security definition that also works in the scenario where there exists a preliminary transmission phase (e.g., the Bounded Storage Model), this is tricky to formalize, as a malicious Bob can proceed with a different choice bit depending on the public random source and the messages exchanged before Alice uses her secrets. We follow the approach of Ding et al. [DHRS04] for defining the security.

In order to have more generality, the main part of the oblivious transfer protocol is divided in two phase: the setup phase, which encompass all communication before

Alice first uses her secrets, and the transfer phase, which happens from that point on. Two pairs of inputs $(s_0, s_1), (s'_0, s'_1)$ are called i-consistent if $s_i = s'_i$. By the end of the setup phase there should exist a random variable I, such that for any two I-consistent pairs of inputs, the resulting view of Bob is statistically close.

Security: A protocol is called $(\lambda_{\mathsf{C}}, \lambda_{\mathsf{B}}, \lambda_{\mathsf{A}})$-secure if it satisfies the following properties:

1. λ_{C}-correct: if Alice and Bob are honest, then

$$\Pr\left[\,\text{no aborts and } s = s_c\,\right] \geq 1 - \lambda_{\mathsf{C}}.$$

2. λ_{B}-secure for Bob: for any strategy Alice used by Alice,

$$\left\|\left\{\mathsf{view}^{\mathsf{OT}}_{\mathsf{Alice}}(s_0, s_1; 0)\right\} - \left\{\mathsf{view}^{\mathsf{OT}}_{\mathsf{Alice}}(s_0, s_1; 1)\right\}\right\| \leq \lambda_{\mathsf{B}}.$$

3. λ_{A}-secure for Alice: for any strategy Bob used by Bob with input c, there exists a random variable I, defined at the end of the setup stage, such that for every two I-consistent pairs $(s_0, s_1), (s'_0, s'_1)$, we have

$$\left\|\left\{\mathsf{view}^{\mathsf{OT}}_{\mathsf{Bob}}(s_0, s_1; c)\right\} - \left\{\mathsf{view}^{\mathsf{OT}}_{\mathsf{Bob}}(s'_0, s'_1; c)\right\}\right\| \leq \lambda_{\mathsf{A}}.$$

2.7 Interactive Hashing and Binary Encoding of Subsets

Interactive hashing was initially introduced in the context of computationally secure cryptography [OVY93], but was later generalized to the information-theoretic setting, and is particularly useful in the context of designing oblivious transfer [CCM98, DHRS04, CS06, Sav07, PDMN11] and commitment protocols [SY11] with unconditional security. In this primitive Bob inputs a string $v \in \{0,1\}^\ell$ and both Alice and Bob receive as output two strings $v_0, v_1 \in \{0,1\}^\ell$ such that $v_0 \neq v_1$. The correctness requirement is that one of the two output strings, v_d, should be equal to v. The security guarantee for Alice is that one of the strings should be effectively beyond the control of (a malicious) Bob. On the other hand, the security guarantee for Bob states that (a malicious) Alice should not be able to learn d.

A variety of protocols for realizing interactive hashing have been proposed [CCM98, DHRS04, NOVY98]. In this work it is used as a black box and the security of the designed protocols only depend on the interactive hashing security properties.

Definition 2.23 (Interactive hashing) *Interactive hashing is a protocol between Alice and Bob in which only Bob has an input $v \in \{0,1\}^\ell$, and both parties output $v_0, v_1 \in \{0,1\}^\ell$ such that $v_d = v$ for some $d \in \{0,1\}$. The protocol is called an* η-uniform (t, θ)-secure interactive hashing protocol *if:*

1. *If both parties are honest, then the random variable V_{1-d} is close to completely random, i.e., V_{1-d} is η-close to the uniform distribution on the $2^\ell - 1$ strings different from v_d.*

2. *Alice's view of the protocol is independent of* d. *Let* Alice *be a strategy for Alice and* $\mathsf{view}^{\mathsf{IH}}_{\mathsf{Alice}}(V)$ *be Alice's view when the input is the random variable* V. *Then*

$$\left\{\mathsf{view}^{\mathsf{IH}}_{\mathsf{Alice}}(V)|V = V_0\right\} = \left\{\mathsf{view}^{\mathsf{IH}}_{\mathsf{Alice}}(V)|V = V_1\right\}.$$

3. *For any* $\mathcal{T} \subset \{0,1\}^{\ell}$ *such that* $|\mathcal{T}| \leq 2^t$, *it should hold that after the protocol execution between an honest Alice and a possibly malicious Bob,*

$$\Pr\left[\, V_0, V_1 \in \mathcal{T}\,\right] \leq \theta,$$

where the probability is over the parties' randomness.

By only requiring V_{1-d} to be close to the uniform distribution, this definition is weaker than others given in the literature [Sav07]. However, it is sufficient to prove our protocols' security and allows for the possibility of using the constant-round interactive hashing protocol of Ding et al. [DHRS04].

Lemma 2.24 ([DHRS04]) *Let* t, ℓ *be positive integers such that* $t \geq \log \ell + 2$. *Then there exists a four-message* η*-uniform* $(t, 2^{-(\ell-t)+O(\log \ell)})$*-secure interactive hashing protocol for* $\eta < 2^{-\ell}$.

The following lemma by Naor et al. [NOVY98] gives an 0-uniform interactive hashing protocol, i.e., V_{1-d} is uniformly distributed, that achieves near-optimal security [Sav07]. Having $\ell - 1$ rounds is its disadvantage.

Lemma 2.25 ([NOVY98]) *There exists a* 0*-uniform* $(t, \psi \cdot 2^{-(\ell-t)})$*-secure interactive hashing protocol for some constant* $\psi > 0$.

The interactive hashing security properties guarantee that one of the outputs is random; however, usually the two binary strings are not used directly in the protocols, but as encodings of subsets of the positions of a tuple. Thus for the protocol to succeed, both outputs v_0 and v_1 need to be valid encodings of subsets containing ℓ_{sub} elements out of the ℓ_{tup} elements of the tuple. Cover showed [Cov73] the existence of an efficiently computable one to one mapping $F : \binom{[\ell_{\text{tup}}]}{\ell_{\text{sub}}} \to [\binom{\ell_{\text{tup}}}{\ell_{\text{sub}}}]$ for every integer $\ell_{\text{sub}} \leq \ell_{\text{tup}}$; thus making it possible to encode the set $\binom{[\ell_{\text{tup}}]}{\ell_{\text{sub}}}$ in binary strings of length $\ell = \lceil \log \binom{\ell_{\text{tup}}}{\ell_{\text{sub}}} \rceil$. However, using such mapping in a straight way may result in only slightly more than half of the strings being valid encodings and thus in several repetitions of the protocol in order to guarantee correctness (as in Cachin et al. [CCM98] for instance). Ding et al. [DHRS04] proposed a "dense" encoding of subsets, ensuring that most ℓ-bit strings are valid encodings. More precisely, they showed the following result.

Lemma 2.26 ([DHRS04]) *Let* $\ell_{sub} \leq \ell_{tup}$, $\ell \geq \lceil \log \binom{\ell_{tup}}{\ell_{sub}} \rceil$, $i = \lfloor 2^{\ell} / \binom{\ell_{tup}}{\ell_{sub}} \rfloor$. *Then there exists an injective mapping* $F\colon \binom{[\ell_{tup}]}{\ell_{sub}} \times [i] \to [2^{\ell}]$ *with* $|\operatorname{Im}(F)| > 2^{\ell} - \binom{\ell_{tup}}{\ell_{sub}}$.

Another possibility is using the modified encoding of Savvides [Sav07], in which each string $v \in \{0,1\}^{\ell}$ encodes the same subset as $v \mod \binom{\ell_{\text{tup}}}{\ell_{\text{sub}}}$, thus implying that all strings always encode valid subsets. With this encoding, each subset corresponds to either 1 or 2 strings in $\{0,1\}^{\ell}$. Therefore the fraction of Bob's subsets of interest is at most the double of the fraction of his strings of interest.

2.8 UC Framework

Here we present a brief discussion of the Universal Composability (UC) framework of Canetti [Can01], please refer to [Can01, CR03] for further details. The main goal of the UC framework is to analyze the security of cryptographic protocols under arbitrary composition, i.e., it takes into consideration scenarios where many copies of a protocol are executed concurrently with themselves and other protocols in a complex environment, such as the Internet. The composition theorem ensures that any protocol proven to be UC-secure can also be securely composed with copies of itself and other protocols. Apart from guaranteeing security in a realistic scenario, the UC framework also enables the modular design of complex applications.

A set of parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$, an adversary $\mathcal{A}$ and an *environment* $\mathcal{Z}$ interact with each other. The environment is responsible for providing the inputs for the parties and $\mathcal{A}$, and for receiving their outputs. The adversary $\mathcal{A}$ is responsible for delivering the messages between the parties (thus modeling that the adversary controls the network scheduling) and may also choose to corrupt a set of parties, in which case he gains control over them. All entities are modeled as Interactive Turing Machines.

The main insight of the UC framework is that $\mathcal{Z}$ captures all activity external to the current execution of the protocol. To prove the security of a protocol, one first defines an idealized version $\mathcal{F}$ of the functionality that the protocol is supposed to perform. Then one shows that for every adversary $\mathcal{A}$ there exists a simulator $\mathcal{S}$ such that no environment $\mathcal{Z}$ can distinguish between an execution of the protocol π with the parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$ and $\mathcal{A}$, and an ideal execution with dummy parties that only forward inputs/outputs, $\mathcal{F}$ and $\mathcal{S}$. The ideal functionality $\mathcal{F}$ does what the protocol should do in a black box manner, i.e., given the inputs, the ideal functionality follows the primitive specification and returns the output as specified; however, the functionality must also deal with the actions of corrupted parties, such as invalid inputs and deviations from the protocol. Some interesting points are: $\mathcal{S}$ has no access to the contents of the messages sent between a party and $\mathcal{F}$ if the party is not corrupted; $\mathcal{Z}$ cannot see the messages sent between the parties and $\mathcal{F}$ and also cannot see the messages sent between the parties in the real protocol execution. A protocol π securely UC-realizes an ideal functionality $\mathcal{F}$ if for every adversary $\mathcal{A}$ in the real world there exists a simulator $\mathcal{S}$ in the ideal world such that no $\mathcal{Z}$ can distinguish an execution of the protocol π with the parties and $\mathcal{A}$ from an execution of the ideal functionality $\mathcal{F}$ with the dummy parties and $\mathcal{S}$. This is stated formally in the following definition.

Definition 2.27 ([Can01]) *A protocol π is said to UC-realize an ideal functionality $\mathcal{F}$ if, for every adversary $\mathcal{A}$, there exists a simulator $\mathcal{S}$ such that, for every environment $\mathcal{Z}$, the following holds:*

$$\mathsf{EXEC}_{\pi,\mathcal{A},\mathcal{Z}} \overset{c}{\approx} \mathsf{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}$$

where $\mathsf{EXEC}_{\pi,\mathcal{A},\mathcal{Z}}(n)$ represents the view of $\mathcal{Z}$ in the real protocol execution with $\mathcal{A}$ and the parties (with security parameter n) and $\mathsf{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}(n)$ represents the view of $\mathcal{Z}$ in the ideal execution with the functionality $\mathcal{F}$, the simulator $\mathcal{S}$ and the dummy parties. The probability distribution is taken over the randomness of the parties.

Obtaining computational indistinguishability between real and ideal executions guarantees that the protocol is secure against probabilistic polynomial time adversaries. Even though this is enough for the security requirements of many protocols and applications, it is interesting to achieve security against computationally unbounded adversaries. In this thesis some of our protocols UC-realize the respective ideal functionalities with *statistically* indistinguishable, thus providing security against attackers that have unlimited computational power.

It is a well-known fact that two-party computation and multi-party computation with dishonest majority is only possible with additional assumptions, either about bounds on resources available to the parties (e.g. polynomial time), or some setup assumption (e.g. existence of a noisy channel or oblivious transfer, or the help of some trusted third party). In the case of UC-secure protocols, the scenario is even more strict: non-trivial two-party and multi-party functionalities cannot be realized without setup assumptions [CF01, CLOS02]. Setup assumptions allowing the realization of non-trivial functionalities include: existence of a common reference string [CF01, CLOS02, PVW08], a public-key infrastructure [BCNP04] or noisy-channels [DMQN08, DvdGMQN13], the random oracle model [HMQ04], signature cards [HMQU05] and tamper-proof hardware [Kat07, DKMQ11, DMQN15]. Pre-distributed correlated randomness, i.e., the commodity-based model, constitutes an attractive setup assumption and is the one focused on this work.

We consider security against static adversaries, i.e., the set of corrupted parties is fixed before the protocol execution and remains unchanged during the execution. In the ideal functionalities the messages are public delayed outputs, meaning that the simulator is first asked whether they should be delivered or not (this is due to the modeling that the adversary controls the network scheduling). This fact as well as the session identifications are omitted from our functionalities' descriptions for the sake of readability.

2.9 Commodity-based Cryptography

Inspired by the client-server distributed computation model, in which a powerful server performs part of the most complex tasks for the clients, Beaver introduced the commodity-based model [Bea97, Bea98b, Bea95], which is a setup assumption and an alternative for obtaining highly efficient, unconditionally secure multi-party computation. In this model there is a setup phase that is independent from the protocol inputs and is performed prior to the protocol execution, possibly long before the protocol inputs are even fixed. In this phase a trusted initializer pre-distributes correlated data to the protocol participants. Thereafter, the trusted initializer does not take part in the protocol execution and, in particular, he does not learn the parties' inputs. The trusted initializer is modeled here by an ideal functionality $\mathcal{F}_{\mathsf{TI}}^{\mathcal{D}}$, which is parametrized by an algorithm $\mathcal{D}$ that samples the correlated data to be pre-distributed to the parties, see Figure 2.1 for details.

The main advantage of this model is that, for many problems, it allows to obtain very efficient solutions while achieving unconditional security (in many cases even perfect security). This comes essentially from the fact that, in these problems, the most complex operations can be delegated to the trusted initializer who performs them locally. More specifically, for those problems, the trusted initializer normally pre-distributes instances of the desired functionalities computed on random inputs, which the parties later on only have to derandomize to match their actual inputs. Notably, this model has been used to construct very efficient protocols for primi-

Functionality $\mathcal{F}_{\mathsf{TI}}^{\mathcal{D}}$

$\mathcal{F}_{\mathsf{TI}}^{\mathcal{D}}$ runs with the parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$ and is parametrized by an algorithm $\mathcal{D}$.

Upon initialization run $(D_1, \ldots, D_u) \leftarrow \mathcal{D}$. For $i = 1, \ldots, u$, deliver D_i to $\mathcal{P}_i$.

Figure 2.1: The Trusted Initializer functionality.

tives as commitments [Riv99, BMSW02, NMQO+03], OT [Bea95, Bea97, Riv99], verifiable secret sharing [NMQO+04, DMQO+11], inner product [DGMN11], string equality [IKM+13], set intersection [IKM+13], secure linear algebra [DDvdG+16] and oblivious polynomial evaluation [TND+15].

In the commodity-based model it is possible to obtain point-to-point secure authenticated channels by using pre-distributed one-time pads and unconditionally secure message authentication codes, as well as broadcast channels by using the techniques of Pfitzmann and Waidner [PW96]. Given that OT is complete [Kil88], one possibility is to use OT-based protocols to obtain general multi-party computation protocols; but this would imply a large overhead. Lower bounds on the OT reductions are known both for the case of perfect security [Bea96, DM99] and statistical security [WW10]. Fitzi et al. [FGMv02, FWW04] investigated the types of correlated data that can be useful for obtaining secure broadcast in the presence of an honest majority, and thus also for secure multi-party computation with honest majority [Bea90, RBO89, CDD+99]. Ishai et al. [IKM+13] studied the communication complexity of protocols based on correlated data and also presented general possibility results for protocols with perfect security.

In practice this correlated data can be obtained in different ways: (1) it can be distributed by a single trusted center that runs locally the sampler $\mathcal{D}$ during the setup phase and delivers the data to the protocol participants; (2) it can be predistributed by many, not entirely trusted centers that do not interact with each other and do not need to know about each other existence. In this case one only needs a majority of the centers to be honest [Bea97, Bea98b]; (3) or it can be pre-computed by the parties using a multi-party computation protocol in order to emulate the trusted initializer. In this case the overall security is only computational; the main advantage is offloading the heavy computational steps to an offline phase that can be executed at any time.

2.10 Matrix Multiplication

Given the operations that can be performed locally with the secret sharings, one important operation that is missing is the secure multiplication of secret shared values. While this operation can be complex to perform in the plain model, in the commodity-based model there is a very efficient and simple protocol due to Beaver [Bea92, Bea98a]. In this section we present a generalization of his solution that can deal with the multi-party, distributed multiplication of matrices which are repre-

Functionality $\mathcal{F}_{\mathsf{DMM}}$

$\mathcal{F}_{\mathsf{DMM}}$ runs with parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$ and is parametrized by the size q of the ring and the dimensions ℓ_1, ℓ_2 and ℓ_3 of the matrices to be multiplied.

Input: Upon receiving a message from a party with its shares of $[\![\mathbf{X}]\!]_q$ and $[\![\mathbf{Y}]\!]_q$, verify if the share of $\mathbf{X}$ is in $\mathbb{Z}_q^{\ell_1 \times \ell_2}$ and the share of $\mathbf{Y}$ is in $\mathbb{Z}_q^{\ell_2 \times \ell_3}$. If it is not, abort. Otherwise, record the shares, ignore any subsequent message from that party and inform the other parties about the receipt.

Output: Upon receipt of the shares from all parties, reconstruct $\mathbf{X}$ and $\mathbf{Y}$ from the shares, compute $\mathbf{Z} = \mathbf{XY}$ and create a secret sharing $[\![\mathbf{Z}]\!]_q$ to distribute to the parties: the corrupt parties fix their shares of the output to any constant values and the shares of the uncorrupted parties are then created by picking uniformly random values subject to the correctness constraint. The shares of the uncorrupted parties are only delivered if the adversary allows.

Figure 2.2: The distributed matrix multiplication functionality.

sented in the form of element-wise secret sharings. The parties have as inputs secret sharings $[\![\mathbf{X}]\!]_q$ with $\mathbf{X} \in \mathbb{Z}_q^{\ell_1 \times \ell_2}$ and $[\![\mathbf{Y}]\!]_q$ with $\mathbf{Y} \in \mathbb{Z}_q^{\ell_2 \times \ell_3}$, and want to obtain as output a secret sharing $[\![\mathbf{Z}]\!]_q$ corresponding to $\mathbf{Z} = \mathbf{XY}$ while leaking information from neither the input values $\mathbf{X}$ and $\mathbf{Y}$ nor the output value $\mathbf{Z}$. The trusted initializer pre-distributes a random matrix multiplication triple to the parties, i.e., secret sharings $[\![\mathbf{U}]\!]_q, [\![\mathbf{V}]\!]_q, [\![\mathbf{W}]\!]_q$ for $\mathbf{U}$ and $\mathbf{V}$ uniformly random in $\mathbb{Z}_q^{\ell_1 \times \ell_2}$ and $\mathbb{Z}_q^{\ell_2 \times \ell_3}$, respectively, and $\mathbf{W} = \mathbf{UV}$. The parties then derandomize the random matrix multiplication triple during the protocol execution in order to compute the secret sharing $[\![\mathbf{Z}]\!]_q$. Figure 2.2 describes the distributed matrix multiplication functionality $\mathcal{F}_{\mathsf{DMM}}$ that is considered and Figure 2.3 presents the protocol π_{DMM} that implements such functionality.

Theorem 2.28 *For static malicious adversaries corrupting any number of parties, the distributed matrix multiplication protocol π_{DMM} UC-realizes with perfect security the functionality $\mathcal{F}_{\mathsf{DMM}}$ in the commodity-based model.*

Proof: For verifying the correctness, first notice that

$$\mathbf{Z} = \mathbf{XY} = (\mathbf{U} + \mathbf{D})(\mathbf{V} + \mathbf{E}) = \mathbf{UV} + \mathbf{UE} + \mathbf{DV} + \mathbf{DE} = \mathbf{W} + \mathbf{UE} + \mathbf{DV} + \mathbf{DE}$$

and therefore $[\![\mathbf{Z}]\!]_q \leftarrow [\![\mathbf{W}]\!]_q + \mathbf{E}[\![\mathbf{U}]\!]_q + \mathbf{D}[\![\mathbf{V}]\!]_q + \mathbf{DE}$ really obtains a secret sharing corresponding to $\mathbf{Z} = \mathbf{XY}$. The fact that the resulting shares are uniformly random subject to the correctness conditions follows trivially from the properties of the pre-distributed matrix multiplication triple.

The simulation is very simple and proceeds as follows. The simulator $\mathcal{S}$ runs internally a copy of the adversary $\mathcal{A}$ and reproduces the real world protocol execution

Secure Distributed Matrix Multiplication Protocol π_{DMM}

The protocol runs with parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$ and is parametrized by the size q of the ring and the dimensions ℓ_1, ℓ_2 and ℓ_3 of the matrices to be multiplied. The trusted initializer chooses uniformly random $\mathbf{U} \in \mathbb{Z}_q^{\ell_1 \times \ell_2}$ and $\mathbf{V} \in \mathbb{Z}_q^{\ell_2 \times \ell_3}$, computes $\mathbf{W} = \mathbf{U}\mathbf{V}$ and pre-distributes secret sharings $\llbracket \mathbf{U} \rrbracket_q, \llbracket \mathbf{V} \rrbracket_q, \llbracket \mathbf{W} \rrbracket_q$ to the parties. The parties have inputs $\llbracket \mathbf{X} \rrbracket_q$ with $\mathbf{X} \in \mathbb{Z}_q^{\ell_1 \times \ell_2}$ and $\llbracket \mathbf{Y} \rrbracket_q$ with $\mathbf{Y} \in \mathbb{Z}_q^{\ell_2 \times \ell_3}$, and interact as follows:

1. Locally compute $\llbracket \mathbf{D} \rrbracket_q \leftarrow \llbracket \mathbf{X} \rrbracket_q - \llbracket \mathbf{U} \rrbracket_q$ and $\llbracket \mathbf{E} \rrbracket_q \leftarrow \llbracket \mathbf{Y} \rrbracket_q - \llbracket \mathbf{V} \rrbracket_q$, then open $\mathbf{D}$ and $\mathbf{E}$ using a broadcast channel. A party aborts if it receives a message with invalid format during the opening.

2. Locally compute the output $\llbracket \mathbf{Z} \rrbracket_q \leftarrow \llbracket \mathbf{W} \rrbracket_q + \mathbf{E}\llbracket \mathbf{U} \rrbracket_q + \mathbf{D}\llbracket \mathbf{V} \rrbracket_q + \mathbf{D}\mathbf{E}$.

Figure 2.3: The protocol for secure distributed matrix multiplication.

perfectly for $\mathcal{A}$. For that, it simulates the protocol execution with dummy inputs for the uncorrupted parties. The leverage of the simulator is the fact that it can simulate the trusted initializer functionality $\mathcal{F}_{\mathsf{TI}}^{\mathcal{D}}$ for $\mathcal{A}$. Using this leverage, whenever a corrupted party broadcasts its shares of $\mathbf{D}$ and $\mathbf{E}$ in the simulated protocol execution, $\mathcal{S}$ can extract the respective shares of $\mathbf{X}$ and $\mathbf{Y}$ to give to the distributed matrix multiplication functionality $\mathcal{F}_{\mathsf{DMM}}$. And whenever an honest party sends its shares to the functionality, $\mathcal{S}$ simulates the broadcast messages for $\mathcal{A}$ by sending random messages, which from $\mathcal{A}$'s point of view are indistinguishable from the messages in the real protocol execution as the shares of $\mathbf{U}$ and $\mathbf{V}$ are uniformly random and unknown to $\mathcal{A}$. Given its knowledge about $\llbracket \mathbf{U} \rrbracket_q, \llbracket \mathbf{V} \rrbracket_q, \llbracket \mathbf{W} \rrbracket_q, \mathbf{D}$ and $\mathbf{E}$ by the end of the simulated execution, $\mathcal{S}$ knows, for each corrupted party, which value its share of the output is supposed to take, and therefore $\mathcal{S}$ can fix these values in $\mathcal{F}_{\mathsf{DMM}}$ so that the sum of the uncorrupted parties' shares is compatible with the simulated execution. If the uncorrupted parties get their shares of the output in the simulated protocol, $\mathcal{S}$ allows $\mathcal{F}_{\mathsf{DMM}}$ to deliver their shares. ∎

Notation: We denote by π_{DM} and π_{IP} the protocol for the special cases of multiplication of single elements and inner-product computation, respectively.

Broadcast Channel: In the two-party case or if we restrict to honest-but-curious adversaries, broadcast is trivially not necessary. In the remaining cases with dishonest majority, no termination can be guaranteed for the final protocols, so no termination for the broadcast protocol is needed and we can use the simple protocol of Damgård et al. [DPSZ11, Section A.3].

If only a simplified version of the protocol is needed in which the output is still distributed as a secret sharing, but the inputs are hold entirely by single parties, lets say $\mathbf{X}$ by $\mathcal{P}_1$ and $\mathbf{Y}$ by $\mathcal{P}_2$, then it is possible to use the even more efficient

Simplified Secure Distributed Matrix Multiplication Protocol π'_{DMM}

The protocol runs with parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$ and is parametrized by the size q of the finite field and the dimensions ℓ_1, ℓ_2 and ℓ_3 of the matrices to be multiplied. The trusted initializer chooses uniformly random $\mathbf{U} \in \mathbb{Z}_q^{\ell_1 \times \ell_2}$ and $\mathbf{V} \in \mathbb{Z}_q^{\ell_2 \times \ell_3}$, computes $\mathbf{W} = \mathbf{UV}$ and pre-distributes the secret sharing $[\![\mathbf{W}]\!]_q$ to the parties, $\mathbf{U}$ to $\mathcal{P}_1$ and $\mathbf{V}$ to $\mathcal{P}_2$. $\mathcal{P}_1$ has input $\mathbf{X} \in \mathbb{Z}_q^{\ell_1 \times \ell_2}$ and $\mathcal{P}_2$ has input $\mathbf{Y} \in \mathbb{Z}_q^{\ell_2 \times \ell_3}$. The parties interact as follows:

1. $\mathcal{P}_1$ locally computes $\mathbf{D} \leftarrow \mathbf{X} - \mathbf{U}$ and sends it to $\mathcal{P}_2$, who in turn computes $\mathbf{E} \leftarrow \mathbf{Y} - \mathbf{V}$ and sends it to $\mathcal{P}_1$. A party aborts if it receives a message with invalid format.

2. Locally compute the output $[\![\mathbf{Z}]\!]_q \leftarrow [\![\mathbf{W}]\!]_q + \mathbf{EU} + \mathbf{DV} + \mathbf{DE}$, where the second term is added locally by $\mathcal{P}_1$ and the last two terms locally by $\mathcal{P}_2$.

Figure 2.4: The simplified protocol for secure distributed matrix multiplication.

protocol π'_{DMM} that is described in Figure 2.4. Table 2.1 presents a comparison of the complexity in terms of the pre-distributed data, communication, and number of additions and matrix multiplications performed between the two protocols.

	Protocol π_{DMM}	Protocol π'_{DMM}
Pre-distributed Data	$u(\ell_1\ell_2 + \ell_2\ell_3 + \ell_1\ell_3)$	$\ell_1\ell_2 + \ell_2\ell_3 + u\ell_1\ell_3$
Communication	$u(\ell_1\ell_2 + \ell_2\ell_3)$ broadcast	$\ell_1\ell_2 + \ell_2\ell_3$ point-to-point
Matrix Multiplication	$2u + 1$	2
Additions	$u(\ell_1\ell_2 + \ell_2\ell_3 + 2\ell_1\ell_3) + \ell_1\ell_3$	$\ell_1\ell_2 + \ell_2\ell_3 + 2\ell_1\ell_3$

Table 2.1: Complexity of both distributed matrix multiplication protocols when executed with u parties to multiply a matrix of dimension $\ell_1 \times \ell_2$ by a matrix of dimension $\ell_2 \times \ell_3$. The pre-distributed data and communication complexities are expressed in terms of ring elements. The multiplications are in terms of matrix multiplications with dimensions $\ell_1 \times \ell_2$ and $\ell_2 \times \ell_3$. And the additions are in terms of ring element additions.

2.11 Other Technical Lemmas

This section comes from Dowsley et al. [DLN14, DLN15] and presents some useful supporting lemmas that are used in the security proofs of our commitment and OT protocols in the Bounded Storage Model with Errors, which are described in Chapter 5.

The following is a basic fact that follows from simple counting.

Lemma 2.29 *Let $0 \leq \delta < 1/2$ and let $x, y \in \{0,1\}^\ell$ be such that $\mathsf{HD}(x,y) \leq \delta\ell$ and $H_\infty(X) \geq \alpha\ell$ where $0 < \alpha < 1$. Then $H_\infty(Y) \geq (\alpha - h(\delta))\ell$.*

The next lemma shows that the restrictions of two tuples to random subsets of their positions have relative Hamming distances that are close to the one between the entire tuples.

Lemma 2.30 ([DLN15]) *Let $x, y \in \{0,1\}^\ell$. Let $\mathcal{S}$ be a random subset of $[\ell]$ of size t and consider any $0 < \nu$. On one hand, if $\mathsf{HD}(x,y) \leq \delta\ell$, then $\mathsf{HD}(x^{\mathcal{S}}, y^{\mathcal{S}}) < (\delta+\nu)t$ except with probability $e^{-t\nu^2/2}$. On the other hand, if $\mathsf{HD}(x,y) \geq \delta\ell$, then $\mathsf{HD}(x^{\mathcal{S}}, y^{\mathcal{S}}) > (\delta-\nu)t$ except with probability $e^{-t\nu^2/2}$.*

Proof: Lets begin with the first part of the lemma. By Lemma 2.10, a random subset sampler is an $(\mu, \nu, e^{-t\nu^2/2})$-averaging sampler for any $\mu, \nu > 0$. Hence for any $f\colon [\ell] \to [0,1]$ with $\frac{1}{\ell}\sum_{i=1}^{\ell} f(i) \geq \mu$

$$\Pr\left[\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} f(i) \leq \mu - \nu\right] \leq e^{-t\nu^2/2}, \tag{2.2}$$

Let

$$f(i) = \begin{cases} 0, & \text{if } x_i \neq y_i, \\ 1, & \text{otherwise.} \end{cases}$$

Fix $\mu = 1-\delta$. Note that $\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} f(i) = 1 - \frac{\mathsf{HD}(x^{\mathcal{S}}, y^{\mathcal{S}})}{t}$ and $\frac{1}{\ell}\sum_{i=1}^{\ell} f(i) = 1 - \frac{\mathsf{HD}(x,y)}{\ell} \geq \mu$. Thus by Equation (2.2)

$$\begin{aligned} e^{-t\nu^2/2} &\geq \Pr\left[\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} f(i) \leq \mu - \nu\right] \\ &= \Pr\left[1 - \frac{\mathsf{HD}(x^{\mathcal{S}}, y^{\mathcal{S}})}{t} \leq 1 - \delta - \nu\right] \\ &= \Pr\left[\mathsf{HD}(x^{\mathcal{S}}, y^{\mathcal{S}}) \geq (\delta+\nu)t\right] \end{aligned}$$

which proves the first part of the lemma.

The second part of the lemma uses the same idea, but now the function f is

$$f(i) = \begin{cases} 0, & \text{if } x_i = y_i, \\ 1, & \text{otherwise.} \end{cases}$$

Fixing $\mu = \delta$ it holds that $\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} f(i) = \frac{\mathsf{HD}(x^{\mathcal{S}}, y^{\mathcal{S}})}{t}$ and $\frac{1}{\ell}\sum_{i=1}^{\ell} f(i) = \frac{\mathsf{HD}(x,y)}{\ell} \geq \mu$ and hence

$$\begin{aligned} e^{-t\nu^2/2} &\geq \Pr\left[\frac{1}{|\mathcal{S}|}\sum_{i\in\mathcal{S}} f(i) \leq \mu - \nu\right] \\ &= \Pr\left[\frac{\mathsf{HD}(x^{\mathcal{S}}, y^{\mathcal{S}})}{t} \leq \delta - \nu\right] \\ &= \Pr\left[\mathsf{HD}(x^{\mathcal{S}}, y^{\mathcal{S}}) \leq (\delta-\nu)t\right] \end{aligned}$$

which finishes the proof of the lemma. ▌

The following statement of the birthday paradox is standard.

Lemma 2.31 *Let $\mathcal{M}, \mathcal{N} \subset [\ell]$ be chosen independently at random with $|\mathcal{M}| = |\mathcal{N}| = 2\sqrt{t\ell}$. Then*

$$\Pr[\,|\mathcal{M} \cap \mathcal{N}| < t\,] < e^{-t/4}$$

Proof: See corollary 3 in Ding [Din01] for example. ▮

The following useful lemma will also be employed in Chapter 5.

Lemma 2.32 ([AS04]) *Let $0 < \delta < 1/2$. Then*

$$\sum_{i=0}^{\delta\ell} \binom{\ell}{i} \leq 2^{h(\delta)\ell}.$$

Proof: It holds that

$$\begin{aligned} 2^{-h(\delta)\ell} &= 2^{(\delta \log \delta + (1-\delta)\log(1-\delta))\ell} \\ &= \delta^{\delta\ell}(1-\delta)^{(1-\delta)\ell} \\ &\leq \delta^{i}(1-\delta)^{\ell-i} \quad \text{for } i = 0, \ldots, \delta\ell. \end{aligned}$$

where the last inequality is valid for $\delta < 1/2$.

Hence

$$2^{-h(\delta)\ell}\sum_{i=0}^{\delta\ell} \binom{\ell}{i} \leq \sum_{i=0}^{\delta\ell} \binom{\ell}{i}\delta^{i}(1-\delta)^{\ell-i} = 1,$$

and this finishes the proof. ▮

The following lemma by Rompel [Rom90] will be also useful.

Lemma 2.33 ([Rom90]) *Suppose t is a positive even integer, $X_1, \cdots, X_u$ are t-wise independent random variables taking values in the range $[0,1]$, $X = \sum_{i=1}^{u} X_i$, $\mu = E[X]$, and $A > 0$. Then*

$$\Pr[\,|X - \mu| > A\,] < O\left(\left(\frac{t\mu + t^2}{A^2}\right)^{t/2}\right).$$

3. On the OT Capacity of GEC Against Malicious Adversaries

A Generalized Erasure Channel is a combination of a discrete memoryless channel and an erasure channel: independently from the input symbol, the output of each transmission is an erasure with probability $p_\epsilon > 0$. It can ben formally defined as follows:

Definition 3.1 (Generalized Erasure Channel) *A discrete memoryless channel $\{W : \mathcal{X} \to \mathcal{Y}\}$ is called a* Generalized Erasure Channel (GEC) *if the output alphabet $\mathcal{Y}$ can be decomposed as $\mathcal{Y}_0 \cup \mathcal{Y}^*$ such that $W(y|x)$ does not depend on $x \in \mathcal{X}$ if $y \in \mathcal{Y}^*$. For a GEC and for all $x \in \mathcal{X}$, $y \in \mathcal{Y}_0$, we denote $W_0(y|x) = \frac{1}{1-p_\epsilon} W(y|x)$ where p_ϵ is the sum of $W(y|x)$ for $y \in \mathcal{Y}^*$.*

GECs represent a very special case in the study of OT protocols based on noisy channels. In general, the known techniques to implement OT from noisy channels first use the noisy channel to emulate a GEC (in case it is not one) and then use the GEC in the rest of the protocol; the exception being the very recent work of Khurana et al. [KMS16]. Therefore, clarifying the OT capacity of GEC is a central question.

Ahlswede and Csiszár [AC07, AC13] obtained upper and lower bounds on the OT capacity of GECs against honest-but-curious adversaries. In the case of GECs with erasure probability $p_\epsilon \geq 1/2$, the upper and lower bounds match and therefore the OT capacity is determined. Pinto et al. [PDMN11] proved that for this case these bounds can also be extend for the malicious adversaries case[1], thus completely characterizing the OT capacity of GECs with $p_\epsilon \geq 1/2$. In the case of GEC with erasure probability $p_\epsilon < 1/2$, the lower bound against honest-but-curious adversaries established by Ahlswede and Csiszár does not match their upper bound and it was unknown whether this OT rate could be achieved also when considering malicious adversaries.

This chapter is based on [DN14] and shows the existence of an OT protocol based on GECs with $p_\epsilon < 1/2$ that is secure against malicious adversaries and achieves

[1] Of course, any upper bounds automatically extends from the honest-but-curious case to the malicious one.

the same OT rate as Ahlswede and Csiszár's scheme against honest-but-curious adversaries. In order to obtain our result we introduce a novel use of interactive hashing that is suitable for dealing with the case of low erasure probability. The techniques used by Pinto et al. [PDMN11] clearly do not apply in the case that $p_\epsilon < 1/2$ as they explicitly use the fact that the majority of the symbols received by Bob are erasures.

3.1 Problem Statement

We consider malicious adversaries that can act arbitrarily. As Beaver [Bea95] presented an extremely efficient reduction from randomized OT to standard OT, for simplicity, we consider OT with random inputs in this chapter. The input messages come from $\mathcal{M} = \{0,1\}^m$. In addition to the bidirectional noiseless channel, the parties are connected by a GEC from Alice to Bob. The security definition used was introduced in Section 2.6. The security parameter n determines the number of times that the GEC W is used.

The *OT rate* of a protocol is given by $\mathsf{R_{OT}} = \frac{m}{n}$. The *OT capacity* $\mathsf{C_{OT}}(W)$ of the channel W is the supremum of the achievable rates for protocols that use W and are $(\lambda_\mathsf{C}, \lambda_\mathsf{B}, \lambda_\mathsf{A})$-secure with $\lambda_\mathsf{C}, \lambda_\mathsf{B}$ and λ_A negligible in n. For a GEC $\{W : \mathcal{X} \to \mathcal{Y}\}$, let $\mathsf{C_S}(W_0)$ denote the Shannon capacity of the discrete memoryless channel $\{W_0 : \mathcal{X} \to \mathcal{Y}_0\}$. The OT capacity of GECs with $p_\epsilon \geq \frac{1}{2}$ was determined by Ahlswede and Csiszár [AC07, AC13] for the case of honest-but-curious adversaries, and by Pinto et al. [PDMN11] for malicious adversaries.

Theorem 3.2 ([AC07, PDMN11, AC13]) *For a Generalized Erasure Channel W with $p_\epsilon \geq \frac{1}{2}$, the OT capacity both in the case of honest-but-curious adversaries as well as in the case of malicious adversaries is* $\mathsf{C_{OT}}(W) = (1 - p_\epsilon)\mathsf{C_S}(W_0)$.

For the case of GECs with $p_\epsilon < \frac{1}{2}$, a lower bound on the OT capacity against honest-but-curious adversaries was obtained by Ahlswede and Csiszár [AC07, AC13].

Theorem 3.3 ([AC07, AC13]) *For a Generalized Erasure Channel with $p_\epsilon < \frac{1}{2}$, a lower bound on the OT capacity in the case of honest-but-curious adversaries is* $\mathsf{C_{OT}}(W) \geq p_\epsilon \mathsf{C_S}(W_0)$.

Here we prove that the same OT rate can also be achieved against malicious adversaries.

3.2 Our Lower Bound on the OT Capacity of GEC

Theorem 3.4 ([DN14]) *For a Generalized Erasure Channel with $p_\epsilon < \frac{1}{2}$, a lower bound on the OT capacity for* malicious adversaries *is* $\mathsf{C_{OT}}(W) \geq p_\epsilon \mathsf{C_S}(W_0)$.

The secure OT protocol that achieves such rate is presented below and belongs to the lineage of protocols initiated by Crépeau and Savvides [CS06, Sav07], which use interactive hashing as a central, efficient mechanism to ensure that (a malicious) Bob is following the protocol rules without revealing to Alice his choice bit. Due to the fact that in our case the non-erasure positions are the majority, our usage of the interactive hashing protocol is different from the previous protocols. The description of the OT protocol π_{OTGEC} is in Figure 3.1. The bit length of the OT input strings is $m = n[(\mu - 5\alpha)H(X) - \mu H(X|Y \in \mathcal{Y}_0) - \mu\varepsilon - \gamma]$, where the constant are clarified in the protocol description.

Theorem 3.5 ([DN14]) *The OT protocol* π_{OTGEC} *is* $(\lambda_{\mathsf{C}}, \lambda_{\mathsf{B}}, \lambda_{\mathsf{A}})$*-secure with* $\lambda_{\mathsf{C}}, \lambda_{\mathsf{B}}$ *and* λ_{A} *negligible in* n*.*

Proof: Correctness: If both Alice and Bob are honest, Bob gets the correct output value unless he aborts in the Good/Bad Sets step or he does not recover exactly $\widetilde{x}^{\mathcal{Q}_c} = x^{\mathcal{Q}_c}$ in the Output step. But the probability that Bob has to abort in the Good/Bad Sets step is a negligible function of the security parameter n due to the Chernoff bound [Che52]. Bob does not recover the correct $\widetilde{x}^{\mathcal{Q}_c} = x^{\mathcal{Q}_c}$ if either $x^{\mathcal{Q}_c}$ is not jointly typical with $y^{\mathcal{Q}_c}$ or if there exists another $\overline{x}^{\mathcal{Q}_c}$ that has $g_c(\overline{x}^{\mathcal{Q}_c}) = g_c(x^{\mathcal{Q}_c})$ and is jointly typical with $y^{\mathcal{Q}_c}$. The former case only occurs with negligible probability due to the definition of joint typicality. For the latter case, an upper bound on the number of $\overline{x}^{\mathcal{Q}_c}$ that are jointly typical with $y^{\mathcal{Q}_c}$ is $2^{\mu n[H(X|Y \in \mathcal{Y}_0) + \varepsilon']}$, for $0 < \varepsilon' < \varepsilon$ and n sufficiently large. Therefore according to Lemma 2.14, for n sufficiently large, with overwhelming probability $g_c(\overline{x}^{\mathcal{Q}_c}) \neq g_c(x^{\mathcal{Q}_c})$ for all these other $\overline{x}^{\mathcal{Q}_c}$ that are jointly typical with $y^{\mathcal{Q}_c}$. As all events that result in Bob not obtaining the correct output only occur with negligible probability in n, the protocol is correct.

Security for Bob: In a GEC each input symbol x is erased with the same probability p_ϵ. Therefore Alice has no knowledge about the erasures and thus from Alice's point of view the tuples $\mathcal{R}_0$ and $\mathcal{R}_1$ are independent from the choice bit c. The only other point where the bit c is used is to compute $e = d \oplus c$ in the Checking the Partitioning step, but there it is xored with the bit d and Alice's has no information about d due to the security properties of the interactive hashing protocol.

Security for Alice: The proof of security for Alice follows the lines of Savvides' proof [Sav07, Section 5.1], but we use new variants of the supporting definitions and lemmas due to the fact that we use the interactive hashing protocol in a different way.

Definition 3.6 *For a tuple* $\mathcal{R}$ *of indices in* $[n]$*, let* $u(\mathcal{R})$ *be the number of indices whose corresponding output was an erasure.*

Definition 3.7 *For a tuple of indices* $\mathcal{R}$ *and a subset* $\mathcal{T}$ *of the elements of* $\mathcal{R}$*, let* $\mathcal{K}$ *denote the corresponding indices.* $\mathcal{T}$ *is called* good *for* $\mathcal{R}$ *if* $u(\mathcal{K}) < \alpha n$*, otherwise it is called* bad *for* $\mathcal{R}$*.*

The proof is divided in two cases as follows: (1) both $u(\mathcal{R}_0), u(\mathcal{R}_1) \geq 2\alpha n$, (2) either $u(\mathcal{R}_0)$ or $u(\mathcal{R}_1)$ is less than $2\alpha n$.

Case 1: For proving Alice's security in the first case we will need the following lemmas.

Secure OT Protocol π_{OTGEC} based on GEC

1. (Parameter Setting) Alice and Bob select a positive constant α such that $3\alpha < 1/2 - p_\epsilon$ and set $\beta = 1/2 - p_\epsilon - \alpha$. Note that $\beta > 2\alpha$.

2. (GEC Usage) Alice chooses x^n randomly according to the probability distribution that achieves the Shannon capacity of W_0. She sends x^n to Bob using the GEC, who receives the string y^n.

3. (Good/Bad Sets) Bob divides the string y^n into a set $\mathcal{G}$ of good indices (those with $y \in \mathcal{Y}_0$) and a set $\mathcal{B}$ of bad indices (those corresponding to erasures). The protocol is aborted if $|\mathcal{G}| < (1 - p_\epsilon - \alpha)n$.

4. (Partitioning) Bob chooses uniformly randomly a bit c and a ℓ-bit string v with $\ell = \lceil \log \binom{n/2}{\beta n} \rceil$. He decodes v into a subset $\mathcal{T}$ of cardinality βn out of $n/2$ elements using Savvides' encoding scheme that is described in Section 2.7. Then he partitions the n indices into two tuples of length $n/2$. For the tuple $\mathcal{R}_c$ he picks randomly $n/2$ indices from $\mathcal{G}$. For $\mathcal{R}_{1-c}$, he first fills the positions of $\mathcal{R}_{1-c}$ specified by the subset $\mathcal{T}$ randomly with unused indices from $\mathcal{G}$ and then fills the rest of $\mathcal{R}_{1-c}$ randomly with the $n/2 - \beta n$ indices that are still unused. Bob sends the descriptions of $\mathcal{R}_0$ and $\mathcal{R}_1$ to Alice, who aborts if there are repeated indices.

5. (Interactive Hashing) Bob sends v to Alice using the interactive hashing protocol from Lemma 2.24. Let v_0 and v_1 be the output strings, $\mathcal{T}_0$ and $\mathcal{T}_1$ the decoded subsets and d be such that $v_d = v$.

6. (Checking the Partitioning) Let $e = d \oplus c$. Define $\mathcal{K}_0$ as the indices contained in $\mathcal{R}_0$ that are selected by $\mathcal{T}_{1-e}$ and $\mathcal{Q}_0$ as the remaining indices in $\mathcal{R}_0$. Similarly, define $\mathcal{K}_1$ as the indices contained in $\mathcal{R}_1$ that are selected by $\mathcal{T}_e$ and $\mathcal{Q}_1$ as the remaining indices in $\mathcal{R}_1$. Bob announces e, $y^{\mathcal{K}_0}$ and $y^{\mathcal{K}_1}$. For a fixed $\widehat{\varepsilon} > 0$ Alice verifies if $y^{\mathcal{K}_0}$ and $y^{\mathcal{K}_1}$ are $2\widehat{\varepsilon}$-jointly typical for the channel $\{W_0 : \mathcal{X} \to \mathcal{Y}_0\}$ with her input on these indices (see Section 2.4 for the considered definitions of typicality); aborting if this is not the case.

7. (Strings Transmission) Let $\mu = p_\epsilon + \alpha$. Alice randomly chooses 2-universal hash functions $g_0, g_1 : \mathcal{X}^{\mu n} \to \{0,1\}^{\mu n [H(X|Y \in \mathcal{Y}_0) + \varepsilon]}$ with $\varepsilon > 0$ such that the output length is integer and computes $g_0(x^{\mathcal{Q}_0})$ and $g_1(x^{\mathcal{Q}_1})$. In addition she also randomly chooses 2-universal hash functions $h_0, h_1 : \mathcal{X}^{\mu n} \to \{0,1\}^{\psi n}$ with $\psi = (\mu - 5\alpha)H(X) - \mu(H(X|Y \in \mathcal{Y}_0) + \varepsilon) - \gamma$ and $\gamma > 0$ is such that the output length is integer. Alice sends Bob $g_0(x^{\mathcal{Q}_0})$, $g_1(x^{\mathcal{Q}_1})$ and the descriptions of g_0, g_1, h_0, h_1. She outputs $s_0 = h_0(x^{\mathcal{Q}_0})$ and $s_1 = h_1(x^{\mathcal{Q}_1})$.

8. (Output) Bob computes all possible $\widetilde{x}^{\mathcal{Q}_c}$ that are jointly typical with $y^{\mathcal{Q}_c}$ and satisfy $g_c(\widetilde{x}^{\mathcal{Q}_c}) = g_c(x^{\mathcal{Q}_c})$. If there exists exactly one such $\widetilde{x}^{\mathcal{Q}_c}$, then Bob outputs $s_c = h_c(\widetilde{x}^{\mathcal{Q}_c})$; otherwise $s_c = 0^{\psi n}$.

Figure 3.1: OT protocol based on GEC.

Lemma 3.8 ([DN14]) *Let $\mathcal{R}$ be a tuple with $n/2$ distinct indices in $[n]$ such that $u(\mathcal{R}) \geq 2\alpha n$. The fraction f of subsets $\mathcal{T}$ of cardinality βn that are good for $\mathcal{R}$ satisfies $f < (1-2\alpha)^{\alpha n}$.*

Proof: Using the probabilistic method, we prove that a subset $\mathcal{T}$ chosen uniformly at random will be good for $\mathcal{R}$ with probability smaller than $(1-2\alpha)^{\alpha n}$. One way of choosing the resulting set of indices $\mathcal{K}$ is by picking sequentially at random, and without replacement, βn indices out of the $n/2$ indices in $\mathcal{R}$. For $1 < i < \beta n$, the probability p_i that the i-th chosen index corresponds to a non-erasure given that $\mathcal{K}$ does not have enough erasure indices so far for $\mathcal{T}$ to be considered bad for $\mathcal{R}$ (i.e., less than αn erasures) is upper bounded by

$$p_i < 1 - \frac{2\alpha n - \alpha n}{n/2} = 1 - 2\alpha.$$

Since for a subset $\mathcal{T}$ to be considered good for $\mathcal{R}$ it needs to correspond to at least $\beta n - \alpha n$ non-erasure indices, we have that

$$\Pr[\,\mathcal{T} \text{ is good for } \mathcal{R}\,] < (1-2\alpha)^{\beta n - \alpha n} < (1-2\alpha)^{\alpha n}.$$

where the last inequality holds because $\beta > 2\alpha$. ▌

Lemma 3.9 ([DN14]) *Let $\mathcal{R}_0, \mathcal{R}_1$ be tuples with $n/2$ distinct positions each such that $u(\mathcal{R}_0) \geq 2\alpha n$ and $u(\mathcal{R}_1) \geq 2\alpha n$. The fraction of strings $v \in \{0,1\}^\ell$ that decode to subsets $\mathcal{T}$ that are good for either $\mathcal{R}_0$ or $\mathcal{R}_1$ is no larger than $4(1-2\alpha)^{\alpha n}$.*

Proof: It follows from the previous lemma and the union bound that the fraction f of subsets $\mathcal{T}$ that are good for either $\mathcal{R}_0$ or $\mathcal{R}_1$ is smaller than $2(1-2\alpha)^{\alpha n}$. Then the lemma follows straightforwardly from the fact that in the encoding scheme used there are either one or two strings mapping to each set. ▌

Since the fraction of the strings $v \in \{0,1\}^\ell$ that are good for either $\mathcal{R}_0$ or $\mathcal{R}_1$ is no larger than $4(1-2\alpha)^{\alpha n}$, we can set the security parameter t of the interactive hashing protocol as

$$t = \log(4(1-2\alpha)^{\alpha n} 2^\ell) = \ell + \alpha n \log(1-2\alpha) + 2$$

and thus have by Lemma 2.24 that the interactive hashing protocol is $(2^{-\ell})$-uniform (t,θ)-secure for

$$\theta = 2^{-(\ell - t) + O(\log \ell)} = 2^{\alpha n \log(1-2\alpha) + O(\log n)}.$$

Hence, by the security of the interactive hashing protocol, the probability that both v_0 and v_1 are good for either $\mathcal{R}_0$ or $\mathcal{R}_1$ is a negligible function of n, and so with overwhelming probability one of the tuples (w.l.o.g. $\mathcal{R}_0$) will be such that $u(\mathcal{K}_0) \geq \alpha n$.

By Lemma 2.22, two n long strings are not jointly typical if they are not jointly typical at a uniformly randomly chosen linear fraction of their positions. Hence Bob only succeeds in Alice's test in the Checking the Partitioning step (i.e., he can only find $y^{\mathcal{K}_0}$ that is jointly typical with Alice's input) if he can correctly guess y's values for the erasure positions that are jointly typical with Alice's input on these positions. Fixing an arbitrarily small $\widehat{\varepsilon}$ with $\mathsf{C_S}(W_0)/2 > \widehat{\varepsilon} > 0$, for n sufficiently large, there are for these positions at most $2^{\alpha n[H(Y \in \mathcal{Y}_0|X)+\widehat{\varepsilon}]}$ sequences of y's values that are jointly typical with Alice's input, and there are at least $2^{\alpha n[H(Y \in \mathcal{Y}_0)-\widehat{\varepsilon}]}$ typical sequences for the y's values, thus Bob's success probability is less than

$$2^{\alpha n[H(Y \in \mathcal{Y}_0|X)-H(Y \in \mathcal{Y}_0)+2\widehat{\varepsilon}]} = 2^{-\alpha n[\mathsf{C_S}(W_0)-2\widehat{\varepsilon}]},$$

which is a negligible function of n. Since Bob can only cheat with negligible probability in the case that both $u(\mathcal{R}_0), u(\mathcal{R}_1) \geq 2\alpha n$, the protocol is secure for Alice in this case.

Case 2: We assume w.l.o.g. that $\mathcal{R}_0$ is the one with $u(\mathcal{R}_0) < 2\alpha n$. The Chernoff bound guarantees that $|\mathcal{B}| > (p_\epsilon - \alpha)n$ with overwhelming probability. If $\mathcal{T}_e$ is bad for $\mathcal{R}_1$, then, by the same reasons as above, we have that Bob can only successfully pass the test performed by Alice in the Checking the Partitioning step (i.e., finding $y^{\mathcal{K}_1}$ that is jointly typical with Alice's input) with negligible probability. But if $u(\mathcal{R}_0) < 2\alpha n$, $u(\mathcal{K}_1) < \alpha n$ and $|\mathcal{B}| > (p_\epsilon - \alpha)n$, then $u(\mathcal{Q}_1) \geq (p_\epsilon - 4\alpha)n$. Then from Bob's point of view, at least $(p_\epsilon - 4\alpha)n = (\mu - 5\alpha)n$ of the positions in $\mathcal{Q}_1$ are erasures and Alice only sends him $\mu n[H(X|Y \in \mathcal{Y}_0) + \varepsilon]$ bits of information about $x^{\mathcal{Q}_1}$ through the output of g_1. Hence

$$H_\infty(X^{\mathcal{Q}_1}|\mathsf{VIEW}^{\mathsf{OT}}_{\widetilde{\mathsf{Bob}}}) > n[(\mu - 5\alpha)H(X) - \mu H(X|Y \in \mathcal{Y}_0) - \mu\varepsilon]$$

and so the use of the 2-universal hash function h_1 for extracting $n[(\mu - 5\alpha)H(X) - \mu H(X|Y \in \mathcal{Y}_0) - \mu\varepsilon - \gamma]$ bits is secure according to Lemma 2.14. Therefore the protocol is secure for Alice in this case as well. ▮

Maximizing the OT rate: For n sufficiently large, α, ε and γ can be made arbitrarily small without compromising the security, thus in the limit the strings' length can be up to

$$m = np_\epsilon[H(X) - H(X|Y \in \mathcal{Y}_0)].$$

Since the probability distribution used for X is the one achieving the Shannon capacity of W_0, this is equal to $np_\epsilon\mathsf{C_S}(W_0)$, thus proving Theorem 3.4.

3.3 Discussion

In this chapter it was proven that for OT protocols based on GECs with error probability $p_\epsilon < 1/2$ the known lower bound on the OT capacity against honest-but-curious adversaries also holds in the case of malicious adversaries. In order to prove this result, a novel usage of the interactive hashing technique suitable for channels with low erasure probability was established, which can be of interest in other scenarios. The question of determining the exact OT capacity of the generalized erasure channels with low erasure probability remains open, even for honest-but-curious adversaries, and would be an interesting direction for future research given the pivotal role of these channels in the known constructions of OT from noisy channels. Another interesting line of research would be developing new methodologies for obtaining OT from noisy channels which circumvent the need of emulating a GEC as a first step, as done in the very recent work of Khurana et al. [KMS16].

4. On the Commitment Capacity of Unfair Noisy Channels

As discussed in the introduction, the existence of noisy channels is one of the physical assumptions that enables to obtain unconditionally secure commitment protocols. However, as pointed out by Damgård et al. [DKS99], from a cryptographic perspective most of the protocols based on noisy channels have the disadvantage that they rely on the assumption that a malicious party do not interfere with the channel to try to modify its error probability.

Given this state of affairs, Damgård et al. [DKS99] introduced the more realistic channel named *Unfair Noisy Channel*, which is an generalization of the Binary Symmetric Channel and deal with the previous problem. An Unfair Noisy Channel is specified by two parameters, γ and δ, such that $0 < \gamma < \delta < \frac{1}{2}$ and is denoted as (γ, δ)-UNC. The honest parties only have the guarantee that the error probability lies in the interval $[\gamma, \delta]$, but do not know the exact value; while a malicious party can fix the error probability to any value in that range. This implies that any protocol based on a (γ, δ)-UNC has to keep its correctness and security guarantees for any error probability in the interval $[\gamma, \delta]$. This channel can be formalized as follows:

Definition 4.1 (Unfair Noisy Channels [DKS99]) *The (γ, δ)-UNC receives as input a bit x and outputs a bit y. The transition probability of the (γ, δ)-UNC is determined by an auxiliary parameter t whose alphabet are the real numbers in the interval $[\gamma, \delta]$. If the transmitter or the receiver is malicious, he can choose the value of t; otherwise it is randomly chosen and is not revealed to the parties. The transition probability is given by $P_{Y|XT}(y, x, t) = 1-t$ if $y = x$ and $P_{Y|XS}(y, x, t) = t$ if $y \neq x$.*

A (γ, δ)-UNC can equivalently be seen as the concatenation of two Binary Symmetric Channels, W_F with error probability γ and W_V with error probability θ for $0 \leq \theta \leq \frac{\delta-\gamma}{1-2\gamma}$. The error probability of the channel W_V can be controlled by a malicious party and it is unknown in the case that both parties are honest.

Damgård et al. [DKS99] proved that, on one hand, if $\delta \geq 2\gamma(1-\gamma)$ then the (γ, δ)-UNC is trivial and does not allow to build secure commitment protocols. On the other hand, if $\delta < 2\gamma(1-\gamma)$ then there is a commitment protocol based on the (γ, δ)-

UNC. In this chapter, which is based on [CDN16], we determine the commitment capacity of the Unfair Noisy Channels.

Recently a variant of UNC known as Elastic Channel has been studied. On one hand, it has two restrictions with relation to UNC: (1) only a corrupt receiver can manipulate the crossover probability to any value in the range $[\gamma, \delta]$; (2) when both parties are honest the crossover probability is always δ. On the other hand, commitment (and even oblivious transfer) can be obtained for all parameters $0 < \gamma < \delta < 1/2$ [KMS16, CDLR16].

4.1 Problem Statement

Given the value of the (γ, δ)-UNC as a cryptographic resource, it is important to study the most efficient way of using them and thus the question of investigating their commitment capacity arises. Our goal here is to determine the commitment capacity of Unfair Noisy Channels in the same way that Winter et al. [WNI03] did for discrete memoryless channels and Nascimento et al. [NBSI08] did for the Gaussian channels.

In our scenario, in addition to the bidirectional noiseless channel, the parties also have available a (γ, δ)-UNC from Alice to Bob that is used n times during the commitment phase, where n is the security parameter. In each of this n uses, Alice inputs a symbol x_i to the (γ, δ)-UNC and an output y_i is delivered to Bob. Let X^n be the random variable denoting the bit string sent through the (γ, δ)-UNC and Y^n the bit string received through the (γ, δ)-UNC.

Remark 4.2 We restrict our model to protocols where the public conversation does not depend on the channel output Y^n. This is indeed the case for all the protocols in the literature. Moreover, the public communication is used solely to prevent Alice from cheating, thus we see no reason for a commitment protocol based on noisy channels to have its public communication depending on the channel output.

Definition 4.3 *We say that two strings x^n and y^n are* ϵ-compatible *with a (γ, δ)-UNC if, for $\epsilon > 0$, $\mathsf{HD}(x^n, y^n) \leq \delta n + \epsilon$. Similarly, two random variables X^n and Y^n are said to be compatible with a (γ, δ)-UNC if* $\Pr[\mathsf{HD}(X^n, Y^n) > \delta n]$ *is negligible in n.*

The security definition considered is the one in Section 2.5, but the correctness is required to hold for any possible error probability of the UNC (i.e., for any compatible input and output of the UNC).[1] The *commitment rate* of the protocol is given by

$$\mathsf{R}_{\mathsf{com}} = \frac{\log |\mathcal{M}|}{n}.$$

where $\mathcal{M}$ is the commitment message space.

A commitment rate is said to be achievable if, for $\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}$ and λ_{B} negligible in n, there exists a $(\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}, \lambda_{\mathsf{B}})$-secure commitment protocol that achieves this rate. The *commitment capacity* of a (γ, δ)-UNC is the supremum of the achievable rates. Our

[1] For easiness of presentation the security of our protocol is argued in the stand-alone model, i.e., there is only one execution of the protocol. But the security of the commitment protocols based on noisy channels can be extended to the UC framework [Can01] in which the protocols can be composed and arbitrary protocols can be executed in parallel [DvdGMQN13, DMQN08].

result is presented below and states the commitment capacity of the (γ, δ)-UNC. The proof appears in sections 4.2 and 4.3.

Theorem 4.4 ([CDN16]) *The commitment capacity of any non-trivial (γ, δ)-unfair noisy channel is given by*

$$h(\gamma) - h(\theta), \text{ for } \theta = \frac{\delta - \gamma}{1 - 2\gamma}.$$

4.2 Protocol - Direct Part

We first prove the direct part of the theorem, i.e., we describe the protocol that achieves the commitment capacity and prove its security. The protocol follows the approach of Damgård et al. [DKS99] and uses two rounds of hash challenge-response in order to guarantee the binding property: the intuition is that the first round reduces the number of inputs that Alice can successfully open to be polynomial in the security parameter. The second round then binds Alice to one specific input. The concealing condition is achieved using a 2-universal hash function Ext chosen by Alice that is used to generate a secure key which is then applied as a one-time pad to cipher c.

Let $\theta = \frac{\delta-\gamma}{1-2\gamma}$ and let $\nu > 0$ be a parameter of the protocol. Let α_1, α_2, β be parameters such that $\alpha_1, \alpha_2 > 0$, $\beta > \alpha_1 + \alpha_2$, and $n(h(\theta) + \alpha_1)$, $n\alpha_2$ and $n(h(\gamma) - h(\theta) - \beta)$ are integers. In the following commitment protocol the message space is $\mathcal{M} = \{0, 1\}^{n(h(\gamma)-h(\theta)-\beta)}$. The protocol π_{ComUNC} is described in Figure 4.1.

Theorem 4.5 ([CDN16]) *For any $h(\gamma) - h(\theta) > \beta > 0$, by setting the other parameters appropriately and having n sufficiently large, the protocol π_{ComUNC} is $(\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}, \lambda_{\mathsf{B}})$-secure with $\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}$ and λ_{B} negligible in n and can achieve the commitment rate $h(\gamma) - h(\theta) - \beta$.*

Proof: Correctness: The protocol fails for honest parties only if $\mathsf{HD}(x^n, y^n) > \delta n + \nu n$ or $\mathsf{HD}(x^n, y^n) < \gamma n - \nu n$. As the (γ, δ)-UNC has error probability less than or equal to δ, the expectation of $\mathsf{HD}(x^n, y^n)$ is less than or equal to δn. Thus the Chernoff bound guarantees that

$$\Pr\left[\,\mathsf{HD}(x^n, y^n) > \delta n + \nu n\,\right]$$

is a negligible function of n. For similar reasons,

$$\Pr\left[\,\mathsf{HD}(x^n, y^n) < \gamma n - \nu n\,\right]$$

is a negligible function of n.

Hiding: For any $\eta > 0$ and n sufficiently large, we have that

$$\begin{aligned}
H_\infty^\varepsilon(X^n|G_1(X^n), G_2(X^n), Y^n, G_1, G_2) &\geq H_\infty(X^n, G_1(X^n), G_2(X^n)|Y^n, G_1, G_2) \\
&\quad - H_0(G_1(X^n), G_2(X^n)|Y^n, G_1, G_2) - \log(\varepsilon^{-1}) \\
&= H_\infty(X^n|Y^n, G_1, G_2) - H_0(G_1(X^n), G_2(X^n)|Y^n, G_1, G_2) - \log(\varepsilon^{-1}) \\
&= H_\infty(X^n|Y^n) - H_0(G_1(X^n), G_2(X^n)|Y^n, G_1, G_2) - \log(\varepsilon^{-1}) \\
&\geq n(h(\gamma) - \eta) - n(h(\theta) + \alpha_1 + \alpha_2) - \log(\varepsilon^{-1}) \\
&= n(h(\gamma) - h(\theta) - \eta - \alpha_1 - \alpha_2) - \log(\varepsilon^{-1})
\end{aligned}$$

Secure Commitment Protocol π_{ComUNC}

Commitment Phase: Alice wants to commit to the binary string $m \in \mathcal{M}$. The parties proceed as follows:

1. Alice chooses a random binary string $x^n = (x_1, \ldots, x_n)$ of dimension n and for $1 \leq i \leq n$ sends the bit x_i to Bob over the (γ, δ)-UNC.

2. Bob receives the string $y^n = (y_1, \ldots, y_n)$ sent over the (γ, δ)-UNC, chooses uniformly at random a function g_1 of the class of $4n$-universal hash functions $\mathcal{G}_1 : \{0,1\}^n \rightarrow \{0,1\}^{n(h(\theta)+\alpha_1)}$, and sends the description of g_1 to Alice over the noiseless channel.

3. Alice computes $e_1 = g_1(x^n)$ and sends it to Bob.

4. Bob chooses uniformly at random a function g_2 of the class of 2-universal hash functions $\mathcal{G}_2 : \{0,1\}^n \rightarrow \{0,1\}^{n\alpha_2}$, and sends its description to Alice over the noiseless channel.

5. Alice chooses uniformly at random a two-universal hash function $\mathsf{Ext} : \{0,1\}^n \rightarrow \{0,1\}^{n(h(\gamma)-h(\theta)-\beta)}$, computes $d = m \oplus \mathsf{Ext}(x^n)$ and $e_2 = g_2(x^n)$, and sends d, e_2 and the description of Ext to Bob over the noiseless channel.

Opening Phase: Alice wants to reveal the value of $\tilde{m}$ to Bob. The parties proceeds as follow:

1. Alice sends to Bob over the noiseless channel the values $\tilde{x}^n$ and $\tilde{m}$ that she claims to be the ones used in the commitment phase.

2. Bob checks if $\gamma n - \nu n \leq \mathsf{HD}(\tilde{x}^n, y^n) \leq \delta n + \nu n$, if $g_1(\tilde{x}^n) = e_1$, $g_2(\tilde{x}^n) = e_2$ and if $\tilde{m} = \mathsf{Ext}(\tilde{x}^n) \oplus d$. Bob accepts if there are no problems in the tests.

Figure 4.1: The commitment protocol π_{ComUNC}.

where the first inequality follows from the chain rule for smooth entropy, the first equality from the fact that $G_1(X^n), G_2(X^n)$ are functions of G_1, G_2 and X^n, the second equality from the fact that X^n is independent of G_1, G_2 given Y^n and the last inequality follows from the facts that the error probability of the channel is at least γ, the range of G_1 has cardinality $2^{n(h(\theta)+\alpha_1)}$ and the range of G_2 has cardinality $2^{n\alpha_2}$.

Setting $\varepsilon = 2^{-\psi n}$ (with $\psi > 0$), for n sufficiently large, the security of the key obtained by applying the hash function $\mathsf{Ext} : \{0,1\}^n \rightarrow \{0,1\}^{n(h(\gamma)-h(\theta)-\beta)}$ to x follows from Lemma 2.14 as $\beta > \alpha_1 + \alpha_2$ and ψ and η can be arbitrarily small for n sufficiently large. Note that having a negligible statistical distance is equivalent to having a negligible mutual information [DPP98].

Binding: A dishonest Alice can cheat successfully only if she finds two different strings $\overline{x}^n$ and $\widetilde{x}^n$ such that $\gamma n - \nu n \leq \mathsf{HD}(\overline{x}^n, y^n) \leq \delta n + \nu n$, $\gamma n - \nu n \leq \mathsf{HD}(\widetilde{x}^n, y^n) \leq \delta n + \nu n$, and both pass the sequentially performed hash challenge-response tests, for arbitrarily small ν and sufficiently large n. We can assume without loss of generality that Alice sets the error probability of the channel to γ when she sends x^n. In the typicality test an honest Bob accepts any string that is jointly typical with y^n for any error probability $\gamma \leq \rho \leq \delta$. So Alice can cheat only if she finds two strings $\overline{x}^n$ and $\widetilde{x}^n$ so that both pass the hash tests and are jointly typical with x^n for Binary Symmetric Channels with error probabilities $0 \leq \overline{\tau} \leq \theta$ and $0 \leq \widetilde{\tau} \leq \theta$. The number of such jointly typical strings is upper bounded by $2^{n(h(\theta)+\varepsilon')}$ for any $\varepsilon' > 0$ and n sufficiently large. We fix $\alpha_1 > \varepsilon'$.

Let the viable set denote the channel inputs that Alice can possibly open to Bob and he would accept. If there were no hash checks, the viable set would have at most $2^{n(h(\theta)+\varepsilon')}$ elements. Lets consider this initial viable set. The goal of the first round of hash challenge-response is to, with overwhelming probability, reduce the number of elements of the viable set to at most $8n+1$. In this first round, Alice has to commit to one arbitrary value e_1 for the output of the hash function g_1. Considering the j-th viable string before this first round, we define I_j as 1 if that string is mapped to e_1 by g_1; and $I_j = 0$ otherwise. Let $I = \sum_j I_j$. Clearly $\mu = E[I] < 1$ as $\alpha_1 > \varepsilon'$. Let e_1 be considered bad if I is bigger than $8n+1$. Given that g_1 is $4n$-wise independent, by applying Lemma 2.33 with $t = 4n$ and $A = 2t = 8n$, we get

$$\Pr[I > 8n+1] \; < \; O\left(\left(\frac{t\mu + t^2}{(2t)^2}\right)^{t/2}\right) < O\left(\left(\frac{1+t}{4t}\right)^{t/2}\right) < O\left(2^{-t/2}\right).$$

Then the probability that any e_1 is bad is upper bounded by

$$O\left(2^{n(h(\theta)+\alpha_1)}2^{-t/2}\right) < O\left(2^{-n}\right).$$

But if the viable set contains at most $8n+1$ elements after the first hash challenge-response round, the probability that some of those collide in the second hash challenge-response round is upper bounded by

$$(8n+1)^2\, 2^{-\alpha_2 n},$$

which is negligible in n.

Commitment Rate: For n sufficiently large, α_1 and α_2 can be made arbitrarily small, and thus β can also be made arbitrarily small while preserving the security of the protocol. Therefore it is possible to achieve the commitment rate $h(\gamma) - h(\theta) - \beta$ for any $h(\gamma) - h(\theta) > \beta > 0$. ▌

4.3 Converse

For proving the converse, we will assume a specific cheating behavior for Alice. As we are interested in proving an upper bound in the commitment capacity, restricting Alice's behavior will only strength our result. Let $k = \log |\mathcal{M}|$ and M be uniformly random over $\mathcal{M}$. Let X^n be a random variable representing the data Alice inputs into the Unfair Noisy Channel. Assume, Alice sets the noise level of the Unfair Noisy Channel connecting her to Bob to γ. Let Y^n be a random variable obtained by passing X^n through the Unfair Noisy Channel (Channel 1). Let Z^n be a random variable obtained by passing X^n through a Binary Symmetric Channel with error probability equal to θ with $0 \leq \theta \leq \frac{\delta-\gamma}{1-2\gamma}$ (Channel 2). Denote the conversation over the public authenticated and noiseless channel by T.

In the case of commitments based on fair noisy channels, it was proved by Winter et al. [WNI03] that after the commit phase is finished, if Bob is presented with Alice's inputs to the channel, X^n, he is able to obtain almost complete knowledge on the committed value M. Here we will prove that in the case of Unfair Noisy Channels if Bob is presented with a noisy version of X^n he is still able to compute the committed value M with high probability.

Lemma 4.6 *$H(M|Z^nT) \leq 1 + kp$ for p negligible in n.*

Proof: Let M, X^n, Y^n, Z^n and T be defined as above. We first give a procedure so that the commitment M can be estimated with high probability from Z^n, Y^n and T.

The procedure is as follows. Let test be the test Bob performs during the opening phase. Given Z^n, Y^n and T, compute the value m that maximizes

$$\Pr\left[\mathsf{test}(Z^n, Y^n, T, m) = 1\right],$$

breaking ties in an arbitrary way. Because of the binding condition, we know that no two different values $\overline{m}$ and $\widetilde{m}$ will have

$$\Pr\left[\mathsf{test}\left(\overline{X}^n, Y^n, T, \overline{m}\right) = 1\right] \geq \lambda_{\mathsf{B}}$$

and

$$\Pr\left[\mathsf{test}\left(\widetilde{X}^n, Y^n, T, \widetilde{m}\right) = 1\right] \geq \lambda_{\mathsf{B}}$$

for all $\widetilde{X}^n$ and $\overline{X}^n$ compatible with Y^n.

Moreover, from the correctness property of the protocol and from the fact that Z^n and Y^n are compatible for the Unfair Noisy Channel in question, we know that for the correct value m we have

$$\Pr\left[\mathsf{test}(Z^n, Y^n, T, m) = 1\right] \geq 1 - \lambda_{\mathsf{C}}.$$

Thus, with high probability this procedure will give us the right committed value m. Let p be the probability that this procedure returns a wrong value. Using Fano's inequality we get

$$\begin{aligned} H(M|Z^nY^nT) &\leq h(p) + p\log|\mathcal{M}| \\ &\leq 1 + p\log|\mathcal{M}| \\ &\leq 1 + kp. \end{aligned}$$

One can prove that the output of the channel Y^n is not needed in the above described procedure. Given the assumed independence of the public conversation T and Y^n, we have that given Z^n one can locally simulate Y^n by passing Z^n through a Binary Symmetric Channel with error probability γ. Denote the output of the simulated channel by $\check{Y}^n$. Note that $\check{Y}^n$ and X^n are compatible. Moreover, given the fact that the public conversation is independent of Y^n one has, from the correctness property, that

$$\Pr\left[\mathsf{test}(Z^n, \check{Y}^n, T, m) = 1\right] \geq 1 - \lambda_{\mathsf{C}}.$$

From bindingness we know that no two different values $\overline{m}$ and $\widetilde{m}$ will have

$$\Pr\left[\mathsf{test}\left(\overline{X}^n, \check{Y}^n, T, \overline{m}\right) = 1\right] \geq \lambda_{\mathsf{B}}$$

and

$$\Pr\left[\mathsf{test}\left(\widetilde{X}^n, \check{Y}^n, T, \widetilde{m}\right) = 1\right] \geq \lambda_{\mathsf{B}}$$

for all $\widetilde{X}^n$ and $\overline{X}^n$ compatible with $\check{Y}^n$.

Again, using Fano's inequality we get

$$\begin{aligned} H(M|Z^n\check{Y}^nT) &\leq h(p) + p\log|\mathcal{M}| \\ &\leq 1 + p\log|\mathcal{M}| \\ &\leq 1 + kp. \end{aligned}$$

Because the Markov chain $MX^n \leftrightarrow Z^n \leftrightarrow \check{Y}^n$ holds, we have that $H(M|Z^n\check{Y}^nT) = H(M|Z^nT)$, which proves our result. ▮

We have that

$$\begin{aligned} k &\leq H(M|Y^nT) + \lambda_{\mathsf{H}} \\ &= H(M|Y^nT) - H(M|Z^nT) + H(M|Z^nT) + \lambda_{\mathsf{H}} \\ &\leq H(M|Y^nT) - H(M|Z^nT) + 1 + kp + \lambda_{\mathsf{H}} \\ &= H(M|Y^nT) - H(M|Z^nT) - H(M|T) + H(M|T) + 1 + kp + \lambda_{\mathsf{H}} \\ &= I(M;Z^n|T) - I(M;Y^n|T) + 1 + kp + \lambda_{\mathsf{H}} \end{aligned}$$

where the first inequality comes from the λ_{H}-hiding requirement and the second from the previous lemma.

The expression $I(M;Z^n|T) - I(M;Y^n|T)$ is then developed using the same steps as in Section V of the seminal work of Csiszár and Körner [CK78]; the details are included for the sake of completeness. Let Z^i denote $Z_1 \ldots Z_i$ and $\widehat{Y}^i$ denote

$Y_i \dots Y_n$. We expand $I(M;Z^n|T)$ starting from $I(M;Z_1|T)$ and $I(M;Y^n|T)$ starting from $I(M;Y_n|T)$

$$\begin{aligned}
I(M;Z^n|T) &= \sum_{i=1}^{n} I(M;Z_i|TZ^{i-1}) \\
&= \sum_{i=1}^{n} \Big[H(Z_i|TZ^{i-1}) - H(Z_i|TZ^{i-1}M) \\
&\qquad -H(Z_i|TZ^{i-1}M\widehat{Y}^{i+1}) + H(Z_i|TZ^{i-1}M\widehat{Y}^{i+1})\Big] \\
&= \sum_{i=1}^{n} \Big[I(M\widehat{Y}^{i+1};Z_i|TZ^{i-1}) - I(\widehat{Y}^{i+1};Z_i|TZ^{i-1}M)\Big] \\
&= \sum_{i=1}^{n} \Big[I(M;Z_i|TZ^{i-1}\widehat{Y}^{i+1}) \\
&\qquad +I(\widehat{Y}^{i+1};Z_i|TZ^{i-1}) - I(\widehat{Y}^{i+1};Z_i|TZ^{i-1}M)\Big].
\end{aligned}$$

Similarly we obtain

$$\begin{aligned}
I(M;Y^n|T) &= \sum_{i=1}^{n} \Big[I(M;Y_i|TZ^{i-1}\widehat{Y}^{i+1}) \\
&\qquad +I(Z^{i-1};Y_i|T\widehat{Y}^{i+1}) - I(Z^{i-1};Y_i|T\widehat{Y}^{i+1}M)\Big].
\end{aligned}$$

We have that

$$\begin{aligned}
\sum_{i=1}^{n} I(\widehat{Y}^{i+1};Z_i|TZ^{i-1}) &= \sum_{i=1}^{n}\sum_{j=i+1}^{n} I(Y_j;Z_i|TZ^{i-1}\widehat{Y}^{j+1}) \\
&= \sum_{j=2}^{n}\sum_{i=1}^{j-1} I(Y_j;Z_i|TZ^{i-1}\widehat{Y}^{j+1}) \\
&= \sum_{j=1}^{n} I(Z^{j-1};Y_j|T\widehat{Y}^{j+1}).
\end{aligned}$$

Similarly we can get that

$$\sum_{i=1}^{n} I(\widehat{Y}^{i+1};Z_i|TZ^{i-1}M) = \sum_{j=1}^{n} I(Z^{j-1};Y_j|T\widehat{Y}^{j+1}M).$$

Therefore

$$I(M;Z^n|T) - I(M;Y^n|T) = \sum_{i=1}^{n} \Big[I(M;Z_i|TZ^{i-1}\widehat{Y}^{i+1}) - I(M;Y_i|TZ^{i-1}\widehat{Y}^{i+1})\Big].$$

Letting L be a random variable uniformly distributed in $\{1,\dots,n\}$ and independent of $MTX^nY^nZ^n$, and setting $U \triangleq TZ^{L-1}\widehat{Y}^{L+1}L$, $V \triangleq UM$, $X \triangleq X_L$, $Y \triangleq Y_L$ and $Z \triangleq Z_L$ we get that $U \leftrightarrow V \leftrightarrow X \leftrightarrow YZ$ form a Markov chain and

$$\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n} \Big[I(M;Z_i|TZ^{i-1}\widehat{Y}^{i+1}) - I(M;Y_i|TZ^{i-1}\widehat{Y}^{i+1})\Big] &= I(M;Z|U) - I(M;Y|U) \\
&= I(V;Z|U) - I(V;Y|U).
\end{aligned}$$

Putting everything together, for any $(\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}, \lambda_{\mathsf{B}})$-secure commitment protocol with $\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}$ and λ_{B} negligible in n, there are $U \leftrightarrow V \leftrightarrow X \leftrightarrow YZ$ such that

$$\frac{k}{n} \leq I(V;Z|U) - I(V;Y|U) + \varepsilon$$

where $\varepsilon = \frac{1+kp+\lambda_{\mathsf{H}}}{n}$ goes to 0 for n sufficiently large.

We now set $\theta = \frac{\delta-\gamma}{1-2\gamma}$. In our case channel 2 is less noisy than channel 1, therefore maximizing over all $U \leftrightarrow V \leftrightarrow X \leftrightarrow YZ$ we get

$$\begin{aligned} I(V;Z|U) - I(V;Y|U) &= I(V;Z) - I(V;Y) - [I(U;Z) - I(U;Y)] \\ &= I(X;Z) - I(X;Y) \\ &\quad -[I(X;Z|V) - I(X;Y|V)] - [I(U;Z) - I(U;Y)] \\ &\leq I(X;Z) - I(X;Y) \\ &\leq h(\gamma) - h(\theta) \end{aligned}$$

where the first inequality comes from the fact that both expressions in the brackets are non-negative since channel 2 is less noisy than channel 1 and the second inequality follows taking the maximum over X. Hence

$$\frac{k}{n} \leq h(\gamma) - h(\theta) + \varepsilon,$$

where ε goes to 0 for n sufficiently large and this completes the proof of the reverse.

4.4 Discussion

In this chapter we obtained the commitment capacity of the Unfair Noisy Channels. Other open problems are to determine the range of parameters for which Unfair Noisy Channels are non-trivial for performing OT as well as to obtain their OT capacity. Deriving the commitment capacity of Weak Channels [Wul09] is also an interesting open problem. In the case of Elastic Channels, for commitments from Alice to Bob, the channel is essentially degraded to a Binary Symmetric Channel with crossover probability γ and therefore the commitment capacity is $h(\gamma)$. On the other hand, we conjecture that the commitment capacity for commitments from Bob to Alice is $h(\delta) - h(\theta)$ for $\theta = \frac{\delta-\gamma}{1-2\gamma}$. However, if either of the two restrictions of the Elastic Channels in relation to unfair noisy channels is discarded (i.e., only the receiver being able to set the crossover probability; and the crossover probability being fixed to δ when both parties are honest), then we get back to the same commitment capacity as for the Unfair Noisy Channels.

5. Commitment and OT in the Bounded Storage Model with Errors

The Bounded Storage Model (BSM) is an interesting model that allows to achieve unconditionally secure protocols by making the assumption that the memory of the parties (even the adversarial ones) are bounded and that a public random string is available to the parties during an initial transmission phase. The security holds even if the parties get infinite storage capacity after this transmission phase. Protocols for many cryptographic tasks such as key agreement [CM97, DM08], OT [CCM98, Din01, HCR02, DHRS04] and commitment [SY11, Alv10] were obtained. The weakness of the BSM is that it assumes that exactly the same random source is available to all the protocol participants, but reliably broadcasting can be hard to realize in practice. In this chapter we consider the more realistic BSM with errors, in which errors can be present in the public random string in arbitrary positions (either due to errors in the channel or deliberately introduced by an adversary); the only guarantee is that the error frequency is not too big. Ding [Din05] developed the first key agreement protocol in this model. This chapter is based on [DLN14, DLN15] and presents the first protocols for commitment and OT in the BSM with errors.

5.1 Problem Statement

In the Bounded Storage Model with errors, a transmission phase is executed prior to the realization of the protocol's main part.

Transmission Phase: In this phase, Alice has access to a sample $x \in \{0,1\}^\ell$ of an $\alpha\ell$-source X with $0 < \alpha < 1$ and Bob to $\widetilde{x} \in \{0,1\}^\ell$ such that $\mathsf{HD}(x, \widetilde{x}) \leq \delta\ell$. Note that this captures both the situation where the source is noisy and the situation where the adversary controls part of the source. The classical assumption in the Bounded Storage Model is that both parties have a memory bound during this phase, but here we will be able to prove the security of our protocols even with a weaker assumption that only bounds the memory of one of the parties[1] (which one specifically depends on the protocol). If the memory bound is on Bob, for a fixed $\gamma < \alpha$, Bob computes a randomized function $\widetilde{f}(\widetilde{x})$ with output size at most $\gamma\ell$, stores its output and discards $\widetilde{x}$. Similarly if the memory bound is on Alice, she computes a randomized function $f(x)$ with output size at most $\gamma\ell$, stores its output and discards x.

We show that techniques for the Bounded Storage Model with errors that Ding [Din05] originally introduced in the context of key extension can also give us efficient protocols for implementing OT. Our OT protocol only assumes a memory bound on Bob. It is based on an efficient linear error correcting code proposed by Guruswami and Indyk [GI02] that has rate σ and achieves the Zyablov bound. We show that as long as $\sigma > 1 - \alpha - \gamma$ the protocol works for noise levels δ as severe as

$$\max_{\sigma < \widetilde{\sigma} < 1} \frac{(1-\widetilde{\sigma})y}{2},$$

where y is the unique value in $[0, 1/2]$ so that $h(y) = 1 - \sigma/\widetilde{\sigma}$. If a random linear error correcting code is used, an improved noise level can be tolerated

$$h(2\delta) < \alpha - \gamma,$$

but this improvement in the resilience comes at the price of making the protocol inefficient from a computational complexity point of view as the problem of decoding random linear codes is intractable. The protocol is described in Section 5.5.

Given that OT is complete, the OT protocol immediately gives us commitment schemes. However, this is not the most desirable solution as the communication, round and computational complexities of OT protocols are usually much higher than the ones for commitment schemes. Moreover, it could be the case that commitment protocols could work for different ranges of noise δ.

For the above reasons, we also design a direct construction of a commitment protocol that does not rely on the framework proposed by Ding [Din05], does not use error correcting codes at all, implements *string* commitment and has only one message from Bob to Alice. Again, we assume that Bob has limited memory; no limitations are impose on Alice whatsoever. The protocol, which is specified in Section 5.2, is very efficient and simple and works for

$$h(\delta) < \frac{\alpha - \gamma}{2}.$$

We then show in Section 5.3 that it is possible to obtain a protocol that works for a much larger range of noise

$$h(\delta) < \alpha - \gamma$$

at the cost of having one additional message in each direction and by using a family of $4k$-universal hash functions. Finally, we show in Section 5.4 that the use of families of $4k$-universal hash functions can be avoided by imposing a memory bound on Alice, instead of Bob. This protocol is based on the interactive hashing protocol of [DHRS04] (which was also used by Shikata and Yamanaka [SY11] for obtaining commitment in the Bounded Storage Model without errors) and also works for

$$h(\delta) < \alpha - \gamma,$$

[1] In our protocols the honest parties do not need unlimited memory.

Secure Commitment Protocol π_{ComBSM}

Transmission phase:

1. Alice chooses k positions from x uniformly at random. Similarly, Bob samples k positions from $\widetilde{x}$. Let $\mathcal{A}$ and $\mathcal{B}$ denote their sets of positions.

Commitment phase:

1. Alice announces $\mathcal{A}$ to Bob.
2. Bob chooses $g \xleftarrow{\$} \mathcal{G}$ and sends its description to Alice.
3. Alice computes $p \leftarrow g(x^{\mathcal{A}})$, $u \xleftarrow{\$} \{0,1\}^r$, $y \leftarrow \mathsf{Ext}(x^{\mathcal{A}}, u)$ and $z = m \oplus y$, and sends (z, p, u) to Bob in order to commit to m.

Opening phase:

1. Alice sends m' and w to Bob, which are defined as $m' = m$ and $w = x^{\mathcal{A}}$ in the case that she is honest.
2. Let $\mathcal{C} = \mathcal{A} \cap \mathcal{B}$, $c = |\mathcal{C}|$. Bob verifies whether $c \geq n$, $\mathsf{HD}(w^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) \leq (\delta + \zeta)c$, $p = g(w)$ and $m' = \mathsf{Ext}(w, u) \oplus z$. If any verification fails Bob outputs 0, otherwise he outputs 1.

Figure 5.1: The commitment protocol π_{ComBSM}.

but has extra rounds of communication and implements *bit* rather than string commitment.

The techniques we use in our results are standard in the field: extractors, error-correcting codes, typicality tests, sampling, etc. However, to the best of our knowledge, this is the first time that these techniques are combined to obtain commitment and OT protocols in the memory bounded model with errors. Moreover, the study of how much adversarial noise can be tolerated in this model and its relation to round complexity is also original, as far as we know. Interestingly, the noise levels tolerated by our protocols are different for OT and commitment schemes. This contrasts sharply with the noiseless situation where either one has every possible secure two-party computation or nothing at all.

5.2 A Simple String Commitment Protocol

Next we present a quite simple string commitment protocol that only involves one message from Bob to Alice. The security definition for commitment that is considered is the one discussed in Section 2.5 and a memory bound is imposed on Bob. The idea of our commitment protocol is the following. First, Alice and Bob sample a number of bits from the public random source. Alice then extracts some private randomness from her sample and uses it as an one-time pad to conceal her commitment before sending it to Bob; thus guaranteeing the hiding property. Additionally she computes a hash of her sample, where the hash function is chosen by Bob, and sends it to Bob; this hash together with the tests performed by Bob during the opening phase guarantee the binding property. In the opening phase, Alice sends her committed value and her sampled string. Bob then performs a number of checks for consistency.

The security parameter is n and k is set as $k = 2\sqrt{\ell n}$. Fix $\varepsilon' > 0$ and let $\rho = \alpha - \gamma - \frac{1+\log(1/\varepsilon')}{\ell}$. Fix τ such that $\frac{\rho}{3} \geq \tau > 0$, and $1 > \omega, \zeta > 0$ such that $\rho - 3\tau > \omega > 2h(\delta + \zeta)$ and $\delta + \zeta < 1/2$. Let $\kappa = (\rho - 3\tau - \omega)k$ and $\ell_m = (1 - \psi)\kappa$ for $\psi > 0$. The commitment message space is $\mathcal{M} = \{0,1\}^{\ell_m}$. The protocol π_{ComBSM}, whose detailed description is in Figure 5.1, assumes that the following functionalities, which are possible due to the lemmas in Section 2.3, are available to the parties:

- A family $\mathcal{G}$ of 2-universal hash functions $g\colon \{0,1\}^k \to \{0,1\}^{\omega k}$.
- A $(\kappa, \varepsilon_{\mathsf{Ext}})$-strong extractor $\mathsf{Ext}\colon \{0,1\}^k \times \{0,1\}^r \to \{0,1\}^{\ell_m}$, for an arbitrary $\varepsilon_{\mathsf{Ext}} > e^{-k/2^{O(\log^* k)}}$.

Remark 5.1 Note that it should hold that $2h(\delta) < \omega + 3\tau < \rho < \alpha - \gamma$, so the protocol is only possible if $2h(\delta) < \alpha - \gamma$.

Theorem 5.2 ([DLN15]) *The protocol π_{ComBSM} is $(\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}, \lambda_{\mathsf{B}})$-secure for $\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}$ and λ_{B} negligible in n.*

Proof: Correctness: It is clear that if both Alice and Bob are honest, the protocol fails only in the case that $c < n$ or $\mathsf{HD}(x^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) > (\delta + \zeta)c$. By Lemma 2.31, $c \geq n$ except with probability at most $e^{-n/4}$. By Lemma 2.30, $\mathsf{HD}(x^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) \leq (\delta + \zeta)c$ except with probability at most $e^{-c\zeta^2/2}$, which is negligible in n if $c \geq n$.

Hiding: After the commitment phase, (a possibly malicious) Bob possesses $(z, p, u, \mathcal{A})$, g and the output of a function $\widetilde{f}(\cdot)$ of $\widetilde{x}$ with $|\widetilde{f}(\widetilde{x})| \leq \gamma n$ for $\gamma < \alpha$. The only random variable that can provide mutual information about M when conditioned on $\widetilde{X}$ is Z, but we prove below that Z is almost uniform from Bob's point of view, and so it works as an one-time pad and only negligible information can be leaked.

By Lemma 2.6,

$$R_\infty^{\varepsilon'}(X|\widetilde{f}(\widetilde{X})) \geq \alpha - \gamma - \frac{1 + \log(1/\varepsilon')}{\ell} = \rho.$$

Since Alice chooses $\mathcal{A}$ randomly and this is an $(\mu, \nu, e^{-k\nu^2/2})$-averaging sampler for any $\mu, \nu > 0$ according to Lemma 2.10, by setting $\mu = \frac{\rho - 2\tau}{\log(1/\tau)}$, $\nu = \frac{\tau}{\log(1/\tau)}$, we have by Lemma 2.9 that

$$R_\infty^{\varepsilon''+\varepsilon'}(X^{\mathcal{A}}|\mathcal{A}, \widetilde{f}(\widetilde{X})) \geq \rho - 3\tau,$$

where ε'' is a negligible function of k.

It holds that

$$\begin{aligned}
H_\infty^{\varepsilon''+\varepsilon'}(X^{\mathcal{A}}|G(X^{\mathcal{A}}), \mathcal{A}, U, G, \widetilde{f}(\widetilde{X})) &= H_\infty^{\varepsilon''+\varepsilon'}(X^{\mathcal{A}}|G(X^{\mathcal{A}}), \mathcal{A}, \widetilde{f}(\widetilde{X})) \\
&\geq H_\infty^{\varepsilon''+\varepsilon'}(X^{\mathcal{A}}|\mathcal{A}, \widetilde{f}(\widetilde{X})) - H_0(G(X^{\mathcal{A}})) \\
&\geq (\rho - 3\tau - \omega)k \\
&= \kappa.
\end{aligned}$$

Therefore, setting ε' and $\varepsilon_{\mathsf{Ext}}$ to be negligible in n, the use of the strong extractor to obtain y that is xored with the message guarantees that only negligible information about the committed message can be leaked according to Definition 2.11.

Binding: The protocol is binding if, after the commitment phase, Alice cannot choose between two different values to successfully open. Let $\sigma = \delta + \zeta$. The only way Alice can cheat is if she can come up with two strings w, w' such that $g(w) = g(w')$, $\mathsf{HD}(w^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) \leq \sigma c$ and $\mathsf{HD}(w'^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) \leq \sigma c$ (with $c \geq n$). If this happens, it holds that either there are two strings w, w' such that $g(w) = g(w')$, $\mathsf{HD}(w, \widetilde{x}^{\mathcal{A}}) \leq \sigma k$ and $\mathsf{HD}(w', \widetilde{x}^{\mathcal{A}}) \leq \sigma k$; or Alice without knowing the set $\mathcal{B}$ that together with $\mathcal{A}$ determines $\mathcal{C}$ can compute w such that $\mathsf{HD}(w, \widetilde{w}^{\mathcal{A}}) > \sigma k$ and $\mathsf{HD}(w^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) \leq \sigma c$. We prove below that the probability that Alice succeeds in cheating decreases exponentially with the security parameter n (or, equivalently in k, c). First the probability that there exists two different strings w, w' both within Hamming distance σk from $\widetilde{x}^{\mathcal{A}}$ and such that $g(w) = g(w')$ is upper bounded by

$$\begin{aligned}\Pr\left[\exists w, w' \text{ s.t. } \begin{cases} w \neq w' \\ g(w) = g(w') \\ \mathsf{HD}(w, \widetilde{x}^{\mathcal{A}}) \leq \sigma k \\ \mathsf{HD}(w', \widetilde{x}^{\mathcal{A}}) \leq \sigma k \end{cases}\right] &= \sum_{w\,:\,\mathsf{HD}(w,\widetilde{x}^{\mathcal{A}}) \leq \sigma k} \;\; \sum_{w' \neq w\,:\,\mathsf{HD}(w',\widetilde{x}^{\mathcal{A}}) \leq \sigma k} 2^{-\omega k} \\ &\leq 2^{-(\omega - 2h(\sigma))k},\end{aligned}$$

where Lemma 2.32 was used to obtain the inequality. By design, it holds that $\omega > 2h(\sigma)$ and therefore the probability that Alice successfully cheats by finding two strings that are at distance at most σk from $\widetilde{x}^{\mathcal{A}}$ and hash to the same value is negligible in k.

Now considering the second case, by assumption w has Hamming distance $(\sigma + \psi)k$ from $\widetilde{x}^{\mathcal{A}}$ for some $\psi > 0$. Since Bob is honest, $\mathcal{B}$ is chosen randomly. Hence Lemma 2.30 can be applied and thus the probability that $\mathsf{HD}(w^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) \leq \sigma c$ is smaller than $e^{-c\psi^2/2}$. ▌

Remark 5.3 In the case of non-rushing adversaries that behave honestly in the commitment phase, it is possible to relax the condition $2h(\delta) < \alpha - \gamma$ to $h(\delta) < \alpha - \gamma$. This follows from the fact that, in order to break the binding condition, the adversary has to find a string that has small Hamming distance and hashes to one *specific* value, instead of finding any two strings with small Hamming distance that hash to the same value.

5.3 Extending the Feasibility Region

We next present a more elaborate version of the protocol that has more rounds of communication, but works for $h(\delta) < \alpha - \gamma$ even if the adversaries are not honest during the commitment phase. The memory bound is still on Bob. The idea for guaranteeing the binding property is to use two rounds of hash challenge-responses in order to guarantee the binding condition. Consider the initial set of viable strings that Alice can possibly send to Bob during the commitment phase and would pass the Hamming distance test. The first hash challenge-response round binds Alice to

Secure Commitment Protocol π_{ComBSME}

Transmission phase:

1. Alice chooses uniformly k positions from X. Similarly, Bob samples k positions from $\widetilde{X}$. We call their sets of positions $\mathcal{A}$ and $\mathcal{B}$, respectively.

Commitment phase:

1. Alice announces $\mathcal{A}$ to Bob.
2. Bob chooses $g_1 \xleftarrow{\$} \mathcal{G}_1$ and sends its description to Alice.
3. Alice computes $p_1 \leftarrow g_1(x^{\mathcal{A}})$ and sends it to Bob.
4. Bob chooses $g_2 \xleftarrow{\$} \mathcal{G}_2$ and sends its description to Alice.
5. Alice computes $p_2 \leftarrow g_2(x^{\mathcal{A}})$, $u \xleftarrow{\$} \{0,1\}^r$, and $y \leftarrow \mathsf{Ext}(x^{\mathcal{A}}, u)$. She then computes $z = m \oplus y$ and sends (z, p_2, u) to Bob in order to commit to m.

Opening phase:

1. Alice sends m' and w to Bob, which are defined as $m' = m$ and $w = x^{\mathcal{A}}$ in the case that she is honest.
2. Let $\mathcal{C} = \mathcal{A} \cap \mathcal{B}$, $c = |\mathcal{C}|$ and $w^{\mathcal{C}}$ be the restriction of w to the positions corresponding to the set $\mathcal{C}$. Bob verifies whether $c \geq \ell$, $\mathsf{HD}(w^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) \leq (\delta + \zeta)c$, $p_1 = g_1(w)$, $p_2 = g_2(w)$ and $m' = \mathsf{Ext}(w, u) \oplus z$. If any verification fails Bob outputs 0, otherwise he outputs 1.

Figure 5.2: The commitment protocol π_{ComBSME}.

one specific output of the hash function, and thus restrict the set of viable strings to be polynomial in the security parameter. The second hash challenge-response round then binds Alice to one specific value for the commitment. This approach was used before in a different context [DKS99].

The security parameter is n and k is set as $k = 2\sqrt{\ell n}$. Fix $\varepsilon' > 0$ and let $\rho = \alpha - \gamma - \frac{1+\log(1/\varepsilon')}{\ell}$. Fix τ such that $\frac{\rho}{3} \geq \tau > 0$, and $\omega_1, \omega_2, \zeta > 0$ such that $\rho - 3\tau > \omega_1 + \omega_2$, $\omega_1 > h(\delta + \zeta)$, and $\delta + \zeta < 1/2$. Let $\kappa = (\rho - 3\tau - \omega_1 - \omega_2)k$ and for $\psi > 0$, $\ell_m = (1 - \psi)\kappa$. The message space is $\mathcal{V} = \{0,1\}^{\ell_m}$. The protocol π_{ComBSME}, whose detailed description is in Figure 5.2, assumes that the following functionalities, which are possible due to the lemmas in Section 2.3, are available to the parties:

- A family $\mathcal{G}_1$ of $4k$-universal hash functions $g_1 \colon \{0,1\}^k \to \{0,1\}^{\omega_1 k}$.
- A family $\mathcal{G}_2$ of 2-universal hash functions $g_2 \colon \{0,1\}^k \to \{0,1\}^{\omega_2 k}$.
- A (κ, ε_E)-strong extractor $\mathsf{Ext} \colon \{0,1\}^k \times \{0,1\}^r \to \{0,1\}^{\ell_m}$, for an arbitrary

$\varepsilon_E > e^{-k/2^{O(\log^* k)}}$.

Remark 5.4 Note that it should hold that $h(\delta) < \omega_1 + 3\tau < \rho < \alpha - \gamma$, so the protocol is only possible if $h(\delta) < \alpha - \gamma$.

Theorem 5.5 ([DLN15]) *The protocol π_{ComBSME} is $(\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}, \lambda_{\mathsf{B}})$-secure for $\lambda_{\mathsf{C}}, \lambda_{\mathsf{H}}$ and λ_{B} negligible in ℓ.*

Proof: Correctness: Same as in Theorem 5.2.

Hiding: Follows the same lines as in Theorem 5.2. The difference is that here $\kappa = (\rho - 3\tau - \omega_1 - \omega_2)k$ in order to account for the entropy loss due to the output of both hash functions g_1 and g_2 (instead of $\kappa = (\rho - 3\tau - \omega)$ in Theorem 5.2 that accounts for the output of a single hash function g).

Binding: The protocol is binding if, after the commit phase, Alice cannot choose between two different values to successfully open. Let $\sigma = \delta + \zeta$. The only way Alice can cheat is if she can come up with two different strings w, w' that pass all tests performed by Bob during the opening phase. Either $\mathsf{HD}(w, \widetilde{x}^{\mathcal{A}}) \leq \sigma k$ and $\mathsf{HD}(w', \widetilde{x}^{\mathcal{A}}) \leq \sigma k$; or Alice can compute w (without knowing the set $\mathcal{B}$ that together with $\mathcal{A}$ determines $\mathcal{C}$) such that $\mathsf{HD}(w, \widetilde{x}^{\mathcal{A}}) > \sigma k$ and $\mathsf{HD}(w^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) \leq \sigma c$. The probability that Alice succeeds in cheating in the latter case can be upper bounded as in Theorem 5.2. Below we upper bound her cheating success probability in the former case and prove that it decreases exponentially with the security parameter n (or, equivalently in k).

Let the viable set dynamically denote the strings that Alice can possibly send to Bob with non-negligible probability of successful opening. Before the first round of hash challenge-response, the viable set consists of all w such that $\mathsf{HD}(w, \widetilde{x}^{\mathcal{A}}) \leq \sigma k$. Now lets consider an arbitrary fixed value p_1 for the output of the first hash. Considering the j-th viable string before the first hash challenge-response round, define I_j as 1 if the j-th viable string is mapped by g_1 to p_1; otherwise $I_j = 0$. And define $I = \sum_j I_j$. Clearly $\mu = E[I] < 1$, as g_1 is chosen from a $4k$-universal family of hash functions with range of size $\{0,1\}^{\omega_1 k}$ for $\omega_1 > h(\delta + \zeta)$. Let p_1 be called *bad* if I is bigger than $8k+1$. Using the fact that g_1 is $4k$-wise independent and applying Lemma 2.33 with $t = 4k$ and $A = 2t = 8k$, we get

$$\Pr\left[\, I > 8k+1 \,\right] < O\left(\left(\frac{t\mu + t^2}{(2t)^2}\right)^{t/2}\right) < O\left(\left(\frac{1+t}{4t}\right)^{t/2}\right) < O\left(2^{-t/2}\right).$$

Then the probability that any p_1 is bad is upper bounded by

$$O\left(2^{\omega_1 k} 2^{-t/2}\right) < O\left(2^{-k}\right).$$

If the viable set is reduced to at most $8k+1$ elements after the first hash challenge-response round, then the probability that some of those collide in the second hash challenge-response round is upper bounded by

$$(8k+1)^2 \, 2^{-\omega_2 k},$$

which is negligible in k. ▮

Secure Bit Commitment Protocol π'_{ComBSM}

Transmission phase:

1. Alice chooses uniformly randomly k positions from x. Similarly, Bob samples k positions from $\widetilde{x}$. We call their sets of positions $\mathcal{A}$ and $\mathcal{B}$, respectively.

Commit phase:

1. Bob announces $\mathcal{B}$ to Alice. Alice computes $\mathcal{D} = \mathcal{A} \cap \mathcal{B}$. If $|\mathcal{D}| < n$, Alice aborts. Otherwise, Alice picks a random subset $\mathcal{C}$ of $\mathcal{D}$ of size n.
2. Alice computes the dense encoding v of $\mathcal{C}$ (as a subset of $\mathcal{B}$). Alice and Bob interactively hash v, producing two strings v_0, v_1. They compute the subsets $\mathcal{C}_0, \mathcal{C}_1 \subset \mathcal{B}$ that are respectively encoded in v_0, v_1. If either encoding is invalid, they abort.
3. Alice sends $p = m \oplus d$ to Bob, where d is such that $v_d = v$.

Open phase:

1. Alice sends m' and $x'^{\mathcal{C}'}$ to Bob, which are defined as $m' = m$ and $x'^{\mathcal{C}'} = x^{\mathcal{C}}$ in the case that she is honest.
2. Bob computes $d' = p \oplus m'$ and checks whether $\mathsf{HD}(x'^{\mathcal{C}'}, \widetilde{x}^{\mathcal{C}_{d'}}) \leq (\delta + \xi)n$. If the verification fails Bob outputs 0, otherwise he outputs 1.

Figure 5.3: The alternative bit commitment protocol π'_{ComBSM}.

5.4 Alternative Bit Commitment Protocol

In this section we impose a memory bound on Alice instead of Bob and design a *bit* commitment protocol which works for $h(\delta) < \alpha - \gamma$ even against adversaries that misbehave in the commitment phase. The central idea is to use interactive hashing to perform the bit commitment in a similar way to what was done by Shikata and Yamanaka [SY11] in the case of the Bounded Storage Model without errors.

The security parameter is n and k is set as $k = 2\sqrt{\ell n}$. The commitment message space is $\mathcal{M} = \{0, 1\}$. Fix $\varepsilon' > 0$ and $\xi > 0$ such that $\delta + \xi < 1/2$, and let $\rho = \alpha - \gamma - \frac{1+\log(1/\varepsilon')}{\ell}$. Fix $0 < \zeta < 1$ and τ such that $\frac{\rho}{3} \geq \tau > 0$. Let $\mu = \frac{\rho - 2\tau}{\log(1/\tau)}$, $\nu = \frac{\tau}{\log(1/\tau)}$ and $\varepsilon'' = e^{-n\nu^2/2} - 2^{-\Omega(\tau\ell)}$, where the last term comes from Lemma 2.9. Fix $j \geq n\,(\log k + 1)$ and $j - O(n) \geq t \geq j - \zeta \log(1/(\varepsilon' + \varepsilon''))$. The protocol π'_{ComBSM}, which is described in Figure 5.3, works if $h(\delta + \xi) < \rho - 3\tau$ and uses the following functionality, which is possible due to the lemmas in Section 2.7:

- An 2^{-j}-uniform $(t, 2^{-(j-t)+O(\log j)})$-secure interactive hashing protocol with input domain $\mathcal{V} = \{0, 1\}^j$ and an associated dense encoding of subsets F for tuples of size k and subsets of size n.

Theorem 5.6 ([DLN15]) *The protocol π'_{ComBSM} is $(\lambda_{\mathsf{C}}, 0, \lambda_{\mathsf{B}})$-secure for λ_{C} and λ_{B} negligible in n.*

Proof: Correctness: If both participants are honest, the protocol fails only in the following cases: (1) $|\mathcal{D}| < n$; (2) $\mathsf{HD}(x^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) > (\delta + \xi)n$ or (3) either v_0 or v_1 is an invalid encoding of a subset. By Lemma 2.31, $|\mathcal{D}| \geq n$ except with probability at most $e^{-n/4}$. By Lemma 2.30, $\mathsf{HD}(x^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) \leq (\delta + \xi)n$ except with probability at most $e^{-n\xi^2/2}$. Finally, since $v_d = v$ is the encoding of $\mathcal{C}$, one of the two outputs of the interactive hashing protocol is always a valid encoding. The other output v_{1-d} is 2^{-j}-close to distributed uniformly over the $2^{-j} - 1$ strings different from v_d. Since it is a dense encoding, Lemma 2.26 implies that the probability that it is not a valid encoding is thus less than or equal to

$$2^{-j} + \frac{\binom{k}{n}}{2^j - 1} \leq 2^{-j} + 2^{n \log k - j + 1} \leq 2^{-n \log k - n} + 2^{-n+1} \leq 2^{-n+2}$$

for $j \geq n\,(\log k + 1)$. Putting everything together this proves the correctness.

Hiding: There are two possibilities: either the protocol does not abort; or it aborts due to $|\mathcal{D}| < n$ or an invalid encoding. If the protocol aborts, Alice still has not sent $p = m \oplus d$, so Bob's view is independent from M. On the other hand, if the protocol does not abort, then v_{1-d} is a valid encoding of some set $\mathcal{C}'$. Due to the security properties of the interactive hashing protocol and the dense encoding of subsets, Bob's view is then consistent with both

1. Alice committing to m and $\mathcal{C}$ being the subset for which she knows the positions of x, and
2. Alice committing to $1 - m$ and $\mathcal{C}'$ being the subset for which she knows the positions of x.

Hence Bob's view is independent of M and the protocol is 0-hiding.

Binding: The strategy of the proof is to show that there exists $i \in \{0, 1\}$ such that $X^{\mathcal{C}_i}$ has high enough min-entropy from Alice's point of view so that she cannot guess a string $\widehat{x}^{\mathcal{C}_i}$ that is close enough to $\widetilde{x}^{\mathcal{C}_i}$ with non-negligible probability. In that case she will not be able to successfully use this output of the interactive hashing during the opening phase and will be forced to use the other output. By the bounded storage assumption, the bounded information $f(X)$ stored by Alice is such that $|f(X)| \leq \gamma\ell$ with $\gamma < \alpha$. Then, by Lemma 2.6,

$$R_{\infty}^{\varepsilon'}(X|f(X)) \geq \alpha - \gamma - \frac{1 + \log(1/\varepsilon')}{\ell} = \rho.$$

Since Bob is honest, $\mathcal{B}$ is randomly chosen. Lets consider a random subset $\widetilde{\mathcal{C}}$ of $\mathcal{B}$ such that $|\widetilde{\mathcal{C}}| = n$. This is an $(\mu, \nu, e^{-n\nu^2/2})$-averaging sampler for any $\mu, \nu > 0$ according to Lemma 2.10. By setting $\mu = \frac{\rho - 2\tau}{\log(1/\tau)}$, $\nu = \frac{\tau}{\log(1/\tau)}$, we have by Lemma 2.9 that

$$R_{\infty}^{\varepsilon' + \varepsilon''}(X^{\widetilde{\mathcal{C}}}|\mathcal{B}, \widetilde{\mathcal{C}}, f(X)) \geq \rho - 3\tau,$$

for $\varepsilon'' = e^{-n\nu^2/2} - 2^{-\Omega(\tau\ell)}$. For $\widetilde{\varepsilon} = (\varepsilon' + \varepsilon'')^{1-\zeta}$, let $\mathcal{BAD}$ be the set of $\widetilde{\mathcal{C}}$'s such that $R_{\infty}(X^{\widetilde{\mathcal{C}}}|\mathcal{B}, \widetilde{\mathcal{C}}, f(X))$ is not $\widetilde{\varepsilon}$-close to $(\rho - 3\tau)$-min entropy rate. Due to the above equation the density of $\mathcal{BAD}$ is at most $(\varepsilon' + \varepsilon'')^\zeta$. Then the size of the set $\mathcal{T} \subset \{0,1\}^j$ of strings that maps (using the encoding scheme) to subsets in $\mathcal{BAD}$ is at most $(\varepsilon' + \varepsilon'')^\zeta 2^j \leq 2^t$. Hence the properties of the interactive hashing protocol guarantee that with overwhelming probability there will be an i such that

$$R_\infty^{\widetilde{\varepsilon}}(X^{\mathcal{C}_i} | \mathcal{B}, \mathcal{C}_i, f(X), T_{IH}) \geq \rho - 3\tau,$$

where T_{IH} are the messages exchanged during the interactive hashing protocol.

However, if $h(\delta + \xi) < \rho - 3\tau$ and the min-entropy rate is at least $\rho - 3\tau$, then fixing $0 < \widehat{\varepsilon} < \rho - 3\tau - h(\delta + \xi)$, for large enough n, the probability that Alice guesses one of the strings $\widehat{x}^{\mathcal{C}_i}$ that would be accepted by Bob as being close enough to $\widetilde{x}^{\mathcal{C}_i}$ is upper bounded by

$$2^{(h(\delta+\xi)-\rho+3\tau-\widehat{\varepsilon})n}$$

which is a negligible function of n. ▮

By fixing the parameters as small as possible, in the limit for large enough n the protocol works for values α, γ, δ which satisfy $h(\delta) < \alpha - \gamma$.

5.5 Oblivious Transfer Protocol

For our OT protocol the memory bound is on Bob. The idea of the protocol is that initially both parties samples some positions from the public random source. Then an interactive hashing protocol (with an associated dense encoding) is used to select two subsets of the positions sampled by Alice. Bob's input to the interactive hashing is a subset of positions for which he has also sampled the public random source. The other subset is out of Bob's control due to the security properties of the interactive hashing protocol. Finally the positions specified by the two subsets are used as input to a fuzzy extractor in order to obtain one-time pads. Bob sends one bit indicating which input string should be xored with which one-time pad. Intuitively, the security for Alice is guaranteed by the fact that one of the subsets is out of Bob's control and will have high min-entropy given his view, thus resulting in a good one-time pad; the security for Bob follows from the security of the interactive hashing.

The OT protocol is defined in Figure 5.4. The security parameter is n and k is set as $k = 2\sqrt{\ell n}$. Fix $\varepsilon', \widehat{\varepsilon} > 0$ and $\xi > 0$ such that $1/4 > \delta + \xi > 0$ and let $\rho = \alpha - \gamma - \frac{1+\log(1/\varepsilon')}{\ell}$. Fix $0 < \zeta < 1$ and τ such that $\frac{\rho}{3} \geq \tau > 0$. Let $\mu = \frac{\rho - 2\tau}{\log(1/\tau)}$, $\nu = \frac{\tau}{\log(1/\tau)}$ and $\varepsilon'' = e^{-n\nu^2/2} - 2^{-\Omega(\tau\ell)}$, where the last term comes from Lemma 2.9. Fix $j \geq n\,(\log k + 1)$ and $j - O(n) \geq t \geq j - \zeta \log(1/(\varepsilon' + \varepsilon''))$. For σ depending on $\delta + \xi$ (see code rate comments below), fix κ and m_F such that $\kappa = \rho + \sigma - 3\tau - 2m_F - 1 - \frac{1+\log(1/\widehat{\varepsilon})}{n}$ and $0 < m_F < \kappa$. The OT message space is $\mathcal{M} = \{0,1\}^{m_F n}$. It is assumed that the following functionalities, which are possible due to the lemmas in Sections 2.3 and 2.7, are available to the parties:

- A pair of functions $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^{m_F n} \times \{0,1\}^q$ and $\mathsf{Rec}\colon \{0,1\}^n \times \{0,1\}^r \times \{0,1\}^q \to \{0,1\}^{m_F n}$ that constitutes an $(\kappa n, \varepsilon_{\mathsf{Ext}}, \delta + \xi, 0)$-fuzzy extractor where $q = (1-\sigma)n$, $\varepsilon_{\mathsf{Ext}}$ is an arbitrary number with $\varepsilon_{\mathsf{Ext}} > e^{-n/2^{O(\log^* n)}}$.
- An 2^{-j}-uniform $(t, 2^{-(j-t)+O(\log j)})$-secure interactive hashing protocol with input domain $\mathcal{V} = \{0,1\}^j$ and an associated dense encoding of subsets F for tuples of size k and subsets of size n.

Secure OT Protocol π_{OTBSM}

Transmission phase:

1. Alice picks uniformly randomly k positions from x. Similarly, Bob samples k positions from $\widetilde{x}$. We call their sets of positions $\mathcal{A}$ and $\mathcal{B}$, respectively.

Setup phase:

1. Alice sends $\mathcal{A}$ to Bob. Bob computes $\mathcal{D} = \mathcal{A} \cap \mathcal{B}$. If $|\mathcal{D}| < n$, Bob aborts. Otherwise, Bob picks a random subset $\mathcal{C}$ of $\mathcal{D}$ of size n.

2. Bob computes the encoding v of $\mathcal{C}$ as a subset of $\mathcal{A}$. Alice and Bob interactively hash v, producing two strings v_0, v_1. They compute the subsets $\mathcal{C}_0, \mathcal{C}_1 \subset \mathcal{A}$ that are respectively encoded in v_0, v_1. If either encoding is invalid, they abort.

Transfer phase:

1. Bob sends $p = c \oplus d$, where d is such that $v_d = v$.

2. For $i \in \{0,1\}$, Alice picks $r_i \xleftarrow{\$} \{0,1\}^r$, computes $(y_i, q_i) \leftarrow \mathsf{Ext}(x^{\mathcal{C}_i}, r_i)$ and $z_i = s_{i \oplus p} \oplus y_i$, and sends (z_i, r_i, q_i) to Bob.

3. Bob computes $y' \leftarrow \mathsf{Rec}(\widetilde{x}^{\mathcal{C}}, r_d, q_d)$ and outputs $s = y' \oplus z_d$.

Figure 5.4: The OT protocol π_{OTBSM}.

Recall (Remark 2.17) that there is a tradeoff between the fraction of errors $\delta + \xi$ that the fuzzy extractor can tolerate and the rate σ of the code used in the construction. The construction given in Theorem 4 of [GI02] has linear-time encoding and decoding and achieves the Zyablov bound: for given $1 > \sigma > 0$ and $\mu > 0$, the code has rate σ and

$$\delta + \xi \geq \max_{\sigma < \widetilde{\sigma} < 1} \frac{(1 - \widetilde{\sigma} - \mu)y}{2} \tag{5.1}$$

where y is the unique number in $[0, 1/2]$ with $h(y) = 1 - \sigma/\widetilde{\sigma}$ and $\delta + \xi$ the amount of errors that can be corrected by the code. In order for κ to be positive, we need to have $\rho + \sigma > 1$. Since ρ approaches $\alpha - \gamma$ from below in the asymptotic limit, an upper bound for δ is obtained by setting $\sigma > 1 - \alpha + \gamma$ and $\mu = 0$ in Equation (5.1).

Alternatively, there is a construction based on random linear codes which achieves a better bound, namely, the Gilbert-Varshamov bound: for a given relative distance v and $\mu > 0$, the code has rate $\sigma \geq 1 - h(v) - \mu$. Applying again the constraint that $\rho + \sigma > 1$ and that $\rho \to \alpha - \gamma$ in the asymptotic limit, and using the fact that a code that can correct $\delta\ell$ errors has relative distance $v = 2\delta + 1/\ell \to 2\delta$, this gives an upper bound for δ: we must have $h(2\delta) < \alpha - \gamma$. However, as noted in Remark 2.17, the random linear code construction does not have efficient decoding. It is an open question whether an efficient construction can achieve better parameters than the one from [GI02].

Theorem 5.7 ([DLN15]) *The protocol π_{OTBSM} is $(\lambda_{\mathsf{C}}, 0, \lambda_{\mathsf{A}})$-secure for λ_{C} and λ_{A} negligible in n.*

Proof: Correctness: The probability of an abort is analyzed first. The protocol will abort if either $|\mathcal{D}| < n$, or if one string obtained in the interactive hashing protocol is an invalid encoding of subsets of $\mathcal{A}$. By Lemma 2.31, $\Pr[\,|\mathcal{D}| < n\,] < e^{-n/4}$. Since $v_d = v$, which is the encoding of $\mathcal{C}$, one of the two outputs of the interactive hashing protocol is always a valid encoding. The other output v_{1-d} is 2^{-j}-close to distributed uniformly over the $2^{-j} - 1$ strings different from v_d. Since it is a dense encoding, Lemma 2.26 implies that the probability that it is not a valid encoding is thus less than or equal to

$$2^{-j} + \frac{\binom{k}{n}}{2^j - 1} \leq 2^{-j} + 2^{n \log k - j + 1} \leq 2^{-n \log k - n} + 2^{-n+1} \leq 2^{-n+2}$$

for $j \geq n\,(\log k + 1)$. If no aborts happens and both parties are honest, then $s = s_c$ if and only if $\mathsf{Rec}(\widetilde{x}^{\mathcal{C}}, r_d, q_d) = y_d$. By the properties of the employed fuzzy extractor, this last event happens only if $\mathsf{HD}(x^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) \leq (\delta + \xi)n$. By Lemma 2.30, $\mathsf{HD}(x^{\mathcal{C}}, \widetilde{x}^{\mathcal{C}}) > (\delta + \xi)n$ with probability at most $e^{-\xi^2 n/2}$. Putting everything together this proves the correctness.

Security for Bob: There are two possibilities: either the protocol aborts or not. If the protocol aborts in the setup phase, Bob still has not sent $p = c \oplus d$, so Alice's view is independent from C. On the other hand, if the protocol does not abort, then v_{1-d} is a valid encoding of some set $\mathcal{C}'$. Due to the properties of the interactive hashing protocol, Alice's view is then consistent with both

1. Bob choosing c and $\mathcal{C}$, and
2. Bob choosing $1 - c$ and $\mathcal{C}'$.

Hence Alice's view is independent of C and the protocol is perfectly secure for Bob.

Security for Alice: There should be an index i (determined at the setup stage) such that for any two pairs $(s_0, s_1), (s_0', s_1')$ with $s_i = s_i'$, Bob's view of the protocol executed with (s_0, s_1) is close to his view of the protocol executed with (s_0', s_1'). The view of Bob is given by the function computed from the public random source $\widetilde{f}(\widetilde{x})$ along with all the messages exchanged and his local randomness.

The proof's strategy is to show that for i, $X^{\mathcal{C}_{1-i}}$ has high enough min-entropy, given Bob's view of the protocol, in such a way that Y_{1-i} is indistinguishable from a uniform distribution. Indistinguishability of Bob's views will then follow.

By the bounded storage assumption, $|\widetilde{f}(\widetilde{x})| \leq \gamma\ell$ with $\gamma < \alpha$. Then, by Lemma 2.6,

$$R_{\infty}^{\varepsilon'}(X|\widetilde{f}(\widetilde{X})) \geq \alpha - \gamma - \frac{1 + \log(1/\varepsilon')}{\ell} = \rho.$$

Since Alice is honest, $\mathcal{A}$ is randomly chosen. Lets consider a random subset $\widetilde{\mathcal{C}}$ of $\mathcal{A}$ such that $|\widetilde{\mathcal{C}}| = n$. This is an $(\mu, \nu, e^{-n\nu^2/2})$-averaging sampler for any $\mu, \nu > 0$

according to Lemma 2.10. By setting $\mu = \frac{\rho - 2\tau}{\log(1/\tau)}$, $\nu = \frac{\tau}{\log(1/\tau)}$, we have by Lemma 2.9 that

$$R_{\infty}^{\varepsilon' + \varepsilon''}(X^{\widetilde{\mathcal{C}}} | \mathcal{A}, \widetilde{\mathcal{C}}, \widetilde{f}(\widetilde{X})) \geq \rho - 3\tau,$$

for $\varepsilon'' = e^{-n\nu^2/2} - 2^{-\Omega(\tau\ell)}$. For $\widetilde{\varepsilon} = (\varepsilon' + \varepsilon'')^{1-\zeta}$, let $\mathcal{BAD}$ be the set of $\widetilde{\mathcal{C}}$'s such that $R_{\infty}(X^{\widetilde{\mathcal{C}}} | \mathcal{A}, \widetilde{\mathcal{C}}, \widetilde{f}(\widetilde{X}))$ is not $\widetilde{\varepsilon}$-close to $(\rho - 3\tau)$-min entropy rate. Due to the above equation the density of $\mathcal{BAD}$ is at most $(\varepsilon' + \varepsilon'')^{\zeta}$. Then the size of the set $\mathcal{T} \subset \{0,1\}^j$ of strings that maps (using the encoding scheme) to subsets in $\mathcal{BAD}$ is at most $(\varepsilon' + \varepsilon'')^{\zeta} 2^j \leq 2^t$. Hence the properties of the interactive hashing protocol guarantee that with overwhelming probability there will be an i such that

$$R_{\infty}^{\widetilde{\varepsilon}}(X^{\mathcal{C}_{1-i}} | \mathcal{A}, \mathcal{C}_{1-i}, \widetilde{f}(\widetilde{X}), T_{IH}) \geq \rho - 3\tau,$$

where T_{IH} are the messages exchanged during the interactive hashing protocol. We now show that $X^{\mathcal{C}_{1-i}}$ has high min-entropy even when given Z_i, Y_i, Q_i. We can see (Z_i, Y_i, Q_i) as a random variable over $\{0,1\}^{(2m_F + 1 - \sigma)n}$. Then, by Lemma 2.6,

$$R_{\infty}^{\widehat{\varepsilon} + \sqrt{8\widetilde{\varepsilon}}}(X^{\mathcal{C}_{1-i}} | \mathcal{A}, \mathcal{C}_{1-i}, \widetilde{f}(\widetilde{X}), T_{IH}, Z_i, Y_i, Q_i) \geq \rho + \sigma - 3\tau - 2m_F - 1 - \frac{1 + \log(1/\widehat{\varepsilon})}{n} = \kappa.$$

Thus setting ε' and $\widehat{\varepsilon}$ to be negligible in n, the use of the $(\kappa\ell, \varepsilon_{\mathsf{Ext}}, \delta + \xi, 0)$-fuzzy extractor to obtain Y_i that is used as an one-time pad guarantees that only negligible information about $s_{i \oplus p}$ can be leaked. Thus the protocol is λ_{A}-secure for Alice, for λ_{A} negligible in n. ▌

5.6 Discussion

This chapter presented the first protocols for commitment and oblivious transfer in the Bounded Storage Model with Errors, thus extending the previous results existing in the literature for key agreement [Din05]. As expected, our protocols work for a limited range of values of the noise parameter δ. The allowed range for our commitment schemes is different than the one for the OT protocol. For the case of commitment schemes, the range of noise that could be tolerated depended on the round complexity of the proposed protocols: extra rounds helped tolerating a more severe noise.

There are many open questions that follow our results here:

- To prove the impossibility of commitment protocols when $h(\delta) \geq \alpha - \gamma$.
- To obtain efficient OT protocols that work for the range of noise achieved by our protocols based on random linear codes.
- What is the best range of noise that can be achieved by non-interactive commitment protocols?
- Is there an intrinsic difference in the level of noise tolerated by bit commitment and OT protocols?

6. Privacy-Preserving Learning

The contents of this chapter are based on [CDNN15] and deal with the problem of obtaining a privacy-preserving protocol for computing a linear regression model from a training dataset that is distributed among many parties. Our results are information-theoretically secure and work in the commodity-based model. The online phase of our protocol is extremely efficient, having solely modular additions and multiplications. It improves the execution time from days [HFN11] to seconds. If a trusted initializer is not available or desirable, we present a solution for substituting the trusted initializer by a secure multi-party computation protocol executed among the parties during an offline phase. Note that this offline phase (and so the combined protocol in this case) is only computationally secure. Despite involving more computationally intensive operations, the offline phase consists essentially of independent computations over random data. Therefore it is embarrassingly parallelizable and gains proportional to the number of available cores can be obtained, making even the offline phase practical.

Linear regression models the relationship between some input variables and a real valued outcome. It is a quite popular technique in statistical analysis and machine learning [SSBD14] due to some appealing characteristic that it exhibits such as: intuitiveness, efficiency of the training phase, ability to fit the data reasonably well for many problems, and the simplicity of the model that helps prevent it from overfitting to a specific set of training examples. Similarly to other machine linear models, the standard algorithm for obtaining a linear regression model assumes that all the training data is directly available to the party computing it. However, in many scenarios the training data is distributed among many parties that cannot or will not share their dataset due to many economical reasons or privacy legislation. One prominent example is the healthcare ecosystem. It is widely acknowledged that big data analytics can revolutionize the healthcare industry, among other things, by optimizing healthcare spending at all levels from patients to hospitals to governments while improving overall population health. In practice, however, a major obstacle for applying machine learning techniques in such scenario is the need of data that is split over many different owners – healthcare providers, hospitals, and medical insurance companies – who do not want to or legally cannot share their data with outside entities. Consequently, it is important to develop privacy-preserving machine learning protocols. We deal particularly with privacy-preserving protocols for linear regression.

As already mentioned in the Introduction, many works deal with the topic of secure linear regression, but most of them do not even try to achieve the level of security that is normally considered in modern cryptography. The pioneering work of Hall et al. [HFN11] actually aim at obtaining a high level of security. It considers the framework of secure two-party protocols with simulation-based definitions of security, as in [Gol04]. We should however remark that as some of the building blocks do not realize the exact functionalities, but rather approximations, they should have considered the framework of Feigenbaum et al. [FIM$^+$01, FIM$^+$06] for dealing with secure approximations. We should additionally point out that their truncation protocol has a small (correctable) problem (see [CDNN15] for details). The performance of our solutions compares favorably with theirs. Their computing time for solving the linear regression problem for 51K input vectors, each with 22 features, is two days [HFN11]. The online phase of our protocol solves this problem in a few seconds. Even when considering the running time of the offline phase of our computationally secure protocol, by exploiting its embarrassingly parallelization property, the overall running time is still in the order of minutes for such a number of features and vectors.

Nikolaenko et al. [NWI$^+$13] considered a different scenario in which the parties encrypt the training data and upload the ciphertexts to a third party. This third party, with the help from a semi-honest Crypto Service Provider that performs the heavy cryptographic operations, then computes the regression model. Their solution is based on homomorphic encryption and garbled circuits and assumes that the Crypto Service Provider do not collude with other parties. Note that the Crypto Service Provider actively engages in the protocol execution, in strong contrast with a trusted initializer, which does not engage in the protocol execution after the setup phase. Our online phase is still much faster than the protocol presented by Nikolaenko et al. [NWI$^+$13]. Even when we add up the offline phase and the online phase running times, in the case of our computationally secure protocol, when multiple cores are available for the offline phase computations, the overall running time is less for our protocol.

We assessed our secure linear regression by implementing and analyzing the results using ten real datasets. We chose a variety of different datasets based on their number of features and instances. Some of our datasets have millions of vectors. We are unaware of any other work on secure linear regression where real datasets of this size have been analyzed before. For example, in [HFN11] and in [NWI$^+$13], the real datasets used had thousands of vectors.

Outline

This chapter is structured as follows: after explaining our model in Section 6.1, we present a high level overview of our protocol for secure linear regression in Section 6.2. Next, we provide details on how we deal with real numbers in Section 6.3 and on our secure computation of the inverse of matrices in Section 6.4. In Section 6.5, we summarize how these building blocks fit together in our information-theoretically secure protocol, while in Section 6.6 we explain how to substitute the trusted initializer and obtain a computationally secure protocol. In Section 6.7 we present runtime results of both protocols on ten different datasets, with a varying number of instances and features, showing a substantial speed-up compared to existing work. We conclude with some final remarks in Section 6.8.

6.1 Model

Adversarial Model

In this work we consider the problem in which a set of parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$, given their inputs, want to securely compute a function without leaking any information other than the result. The security is defined by comparing a real world and an ideal world. In the real world the parties executes a protocol π and an adversary $\mathcal{A}$ controls the corrupted parties in the protocol execution. The output of the real world is formed by joining the output of the uncorrupted parties with the protocol view of $\mathcal{A}$. In the ideal world there is an ideal functionality $\mathcal{F}$ which takes the inputs from the parties and gives the outputs to them. And there is a also a simulator $\mathcal{S}$ that by only learning the inputs and outputs of the corrupted parties, generates a simulated view of the protocol execution. The output of the ideal world is formed by joining the output of the uncorrupted parties with the simulated view of $\mathcal{S}$. A protocol π securely computes the functionality $\mathcal{F}$ if for every adversary $\mathcal{A}$, there exists a simulator $\mathcal{S}$, such that the joint outputs of the real and ideal worlds are indistinguishable. The considered indistinguishability can be either statistical or computational; resulting respectively in statistical or computational security. The advantage of defining the security using the simulation paradigm is that it allows for the sequential composition of protocols. We deal with honest-but-curious adversaries, which were the ones considered in other privacy-preserving classification protocols so far.

Secure Approximations

Note that the secure computation of an approximation $\overline{f}$ of a target function f can reveal more information than the target function itself. Imagine for instance the case where the output of $\overline{f}$ is equal to the output of f in all bits except one, in which $\overline{f}$ encodes one bit of the input of one party. To ensure that the approximation $\overline{f}$ does not leak additional information we use the framework of Feigenbaum et al. [FIM$^+$01, FIM$^+$06] for private approximations. Only deterministic target functions f are considered, but the approximations $\overline{f}$ can be randomized.

Definition 6.1 (ε-approximation) *The functionality $\overline{f}$ is an ε-approximation of f if for all possible inputs $\mathbf{x}$, $|f(\mathbf{x}) - \overline{f}(\mathbf{x})| < \varepsilon$.*

Definition 6.2 (Privacy with respect to f) *The functionality $\overline{f}$ is functionally private with respect to f if there is a simulator $\mathcal{S}$ such that for all possible inputs $\mathbf{x}$, $\{\mathcal{S}(f(\mathbf{x}))\} \overset{s}{\approx} \{\overline{f}(\mathbf{x})\}$.*

Note that functional privacy is a property from the functionality $\overline{f}$ itself, and not from any protocol implementing it. It captures the fact that the approximation error is independent from the inputs when conditioned on the output of the exact functionality.

6.2 Overview

Assume that we have a set of training examples (real vectors)

$$(a_1(x_i), a_2(x_i), \ldots, a_m(x_i), y_i),$$

where $a_j(x_i)$ is the value of the input attribute a_j for the training example x_i ($i = 1, \ldots, t$) and y_i is the associated output. The goal is to leverage these training examples to predict the unknown outcome for a previously unseen input as accurately as possible. To this end, we want to learn a linear function

$$y = \beta_1 a_1(x) + \beta_2 a_2(x) + \ldots + \beta_m a_m(x) + b$$

that best approximates the relation between the input variables $a_1(x), a_2(x), \ldots, a_m(x)$ and the response variable y. Throughout this paper we assume that all variables are real numbers and that we aim to find real values for the parameters $\beta_1, \beta_2, \ldots, \beta_m$ and b that minimize the empirical risk function

$$\frac{1}{t}\sum_{i=1}^{t}((\beta_1 a_1(x_i) + \beta_2 a_2(x_i) + \ldots + \beta_m a_m(x_i) + b) - y_i)^2, \tag{6.1}$$

which is the mean squared error over the training instances. For notational convenience, we switch to the homogenous version of the linear function and we use vector notation, i.e. let

- $\mathbf{x_i} = (a_0(x_i), a_1(x_i), a_2(x_i), \ldots, a_m(x_i))$, with $a_0(x_i) = 1$ for all $i \in \{1, \ldots, t\}$ and
- $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_m)$, with $\beta_0 = b$.

Using $\langle \boldsymbol{\beta}, \mathbf{x_i} \rangle$ to denote the dot product of $\boldsymbol{\beta}$ and $\mathbf{x_i}$, minimizing (6.1) amounts to calculating the gradient and comparing it to zero, i.e. solving

$$\frac{2}{t}\sum_{i=1}^{t}(\langle \boldsymbol{\beta}, \mathbf{x_i} \rangle - y_i)\mathbf{x_i} = 0. \tag{6.2}$$

The solution to (6.2) is[1]

$$\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{6.3}$$

with

$$\mathbf{X} = \begin{pmatrix} \mathbf{x_1} \\ \mathbf{x_2} \\ \ldots \\ \mathbf{x_t} \end{pmatrix} \text{ and } \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \ldots \\ y_t \end{pmatrix}.$$

The scenarios that we are interested in are those in which the training data is not owned by a single party but is instead distributed across multiple parties who are not willing to disclose it. Our experiments in Section 6.7 correspond to scenarios in which $\mathbf{X}$ is partitioned column-wise across two honest-but-curious parties, i.e. Alice and Bob have information about different features of the same instances, and Bob has the vector $\mathbf{y}$. However, as will become clear below, our protocols work in all

[1] Assuming that $\mathbf{X}^T\mathbf{X}$ is invertible

scenarios in which the parties hold secret sharings $[\![\mathbf{X}]\!]_q$ and $[\![\mathbf{y}]\!]_q$ as input, regardless of whether they are sliced column-wise, row-wise, or distributed in any other way.

Here we give an overview of our solution. The basic idea is to reduce the problem of securely computing linear regression to the problem of securely computing products of matrices. The protocol for computing products of matrices works only for elements of the matrices belonging to a finite field. Thus, the parties should be able to map their real-valued fixed precision inputs to elements of a finite field (as described in Section 6.3).

1. Offline phase: in the information-theoretically secure protocol, the players receive correlated data from the trusted initializer. In the case of the computationally secure protocol, they run the protocol described in Section 6.6.
2. Online Phase:
   a) The players map their fixed precision real valued inputs to elements of a finite field as described in Section 6.3.
   b) The players compute over their shares using the protocols for matrix multiplication (described in Section 2.10) and for computing the inverse of a Covariance Matrix (described in Section 6.4) in order to obtain shares of the estimated regression coefficient vector.
   c) The players exchange their shares of the estimated regression coefficient vector and reconstruct it.

After presenting the building blocks in Sections 6.3 and 6.4, we reiterate the information-theoretically secure and the computationally secure protocols for linear regression against honest-but-curious adversaries at a more concrete level of detail in Sections 6.5 and 6.6 respectively.

6.3 Dealing with Real Numbers

The security proof of the (matrix) multiplication protocol presented in Section 2.10 essentially relies on the fact that the blinding factors are uniformly random in $\mathbb{Z}_q$. If one tries to design similar protocols working directly with integers or real numbers, there would be a problem, since it is not possible to sample uniformly in $\mathbb{Z}$ or $\mathbb{R}$. Similarly, in protocols that use homomorphic encryption as building blocks, the encryption is normally done for messages which are members of a finite group. But in secure protocols for functionalities such as linear regression one needs to deal with inputs which are real numbers. Thus it is necessary to develop a way to approximate the computations on real numbers by using building blocks which work on fields $\mathbb{Z}_q$.

We adapt the method of Catrina and Saxena [CS10] with a fixed-point representation. Let k, e and f be integers such that $k > 0$, $f \geq 0$ and $e = k - f \geq 0$. Let $\mathbb{Z}_{\langle k \rangle}$ denote the set $\{x \in \mathbb{Z} : -2^{k-1} + 1 \leq x \leq 2^{k-1} - 1\}$. The fixed-point data type with k bits, resolution 2^{-f}, and range 2^e is the set $\mathbb{Q}_{\langle k,f \rangle} = \{\tilde{x} \in \mathbb{Q} : \tilde{x} = \hat{x}2^{-f}, \hat{x} \in \mathbb{Z}_{\langle k \rangle}\}$. The signed integers in $\mathbb{Z}_{\langle k \rangle}$ are then encoded in the field $\mathbb{Z}_q$ (with $q > 2^k$) using the function

$$g\colon\ \mathbb{Z}_{\langle k \rangle} \to \mathbb{Z}_q, g(\hat{x}) = \hat{x} \mod q.$$

In secure computation protocols using secret sharing techniques, the values in $\mathbb{Z}_q$ are actually shared among the parties. Using this encoding, we have that $\hat{x} + \hat{y} =$

Functionality $\mathcal{F}_{\mathsf{Trunc}}$

$\mathcal{F}_{\mathsf{Trunc}}$ is parametrized by the size q of the field and the dimensions ℓ_1, ℓ_2 of the input matrix. It reduces the resolution of each element of the matrix by 2^{-f}.

Input: Upon receiving a message from a party with its shares of $[\![\mathbf{W}]\!]_q$ for the matrix $\mathbf{W}$ whose elements should be truncated, record the share, ignore any subsequent message from that party and inform the other parties about the receipt.

Output: Upon receipt of the shares from all parties, recover $\mathbf{W}$ from the shares. For each element w of $\mathbf{W}$, compute $\hat{w} = g^{-1}(w \mod q)$, sample $u \in \{0, 1\}$ such that $\Pr[u = 1] = (\hat{w} \mod 2^f)/2^f$, and then fix in the output matrix $\mathbf{T}$ the element in the same row and column to $t = g\left(\left\lfloor \frac{\hat{w}}{2^f} \right\rfloor + u\right)$. Create a secret sharing of $[\![\mathbf{T}]\!]_q$ to distribute to the parties. Before the output deliver, the corrupt parties fix their shares of the output to any constant values. The shares of the uncorrupted parties are then created by picking uniformly random values subject to the correctness constraints.

Figure 6.1: The distributed approximate truncation functionality.

Truncation Protocol π_{Trunc}

Let n be a statistical security parameter. The protocol is parametrized by the size $q > 2^{k+f+n+1}$ of the field and the dimensions ℓ_1, ℓ_2 of the input matrix. At the setup, the trusted initializer picks a matrix $\mathbf{R}' \in \mathbb{F}_q^{\ell_1 \times \ell_2}$ with elements uniformly random in $\{0, \ldots, 2^f - 1\}$ and a matrix $\mathbf{R}'' \in \mathbb{F}_q^{\ell_1 \times \ell_2}$ with elements uniformly random in $\{0, \ldots, 2^{k+n} - 1\}$. He then computes $\mathbf{R} = \mathbf{R}'' 2^f + \mathbf{R}'$ and creates secret sharings $[\![\mathbf{R}]\!]_q$ and $[\![\mathbf{R}']\!]_q$ to distribute to the parties. The parties input is $[\![\mathbf{W}]\!]_q$ such that for all elements w of $\mathbf{W}$ it holds that $w \in \{0, 1, \ldots, 2^{k+f-1} - 1\} \cup \{q - 2^{k+f-1} + 1, \ldots, q - 1\}$.

1. Locally compute $[\![\mathbf{Z}]\!]_q \leftarrow [\![\mathbf{W}]\!]_q + [\![\mathbf{R}]\!]_q$ and then open $\mathbf{Z}$.
2. Compute $\mathbf{C} = \mathbf{Z} + 2^{k+f-1}$ and $\mathbf{C}' = \mathbf{C} \mod 2^f$ where these scalar operations are performed element-wise. Then compute the secret sharing $[\![\mathbf{S}]\!]_q \leftarrow [\![\mathbf{W}]\!]_q + [\![\mathbf{R}']\!]_q - \mathbf{C}'$.
3. For $i = ((q + 1)/2)^f$, locally compute $[\![\mathbf{T}]\!]_q \leftarrow i[\![\mathbf{S}]\!]_q$ and output the shares of $\mathbf{T}$.

Figure 6.2: The truncation protocol.

$g^{-1}(g(\hat{x}) + g(\hat{y}))$, where the second addition is in $\mathbb{Z}_q$, i.e., we can compute the addition for signed integers in $\mathbb{Z}_{\langle k \rangle}$ by using the arithmetic in $\mathbb{Z}_q$. The same holds for subtraction and multiplication.

For the fixed-point data type we can do additions using the fact that $\tilde{w} = \tilde{x} + \tilde{y} = (\hat{x} + \hat{y})2^{-f}$, so we can trivially obtain the representation of $\tilde{w}$ with resolution 2^{-f} by computing $\hat{w} = \hat{x} + \hat{y}$, i.e., we can do the addition of the fixed-point type by using the addition in $\mathbb{Z}_{\langle k \rangle}$, which itself can be done by performing the addition in $\mathbb{Z}_q$. The same holds for subtraction. But for multiplication we have that $\tilde{w} = \tilde{x}\tilde{y} = \hat{x}\hat{y}2^{-2f}$, and therefore if we perform the multiplication in $\mathbb{Z}_q$, we will obtain (if no overflow occurred) the representation of $\tilde{w}$ with resolution 2^{-2f}. For the signed integers representation to be independent from the amount of multiplication operations performed with the fixed-point data, we need to scale the resolution of $\tilde{w}$ down to 2^{-f}.

We use a slightly modified version of the truncation protocol of Catrina and Saxena [CS10]. The central idea is to reveal the number w to be truncated, but blinded by a factor r which is from a domain exponentially bigger than the domain of the value w and so statistically hides it. The value r is generated in such way that the parties obtain shares of both r itself as well as of r' that represents the f least significant bits of r. The parties can then reveal $w + r$ and compute shares of the truncated value by using local computations. We describe in Figure 6.2 a truncation protocol π_{Trunc} that is a generalization of this idea to truncate all elements of a matrix at once. The functionality $\mathcal{F}_{\mathsf{Trunc}}$ that captures the approximate truncation without leakage is described in Figure 6.1.

Theorem 6.3 ([CDNN15]) *The truncation protocol π_{Trunc} securely computes the approximate truncation functionality $\mathcal{F}_{\mathsf{Trunc}}$ against honest-but-curious adversaries.*

Proof: Correctness: Note that for any element w of $\mathbf{W}$, the operations performed on it to obtain the truncated value only depends on elements of other matrices that are on the same row and column. Therefore we analyze the correctness by considering one element w of $\mathbf{W}$ and the respective elements of the other matrices, denoted r, r', c and c' respectively. Let $\hat{w} = g^{-1}(w \mod q)$. We have that $\hat{w} \in \{-2^{k+f-1} + 1, -2^{k+f-1} + 2, \ldots, 2^{k+f-1} - 1\}$. Let $b = \hat{w} + 2^{k+f-1}$ and let $b' = b \mod 2^f$. We have that $b \in \{1, \ldots, 2^{k+f} - 1\}$ and since $k > 0$ also that

$$b' = b \mod 2^f = \hat{w} + 2^{k+f-1} \mod 2^f = \hat{w} \mod 2^f.$$

Since $r < 2^{k+f+n}$ and $q > 2^{k+f+n+1}$ we have that $c = b + r$ and thus

$$c' = (b' + r') \mod 2^f = b' + r' - u2^f$$

where $u \in \{0, 1\}$ and $\Pr[u = 1] = \Pr\left[r' \geq 2^f - b'\right] = (\hat{w} \mod 2^f)/2^f$ with the probability over the choices of the random r'. Hence

$$c' - r' = g(\hat{w} \mod 2^f - u2^f),$$

$$w + r' - c' = g(\hat{w} - (\hat{w} \mod 2^f) + u2^f) = g\left(\left\lfloor \frac{\hat{w}}{2^f} \right\rfloor 2^f + u2^f\right),$$

$$(w + r' - c')((q+1)/2)^f = g\left(\left\lfloor \frac{\hat{w}}{2^f} \right\rfloor + u\right),$$

since the multiplicative inverse of 2^f in $\mathbb{Z}_q$ is $((q+1)/2)^f$. Therefore the shares output by the parties are correct.

Security: The only messages exchanged are to open $\mathbf{Z} = \mathbf{W} + \mathbf{R}$, but since $\mathbf{R}$ has elements that are uniformly random in $\{0, \ldots, 2^{k+f+n} - 1\}$ and $\mathbf{W}$'s elements are in $\{0, 1, \ldots, 2^{k+f-1} - 1\} \cup \{q - 2^{k+f-1} + 1, \ldots, q - 1\}$, we have that the statistical distance between the probability distributions of the elements of $\mathbf{Z}$ and $\mathbf{R}$ is at most 2^{-n} and the matrices are statistically indistinguishable. The simulation strategy is very simple and consists of opening a secret sharing of a matrix whose elements are uniformly random in $\{0, \ldots, 2^{k+f+n} - 1\}$. ▌

Theorem 6.4 ([CDNN15]) *$\mathcal{F}_{\mathsf{Trunc}}$ is an 1-approximation*[2] *and is functionally private with respect to an exact truncation functionality that computes the truncation using the floor function.*

Proof: The only difference between the two functionalities is that in the approximate truncation an error factor u is present in the shared output of each element. But note that $u \in \{0, 1\}$ and $\Pr[u = 1] = (\hat{w} \mod 2^f)/2^f$, but u is independent from the specific shares used to encode $g(\hat{w})$. Thus the protocol rounds $\hat{w}/2^f$ to the nearest integer with probability $1 - \alpha$, where α is the distance between the real number $\hat{w}/2^f$ and the nearest integer. ▌

6.4 Computing the Inverse of a Covariance Matrix

In order to be able to compute the linear regression from a design matrix and the response vector we need to compute the inverse of the covariance matrix. Let $\mathbf{A}$ be a covariance matrix. In order to compute $\mathbf{A}^{-1}$ we use a generalization for matrices of the Newton-Raphson division method. The algorithms for division of fixed-point numbers are divided in two main classes: digit recurrence (subtractive division) and functional iteration (multiplicative division). The Newton-Raphson division method is from the functional iteration class, which is more amenable to secure implementation and converges faster. Additionally this method is self correcting, i.e., truncations errors in one iteration decrease quadratically in the next iterations. The inverse of a number a is computed by defining the function $h(x) = x^{-1} - a$ and then applying the Newton-Raphson method for finding successively better approximations to the roots of $h(x)$. The iterations follow the form:

$$x_{s+1} = x_s(2 - ax_s).$$

This algorithm can be generalized for computing the inverse of the matrix $\mathbf{A}$. A numerical stable iteration for computing $\mathbf{A}^{-1}$ is [HFN11, GH06]:

$$\begin{aligned} c &= \mathsf{trace}(\mathbf{A}) \\ \mathbf{X}_0 &= c^{-1}\mathbf{I} \\ \mathbf{X}_{s+1} &= \mathbf{X}_s(2 - \mathbf{A}\mathbf{X}_s) \end{aligned}$$

where $\mathbf{I}$ is the identity matrix with the same dimensions as $\mathbf{A}$. Note that $\mathbf{A}$ is a covariance matrix and thus it is positive semi-definite and the trace of $\mathbf{A}$ dominates

[2] in the representation, 2^{-f} in the fixed-point data type.

Protocol for Computing the Inverse of the Covariance Matrix π_{MatInv}

The protocol is parametrized by the size q of the field. Let $\mathbf{A} \in \mathbb{Z}_q^{\ell \times \ell}$ be the encoding in $\mathbb{Z}_q$ of a covariance matrix where the elements are fixed-point numbers. The parties have as input a secret sharing $[\![\mathbf{A}]\!]_q$.

1. The parties locally compute shares of $c = \mathsf{trace}(\mathbf{A})$ by adding the values of the elements in the main diagonal of their shares of $\mathbf{A}$.

2. The parties use the Newton-Raphson division method to obtain a secret sharing of c^{-1} with resolution 2^{-f}. The subtractions can be performed locally and the multiplications using the distributed (matrix) multiplication functionality $\mathcal{F}_{\mathsf{DMM}}$ followed by the approximate truncation functionality $\mathcal{F}_{\mathsf{Trunc}}$.

3. The parties use the generalized Newton-Raphson method to obtain a secret sharing of $\mathbf{A}^{-1}$ with resolution 2^{-f} for the elements. The subtractions can be performed locally and the multiplications using the distributed matrix functionality $\mathcal{F}_{\mathsf{DMM}}$ followed by the approximate truncation functionality $\mathcal{F}_{\mathsf{Trunc}}$.

Figure 6.3: The protocol for securely and distributively computing the inverse of the covariance matrix.

the largest eigenvalue of $\mathbf{A}$. It is convenient to use $c = \mathsf{trace}(\mathbf{A})$ because the trace of $\mathbf{A}$ can be computed locally by parties that have shares of $\mathbf{A}$. To compute c^{-1} the Newton-Raphson is also used with x_0 set to an arbitrarily small value, as the convergence happens if the magnitude of the initial value is smaller than that of c^{-1}.

Note that in our case we use this method to compute securely the inverse of the covariance matrix, i.e, each party has a share of the covariance matrix as input and should receive as output random shares of its inverse, but no additional information should be learned by the parties. Hence we cannot perform a test after each iteration in order to check if the values already converged with resolution 2^{-f} (and thus stop the iterations at the optimal point) because this would leak information about the input based on how many iterations were performed. We have to use an upper bound γ on the number of iterations such that all possible covariance matrix converges with resolution 2^{-f} in γ iterations. A very conservative upper bound is $\gamma = 2k$ [HFN11].

The parties use the protocol described in Figure 6.3 to securely compute the inverse of a shared covariance matrix. We emphasize that the truncation used is the approximate one, but the Newton-Raphson method is self-correcting.

6.5 Linear Regression

We consider the setting in which there are a design matrix $\widetilde{\mathbf{X}}$ and a response vector $\widetilde{\mathbf{y}}$. We are interested in analyzing the statistical regression model

$$\widetilde{\mathbf{y}} = \widetilde{\mathbf{X}}\widetilde{\boldsymbol{\beta}} + \epsilon,$$

and therefore want to compute the estimated regression coefficient vector

$$\overline{\boldsymbol{\beta}} = (\widetilde{\mathbf{X}}^T\widetilde{\mathbf{X}})^{-1}\widetilde{\mathbf{X}}^T\widetilde{\mathbf{y}}$$

The design matrix is a $t \times m$ matrix where the elements are of the fixed-point data type with precision 2^{-f} and range 2^{k-f} (i.e., $\widetilde{\mathbf{X}} \in \mathbb{Q}^{t\times m}_{\langle k,f\rangle}$) and the response vector $\widetilde{\mathbf{y}} \in \mathbb{Q}^{t}_{\langle k,f\rangle}$. Let $\hat{\mathbf{X}} \in \mathbb{Z}^{t\times m}_{\langle k\rangle}$ be the element-wise representation of $\widetilde{\mathbf{X}}$ as signed integers and let $\mathbf{X} \in \mathbb{Z}^{t\times m}_{q}$ be the element-wise encoding of $\hat{\mathbf{X}}$ as elements of the field $\mathbb{Z}_q$. Define $\hat{\mathbf{y}}$ and $\mathbf{y}$ in the same way from $\widetilde{\mathbf{y}}$.

It is assumed that the parties hold shares of $\mathbf{X}$ and $\mathbf{y}$. They can then use the protocols for matrix multiplication, truncation and covariance matrix inversion that were described in the previous sections in order to compute shares of

$$\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Then they only need to reveal their final shares and convert the result back to the fixed-point data type in order to get $\overline{\boldsymbol{\beta}}$. In more details:

1. Online Phase:
   a) The players map their fixed precision real valued inputs to elements of a finite field as described in Section 6.3 and create the shares of $\mathbf{X}$ as described above.
   b) They compute $\mathbf{X}^T\mathbf{X}$ by using the matrix multiplication protocol π_{DMM} (described in Section 2.10). Once the multiplication is finished they ran the truncation protocol π_{Trunc}.
   c) They compute the inverse of $\mathbf{X}^T\mathbf{X}$ by running the protocol for computing the inverse of a covariance matrix (described in Section 6.4). Within the covariance matrix inversion protocol there are several calls to the matrix multiplication and truncation protocols.
   d) They run the matrix multiplication and the truncation protocols twice to obtain $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ and finally $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$.
   e) The players exchange their shares of the estimated regression coefficient vector and reconstruct it.
   f) The coefficients $\boldsymbol{\beta}$ obtained by the players are mapped back from finite field elements to real values with finite precision.

The security of the composed protocol follows from the secure sequential composition of the subprotocols using the facts that π_{DMM} securely implements the distributed matrix multiplication functionality $\mathcal{F}_{\mathsf{DMM}}$ and π_{Trunc} securely computes the approximate truncation functionality $\mathcal{F}_{\mathsf{Trunc}}$. It is assumed that a big enough k is used so that no overflow occurs and hence the correctness of the protocol follows. The final protocol implements the linear regression functionality $\mathcal{F}_{\mathsf{Reg}}$, described in Figure 6.4, that upon getting the shares of the design matrix $\mathbf{X}$ and the response vector $\mathbf{y}$, compute $\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$ and output $\boldsymbol{\beta}$ to the parties.

Functionality $\mathcal{F}_{\mathsf{Reg}}$

Input: Upon receiving a message from a party with its shares of the design matrix $\mathbf{X}$ and the response vector $\mathbf{y}$, record the shares, ignore any subsequent message from that party and inform the other parties about the receipt.

Output: Upon receipt of the shares from all parties, recover the design matrix $\mathbf{X}$ and the response vector $\mathbf{y}$, compute $\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$ and output $\boldsymbol{\beta}$ to the parties.

Figure 6.4: The linear regression functionality.

6.6 Removing the Trusted Initializer

If a trusted initializer is not desired, it is possible to obtain solutions in which the parties compute themselves the correlated data during an offline phase. Creating multiplication triples is a well studied problem [DPSZ12, DKL+13, DSZ15, Pul13]. Below we present one possible solution for the case of two honest-but-curious parties, which is the case considered in our experiments. The idea is to use the homomorphic properties of the Paillier's encryption scheme [Pai99]. For two large prime numbers p and p', the secret key of the Paillier's cryptosystem is $\mathsf{sk} = (p, p')$. The corresponding public key is $\mathsf{pk} = N = pp'$ and the encryption of a message $x \in \mathbb{Z}_N$ is done by picking a random $r \in \mathbb{Z}^*_{N^2}$ and computing $\mathsf{Enc}(\mathsf{pk}, x) = (N+1)^x r^N \mod N^2$. The following homomorphic properties of the Paillier's encryption scheme are used:

$$\mathsf{Enc}(\mathsf{pk}, x) \cdot \mathsf{Enc}(\mathsf{pk}, y) = \mathsf{Enc}(\mathsf{pk}, x + y \mod N) \text{ and}$$

$$\mathsf{Enc}(\mathsf{pk}, x)^y = \mathsf{Enc}(\mathsf{pk}, xy \mod N).$$

Given two vectors $\mathbf{x} = (x_1, \dots, x_n)$ and $\mathbf{y} = (y_1, \dots, y_n)$ where the second is given in clear and the first is encrypted element-wise (i.e., $\mathsf{Enc}(\mathsf{pk}, x_i)$ are revealed), one can compute a ciphertext corresponding to the inner product:

$$\mathsf{Enc}(\mathsf{pk}, \langle \mathbf{x}, \mathbf{y} \rangle \mod N) = \prod_{i=1}^{n} \mathsf{Enc}(\mathsf{pk}, x_i)^{y_i}.$$

The idea for computing the necessary correlated data for the distributed matrix multiplication protocol is to use the above fact in order to compute the non-local multiplication terms. Bob has a pair of public/secret keys for the Paillier's encryption scheme and sends to Alice the element-wise encryption *under his own public key* of the elements of the column/row that needs to get multiplied. Alice, having the plaintext corresponding to her own values on the appropriate column/row, can compute an encryption of the inner product under Bob's public key. She then adds a random blinding factor and sends the ciphertext to Bob, who can decrypt it, thus yielding distributed shares of the inner product between Alice and Bob. The protocol is described in Figure 6.5. Its security follows trivially from the IND-CPA security of the Paillier's encryption scheme [Pai99].

Pre-distribution Protocol π_{PRED}

The protocol is parametrized by the dimensions n_1, n_2, n_3 of the matrices to be multiplied. Bob holds a Paillier's secret key sk, whose corresponding public-key is pk. For a matrix $\mathbf{X}$, $x[i,j]$ denote the element in the i-th row and j-th column.

1. Bob chooses uniformly random $\mathbf{A}_{\mathsf{Bob}} \in \mathbb{Z}_N^{n_1 \times n_2}$ and $\mathbf{B}_{\mathsf{Bob}} \in \mathbb{Z}_N^{n_2 \times n_3}$, element-wise encrypts them under his own public key and send the ciphertexts to Alice.

2. Alice chooses uniformly random $\mathbf{A}_{\mathsf{Alice}} \in \mathbb{Z}_N^{n_1 \times n_2}$, $\mathbf{B}_{\mathsf{Alice}} \in \mathbb{Z}_N^{n_2 \times n_3}$ and $\mathbf{T} \in \mathbb{Z}_N^{n_1 \times n_3}$. For $i = 1, \ldots, n_1$, $j = 1, \ldots, n_3$, Alice computes the ciphertext

$$\begin{aligned}\tilde{c}[i,j] &= \mathsf{Enc}\left(\mathsf{pk}, t[i,j]\right) \cdot \\ &\cdot \prod_{k=1}^{n_2} \Big(\mathsf{Enc}\left(\mathsf{pk}, b_{\mathsf{Bob}}[k,j]\right)^{a_{\mathsf{Alice}}[i,k]} \cdot \mathsf{Enc}\left(\mathsf{pk}, a_{\mathsf{Bob}}[i,k]\right)^{b_{\mathsf{Alice}}[k,j]}\Big)\end{aligned}$$

and sends them to Bob. Alice outputs $\mathbf{A}_{\mathsf{Alice}}$, $\mathbf{B}_{\mathsf{Alice}}$ and $-\mathbf{T}$.

3. Bob decrypts the ciphertexts in order to get the matrix $\mathbf{C} = (\mathbf{A}_{\mathsf{Alice}}\mathbf{B}_{\mathsf{Bob}} + \mathbf{A}_{\mathsf{Bob}}\mathbf{B}_{\mathsf{Alice}} + \mathbf{T})$. Bob outputs $\mathbf{A}_{\mathsf{Bob}}$, $\mathbf{B}_{\mathsf{Bob}}$ and $\mathbf{C}$.

Figure 6.5: The protocol for pre-distributing the correlated data.

Note that the values r and r' that are distributed by the trusted initializer for performing the truncation protocol can be trivially computed by the parties themselves using distributed multiplications.

6.7 Experiments

We assessed our secure linear regression algorithm by implementing it and analyzing the results using ten real datasets and the case of two honest-but-curious parties, Alice and Bob. We chose a variety of different datasets based on the number of features and the number of instances (see Section 6.7). We used C++ as our programming language which we augmented with the BOOST libraries for functionality such as lexical_cast for type casting and asio for work with sockets. We also made use of the GMP and NTL libraries within C++ to implement our protocols. We built our system on top of a Microsoft Azure G4 series machine with Intel Xeon processor E5 v3 family, 224GB RAM size, 3072GB of disk size and 16 cores. Finally, we chose Ubuntu 12.04 as our operating system. We have merged the matrix multiplication and truncation protocols within one protocol for implementation purposes.

The online phase (Section 6.7) is very fast and capable of handling millions of records within less than an hour, which is a huge improvement to the previous results. We only use addition and multiplication of matrices on our online phase which makes it simple and easy to manage.

In the case when a trusted initializer is not desired one can use our computationally secure protocol, at the cost of having a costier offline phase (Section 6.7). However, because Alice and Bob only work over random inputs during the offline phase, the encryption, decryption and mathematical operations are all embarassingly parallelizable.

Datasets

All our datasets are contained within the UCI repository[3], with the exception of the State Inpatient Database (WA) which is provided by HCUP[4]. The UCI repository includes 48 regression task datasets from which we chose 9. Our datasets range in size from 395 instances to over 4 million and from 7 attributes to 367, and are summarized in Table 6.1.

Gas Sensor Array Under Dynamic Gas Mixtures This dataset represents data from 16 chemical sensors exposed to ethylene and CO mixtures at various concentration levels in the air. We added together the concentration of ethylene and the concentration of CO to create one continuous response variable of gas concentration and removed the time feature from the dataset. We then designated the first 8 sensor readings to Alice and the second 8 to Bob. This left us with a total of 4,208,261 sets of 16 sensor readings to different total concentrations of ethylene and CO.

Communities and Crime We used 122 attributes describing 1,993 communities and their law enforcement departments in 1990 to create this dataset. The goal with this dataset is to predict the number of violent crimes per capita for each community. All missing values present in the dataset (of which there were 36,850 distributed throughout 1,675 different communities and 22 different attributes) were replaced with 0s. These missing values were largely relevant to the communities' police departments. We also removed 5 variables that were present in the original data but described by the UCI documentation as non-predictive, namely state, county, community, community name, and fold. The final 122 attributes were then divided in half between Alice and Bob.

Auto MPG This dataset contains attributes describing 398 automobiles in attempt to predict MPG (miles per gallon) for each. We removed the car name attribute which was present in the original data and were left with 7 predictive features. We then replaced the 6 missing horsepower values with 0s. In the end we designated the cylinders, displacement, and horsepower features to Alice and the weight, acceleration, model year, and origin features to Bob.

BlogFeedback In an attempt to predict the number of comments a blog post will receive in the upcoming 24 hours, this dataset originally had 280 attributes. Since our complete dataset must be linearly independent, to enable the inversion of X^TX required in our protocol, we removed 57 of these original attributes leaving us with 223 predictors describing 52,397 blog posts. An example of such a feature would be the binary indicator of whether or not a

[3] UC Irvine Machine Learning Repository
`https://archive.ics.uci.edu/ml/datasets.html`

[4] `http://www.ahrq.gov/research/data/hcup/`

blog post was published on a Sunday. There are binary indicators of whether publication occurred on any of the other days of the week and therefore this feature, publication on a Sunday, is linearly dependent on the other six. Finally, the dataset was divided column wise, designating 111 attributes to Alice and the other 112 to Bob.

Wine Quality This dataset takes 11 attributes related to the white variant of Portuguese "Vinho Verde" wine which are used to predict the quality score of 4,897 wines. We designated the fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, and free sulfur dioxide features to Alice and the total sulfur dioxide, density, pH, sulphates, and alcohol features to Bob.

Bike Sharing In this dataset we took attributes describing a certain hour and day and attempted to predict the number of users for a bike sharing system. We removed the record index which was present in the original data as well as the count of casual users and the count of registered users and targeted the total rental bikes used ($casual + registered$) for our prediction. We were left with 13 predictors of 17,379 hour/day combinations which were used to model bike use. Alice received information on the dates, seasons, years, months, hours, and holidays while Bob was given information on weekdays, working days, weather situations, temperatures, feel temperatures, humidities, and windspeeds.

Student Performance We used 30 attributes describing 395 students across two schools to create this dataset. The goal with this dataset is to predict the final grade of each student in their math class. We removed two columns from the original dataset – one detailing students' performances in the first period and one detailing their performances in the second period. We identified the student's final grade as our sole response variable. The final 30 attributes were then divided evenly between Alice and Bob.

YearPredictionMSD In this dataset we have attributes describing audio features of 515,344 songs and we aim to predict the release year of each song. We kept all 90 features that were present in the original data provided by the UCI repository. In allocating the data we gave Alice the first 45 features and the second 45 to Bob.

State Inpatient Database (WA) From the HCUP State Inpatient Database (WA) we extracted attributes describing 25,180 beneficiaries who had at least one hospital admission within the state of Washington during the first nine months of the year between the years 2009 and 2012. The goal with this data is to predict the cost each beneficiary will incur in the final three months of the same year. We extracted demographic, medical, and previous cost information from the original data and replaced any missing values with a 0 value. We then designated the age, gender, race, number of chronic conditions, length of stay, and number of admits attributes to Alice. Bob was given a Boolean matrix of comorbidities as well as previous cost information.

Relative Location of CT Slices on Axial Axis In an attempt to predict the relative location of a CT slice on the axial axis of the human body, the original dataset had 384 attributes describing CT images. Since our complete dataset

Dataset Name	Number of Rows	Number of Columns	Training Time: Data Shared in the Clear	Training Time: Using Proposed Secure Protocol
Student Performance	395	30	0.3 sec	11.7 sec
Auto MPG	398	7	0.09 sec	1.2 sec
Communities and Crime	1,993	122	9 sec	147 sec
Wine Quality	4,897	11	0.9 sec	5.2 sec
Bike Sharing	17,379	13	3.7 sec	16.5 sec
State Inpatient Database (WA)	25,180	36	21 sec	93 sec
BlogFeedback	52,397	223	1,800 sec	9,000 sec
Relative Location of CT Slices on Axial Axis	53,500	367	6,000 sec	30,000 sec
YearPredictionMSD	515,344	90	3,800 sec	18,000 sec
Gas Sensor Array Under Dynamic Gas Mixtures	4,208,261	16	1,100 sec	4,500 sec

Table 6.1: Actual time required (in seconds) for the online phase of the secure protocol to build a predictive linear regression model.

must be linearly independent, we removed 17 of these original attributes leaving us with 367 predictors describing 53,500 CT images. We then divided this dataset column wise, designating 183 attributes to Alice and the other 184 to Bob.

Online Phase

We present in Table 6.1 the running times for the online phase of our protocol building a predictive linear regression model. Our online phase is very fast, computing a linear regression model for a matrix of over 4 million rows and 16 columns in under one hour. The regression coefficients computed with our secure protocol agree to the 5th decimal digit with regression coefficients computed without any security.

We briefly work out the theoretical complexity of computing $\boldsymbol{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$ with our online protocol. If our dataset (which is denoted by $\mathbf{X}$ in this formula), has m features and t records, then the total runtime for computing the $\boldsymbol{\beta}$ values is $O(tm^2)$ which means that the number of records in the dataset has only a linear effect on the run time of our implementation.

We used NTL for matrix multiplication with modular arithmetic. We also used GMP (the GNU Multi-Precision library) in conjunction with NTL to increase our performance. In the NTL library, the basic algorithm is used.

Note that $(\mathbf{X}^T\mathbf{X})$ is a square matrix with both dimensions equal to m, and for datasets in which the number of features is small relative to the number of records, computing $(\mathbf{X}^T\mathbf{X})^{-1}$ is very fast and negligible in respect to, for example, computing $\mathbf{X}^T\mathbf{X}$. Our online phase is faster and independent from the trusted party unlike similar implementations, such as Nikolaenko et al.'s implementation [NWI+13].

Computationally Secure Offline Phase

In the pre-processing of the computationally secure offline phase of the matrix multiplication protocol π_{DMM}, we use Paillier for encryption and decryption, but any additive homomorphic encryption scheme can be used. The downside of these schemes is that their encryption and decryption times are computationally intensive and, if the given dataset is large, the pre-processing phase can take a long time. This issue can be tackled by noticing that Alice and Bob, during this phase, only perform simple computations over random, independent data and thus one can use heavy parallelization to speed-up the running time.

Dataset Name	Number of Rows	Number of Columns	Offline Time With 16 Cores	Offline Time With 64 Cores	Offline Time With 256 Cores
Student Performance	395	30	20 sec	6 sec	2 sec
Auto MPG	398	7	4 sec	1 sec	0.3 sec
Communities and Crime	1,993	122	400 sec	100 sec	30 sec
Wine Quality	4,897	11	100 sec	30 sec	10 sec
Bike Sharing	17,379	13	350 sec	100 sec	30 sec
State Inpatient Database (WA)	25,180	36	1,500 sec	400 sec	100 sec
BlogFeedback	52,397	223	15,000 sec	4,000 sec	1,000 sec
Relative Location of CT Slices on Axial Axis	53,500	367	30,000 sec	10,000 sec	3,000 sec
YearPredictionMSD	515,344	90	70,000 sec	20,000 sec	6,000 sec
Gas Sensor Array Under Dynamic Gas Mixtures	4,208,261	16	100,000 sec	30,000 sec	10,000 sec

Table 6.2: Estimated time required (in seconds) for the offline phase of the secure protocol to build a predictive linear regression model.

For a dataset with t records and m features, in order to get coefficients securely and correctly, we use $i = 50$ iterations in the computation of inversion. Overall, we need $5tm + (3i+3)m^2 + m + 3i$ encryptions and $tm + (i+1)m^2 + t + i$ decryptions. We also have two matrix multiplications and 3 matrix additions between encryption and decryption. Each encryption in an Azure VM takes about 0.005 seconds for each core. It is then easy to see that the encryption phase is the bottleneck of the pre-processing phase and easily parallelizable. Since we have $5tm$ number of encryptions, by multiplying this number to the runtime of a single encryption time divided by number of cores, a good estimate of the pre-processing phase is achievable.

The estimated running time for the offline phase is given in Table 6.2. These estimated results are a huge improvement when compared to the previous result [HFN11] which took two days given a dataset with only about 50,000 records and are comparable to the total running time presented in [NWI+13] in the case of 256 cores.

6.8 Discussion

In this chapter we presented an information-theoretically secure protocol for privacy-preserving linear regression in the commodity-based model. The protocol has an offline phase where a trusted initializer pre-distributes random correlated data to the parties. The trusted initializer never engages again in the protocol. The online phase is orders of magnitude faster when compared to previous solutions in the literature [NWI+13, HFN11]. When a trusted initializer is not desirable or available, it is possible to substitute it with an computationally secure offline phase that is run between the parties. This offline phase is completely parallelizable, making it practical when a large number of cores is available.

One interesting direction for future work is trying to obtain practical protocols for linear regression that are secure against stronger types of adversaries, such as covert and malicious adversaries. One technique that can possibly help in achieving such goal is the compute with MACs approach [BDOZ11, DPSZ12, DKL+13].

7. Privacy-Preserving Classifiers

Data-driven machine learning classification has the ability to vastly improve the quality of our daily lives, but raises privacy issues from the point of view of both the user Alice as well as the model owner Bob. In this chapter, whose contents are based on [CDH$^+$16, DDKN15], we deal with the scenario in which Alice holds some data that she wants to classify using a model hold by Bob, but the protocol should be such that Bob learns nothing about Alice's data and Alice learns as little as possible about Bob's model. In other words, we tackle the problem of evaluating machine learning classifiers in a privacy-preserving way.

Our contributions

We show new privacy-preserving protocols for evaluating decision trees, hyperplane-based and Naive Bayes classifiers. Our protocols compare favorably against previous results [BPTG15, BPTG14, WFNL15]. The results are proven in the so-called commodity-based model and our online phase is unconditionally secure; that is, if the commodities are provided in an information theoretically secure fashion, the overall protocol will be information theoretically secure. Finally, differently from previously proposed solutions [BPTG15, BPTG14, WFNL15] the online phase of our protocols solely use modular additions and multiplications. No modular exponentiations are ever required. Our solutions are secure in the honest-but-curious model, which is consistent with the previous works in the area.

The main idea behind our solutions is to decompose the problem of obtaining privacy-preserving classifiers into the problem of obtaining secure versions of a few building blocks: distributed multiplication, distributed comparison, bit-decomposition of shares, distributed inner product and argmax computation, and oblivious input selection. We then either use the most efficient available versions of these protocols or propose new, more efficient ones. In more detail, the main contributions are:

- A novel protocol for computing private scoring of decision trees where Bob learns nothing about Alice's data and Alice learns only the depth of Bob's decision tree. Moreover, only modular additions and multiplications are required. In previous solutions [BPTG15, BPTG14, WFNL15], either modular exponentiations and fully homomorphic encryption are required [BPTG15, BPTG14] or

Paillier encryption-based private comparison schemes and Oblivious Transfer protocols (both requiring modular exponentiations) are required [WFNL15].

- Demonstration that applying our new building blocks to get decision tree and hyperplane-based classifiers improves the efficiency. We implement the cases of decision trees, support vector machines and logistic regression. We evaluate the implementation on 7 real data benchmark datasets from the UCI Machine Learning repository and present of the obtained accuracies and running times.

Model

For the sake of generality, some of the protocols in this chapter are presented considering u parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$, despite the fact that our application in this chapter only deal with two parties. The adversaries considered in this chapter are honest-but-curious and static (as in all other practical privacy-preserving classification protocols so far). Honest-but-curious adversaries follow the protocol instructions correctly, but try to learn additional information. A static adversary means that the set of corrupted parties is fixed before the protocol execution and remains unchanged during the execution. For a version of the UC composition theorem for this scenario please refer to the Theorem 4.20 of Cramer et al. [CDN15].

Outline

Section 7.1 explains the machine learning classifiers that are considered in this work. We then present in Section 7.2 the building blocks that are used in the privacy-preserving classifiers: a secure distributed comparison protocol, a secure argmax protocol, a secure bit-decomposition protocol and an oblivious input selection protocol. After that, Section 7.3 describes the privacy-preserving classifiers and Section 7.4 the experiments that we performed to assess their performance. Section 7.5 explains how the pre-distributed data can be generated by the parties if no trusted initializer is available (or desirable). Finally, Section 7.6 compares our solution with the related work and Section 7.7 presents our concluding remarks.

7.1 Machine Learning Classifiers

In this section we briefly review the machine learning models for which we propose privacy-preserving scoring protocols in Section 7.3. Our presentation and notation is similar to that of Bost et al. [BPTG15, BPTG14].

Decision Trees

Decision trees are non-parametric, discriminative classifiers[1]. Alice holds an input vector $\mathbf{x} = (x_1, \ldots, x_t) \in \mathbb{R}^t$ consisting of t features. The classification algorithm consists of a mapping C: $\mathbb{R}^t \rightarrow \{c_1, \ldots, c_k\}$ on $\mathbf{x}$. The result of the classification $C(\mathbf{x})$ is one of the k possible classes $c_1, \ldots, c_k$. The model is a tree structure and is

[1] Being non-parametric means that the structure of the model is not completely fixed, the model can grow in size to accommodate the complexity of the training data. Being discriminative means that the model learns boundaries between the classes.

held by Bob. Each internal node of the tree structure tests the value of a particular feature against a corresponding threshold and branches according to the results. Each leaf node specifies one of the k classes. The result of the classification is the class associated with the leaf reached from traversing the tree.

In all our secure protocols a full tree is assumed. In the case where a decision tree is not full, one can always fill it with dummy nodes to obtain a full tree. It is assumed, without loss of generality, that the trees are binary.

Bob's model is $D = (d, G, H, \mathbf{w})$, where d is the depth of the tree, $G\colon \ \{1, \ldots, 2^d\} \rightarrow \{1, \ldots, k\}$ is a mapping from the indices of the leaves to the indices of the classes, $H\colon \ \{1, \ldots, 2^d - 1\} \rightarrow \{1, \ldots, t\}$ is a mapping from the indices of the internal nodes (always considered in level-order) to the indices of Alice's input features and $\mathbf{w} = (w_1, \ldots, w_{2^d-1})$ with $w_i \in \mathbb{R}$ contains the thresholds corresponding to each internal node. For each internal node v_i with $1 \leq i \leq 2^d - 1$, let z_i be the Boolean variable denoting the result of comparing $x_{H(i)}$ with w_i, which is one if $x_{H(i)} \geq w_i$ and zero otherwise. The classification process goes as follows:

- Starting from the root node, for the current internal node v_i, evaluate z_i. If $z_i = 1$, take the left branch; otherwise, the right branch.
- The algorithm terminates when a leaf is reached. If the j-th leaf is reached, then the output is $c_{G(j)}$.

Hyperplane-Based Classifiers

Hyperplane-based classifiers are parametric, discriminative classifiers. For a setting with t features[2] and k classes, the model consists of k vectors $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_k)$ with $\mathbf{w}_i \in \mathbb{R}^t$ and the classification result is obtained by determining, for Alice's feature vector $\mathbf{x} \in \mathbb{R}^t$, the index

$$k^* = \operatorname*{argmax}_{i \in [k]} \langle \mathbf{w}_i, \mathbf{x} \rangle,$$

where $\langle \cdot, \cdot \rangle$ is the inner-product.

Hyperplane-based classifiers are very common in machine learning. They can be obtained, for example, through maximizing the margin (as in support vector machines, which are explained below), perceptron learning, Fisher linear discriminant analysis and least squares optimization. All these techniques result in hyperplane-based classifiers for which the privacy-preserving scoring protocols we propose in Section 7.3 are applicable.

Support vector machine (SVM) learning is a method for training classifiers based on different types of kernel functions – polynomial functions, radial basis functions, etc. An SVM is characterized by a linear separating hyperplane which maximizes the margins between the classes [DBK+96]. The decision boundary is maximized with respect to the data points from each class (known as support vectors) that are closest to the decision boundary. Support vector machines are a particular case of hyperplane-based classifiers. For the particular case of an SVM classifier with two classes c^+ and c^-, we can rephrase hyperplane-based classifiers as follows. Alice holds an input vector $\mathbf{x}$, Bob holds a model $(\mathbf{a}, b)$, where $\mathbf{a}$ is an t-dimensional vector (the weight vector) and b is a real number. The result of the classification is obtained by computing

$$\mathsf{sign}\left(\langle \mathbf{x}, \mathbf{a} \rangle + b\right),$$

[2] We can have one of the features being 1 in order to account for constants.

where $\mathsf{sign}(y)$ is $+$ if $y > 0$ and $-$ otherwise.

Logistic Regression is a classifier that models the posterior probability of the class given the input features by fitting a logistic curve to the relationship between them [NJ01]. As such, logistic regression model outputs can be interpreted as probabilities of the occurrence of a class. When the response is a binary variable with class labels c^+ and c^-, then for a new input instance $\mathbf{x}$, a trained logistic regression model outputs the probabilities

$$P_{C|X}(c^-|\mathbf{x}) = \frac{1}{1 + \exp(\langle \mathbf{x}, \mathbf{a} \rangle + b)}$$

and $P_{C|X}(c^+|\mathbf{x}) = 1 - P_{C|X}(c^-|\mathbf{x})$, where the weight vector $\mathbf{a}$ and the real number b are learned during the logistic regression model training process. The class decision for the given probability is then made based on a threshold value which is often set to 0.5: if $P_{C|X}(c^+|\mathbf{x}) \geq 0.5$, then we predict that the instance belongs to the positive class, and otherwise we predict the instance belongs to the negative class. In this case the classification can be done by computing

$$\mathsf{sign}\left(\langle \mathbf{x}, \mathbf{a} \rangle + b\right).$$

Naive Bayes Classifier

The Naive Bayes classifier is a parametric, generative classifier. Generative classifiers try to learn the distribution of the individual classes instead of only boundaries between them (as in discriminative classifiers). The Naive Bayes classifier uses the Bayes' rule and makes a conditional independence assumption that says that the features are independent when conditioned on the class. The model then consists of the probability distribution of the classes P_C and the conditional probability distribution of the each feature conditioned on the classes $P_{X_i|C}$. The classification result is obtained by determining, for Alice's feature vector $\mathbf{x} = (x_1, \ldots, x_t) \in \mathbb{R}^t$, the index

$$k^* = \operatorname*{argmax}_{j \in [k]} \left\{ \log P_C(c_j) + \sum_{i=1}^{t} \log P_{X_i|C}(x_i|c_j) \right\},$$

where the logarithms are used for stability reasons. On one hand, the prediction using this model can be poor compared to discriminative classifiers if the conditional independence assumption does not hold. On the other hand, this model works better with low amount of training data [NJ01] (discriminative classifiers can overfit the data).

7.2 Building Blocks

7.2.1 Secure Distributed Comparison

For performing secure distributed bitwise comparison we use the protocol of Garay et al. [GSV07] with secret sharings in the field $\mathbb{Z}_2$. That protocol has $\lceil \log \ell \rceil + 1$ rounds and uses $3\ell - \lfloor \log \ell \rfloor - 2$ multiplications. The protocol will be denoted by π_{DC} and it securely implements the distributed comparison functionality $\mathcal{F}_{\mathsf{DC}}$ that is described in Figure 7.1. For a detailed description of the protocol see the original paper of Garay et al. [GSV07] or Section 4.3.3 of De Hoogh's PhD thesis [dH12].

Functionality $\mathcal{F}_{\mathsf{DC}}$

$\mathcal{F}_{\mathsf{DC}}$ runs with parties $\mathcal{P}_1, \ldots, \mathcal{P}_n$ and is parametrized by the bit-length ℓ of the values being compared.

Input: Upon receiving a message from a party with its shares of $[\![x_i]\!]_2$ and $[\![y_i]\!]_2$ for all $i \in \{1, \ldots, \ell\}$, record the shares, ignore any subsequent messages from that party and inform the other parties about the receipt.

Output: Upon receipt of the inputs from all parties, reconstruct x and y from the bitwise shares. If $x \geq y$, then create and distribute to the parties the secret sharing $[\![1]\!]_2$; otherwise the secret sharing $[\![0]\!]_2$. Before the deliver of the output shares, the corrupt parties fix their shares of the output to any constant values. In both cases the shares of the uncorrupted parties are then created by picking uniformly random values subject to the correctness constraint.

Figure 7.1: The distributed comparison functionality.

7.2.2 Secure Argmax

Suppose that the parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$ have bitwise shares of a tuple of values $(v_1, \ldots, v_k)$ and want one of them, let's say $\mathcal{P}_1$, to learn all the arguments $m \in \{1, \ldots, k\}$ such that $v_m \geq v_j$ for all $j \in \{1, \ldots, k\}$, but no party should learn any v_j or the relative order between the elements. The parties just want $\mathcal{P}_1$ to learn

$$m = \arg \max_{j \in \{1, \ldots, k\}} v_j.$$

The argmax functionality $\mathcal{F}_{\mathsf{argmax}}$ is described in Figure 7.2. Using our protocol for secure distributed comparison it is possible to give simple and practical solutions for securely computing this function. A first idea, which optimizes the number of communication rounds, is to have the parties comparing in parallel each ordered pair of vectors and then using the result of the comparisons to determine the argmax. Note than when considering all executions of the comparison protocol involving a specific value v_j as the first argument, they will all return one if and only if the value is a maximum. The protocol π_{argmax} is described in Figure 7.3.

Theorem 7.1 ([CDH$^+$16]) *The argmax protocol π_{argmax} UC-realizes the argmax functionality $\mathcal{F}_{\mathsf{argmax}}$ against honest-but-curious adversaries in the commodity-based model.*

Proof: Correctness: The correctness follows trivially as for a maximum value, all comparison involving it as the first argument will return one, and so the product of the comparison results will also be one and the index will be added to the output. For all values which are not a maximum, at least one comparison will return zero, and so the product will be zero and the index will not be added to the output.

Functionality $\mathcal{F}_{\mathsf{argmax}}$

$\mathcal{F}_{\mathsf{argmax}}$ runs with parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$ and is parametrized by the bit-length ℓ of the values being compared and the number k of values being compared.

Input: Upon receiving a message from a party with its bitwise shares of $[\![v_{j,i}]\!]_2$ for all $j \in \{1, \ldots, k\}$ and $i \in \{1, \ldots, \ell\}$, record the shares, ignore any subsequent messages from that party and inform the other parties about the receipt.

Output: Upon receipt of the inputs from all parties, reconstruct the values v_j from the bitwise shares $v_{j,i}$, compute $m = \text{argmax}_{j\in\{1,\ldots,k\}} v_j$, and send m to $\mathcal{P}_1$.

Figure 7.2: The argmax functionality.

Secure Argmax Protocol π_{argmax}

Let ℓ be the bit length of the k values to be compared. The trusted initializer pre-distributes all the correlated randomness necessary for the execution of the instances of the distributed multiplication and comparison protocols. The parties have as input bitwise shares $[\![v_{j,i}]\!]_q$ for all $j \in \{1, \ldots, k\}$, $i \in \{1, \ldots, \ell\}$ and proceed as follows:

1. For all $j = 1, \ldots, k$ and $n \in \{1, \ldots, k\} \setminus j$, the parties compare in parallel $[\![v_{j,i}]\!]_2$ and $[\![v_{n,i}]\!]_2$ $(i = 1, \ldots, \ell)$. Let $[\![w_{j,n}]\!]_2$ denote the output obtained.
2. For all $j = 1, \ldots, k$, the parties computed in parallel $[\![w_j]\!]_2 = \prod_{n\in\{1,\ldots,k\}\setminus j} [\![w_{j,n}]\!]_2$.
3. The parties open w_j for P_1. If $w_j = 1$, P_1 append j to the value to be output in the end.

Figure 7.3: The secure argmax protocol.

Security: The first two steps only involve invocations of the distributed comparison π_{DC} and multiplication π_{DM} protocols, while the last step only opens one bit of information per index, indicating whether it corresponds to a maximum value or not; but this information is exactly the information contained in the output of the functionality $\mathcal{F}_{\mathsf{argmax}}$; hence the security of the protocol follows easily. Using the fact that π_{DC} UC-realizes $\mathcal{F}_{\mathsf{DC}}$ and π_{DM} UC-realizes $\mathcal{F}_{\mathsf{DMM}}$, the simulator $\mathcal{S}$ runs internally a protocol execution for the adversary $\mathcal{A}$ in which he simulates the ideal functionalities and uses dummy inputs for the uncorrupted parties. Using this leverage, it is trivial for $\mathcal{S}$ to extract the inputs of the corrupted parties in order to give to $\mathcal{F}_{\mathsf{argmax}}$. If $\mathcal{P}_1$ is corrupted, $\mathcal{S}$ can then use the output it gets from $\mathcal{F}_{\mathsf{argmax}}$ to adjust the output of the simulated protocol by picking an uncorrupted party and changing its share of each w_j appropriately before the opening. $\mathcal{Z}$ has no advantage in distinguishing the real and ideal worlds. ❚

Optimization: The round complexity for performing the multiplications in the second step can be improved by using a binary tree approach: the multiplicands are inserted as leaves of a binary tree and then we proceed upwards attributing to each internal node the value corresponding to the multiplication of its two children. Using this method the second step can take $\lceil \log k - 1 \rceil$ rounds.

We also present in Figure 7.4 an alternative argmax protocol π'_{argmax} that focus on optimizing the usage of the underlying multiplication and comparison protocols (instead of the number of rounds) and is based on the idea of iterating the comparison protocol over all values while always keeping tracking in form of bitwise secret sharings of the highest value found so far and its argument. The protocol π'_{argmax} realizes a slightly modified version $\mathcal{F}'_{\mathsf{argmax}}$ of the argmax functionality that only outputs the smallest index corresponding to a maximum.

Theorem 7.2 ([CDH$^+$16]) *The argmax protocol π'_{argmax} UC-realizes the argmax functionality $\mathcal{F}'_{\mathsf{argmax}}$ against honest-but-curious adversaries in the commodity-based model.*

Proof: The correctness follows trivially as we are simply iterating over the values while keeping track of the maximum value found so far and its argument, as one would do in the standard algorithm that is not concerned with any secure requirement. The simulation strategy follows the same lines as before. ❚

7.2.3 Secure Bit-Decomposition

This section deals with the problem of converting from shares $[\![x]\!]_q$ of a value x in a large ring $\mathbb{Z}_q$ to shares of $[\![x_i]\!]_2$ in $\mathbb{Z}_2$, where $x_\ell \cdots x_1$ is the binary representation of x. The bit-decomposition functionality $\mathcal{F}_{\mathsf{decomp}}$ is described in Figure 7.5. The usefulness of such functionality comes from the fact that it allows to convert from a representation that allows the efficient execution of algebraic operations to a representation that allows the efficient execution of Boolean operations (such as a comparison). We present in Figure 7.6 a bit-decomposition protocol π_{decomp} that is specialized for the two-party case with $q = 2^\ell$. Alice and Bob know shares a and b, respectively, such that $x = a + b \mod 2^\ell$. Note that Alice also knows the bit string representation of a, i.e., $a_\ell \ldots a_1$, and Bob similarly knows $b_\ell \ldots b_1$. The main

Secure Argmax Protocol π'_{argmax}

Let ℓ be the bit length of the k values $(v_1, \ldots, v_k)$ to be compared. Let $g = \lceil \log_2 k \rceil$. The trusted initializer pre-distributes all the correlated randomness necessary for the executions of the distributed comparison and multiplication protocols. The parties have as input secret sharings $\llbracket v_{j,i} \rrbracket_2$ for all $j \in \{1, \ldots, k\}$ and $i \in \{1, \ldots, \ell\}$. The bit string $max \in \{0,1\}^\ell$ stores the maximum value found until the current point and the bit string $argmax \in \{0,1\}^g$ its argument. Both are stored in the form of bitwise secret sharings and are initialized with the values v_1 and 1 respectively. The protocol proceeds as follows:

1. For $j = 2, \ldots, k$:
   a) Compare the values max and v_j using the bitwise shares and protocol π_{DC}. Let $\llbracket c \rrbracket_2$ denote its output.
   b) For $i \in \{1, \ldots, \ell\}$, compute $\llbracket max_i \rrbracket_2 \leftarrow \llbracket c \rrbracket_2 \llbracket max_i \rrbracket_2 + (1 - \llbracket c \rrbracket_2) \llbracket v_{j,i} \rrbracket_2$.
   c) For $f \in \{1, \ldots, g\}$, compute $\llbracket argmax_f \rrbracket_2 \leftarrow \llbracket c \rrbracket_2 \llbracket argmax_f \rrbracket_2 + j_f(1 - \llbracket c \rrbracket_2)$ where $j_g \ldots j_1$ is the bit string representing the number j.
2. Open the secret sharings $\llbracket argmax_p \rrbracket_2$ to $\mathcal{P}_1$ so that it can recover $argmax$.

Figure 7.4: The alternative secure argmax protocol.

observation is that the difference between the sum of $a = a_\ell \ldots a_1$ and $b = b_\ell \ldots b_1$ modulo 2^ℓ and two bit strings that xor to the bit string $x_\ell \cdots x_1$ is exactly equal to the carry bits.[3] Therefore we use a carry computation to obtain the bitwise secret sharings $\llbracket x_i \rrbracket_2$ starting from $a_\ell \ldots a_1$ and $b_\ell \ldots b_1$.

Theorem 7.3 ([CDH$^+$16]) *Over a ring $\mathbb{Z}_{2^\ell}$, the bit-decomposition protocol π_{decomp} UC-realizes the bit-decomposition functionality $\mathcal{F}_{\mathsf{decomp}}$ for the special case of two players against honest-but-curious adversaries in the commodity-based model.*

Proof: Correctness: The protocol implements a full adder logic $c_i = (a_i \wedge b_i) \vee ((a_i \oplus b_i) \wedge c_{i-1})$, which can be similarly expressed as $c_i = \neg(\neg(a_i \wedge b_i) \wedge \neg((a_i \oplus b_i) \wedge c_{i-1}))$ to obtain the carry bit string. By adding c_{i-1} into y_i, we convert from bit strings that sum to x modulo 2^ℓ to bit strings that xor to x, thus obtaining the shares of x_i modulo 2.

Security: The only non-local operations are the invocations of the distributed multiplication protocol π_{DM}, which UC-realizes $\mathcal{F}_{\mathsf{DMM}}$. Therefore the security follows essentially from the security of that protocol. $\mathcal{S}$ runs a copy of $\mathcal{A}$ and simulates an execution of the protocol using dummy inputs for the uncorrupted party. Since $\mathcal{S}$ is the one simulating the distributed multiplication functionality $\mathcal{F}_{\mathsf{DMM}}$, it can easily extract the corrupted party's share of the input in order to give it to $\mathcal{F}_{\mathsf{decomp}}$

[3] The protocol is similar to the one of Laud and Randmets [LR15], see the related works in Section 7.6 for more details.

Functionality $\mathcal{F}_{\mathsf{decomp}}$

$\mathcal{F}_{\mathsf{decomp}}$ runs with parties $\mathcal{P}_1, \ldots, \mathcal{P}_u$ and is parametrized by the bit-length ℓ of the value x being converted from additive sharings $\llbracket x \rrbracket_q$ in $\mathbb{Z}_q$ to additive bitwise sharings $\llbracket x_i \rrbracket_2$ in $\mathbb{Z}_2$ such that $x = x_\ell \cdots x_1$.

Input: Upon receiving a message from a party with its share of $\llbracket x \rrbracket_q$, record the share, ignore any subsequent messages from that party and inform the other parties about the receipt.

Output: Upon receipt of the inputs from all parties, reconstruct the value $x = x_\ell \cdots x_1$ from the shares, and for $i \in \{1, \ldots, \ell\}$ distribute new sharings $\llbracket x_i \rrbracket_2$ of the bit x_i. Before the output deliver, the corrupt parties fix their shares of the outputs to any constant values. The shares of the uncorrupted parties are then created by picking uniformly random values subject to the correctness constraints.

Figure 7.5: The bit-decomposition functionality.

Secure Two-Party Bit-Decomposition Protocol π_{decomp}

Let ℓ be the bit length of the value x to be reshared. All distributed multiplications using protocol π_{DM} will be over $\mathbb{Z}_2$ and the required correlated randomness is predistributed by the trusted initializer. The parties, Alice and Bob, have as input $\llbracket x \rrbracket_q$ for $q = 2^\ell$ and proceed as follows:

1. Let a denote Alice's share of x, which corresponds to the bit string $a_\ell \ldots a_1$. Similarly, let b denote Bob's share of x, which corresponds to the bit string $b_\ell \ldots b_1$. Define the secret sharings $\llbracket y_i \rrbracket_2$ as the pair of shares (a_i, b_i) for $y_i = a_i + b_i \mod 2$, $\llbracket a_i \rrbracket_2 \leftarrow a_i$ and $\llbracket b_i \rrbracket_2 \leftarrow b_i$.
2. Compute $\llbracket c_1 \rrbracket_2 \leftarrow \llbracket a_1 \rrbracket_2 \llbracket b_1 \rrbracket_2$ using π_{DM} and locally set $\llbracket x_1 \rrbracket_2 \leftarrow \llbracket y_1 \rrbracket_2$.
3. For $i = 2, \ldots, \ell$:
   a) Compute $\llbracket d_i \rrbracket_2 \leftarrow \llbracket a_i \rrbracket_2 \llbracket b_i \rrbracket_2 + 1$
   b) $\llbracket e_i \rrbracket_2 \leftarrow \llbracket y_i \rrbracket_2 \llbracket c_{i-1} \rrbracket_2 + 1$
   c) $\llbracket c_i \rrbracket_2 \leftarrow \llbracket e_i \rrbracket_2 \llbracket d_i \rrbracket_2 + 1$
   d) $\llbracket x_i \rrbracket_2 \leftarrow \llbracket y_i \rrbracket_2 + \llbracket c_{i-1} \rrbracket_2$
4. Output $\llbracket x_i \rrbracket_2$ for $i \in \{1, \ldots, \ell\}$.

Figure 7.6: The secure two-party bit-decomposition protocol.

and also derive the corrupted party's shares of the outputs in order to fix then in $\mathcal{F}_{\mathsf{decomp}}$. Consequently $\mathcal{Z}$ is unable to distinguish this ideal world from the real world interaction with $\mathcal{A}$ and parties executing π_{decomp}. ∎

Optimization: The idea to optimize the number of rounds to logarithmic is to compute speculatively (using secret sharings). In the first iteration the bit strings are divided in blocks of size 1 and the values of x_i and c_i are computed speculatively using both $c_{i-1} = 1$ and $c_{i-1} = 0$ for all i except $i = 1$, for which we know that there is no carry in and so only one computation is needed. The second iteration divides the bit strings in blocks of size 2 and uses the information from the previous iteration to compute $x_{i+1}x_i$ and $c_{i+1}c_i$ speculatively using both $c_{i-1} = 1$ and $c_{i-1} = 0$ (except for the least significant block that only needs one computation). The third iteration proceeds analogously with blocks of size 4 by joining the blocks of size 2, and so on. After $\lceil \log \ell \rceil + 1$ iterations one gets the desired bit strings $x_\ell \dots x_1$ and $c_\ell \dots c_1$. The first iteration uses 3ℓ instances of the multiplication protocol and needs two rounds of communication as there are pairs of sequential multiplications, all other iterations only need one round of communication and use 2ℓ multiplications each. Therefore in total the optimized protocol has $2 + \lceil \log \ell \rceil$ rounds and uses $2\ell \lceil \log \ell \rceil + 3\ell$ instance of the multiplication protocol.

7.2.4 Oblivious Input Selection

In our applications there are also circumstances in which Alice holds a vector of inputs $\mathbf{x} = (x_1, \dots, x_t)$ and Bob holds an index k, and they want to obtain bitwise secret sharings of x_k for further uses in the protocol, but without revealing any information about the inputs or k. The oblivious input selection functionality $\mathcal{F}_{\mathsf{OIS}}$, which captures this task, is described in Figure 7.7. In Figure 7.8 a protocol π_{OIS} realizing this functionality is explained. This idea was previously used by Toft [Tof07, Tof09b], where it was called "secret indexing".

Theorem 7.4 ([CDH$^+$16]) *The oblivious input selection protocol π_{OIS} UC-realizes the oblivious input selection functionality $\mathcal{F}_{\mathsf{OIS}}$ against honest-but-curious adversaries in the commodity-based model.*

Proof: Correctness: Straightforward to verify.

Security: Similarly to the previous proofs, $\mathcal{S}$ uses the fact that the only messages exchanged are for performing the distributed multiplications and the leverage of being able to simulate $\mathcal{F}_{\mathsf{DMM}}$ in order to simulate an execution of the protocol to $\mathcal{A}$ and at the same time being able to extract the inputs and the output shares of a corrupted party in order to forward to $\mathcal{F}_{\mathsf{OIS}}$. By doing so, $\mathcal{S}$ makes $\mathcal{Z}$ unable to distinguish this ideal world with interactions with $\mathcal{S}$ and $\mathcal{F}_{\mathsf{OIS}}$ from the real world with $\mathcal{A}$ and the parties executing the protocol π_{OIS}. ∎

7.3 Privacy-Preserving Classifiers

We now present our privacy-preserving classifiers using the building blocks from the previous sections.

Functionality $\mathcal{F}_{\mathsf{OIS}}$

$\mathcal{F}_{\mathsf{OIS}}$ runs with Alice and Bob and is parametrized by the size t of the input vector $\mathbf{x} = (x_1, \dots, x_t)$ and the bit-length ℓ of each input x_j.

Input: Upon receiving a message with the input vector $\mathbf{x} = (x_1, \dots, x_t)$ from Alice, store them, ignore any subsequent messages from her and inform Bob that the inputs were received.

Output: Upon receipt of the selected index $k \in [t]$ from Bob, distribute bitwise sharings $[\![x_{k,i}]\!]_2$ for $i \in \{1, \dots, \ell\}$ and ignore any subsequent messages. Before the output deliver, the corrupt party fix its shares of the outputs to any constant values. The shares of the uncorrupted parties are then created by picking uniformly random values subject to the correctness constraints.

Figure 7.7: The oblivious input selection functionality.

Oblivious Input Selection Protocol π_{OIS}

Let ℓ be the bit length of the inputs to be shared and t the dimension of the input vector. The trusted initializer pre-distributes all the correlated randomness necessary for the executions of π_{DM} over $\mathbb{Z}_2$. Alice has as input a vector of values, $\mathbf{x} = (x_1, \dots, x_t)$, and Bob has as input $k \in [t]$, the index of the desired input value. They proceed as follows:

1. Define $y_k = 1$ and, for $j \in \{1, \dots, t\} \setminus \{k\}$, $y_j = 0$. For $j \in \{1, \dots, t\}$ and $i \in \{1, \dots, \ell\}$, let $x_{j,i}$ denote the i-th bit of x_j. Let $[\![y_j]\!]_2 \leftarrow y_j$ and $[\![x_{j,i}]\!]_2 \leftarrow x_{j,i}$.
2. For $i = 1, \dots, \ell$, compute $[\![z_i]\!]_2 \leftarrow \sum_{j=1}^{t} [\![y_j]\!]_2 [\![x_{j,i}]\!]_2$ using the distributed multiplication π_{DM} over $\mathbb{Z}_2$.
3. Output $[\![z_i]\!]_2$ for $i \in \{1, \dots, \ell\}$.

Figure 7.8: The oblivious input selection protocol.

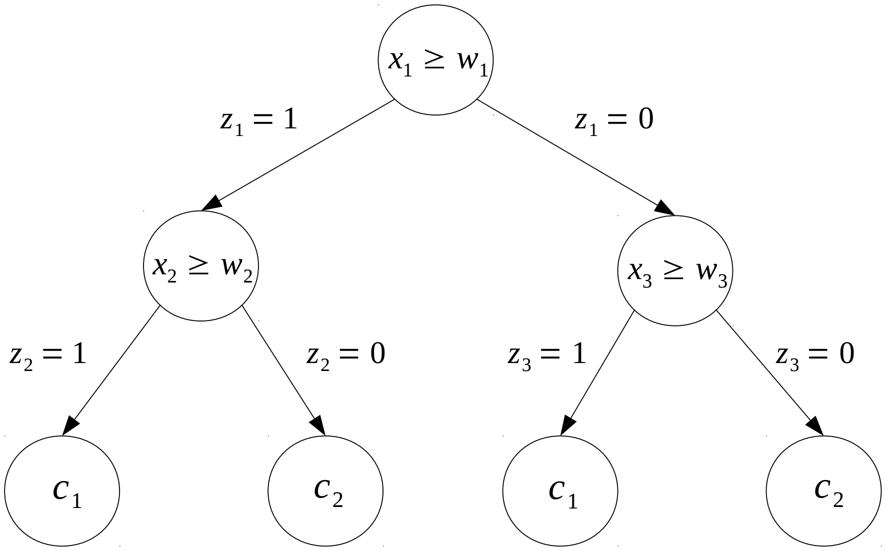

Figure 7.9: Example of decision tree with 7 nodes and 2 classes.

Secure Decision Trees

Here, Alice inputs $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ and the classification algorithm will result in one of the k possible classes $c_1, \ldots, c_k$. Bob holds the model $D = (d, G, H, \mathbf{w})$, where d is the depth of the tree, G maps the leaves to classes, H maps internal nodes (always considered in level-order) to input features and $\mathbf{w}$ is a vector of thresholds. Each internal node of the tree structure tests the value of a particular feature against a corresponding threshold and branches according to the results. Each leaf node specifies a class. In all our secure protocols, we assume without loss of generality that we have a full binary tree. In case a decision tree is not full, one can always fill it with dummy nodes and obtain a full one. Let z_i be the Boolean variable denoting the result of comparing $x_{H(i)}$ with w_i. We recall the classification algorithm:

- Starting from the root node, for the current internal node v_i, evaluate z_i. If $z_i = 1$, take the left branch; otherwise, the right branch.
- The algorithm terminates when a leaf is reached. If the j-th leaf is reached, then the output is $c_{G(j)}$.

Similar to Bost et al. [BPTG15], the classification can be expressed as a polynomial P_G: $\{0,1\}^{2^d-1} \rightarrow \{1, \ldots, k\}$ that depends on the mapping G from the leaves to the classes. On input $\mathbf{z} = (z_1, \ldots, z_{2^d-1})$, P_G gives the classification result. This polynomial is a sum of terms such that each term corresponds to one possible path in the tree: the term corresponding to path taken by $\mathbf{x}$ in the tree evaluates to the classification result (i.e., the class associated to that leaf), while the remaining terms evaluate to zero. For example, for the tree portrayed in Figure 7.9, the polynomial P_G that represents the tree is: $P_G(z_1, z_2, z_3) = z_1 z_2 c_1 + z_1 \bar{z}_2 c_2 + \bar{z}_1 z_3 c_1 + \bar{z}_1 \bar{z}_3 c_2$ where $\bar{x}$ denotes $1 - x$.

The idea of our secure protocol is that, for each internal node, Alice and Bob use the oblivious input selection protocol π_{OIS} to obtain bitwise secret sharings of the value $x_{H(i)}$ that will be compared against the threshold w_i of this node. Note that, as Alice does not learn any information from the execution of π_{OIS}, she does not

Functionality $\mathcal{F}_{\mathsf{DT}}$

$\mathcal{F}_{\mathsf{DT}}$ is parametrized by the tree depth d, which is revealed to Alice.

Input: Upon receiving the feature vector $\mathbf{x}$ from Alice or the decision tree model $D = (d, G, H, \mathbf{w})$ from Bob, store it, ignore any subsequent message from that party, and inform the other party about the receipt.

Output: Upon receipt of the inputs from both parties, evaluate the decision tree D with the input $\mathbf{x}$. Let j be the reached leaf. Output $G(j)$ to Alice.

Figure 7.10: The decision tree functionality.

Secure Decision Tree Protocol π_{DT}

Alice has as input a feature vector $\mathbf{x}$ and Bob has a decision tree model $D = (d, G, H, \mathbf{w})$. Alice and Bob proceed as follows:

1. For $i = 1, \ldots, 2^d - 1$, Alice and Bob obtain bitwise secret sharings of $x_{H(i)}$ by using π_{OIS} with inputs $x_1, \cdots, x_n$ from Alice and input $H(i)$ from Bob.
2. For $i = 1, \ldots, 2^d - 1$, Alice and Bob securely compare $x_{H(i)}$ and w_i. For the input w_i, Bob inputs its bit representation and Alice inputs zeros. Let $[\![z_i]\!]_2$ denote the result.
3. For $j = 0, \ldots, 2^d - 1$, let $j_d \ldots j_1$ be the binary representation of j with d bits and let $b_\alpha \ldots b_1$ for $\alpha = \lceil \log k \rceil$ be the binary representation of $G(j+1) - 1$. For $r = 1, \ldots, \alpha$, initialize $[\![y_{j,r}]\!]_2$ with the shares $(0, b_r)$. Initialize $u = 1$ and $s = d$. While $s > 0$ do:
   a) For $r = 1, \ldots, \alpha$, $[\![y_{j,r}]\!]_2 \leftarrow [\![y_{j,r}]\!]_2([\![z_u]\!]_2 + j_s)$.
   b) Update $u \leftarrow 2u + j_s$ and $s \leftarrow s - 1$.
4. For all $r = 1, \ldots, \alpha$ compute $[\![\sigma_r]\!]_2 \leftarrow \sum_{j=0}^{2^d-1} [\![y_{j,r}]\!]_2$ and open σ_r to Alice. Alice reconstructs σ from the bit string $\sigma_\alpha \ldots \sigma_1$ and outputs $k^* = \sigma + 1$.

Figure 7.11: The protocol for secure evaluation of a decision tree.

know which feature will be used in the comparison at each internal node. Then the comparisons are performed using the secure distributed comparison protocol π_{DC} in order to obtain $\mathbf{z}$, which is then used to evaluate the polynomial P_G using the secure multiplication protocol π_{DM} and local addition of secret sharings. The only information leaked about the tree structure to Alice is its depth d. The decision tree functionality $\mathcal{F}_{\mathsf{DT}}$ is described in Figure 7.10 and a more detailed description of the protocol π_{DT} realizing $\mathcal{F}_{\mathsf{DT}}$ is in Figure 7.11.

Theorem 7.5 ([CDH$^+$16]) *The decision tree protocol π_{DT} UC-realizes the decision tree functionality $\mathcal{F}_{\mathsf{DT}}$ against honest-but-curious adversaries in the commodity-based model.*

Proof: Correctness: For each leaf $j \in \{1, \dots, 2^d\}$, the secret sharings $[\![y_{j-1,r}]\!]_2$ with $r = 1, \dots, \lceil \log k \rceil$ obtained in step 3 correspond to a binary representation of the index of its associated class (offset by 1) if j is the leaf that would be reached by using the model D on input $\mathbf{x}$; otherwise they correspond to zeros as at least one of the terms $[\![z_f]\!]_2 + j_s$ in the multiplication would be zero. Thus in step 4, by summing all $[\![y_{j-1,r}]\!]_2$ for $j \in \{1, \dots, 2^d\}$, opening the results and adding 1, Alice obtains the result of the classification k^*.

Security: Alice learns the depth d of the tree in order to allow the execution, but this is leaked by $\mathcal{F}_{\mathsf{DT}}$ as well. In the first three steps messages are only exchanged in order to execute the sub-protocols π_{OIS}, π_{DC} and π_{DM} respectively, which UC-realize the functionalities $\mathcal{F}_{\mathsf{OIS}}$, $\mathcal{F}_{\mathsf{DC}}$ and $\mathcal{F}_{\mathsf{DMM}}$ respectively. Then the last step simply reveals the bit string encoding the class that was the result of the classification to Alice. The simulation strategy is similar to the one in the previous sections. The simulator $\mathcal{S}$ internally runs a protocol execution for the adversary $\mathcal{A}$ in which $\mathcal{S}$ simulates $\mathcal{F}_{\mathsf{OIS}}$, $\mathcal{F}_{\mathsf{DC}}$ and $\mathcal{F}_{\mathsf{DMM}}$ and uses dummy inputs for the uncorrupted parties. Using this leverage $\mathcal{S}$ can easily extract the inputs of the corrupted party, $\mathbf{x}$ in case Alice is corrupted or $D = (d, G, H, \mathbf{w})$ in case Bob is corrupted, in order to forward to $\mathcal{F}_{\mathsf{DT}}$. In case Alice is corrupted, upon learning the correct output from $\mathcal{F}_{\mathsf{DT}}$, $\mathcal{S}$ can adjust appropriately Bob's shares of σ_r in the simulated protocol in order to match the right result. With this simulation strategy, $\mathcal{Z}$ cannot distinguish the ideal and real worlds. ▌

Optimization: All independent operations are run in parallel and the round complexity of step 3(a) can be reduced using techniques similar to the previous sections.

Secure Hyperplane-Based Classifiers

A privacy-preserving hyperplane-based classifier is easily achievable using our building blocks. The classification result of hyperplane-based classifiers is given by the index

$$k^* = \underset{i \in [k]}{\operatorname{argmax}} \langle \mathbf{w}_i, \mathbf{x} \rangle.$$

Thus, one just needs to represent the model and features in $\mathbb{Z}_q$, compute each inner product between $\mathbf{w}_i$ and $\mathbf{x}$ by using π_{IP}, input the results into the bit-decomposition protocol π_{decomp} and then into the argmax protocol π_{argmax}.

In the specific case of SVM, Alice holds an input vector $\mathbf{x}$, Bob holds a model $(\mathbf{a}, b)$, where $\mathbf{a}$ is an t-dimensional vector (the weight vector) and b is a real number. The result of the classification is obtained by computing

$$\mathsf{sign}\left(\langle \mathbf{x}, \mathbf{a} \rangle + b\right),$$

where $\mathsf{sign}\,(y)$ is $+$ if $y > 0$ and $-$ otherwise. The overall idea for obtaining privacy-preserving SVM classifiers is as follows: Alice inputs her personal vector $\mathbf{x}$ and Bob inputs his model vector $\mathbf{a}$ to the secure distributed inner product protocol π_{IP}. After that, the result is run through the bit-decomposition protocol π_{decomp}. The resultant bitwise shares, together with b, are used in the comparison protocol π_{DC} to determine the final result, which is then opened to Alice as her prediction.

To score a logistic regression classifier with threshold 0.5 one needs to check whether the expression

$$\log\left(\frac{P_{C|X}(c^{+}|\mathbf{x})}{P_{C|X}(c^{-}|\mathbf{x})}\right)$$

is positive or not, where

$$P_{C|X}(c^{-}|\mathbf{x}) = \frac{1}{1+\exp(\langle \mathbf{x}, \mathbf{a} \rangle + b)}.$$

This boils down to computing $\mathsf{sign}\,(\langle \mathbf{x}, \mathbf{a} \rangle + b)$, where $\mathbf{x}$ is the input feature vector, and the $\mathbf{a}$ and b are vectors defining the logistic regression classification model (held by Bob). Therefore, the protocol used to privately evaluate a logistic regression model is exactly the same as the one described in the support vector machines section.

The security of these compositions follows from the security of the sub-protocols and the fact that no values are ever opened before the final result; each party only sees shares, which appear completely random.

Secure Naive Bayes Classifier

In the case of the Naive Bayes classifier the classification is done by computing the index

$$k^{*} = \underset{j\in[k]}{\operatorname{argmax}} \left\{ \log P_C(c_j) + \sum_{i=1}^{t} \log P_{X_i|C}(x_i|c_j) \right\}.$$

If the features have finite alphabets, then a privacy-preserving classifier can be obtained as follows. First Bob expresses the log of the probabilities as bit-strings. The oblivious input selection protocol π_{OIS} is then used to share the appropriate $P_{X_i|C}(x_i|c_j)$ without revealing x_i to Bob (Bob inputs the conditional probability for each possible value of x_i, and Alice selects the appropriated one to be shared). The terms to be given as input to the argmax are then compute locally using secret sharing additions, and finally π_{argmax} is used to compute the output.

7.4 Experiments

For decision trees, SVM and logistic regression models we report accuracy (calculated using 10-fold cross validation) for 7 different datasets within the UCI Repository[4].

[4] UC Irvine Machine Learning Repository `https://archive.ics.uci.edu/ml/datasets.html`

We also report average classification time for an instance in each dataset when following our privacy-preserving protocol as well as average time required when the classification is done in the clear. Note that the bit-length used to express the values should be large enough as not to compromise the accuracy of the algorithms. It is no real gain for applications if the performance is improved at the cost of drastically decreasing the accuracy, therefore the accuracy is also reported.

Support Vector Machine: For this study, we tested SVM with a linear kernel, and we report the results for accuracy for 7 different datasets from the UCI repository. We leveraged the e1071 package within R [MDH$^+$15], setting type to 'C-classification', indicating our problems were classification tasks.

Decision Trees: Our implementation used the classification and regression tree algorithm (CART) [BFOS84] in R [TAR15]. The minimum deviance (mean squared error) is used as the test parameter for proceeding with a new split. That is, adding a node should reduce the error by at least a certain amount. For our models, we set the complexity parameter to 0.01 and report the corresponding accuracy.

Logistic Regression: For our experimentation, we used R's base glm function [R C15], setting the family parameter to binomial(link="logit") to obtain a logistic regression model.

The following datasets were chosen for our experimentation:

1. **Breast Cancer Wisconsin (Diagnostic):** The goal with this dataset is to classify 568 different tumors as malignant or benign. Each tumor is characterized by 30 different continuous features derived from an image of the tumor (i.e. perimeter, area, symmetry, etc.).

2. **Pima Indians Diabetes:** This dataset includes 767 females of at least 21 years of age, all with Pima Indian decent, and we wish to identify those with diabetes. We leverage 8 different continuous features which describe each woman's health (examples: body mass index, diastolic blood pressure).

3. **Parkinsons:** Here, the task is to differentiate between patients with and without Parkinsons. To this end, the dataset includes 22 features, all of which are measures derived from voice recordings of 195 different patients (example: average vocal fundamental frequency).

4. **Connectionist Bench (Sonar, Mines vs. Rocks):** The goal with this dataset is to differentiate whether 207 sonar signals were bounced off of a metal cylinder vs. a roughly cylindrical rock. Each of the 60 features is within the range of 0.0 to 1.0 and represents energy within a particular frequency band over a certain period of time.

5. **Hill-Valley:** The task for this dataset is to identify hills vs. valleys in terrain. Each of the 100 continuous features is a point on a 2-D graph. We chose the dataset which did not contain any noise.

6. **LSVT Voice Rehabilitation:** This dataset includes 126 patients who have undergone voice rehabilitation treatment and we wish to determine the success

of their treatment, i.e. whether their phonations are considered acceptable or unacceptable. To do this, we leverage 312 features, each of which is the result of a different speech signal algorithm.

7. **Spambase:** Here, the goal is to identify 4,600 emails as either spam or not spam. This dataset includes 57 features which describe the contents of each email (examples: word frequencies, number of capital letters).

Implementation Details

To generate preliminary results, the privacy-preserving algorithms were implemented in Java, and compared against a simple implementation without any privacy preservation. For our experiments with the privacy-preserving classifiers, a general bit length, ℓ, of 64 bits was used for representing all the inputs and throughout all calculations, as this allowed for a good trade off between complexity and space for precision. For some trials, a smaller bit length might have served with sufficient precision.

All values had to be converted to integers to properly work in the proposed algorithms. This was accomplished by choosing a multiplier value and applying it to the features and the weights for SVM and logistic regression or the thresholds for decision trees and rounding any remaining decimals. Furthermore, since calculations were done over a ring, any negative values had to be expressed as their additive inverses. This means in addition to precision considerations, the bit length must be selected in such a way that the positive values and negative values will remain distinctly separate in the lower half and upper half of the values, respectively. This allows us to differentiate between positive and negative values by comparing against $2^{\ell-1}$ instead of 0.

Table 7.1 presents the results for the case of decision tree classifiers and Table 7.2 for SVM and logistic regression classifiers. These results were generated using a laptop computer with 16 GB DDR4 RAM at 2133 MHz and an Intel Core i7 6700HQ at 2.6 GHz. For each dataset the average was computed using more than 10000 scorings.

Analysis and Comparisons to Previous Results

Next we analyze our experimental results and compare then with the previous works.

Decision Trees: the computing time for running our protocol for the privacy-preserving evaluation of decision trees is at most 13 milliseconds for trees of depth up to 9. In Bost et al. [BPTG15], for evaluating a tree of depth 4, the computing time is in the order of a few seconds. Our protocol has 11 rounds of communication or less for trees with depth up to 9, while their number of interactions is always over 30, even for trees of depth 4. In the case of the protocols for computing decision trees of Wu et al. [WFNL15], the computing time for a tree with depth 4 is around 100 ms. The communication complexity of our protocol for a decision tree of depth 4 and 8 features is around 3KB, while the results in [WFNL15] are around 100KB and in [BPTG15] are around 3MB for trees of the same dimension. As stated in these previous works, solutions based on general purpose multiparty computation frameworks have a much poorer performance than their specific protocols (and hence

Dataset	Depth of Tree	Number of Features	Accuracy	Classification Time in the Clear (ms)	Classification Time Secure Protocol (ms)	Communication Complexity Uplink+Downlink (kB)
Breast Cancer	4	30	95.95%	0.07 + 1 RTT/2	3.20 + 10 RTT/2	7.96
Diabetes	9	8	77.18%	0.02 + 1 RTT/2	9.11 + 11 RTT/2	95.94
Parkinson's	4	22	88.72%	0.40 + 1 RTT/2	3.62 + 10 RTT/2	6.09
Connectionist Bench	4	60	73.91%	0.10 + 1 RTT/2	9.64 + 10 RTT/2	14.99
Hill-Valley	3	100	49.83%	0.14 + 1 RTT/2	4.85 + 9 RTT/2	11.37
LSVT rehabilitation	3	310	79.37%	0.75 + 1 RTT/2	12.79 + 9 RTT/2	34.34
Spambase	6	57	88.89%	0.10 + 1 RTT/2	9.33 + 11 RTT/2	60.04

Table 7.1: Results of the experiments for the decision tree classifiers. The classification time is given as the computing time plus the number of half roundtrip times (RTT/2).

Dataset	Number of Features	Accuracy	Classification Time in the Clear (ms)	Classification Time Secure Protocol (ms)	Communication Complexity Uplink+Downlink (kB)
SVM					
Breast Cancer	30	97.71%	0.06 + 1 RTT/2	3.47 + 16 RTT/2	0.92
Diabetes	8	77.05%	0.02 + 1 RTT/2	3.04 + 16 RTT/2	0.57
Parkinson's	22	87.18%	0.04 + 1 RTT/2	3.36 + 16 RTT/2	0.79
Connectionist Bench	60	74.70%	0.10 + 1 RTT/2	4.12 + 16 RTT/2	1.39
Hill-Valley	100	57.59%	0.17 + 1 RTT/2	4.89 + 16 RTT/2	2.01
LSVT rehabilitation	310	80.16%	0.51 + 1 RTT/2	9.16 + 16 RTT/2	5.29
Spambase	57	92.72%	0.10 + 1 RTT/2	4.06 + 16 RTT/2	1.34
Logistic Regression					
Breast Cancer	30	95.95%	0.07 + 1 RTT/2	3.55 + 16 RTT/2	0.92
Diabetes	8	77.31%	0.02 + 1 RTT/2	3.06 + 16 RTT/2	0.57
Parkinson's	22	85.13%	0.04 + 1 RTT/2	3.35 + 16 RTT/2	0.79
Connectionist Bench	60	74.40%	0.11 + 1 RTT/2	4.16 + 16 RTT/2	1.39
Hill-Valley	100	60.07%	0.16 + 1 RTT/2	4.97 + 16 RTT/2	2.01
LSVT rehabilitation	310	53.17%	0.49 + 1 RTT/2	9.64 + 16 RTT/2	5.29
Spambase	57	92.70%	0.10 + 1 RTT/2	4.17 + 16 RTT/2	1.34

Table 7.2: Results of the experiments for the SVM and logistic regression classifiers. The classification time is given as the computing time plus the number of half roundtrip times (RTT/2). All datasets only have two classes.

than the solutions presented here as well).

Support Vector Machines: We run the protocols proposed in [DDKN15] with the building blocks presented in this paper. While there are no implementation times given in [DDKN15], it is clear that our implementations have a significant impact in the performance. The number of rounds is usually the most important factor in determining the latency of these protocols and we reduce the round complexity from linear to logarithmic in the input length. Compared to the implementations described in Bost et al. [BPTG15] the computation times are about 50ms for 30 and 47 features. In our case for 30 features, the computing time is less than 4 ms. Our number of rounds is larger: our solution takes 16 rounds, while their solution takes 7 rounds. If the roundtrip time is the major factor in the total time their solution is preferable to ours. The main reason for the elevated round complexity in our solution is the bit decomposition protocol, which is not needed in their work.

Logistic Regression: The efficiency of the logistic regression protocol is the same as the support vector machine one.

7.5 Removing the Trusted Initializer

Our protocols assume that pre-distributed data is made available to the players by a trusted initializer: random binary multiplication triples (binary Beaver triples) in the case of decision trees and Naive Bayes classifier; and random binary multiplication triples and random inner product evaluations for hyperplane-based classifiers.

In case a trusted initializer is not available or desirable, Alice and Bob can run pre-computations during a setup phase. In the case of the protocol evaluating decision trees or Naive Bayes classifier, to obtain the binary random multiplication triples, Alice and Bob can run oblivious transfer protocols on random inputs. The outcome of these evaluations can be easily transformed in the random binary multiplication triples (see, for instance, [NNOB12]). The nice point of this solution is that oblivious transfer can be extended efficiently by using symmetric cryptographic primitives [IKNP03, KK13, ALSZ13]. The online phase of our protocols would remain the same - using solely modular additions and multiplications. Therefore, even considering the offline phase, our protocol would still be substantially more efficient than the protocols proposed in [BPTG15, BPTG14] and in [WFNL15]. We also remark that the protocol for evaluating decision trees in [BPTG15, BPTG14] does not allow its computationally heavy steps (Paillier encryptions and uses of a somewhat homomorphic encryption scheme) to be pre-computed. We also note that while the oblivious transfer executions in [WFNL15] could also be pre-computed, the Paillier encryption scheme would still be needed in the online phase.

7.6 Related Works

In this section we compare our results with the related work.

Privacy-Preserving Scoring of Machine Learning Classifiers: General (non-application specific) privacy-preserving protocols for privately scoring machine learning classifiers were proposed just recently by Bost et al.[BPTG15, BPTG14] for the case of hyperplane-based classifiers, Naive Bayes and decision trees and Wu et al. [WFNL15] for decision trees and random forests.

De Hoogh et al. [dHSCodA14] introduced the most efficient protocol for privacy-preserving training of decision trees with categorical attributes only. They also presented a protocol for privacy-preserving scoring of decision trees. Their protocol is designed for categorical attributes. It does not scale well for fined-grained numerical attributes - the complexity of the protocol increases exponentially on the bit-length representation of a category.

Many classification problems are characterized by numerical attributes, such as age, temperature, or blood test results, or by a combination of numerical and categorical attributes. The well known top down algorithms to induce decision trees from data (ID3, CART) can easily be extended to include numerical attributes as well. This is typically done with a binary split at internal nodes, e.g. instances with “cholesterol level $\leq p$” go down the left branch, and instances with “cholesterol level $> p$” go down the right. The threshold p is chosen dynamically at each node as the tree is grown, and, unlike with categorical attributes, a numerical attribute may appear more than once in the same tree branch, but with different thresholds. For instance, in the branch below the node “cholesterol level $\leq p$”, a new node “cholesterol level $\leq p^*$” may appear, with p^* a smaller threshold than p. The process of dynamically choosing and refining thresholds adds to the expressivity of decision trees with numerical values, making the hypothesis space of such trees far richer than that of decision trees with categorical values.

Bost et al. [BPTG15, BPTG14] implemented hyperplane-based and Naive Bayes classifiers by using a secure protocol for computing the inner product based on the Paillier encryption scheme and a comparison protocol that also relies heavily on that encryption scheme.

The decision tree protocol of Bost et al. [BPTG15, BPTG14] is divided in two phases. In a first stage Paillier-based comparison protocols are run with Alice inputting a vector containing her features and Bob inputting the threshold values of the decision tree. On a second stage, fully homomorphic encryption is used to process the outcomes of the comparison protocols run in the first stage. It is claimed that the protocol leaks nothing about the tree (we will show that in a more realistic attack scenario this is not true) and the second stage is round-optimal. However, the computations to be performed are heavy and the first stage involves many rounds (in total their protocol typically has more rounds than ours). In our solution, we allow the depth of the tree to be leaked, but avoid altogether using Paillier and fully homomorphic encryption. In our solution, the online phase for evaluating decision trees uses solely modular additions and multiplications.

Wu el at. [WFNL15] proposed protocols for decision trees and random forests that are based on an original comparison protocol using Paillier encryption scheme and oblivious transfer. The Paillier encryption scheme uses modular exponentiation and oblivious transfer protocols normally use operations that are as expensive as public-key cryptographic primitives. As already pointed out our solutions use, in the online phase, solely additions and multiplications over a ring.

All the published results for privacy-preserving machine learning classification are secure in the honest-but-curious model.

How much information is leaked about the decision trees in [BPTG15, BPTG14] and in [WFNL15]: In the protocol in [BPTG15, BPTG14], theoretically nothing is ever leaked about the tree. However, if an adversary can measure the time it takes for Bob to do the evaluation of the decision tree protocol, clearly the deeper the tree the longer the computation becomes. Therefore, some information about the depth of the tree is leaked if this side channel attack is considered. Therefore, in our solution we do not loose much by giving away the depth of the tree to an adversary. In [WFNL15], the depth of the tree is also leaked.

Bit-Decomposition Protocols: The best solution for bit-decomposition, in terms of round complexity, is a constant-round solution by Toft [Tof09a], which has round complexity equal to 23. Veugen noted in [Veu15] that for a certain range of practical parameters (number of input bits less than 20), a protocol with a linear number of rounds in the length of the input could outperform the solution presented by Toft [Tof09a]. Veugen proposed a protocol that has a linear number of rounds in ℓ, where ℓ is the length of the input in bits. Veugen also proposed a way to reduce the number of rounds of this protocol by a factor of β, obtaining a round complexity equal to ℓ/β at the cost of performing an exponential (in β) number of multiplications in a pre-processing phase.

The bit-decomposition protocol used in this work is over binary fields and runs in $2+\lceil\log\ell\rceil$ rounds. For practical values of ℓ (less than 100 typically), it is always better than Toft's and Veugen's solutions. The number of multiplications to be performed in our the online phase, $2\ell\lceil\log\ell\rceil + 3\ell$, is less than the $31\ell\lceil\log\ell\rceil + 71\ell + 30\lceil\sqrt{\ell}\rceil$ multiplications in the case of Toft's protocol. While Veugen's protocol can have a fast online phase, requiring only $3\ell - 2\beta$ multiplications for ℓ/β rounds, it requires an exponential (in β) number of multiplications in the offline phase.

The protocol of Schoenmakers and Tuyls [ST06] has the same number of rounds and roughly half as many multiplications as the protocol used here. However the multiplication are in $\mathbb{Z}_q$ for big q while our multiplications are in $\mathbb{Z}_2$. Hence our multiplications are faster and communicate less data. In addition, in our case OT extension can be directly used for the pre-computation if a trusted initializer is not available. For more details about Schoenmakers and Tuyls' protocol see the original paper or Section 4.3.5 of De Hoogh's PhD thesis [dH12].

A restriction of our protocol is that it only works for operations modulo a power of 2. As we need no modular inversions in our privacy-preserving machine learning protocols this imposes no problem at all. The bit-decomposition protocol of Laud and Randmets [LR15] for the case of three parties with at most one corruption is similar to one here. It first reduce the original problem to a new one between two-parties, and then uses the adder idea to obtain bitwise shares. Although the protocol is not fully specified in [LR15], we believe that the authors intended to use the same adder computation as here.

7.7 Discussion

This chapter presented a protocol for privacy-preserving classification of decision trees, and improvements to the performance of previously proposed protocols for general hyperplane-based and Naive Bayes classifiers. Our protocols work in the commodity-based model. The pre-distributed data can be distributed during a setup phase by a trusted initializer to the parties. In the case a trusted initializer is not available or desirable, the parties can pre-compute this data by themselves, during a setup phase, with the help of well-known computationally secure schemes.

Our solutions are very efficient and use solely modular addition and multiplications. We present accuracy and runtime results for 7 classification benchmark datasets from the UCI repository.

8. Conclusion

This thesis exposed some results about the uses of correlated data in cryptography from theoretical and practical points of view.

On the theoretical side, there are still many open questions regarding the OT and commitment capacities of many other noisy resources. For instance, determining the OT capacity of unfair noisy channels, or the capacities of elastic channels. Additionally, there are still open problems regarding the optimal way of using the underlying resourses in order to obtain other cryptographic primitives.

From a practical perspective, correlated data can possibly be used to obtain very efficient protocols for other machine learning problems as well as problems in other fields. This is an interesting direction for future works. One additional inviting direction for further investigation is determining which other forms of correlated data can be used to achieve even more efficient protocols.

Overall, we believe that cryptography based on correlated data is a promising area that deserves further research efforts.

Bibliography

[AB06] Shai Avidan and Moshe Butman. Blind vision. In Aleš Leonardis, Horst Bischof, and Axel Pinz, editors, *Computer Vision – ECCV 2006: 9th European Conference on Computer Vision, Proceedings, Part III*, pages 1–13, Graz, Austria, May 7–13, 2006. Springer Berlin Heidelberg.

[AB07] Shai Avidan and Moshe Butman. Efficient methods for privacy preserving face detection. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 57–64. MIT Press, 2007.

[AC93] Rudolph Ahlswede and Imre Csiszár. Common randomness in information theory and cryptography. i. secret sharing. *Information Theory, IEEE Transactions on*, 39(4):1121–1132, July 1993.

[AC07] Rudolph Ahlswede and Imre Csiszár. On oblivious transfer capacity. In *Information Theory, 2007. ISIT 2007. IEEE International Symposium on*, pages 2061–2064, Nice, France, June 24–29, 2007.

[AC13] Rudolf Ahlswede and Imre Csiszár. On oblivious transfer capacity. In Harout Aydinian, Ferdinando Cicalese, and Christian Deppe, editors, *Information Theory, Combinatorics, and Search Theory*, volume 7777 of *Lecture Notes in Computer Science*, pages 145–166. Springer Berlin Heidelberg, 2013.

[AFJK09] Ehsan Ardestanizadeh, Massimo Franceschetti, Tara Javidi, and Young-Han Kim. Wiretap channel with secure rate-limited feedback. *Information Theory, IEEE Transactions on*, 55(12):5353–5361, December 2009.

[AHMR15] Arash Afshar, Zhangxiang Hu, Payman Mohassel, and Mike Rosulek. How to efficiently evaluate RAM programs with malicious security. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 702–729, Sofia, Bulgaria, April 26–30, 2015. Springer, Heidelberg, Germany.

[ALSZ13] Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. More efficient oblivious transfer and extensions for faster secure computation. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 13: 20th Conference on Computer* *and Communications Security*, pages 535–548, Berlin, Germany, November 4–8, 2013. ACM Press.

[ALSZ15] Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. More efficient oblivious transfer extensions with security for malicious adversaries. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 673–701, Sofia, Bulgaria, April 26–30, 2015. Springer, Heidelberg, Germany.

[Alv10] Vinícius M. Alves. Protocolo de comprometimento de bit eficiente com segurança sequencial baseado no modelo de memória limitada. Master's thesis, Universidade de Brasília, 2010.

[AS04] Noga Alon and Joel H Spencer. *The probabilistic method.* John Wiley & Sons, 2004.

[BBCM95] Charles H. Bennett, Gilles Brassard, Claude Crépeau, and Ueli M. Maurer. Generalized privacy amplification. *Information Theory, IEEE Transactions on*, 41(6):1915–1923, November 1995.

[BBR88] Charles H. Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy amplification by public discussion. *SIAM J. Comput.*, 17(2):210–229, April 1988.

[BCC88] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *J. Comput. Syst. Sci.*, 37(2):156–189, October 1988.

[BCNP04] Boaz Barak, Ran Canetti, Jesper Buus Nielsen, and Rafael Pass. Universally composable protocols with relaxed set-up assumptions. In *45th Annual Symposium on Foundations of Computer Science*, pages 186–195, Rome, Italy, October 17–19, 2004. IEEE Computer Society Press.

[BDOZ11] Rikke Bendlin, Ivan Damgård, Claudio Orlandi, and Sarah Zakarias. Semi-homomorphic encryption and multiparty computation. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 169–188, Tallinn, Estonia, May 15–19, 2011. Springer, Heidelberg, Germany.

[Bea90] Donald Beaver. Multiparty protocols tolerating half faulty processors. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 560–572, Santa Barbara, CA, USA, August 20–24, 1990. Springer, Heidelberg, Germany.

[Bea92] Donald Beaver. Efficient multiparty protocols using circuit randomization. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 420–432, Santa Barbara, CA, USA, August 11–15, 1992. Springer, Heidelberg, Germany.

[Bea95] Donald Beaver. Precomputing oblivious transfer. In Don Coppersmith, editor, *Advances in Cryptology – CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 97–109, Santa Barbara, CA, USA, August 27–31, 1995. Springer, Heidelberg, Germany.

[Bea96] Donald Beaver. Correlated pseudorandomness and the complexity of private computations. In *28th Annual ACM Symposium on Theory of Computing*, pages 479–488, Philadephia, PA, USA, May 22–24, 1996. ACM Press.

[Bea97] Donald Beaver. Commodity-based cryptography (extended abstract). In *29th Annual ACM Symposium on Theory of Computing*, pages 446–455, El Paso, TX, USA, May 4–6, 1997. ACM Press.

[Bea98a] Donald Beaver. One-time tables for two-party computation. In *Computing and Combinatorics*, pages 361–370. Springer, 1998.

[Bea98b] Donald Beaver. Server-assisted cryptography. In *Proceedings of the 1998 workshop on New security paradigms*, NSPW '98, pages 92–106, Charlottesville, Virginia, USA, 1998. ACM, New York, NY, USA.

[BFK$^+$09] Mauro Barni, Pierluigi Failla, Vladimir Kolesnikov, Riccardo Lazzeretti, Ahmad-Reza Sadeghi, and Thomas Schneider. Secure evaluation of private linear branching programs with medical applications. In Michael Backes and Peng Ning, editors, *ESORICS 2009: 14th European Symposium on Research in Computer Security*, volume 5789 of *Lecture Notes in Computer Science*, pages 424–439, Saint-Malo, France, September 21–23, 2009. Springer, Heidelberg, Germany.

[BFL$^+$09] M. Barni, P. Failla, R. Lazzeretti, A. Paus, A. R. Sadeghi, T. Schneider, and V. Kolesnikov. Efficient privacy-preserving classification of ecg signals. In *Information Forensics and Security (WIFS 2009) First IEEE International Workshop on*, pages 91–95, London, United Kingdom, December 6–9, 2009.

[BFL$^+$11] M. Barni, P. Failla, R. Lazzeretti, A. R. Sadeghi, and T. Schneider. Privacy-preserving ecg classification with branching programs and neural networks. *IEEE Transactions on Information Forensics and Security*, 6(2):452–468, June 2011.

[BFOS84] Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth Publishing Company, 1984.

[BH05] László Babai and Thomas P. Hayes. Near-independence of permutations and an almost sure polynomial bound on the diameter of the symmetric group. In *16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1057–1066, Vancouver, BC, Canada, January 23–25, 2005. ACM-SIAM.

[BLN13] Joppe W. Bos, Kristin Lauter, and Michael Naehrig. Private predictive analysis on encrypted medical data. Technical Report MSR-TR-2013-81, Microsoft Research, September 2013.

[Blu83] Manuel Blum. Coin flipping by telephone a protocol for solving impossible problems. *SIGACT News*, 15(1):23–27, January 1983.

[BM90] Mihir Bellare and Silvio Micali. Non-interactive oblivious transfer and spplications. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 547–557, Santa Barbara, CA, USA, August 20–24, 1990. Springer, Heidelberg, Germany.

[BMK09] Ghadamali Bagherikaram, Abolfazl S. Motahari, and Amir K. Khandani. Secrecy capacity region of gaussian broadcast channel. In *Information Sciences and Systems, 2009. CISS 2009. 43rd Annual Conference on*, pages 152–157, March 2009.

[BMSW02] C. Blundo, B. Masucci, D. R. Stinson, and R. Wei. Constructions and bounds for unconditionally secure non-interactive commitment schemes. *Des. Codes Cryptography*, 26(1-3):97–110, June 2002.

[BPTG14] Raphael Bost, Raluca Ada Popa, Stephen Tu, and Shafi Goldwasser. Machine learning classification over encrypted data. Cryptology ePrint Archive, Report 2014/331, 2014. `http://eprint.iacr.org/2014/331`.

[BPTG15] Raphael Bost, Raluca Ada Popa, Stephen Tu, and Shafi Goldwasser. Machine learning classification over encrypted data. In *ISOC Network and Distributed System Security Symposium – NDSS 2015*, San Diego, CA, USA, February 8–11, 2015. The Internet Society.

[BR94] Mihir Bellare and John Rompel. Randomness-efficient oblivious sampling. In *35th Annual Symposium on Foundations of Computer Science*, pages 276–287, Santa Fe, New Mexico, November 20–22, 1994. IEEE Computer Society Press.

[BR06] João Barros and Miguel R. D. Rodrigues. Secrecy capacity of wireless channels. In *Information Theory, 2006 IEEE International Symposium on*, pages 356–360, Seattle, WA, USA, July 9–14, 2006.

[BSMD10] Martin Burkhart, Mario Strasser, Dilip Many, and Xenofontas Dimitropoulos. Sepia: privacy-preserving aggregation of multi-domain network events and statistics. In *Proceedings of the 19th USENIX conference on Security*, USENIX Security'10, pages 15–15, Washington, DC, 2010. USENIX Association, Berkeley, CA, USA.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd Annual Symposium on Foundations of Computer Science*, pages 136–145, Las Vegas, NV, USA, October 14–17, 2001. IEEE Computer Society Press.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *20th Annual ACM Symposium on Theory of Computing*, pages 11–19, Chicago, IL, USA, May 2–4, 1988. ACM Press.

[CCM98] Christian Cachin, Claude Crépeau, and Julien Marcil. Oblivious transfer with a memory-bounded receiver. In *39th Annual Symposium on Foundations of Computer Science*, pages 493–502, Palo Alto, CA, USA, November 8–11, 1998. IEEE Computer Society Press.

[CDD$^+$99] Ronald Cramer, Ivan Damgård, Stefan Dziembowski, Martin Hirt, and Tal Rabin. Efficient multiparty computations secure against an adaptive adversary. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 311–326, Prague, Czech Republic, May 2–6, 1999. Springer, Heidelberg, Germany.

[CDH$^+$16] Martine De Cock, Rafael Dowsley, Caleb Horst, Raj Katti, Anderson C. A. Nascimento, Stacey C. Newman, and Wing-Sea Poon. Efficient and private scoring of decision trees, support vector machines and logistic regression models based on pre-computation. Manuscript, 2016.

[CDLR16] Ignacio Cascudo, Ivan Damgård, Felipe Lacerda, and Samuel Ranellucci. Oblivious transfer from any non-trivial elastic noisy channels via secret key agreement. Cryptology ePrint Archive, Report 2016/120. To appear on TCC 2016b, 2016. `http://eprint.iacr.org/2016/120`.

[CDN15] Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.

[CDN16] Claude Crépeau, Rafael Dowsley, and Anderson C. A. Nascimento. On the commitment capacity of unfair noisy channels. Manuscript, 2016.

[CDNN15] Martine de Cock, Rafael Dowsley, Anderson C.A. Nascimento, and Stacey C. Newman. Fast, privacy preserving linear regression over distributed datasets based on pre-distributed data. In *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*, AISec '15, pages 3–14, Denver, Colorado, USA, 2015. ACM, New York, NY, USA.

[CDv88] David Chaum, Ivan Damgård, and Jeroen van de Graaf. Multiparty computations ensuring privacy of each party's input and correctness of the result. In Carl Pomerance, editor, *Advances in Cryptology – CRYPTO'87*, volume 293 of *Lecture Notes in Computer Science*, pages 87–119, Santa Barbara, CA, USA, August 16–20, 1988. Springer, Heidelberg, Germany.

[CEG95] Ran Canetti, Guy Even, and Oded Goldreich. Lower bounds for sampling algorithms for estimating the average. *Information Processing Letters*, 53(1):17–25, January 13 1995.

[CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 19–40, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany.

[Che52] Herman Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, 23(4):493–507, December 1952.

[CK78] Imre Csiszár and János Körner. Broadcast channels with confidential messages. *Information Theory, IEEE Transactions on*, 24(3):339–348, May 1978.

[CK82] Imre Csiszár and János Körner. *Information Theory: Coding Theorems for Discrete Memoryless Systems.* Academic Press, Inc., Orlando, FL, USA, 1982.

[CK88] Claude Crépeau and Joe Kilian. Achieving oblivious transfer using weakened security assumptions (extended abstract). In *29th Annual Symposium on Foundations of Computer Science*, pages 42–52, White Plains, New York, October 24–26, 1988. IEEE Computer Society Press.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th Annual ACM Symposium on Theory of Computing*, pages 494–503, Montréal, Québec, Canada, May 19–21, 2002. ACM Press.

[CM97] Christian Cachin and Ueli M. Maurer. Unconditional security against memory-bounded adversaries. In Burton S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 292–306, Santa Barbara, CA, USA, August 17–21, 1997. Springer, Heidelberg, Germany.

[CMW05] Claude Crépeau, Kirill Morozov, and Stefan Wolf. Efficient unconditional oblivious transfer from almost any noisy channel. In Carlo Blundo and Stelvio Cimato, editors, *SCN 04: 4th International Conference on Security in Communication Networks*, volume 3352 of *Lecture Notes in Computer Science*, pages 47–59, Amalfi, Italy, September 8–10, 2005. Springer, Heidelberg, Germany.

[CN04] Imre Csiszár and Prakash Narayan. Secrecy capacities for multiple terminals. *Information Theory, IEEE Transactions on*, 50(12):3047–3061, December 2004.

[CN08] Imre Csiszár and Prakash Narayan. Secrecy capacities for multiterminal channel models. *Information Theory, IEEE Transactions on*, 54(6):2437–2452, June 2008.

[Cov73] Thomas M. Cover. Enumerative source encoding. *Information Theory, IEEE Transactions on*, 19(1):73–77, January 1973.

[CR03] Ran Canetti and Tal Rabin. Universal composition with joint state. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 265–281, Santa Barbara, CA, USA, August 17–21, 2003. Springer, Heidelberg, Germany.

[Cré97] Claude Crépeau. Efficient cryptographic protocols based on noisy channels. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 306–317, Konstanz, Germany, May 11–15, 1997. Springer, Heidelberg, Germany.

[CS06] Claude Crépeau and George Savvides. Optimal reductions between oblivious transfers using interactive hashing. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 201–221, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Heidelberg, Germany.

[CS10] Octavian Catrina and Amitabh Saxena. Secure computation with fixed-point numbers. In Radu Sion, editor, *FC 2010: 14th International Conference on Financial Cryptography and Data Security*, volume 6052 of *Lecture Notes in Computer Science*, pages 35–50, Tenerife, Canary Islands, Spain, January 25–28, 2010. Springer, Heidelberg, Germany.

[CvT95] Claude Crépeau, Jeroen van de Graaf, and Alain Tapp. Committed oblivious transfer and private multi-party computation. In Don Coppersmith, editor, *Advances in Cryptology – CRYPTO'95*, volume 963 of *Lecture Notes in Computer Science*, pages 110–123, Santa Barbara, CA, USA, August 27–31, 1995. Springer, Heidelberg, Germany.

[CW79] J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143 – 154, 1979.

[DBK$^+$96] Harris Drucker, Christopher J. C. Burges, Linda Kaufman, Alexander J. Smola, and Vladimir Vapnik. Support vector regression machines. In *Advances in Neural Information Processing Systems 9, Proceedings of the 1996 NIPS conference*, pages 155–161, 1996.

[DDKN15] Bernardo Machado David, Rafael Dowsley, Raj Katti, and Anderson C. A. Nascimento. Efficient unconditionally secure comparison and privacy preserving machine learning classification protocols. In

Man Ho Au and Atsuko Miyaji, editors, *ProvSec 2015: 9th International Conference on Provable Security*, volume 9451 of *Lecture Notes in Computer Science*, pages 354–367, Kanazawa, Japan, November 24–26, 2015. Springer, Heidelberg, Germany.

[DDN14] Bernardo David, Rafael Dowsley, and Anderson C. A. Nascimento. Universally composable oblivious transfer based on a variant of LPN. In Dimitris Gritzalis, Aggelos Kiayias, and Ioannis G. Askoxylakis, editors, *CANS 14: 13th International Conference on Cryptology and Network Security*, volume 8813 of *Lecture Notes in Computer Science*, pages 143–158, Heraklion, Crete, Greece, October 22–24, 2014. Springer, Heidelberg, Germany.

[DDvdG$^+$16] Bernardo David, Rafael Dowsley, Jeroen van de Graaf, Davidson Marques, Anderson C. A. Nascimento, and Adriana C. B. Pinto. Unconditionally secure, universally composable privacy preserving linear algebra. *Information Forensics and Security, IEEE Transactions on*, 11(1):59–73, January 2016.

[DFK$^+$06] Ivan Damgård, Matthias Fitzi, Eike Kiltz, Jesper Buus Nielsen, and Tomas Toft. Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation. In Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, volume 3876 of *Lecture Notes in Computer Science*, pages 285–304, New York, NY, USA, March 4–7, 2006. Springer, Heidelberg, Germany.

[DGMN11] Rafael Dowsley, Jeroen van de Graaf, Davidson Marques, and Anderson C. A. Nascimento. A two-party protocol with trusted initializer for computing the inner product. In Yongwha Chung and Moti Yung, editors, *WISA 10: 11th International Workshop on Information Security Applications*, volume 6513 of *Lecture Notes in Computer Science*, pages 337–350, Jeju Island, Korea, August 24–26, 2011. Springer, Heidelberg, Germany.

[dH12] Sebastiaan Jacobus Antonius de Hoogh. *Design of Large Scale Applications of Secure Multiparty Computation: Secure Linear Programming*. PhD thesis, Technische Universiteit Eindhoven, 2012.

[DHC04] Wenliang Du, Yunghsiang S. Han, and Shigang Chen. Privacy-preserving multivariate statistical analysis: Linear regression and classification. In *In Proceedings of the 4th SIAM International Conference on Data Mining*, pages 222–233, 2004.

[DHRS04] Yan Zong Ding, Danny Harnik, Alon Rosen, and Ronen Shaltiel. Constant-round oblivious transfer in the bounded storage model. In Moni Naor, editor, *TCC 2004: 1st Theory of Cryptography Conference*, volume 2951 of *Lecture Notes in Computer Science*, pages 446–472, Cambridge, MA, USA, February 19–21, 2004. Springer, Heidelberg, Germany.

[dHSCodA14] Sebastiaan de Hoogh, Berry Schoenmakers, Ping Chen, and Harm op den Akker. Practical secure decision tree learning in a teletreatment application. In Nicolas Christin and Reihaneh Safavi-Naini, editors, *FC 2014: 18th International Conference on Financial Cryptography and Data Security*, volume 8437 of *Lecture Notes in Computer Science*, pages 179–194, Christ Church, Barbados, March 3–7, 2014. Springer, Heidelberg, Germany.

[Din01] Yan Zong Ding. Oblivious transfer in the bounded storage model. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 155–170, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Heidelberg, Germany.

[Din05] Yan Zong Ding. Error correction in the bounded storage model. In Joe Kilian, editor, *TCC 2005: 2nd Theory of Cryptography Conference*, volume 3378 of *Lecture Notes in Computer Science*, pages 578–599, Cambridge, MA, USA, February 10–12, 2005. Springer, Heidelberg, Germany.

[DKL^{+}13] Ivan Damgård, Marcel Keller, Enrique Larraia, Valerio Pastro, Peter Scholl, and Nigel P. Smart. Practical covertly secure MPC for dishonest majority - or: Breaking the SPDZ limits. In Jason Crampton, Sushil Jajodia, and Keith Mayes, editors, *ESORICS 2013: 18th European Symposium on Research in Computer Security*, volume 8134 of *Lecture Notes in Computer Science*, pages 1–18, Egham, UK, September 9–13, 2013. Springer, Heidelberg, Germany.

[DKMQ11] Nico Döttling, Daniel Kraschewski, and Jörn Müller-Quade. Unconditional and composable security using a single stateful tamper-proof hardware token. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 164–181, Providence, RI, USA, March 28–30, 2011. Springer, Heidelberg, Germany.

[DKS99] Ivan Damgård, Joe Kilian, and Louis Salvail. On the (im)possibility of basing oblivious transfer and bit commitment on weakened security assumptions. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 56–73, Prague, Czech Republic, May 2–6, 1999. Springer, Heidelberg, Germany.

[DLN14] Rafael Dowsley, Felipe Lacerda, and Anderson C. A. Nascimento. Oblivious transfer in the bounded storage model with errors. In *Information Theory (ISIT), 2014 IEEE International Symposium on*, pages 1623–1627, Honolulu, HI, USA, June 29 – July 4, 2014.

[DLN15] Rafael Dowsley, Felipe Lacerda, and Anderson C. A. Nascimento. Commitment and oblivious transfer in the bounded storage model with errors. Cryptology ePrint Archive, Report 2015/952, 2015. `http://eprint.iacr.org/`.

[DM99] Yevgeniy Dodis and Silvio Micali. Lower bounds for oblivious transfer reductions. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 42–55, Prague, Czech Republic, May 2–6, 1999. Springer, Heidelberg, Germany.

[DM08] Stefan Dziembowski and Ueli M. Maurer. The bare bounded-storage model: The tight bound on the storage requirement for key agreement. *IEEE Transactions on Information Theory*, 54(6):2790–2792, 2008.

[DMQN08] Rafael Dowsley, Jörn Müller-Quade, and Anderson C. A. Nascimento. On the possibility of universally composable commitments based on noisy channels. In André Luiz Moura dos Santos and Marinho Pilla Barcellos, editors, *Anais do VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais, SBSEG 2008*, pages 103–114, Gramado, Brazil, September 1–5, 2008. Sociedade Brasileira de Computação (SBC).

[DMQN15] Rafael Dowsley, Jörn Müller-Quade, and Tobias Nilges. Weakening the isolation assumption of tamper-proof hardware tokens. In Anja Lehmann and Stefan Wolf, editors, *ICITS 15: 8th International Conference on Information Theoretic Security*, volume 9063 of *Lecture Notes in Computer Science*, pages 197–213, Lugano, Switzerland, May 2–5, 2015. Springer, Heidelberg, Germany.

[DMQO$^+$11] Rafael Dowsley, Jörn Müller-Quade, Akira Otsuka, Goichiro Hanaoka, Hideki Imai, and Anderson C. A. Nascimento. Universally composable and statistically secure verifiable secret sharing scheme based on pre-distributed data. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E94-A(2):725–734, 2011.

[DN14] Rafael Dowsley and Anderson C. A. Nascimento. On the oblivious transfer capacity of generalized erasure channels against malicious adversaries. *CoRR*, abs/1410.2862, 2014.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, March 2008.

[DPP98] Ivan B. Damgård, Torben P. Pedersen, and Birgit Pfitzmann. Statistical secrecy and multibit commitments. *Information Theory, IEEE Transactions on*, 44(3):1143–1151, May 1998.

[DPSZ11] I. Damgard, V. Pastro, N.P. Smart, and S. Zakarias. Multiparty computation from somewhat homomorphic encryption. Cryptology ePrint Archive, Report 2011/535, 2011. `http://eprint.iacr.org/2011/535`.

[DPSZ12] Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption.

In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 643–662, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany.

[DRS04] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 523–540, Interlaken, Switzerland, May 2–6, 2004. Springer, Heidelberg, Germany.

[DSZ15] Daniel Demmler, Thomas Schneider, and Michael Zohner. ABY - A framework for efficient mixed-protocol secure two-party computation. In *ISOC Network and Distributed System Security Symposium – NDSS 2015*, San Diego, CA, USA, February 8–11, 2015. The Internet Society.

[DvdGMN08] Rafael Dowsley, Jeroen van de Graaf, Jörn Müller-Quade, and Anderson C. A. Nascimento. Oblivious transfer based on the McEliece assumptions. In Reihaneh Safavi-Naini, editor, *ICITS 08: 3rd International Conference on Information Theoretic Security*, volume 5155 of *Lecture Notes in Computer Science*, pages 107–117, Calgary, Canada, August 10–13, 2008. Springer, Heidelberg, Germany.

[DvdGMQN12] Rafael Dowsley, Jeroen van de Graaf, Jörn Müller-Quade, and Anderson C. A. Nascimento. Oblivious transfer based on the McEliece assumptions. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E95-A(2):567–575, 2012.

[DvdGMQN13] Rafael Dowsley, Jeroen van de Graaf, Jörn Müller-Quade, and Anderson C. A. Nascimento. On the composability of statistically secure bit commitments. *Journal of Internet Technology*, 14(3):509–516, 2013.

[EFG$^+$09] Zekeriya Erkin, Martin Franz, Jorge Guajardo, Stefan Katzenbeisser, Inald Lagendijk, and Tomas Toft. Privacy-preserving face recognition. In Ian Goldberg and Mikhail J. Atallah, editors, *Privacy Enhancing Technologies: 9th International Symposium, PETS 2009, Proceedings*, pages 235–253, Seattle, WA, USA, August 5–7 2009. Springer Berlin Heidelberg.

[EGL85] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, June 1985.

[EU11] Ersen Ekrem and Sennur Ulukus. The secrecy capacity region of the gaussian mimo multi-receiver wiretap channel. *Information Theory, IEEE Transactions on*, 57(4):2083–2114, April 2011.

[Eve81] Shimon Even. Protocol for signing contracts. In Allen Gersho, editor, *Advances in Cryptology – CRYPTO'81*, volume ECE Report

82-04, pages 148–153, Santa Barbara, CA, USA, 1981. U.C. Santa Barbara, Dept. of Elec. and Computer Eng.

[FGMv02] Matthias Fitzi, Nicolas Gisin, Ueli M. Maurer, and Oliver von Rotz. Unconditional byzantine agreement and multi-party computation secure against dishonest minorities from scratch. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 482–501, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Heidelberg, Germany.

[FIM$^+$01] Joan Feigenbaum, Yuval Ishai, Tal Malkin, Kobbi Nissim, Martin Strauss, and Rebecca N. Wright. Secure multiparty computation of approximations. In *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, pages 927–938, 2001.

[FIM$^+$06] Joan Feigenbaum, Yuval Ishai, Tal Malkin, Kobbi Nissim, Martin J. Strauss, and Rebecca N. Wright. Secure multiparty computation of approximations. *ACM Transactions on Algorithms*, 2(3):435–472, 2006.

[FJN$^+$13] Tore Kasper Frederiksen, Thomas Pelle Jakobsen, Jesper Buus Nielsen, Peter Sebastian Nordholt, and Claudio Orlandi. MiniLEGO: Efficient secure two-party computation from general assumptions. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 537–556, Athens, Greece, May 26–30, 2013. Springer, Heidelberg, Germany.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology – CRYPTO'86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194, Santa Barbara, CA, USA, August 1987. Springer, Heidelberg, Germany.

[FWW04] Matthias Fitzi, Stefan Wolf, and Jürg Wullschleger. Pseudo-signatures, broadcast, and multi-party computation from correlated randomness. In Matthew Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 562–578, Santa Barbara, CA, USA, August 15–19, 2004. Springer, Heidelberg, Germany.

[GH06] Chun-Hua Guo and Nicholas J. Higham. A schur-newton method for the matrix p'th root and its inverse. *SIAM Journal On Matrix Analysis and Applications*, 28(3):788–804, October 2006.

[GI02] Venkatesan Guruswami and Piotr Indyk. Near-optimal linear-time codes for unique decoding and new list-decodable codes over smaller alphabets. In *34th Annual ACM Symposium on Theory of Computing*, pages 812–821, Montréal, Québec, Canada, May 19–21, 2002. ACM Press.

[GLEG08] Praveen Kumar Gopala, Lifeng Lai, and Hesham El Gamal. On the secrecy capacity of fading channels. *Information Theory, IEEE Transactions on*, 54(10):4687–4698, October 2008.

[GLLM05] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. On private scalar product computation for privacy-preserving data mining. In Choonsik Park and Seongtaek Chee, editors, *ICISC 04: 7th International Conference on Information Security and Cryptology*, volume 3506 of *Lecture Notes in Computer Science*, pages 104–120, Seoul, Korea, December 2–3, 2005. Springer, Heidelberg, Germany.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th Annual ACM Symposium on Theory of Computing*, pages 218–229, New York City, NY, USA, May 25–27, 1987. ACM Press.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(3):691–729, 1991.

[Gol01] Oded Goldreich. *Foundations of Cryptography: Basic Tools*, volume 1. Cambridge University Press, Cambridge, UK, 2001.

[Gol04] Oded Goldreich. *Foundations of Cryptography: Basic Applications*, volume 2. Cambridge University Press, Cambridge, UK, 2004.

[GSV07] Juan A. Garay, Berry Schoenmakers, and José Villegas. Practical and secure solutions for integer comparison. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *PKC 2007: 10th International Conference on Theory and Practice of Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 330–342, Beijing, China, April 16–20, 2007. Springer, Heidelberg, Germany.

[Hai04] Iftach Haitner. Implementing oblivious transfer using collection of dense trapdoor permutations. In Moni Naor, editor, *TCC 2004: 1st Theory of Cryptography Conference*, volume 2951 of *Lecture Notes in Computer Science*, pages 394–409, Cambridge, MA, USA, February 19–21, 2004. Springer, Heidelberg, Germany.

[HCR02] Dowon Hong, Ku-Young Chang, and Heuisu Ryu. Efficient oblivious transfer in the bounded-storage model. In Yuliang Zheng, editor, *Advances in Cryptology – ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 143–159, Queenstown, New Zealand, December 1–5, 2002. Springer, Heidelberg, Germany.

[HFN11] Rob Hall, Stephen E. Fienberg, and Yuval Nardi. Secure multiple linear regression based on homomorphic encryption. *Journal of Official Statistics*, 27(4):669–691, 2011.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[HMQ04] Dennis Hofheinz and Jörn Müller-Quade. Universally composable commitments using random oracles. In Moni Naor, editor, *TCC 2004: 1st Theory of Cryptography Conference*, volume 2951 of *Lecture Notes in Computer Science*, pages 58–76, Cambridge, MA, USA, February 19–21, 2004. Springer, Heidelberg, Germany.

[HMQU05] Dennis Hofheinz, Jörn Müller-Quade, and Dominique Unruh. Universally composable zero-knowledge arguments and commitments from signature cards. In *In Proceedings of the 5th Central European Conference on Cryptology MoraviaCrypt 2005*, 2005.

[HR05] Thomas Holenstein and Renato Renner. One-way secret-key agreement and applications to circuit polarization and immunization of public-key encryption. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 478–493, Santa Barbara, CA, USA, August 14–18, 2005. Springer, Heidelberg, Germany.

[IKM$^+$13] Yuval Ishai, Eyal Kushilevitz, Sigurd Meldgaard, Claudio Orlandi, and Anat Paskin-Cherniavsky. On the power of correlated randomness in secure computation. In Amit Sahai, editor, *TCC 2013: 10th Theory of Cryptography Conference*, volume 7785 of *Lecture Notes in Computer Science*, pages 600–620, Tokyo, Japan, March 3–6, 2013. Springer, Heidelberg, Germany.

[IKNP03] Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 145–161, Santa Barbara, CA, USA, August 17–21, 2003. Springer, Heidelberg, Germany.

[ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudorandom generation from one-way functions (extended abstracts). In *21st Annual ACM Symposium on Theory of Computing*, pages 12–24, Seattle, WA, USA, May 15–17, 1989. ACM Press.

[IMN06] Hideki Imai, Kirill Morozov, and Anderson C. A. Nascimento. On the oblivious transfer capacity of the erasure channel. In *Information Theory, 2006 IEEE International Symposium on*, pages 1428–1431, Seattle, WA, USA, July 9–14, 2006.

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 572–591, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Heidelberg, Germany.

[Kal05] Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 78–95, Aarhus, Denmark, May 22–26, 2005. Springer, Heidelberg, Germany.

[Kat07] Jonathan Katz. Universally composable multi-party computation using tamper-proof hardware. In Moni Naor, editor, *Advances in Cryptology – EUROCRYPT 2007*, volume 4515 of *Lecture Notes in Computer Science*, pages 115–128, Barcelona, Spain, May 20–24, 2007. Springer, Heidelberg, Germany.

[Kil88] Joe Kilian. Founding cryptography on oblivious transfer. In *20th Annual ACM Symposium on Theory of Computing*, pages 20–31, Chicago, IL, USA, May 2–4, 1988. ACM Press.

[KK13] Vladimir Kolesnikov and Ranjit Kumaresan. Improved OT extension for transferring short secrets. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 54–70, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[KLML05] Eike Kiltz, Gregor Leander, and John Malone-Lee. Secure computation of the mean and related statistics. In Joe Kilian, editor, *TCC 2005: 2nd Theory of Cryptography Conference*, volume 3378 of *Lecture Notes in Computer Science*, pages 283–302, Cambridge, MA, USA, February 10–12, 2005. Springer, Heidelberg, Germany.

[KLSR05] Alan F Karr, Xiaodong Lin, Ashish P Sanil, and Jerome P Reiter. Secure regression on distributed databases. *Journal of Computational and Graphical Statistics*, 14(2):263–279, 2005.

[KLSR09] Alan F Karr, Xiaodong Lin, Ashish P Sanil, and Jerome P Reiter. Privacy-preserving analysis of vertically partitioned data using secure matrix products. *Journal of Official Statistics*, 25(1):125–138, 2009.

[KM01] Valeri Korjik and Kirill Morozov. Generalized oblivious transfer protocols based on noisy channels. In *Proceedings of the International Workshop on Information Assurance in Computer Networks: Methods, Models, and Architectures for Network Security*, MMM-ACNS '01, pages 219–229, St. Petersburg, Russia, May 21–23, 2001. Springer Berlin Heidelberg.

[KMS16] Dakshita Khurana, Hemanta K. Maji, and Amit Sahai. Secure computation from elastic noisy channels. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 184–212, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.

[KR09] Bhavana Kanukurthi and Leonid Reyzin. Key agreement from close secrets over unsecured channels. In Antoine Joux, editor, *Advances in Cryptology – EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 206–223, Cologne, Germany, April 26–30, 2009. Springer, Heidelberg, Germany.

[KSS13] Marcel Keller, Peter Scholl, and Nigel P. Smart. An architecture for practical actively secure MPC with dishonest majority. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 13: 20th Conference on Computer and Communications Security*, pages 549–560, Berlin, Germany, November 4–8, 2013. ACM Press.

[Lin13] Yehuda Lindell. Fast cut-and-choose based protocols for malicious and covert adversaries. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 1–17, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Heidelberg, Germany.

[LPS07] Yingbin Liang, H. Vincent Poor, and Shlomo Shamai. Secrecy capacity region of fading broadcast channels. In *Information Theory, 2007. ISIT 2007. IEEE International Symposium on*, pages 1291–1295, Nice, France, June 24–29, 2007.

[LR14] Yehuda Lindell and Ben Riva. Cut-and-choose Yao-based secure computation in the online/offline and batch settings. In Juan A. Garay and Rosario Gennaro, editors, *Advances in Cryptology – CRYPTO 2014, Part II*, volume 8617 of *Lecture Notes in Computer Science*, pages 476–494, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany.

[LR15] Peeter Laud and Jaak Randmets. A domain-specific language for low-level secure multiparty computation protocols. In Indrajit Ray, Ninghui Li, and Christopher Kruegel:, editors, *ACM CCS 15: 22nd Conference on Computer and Communications Security*, pages 1492–1503, Denver, CO, USA, October 12–16, 2015. ACM Press.

[LYCH78] S. Leung-Yan-Cheong and Martin E. Hellman. The gaussian wire-tap channel. *Information Theory, IEEE Transactions on*, 24(4):451–456, July 1978.

[LYT10] Zang Li, Roy Yates, and Wade Trappe. Secrecy capacity of independent parallel channels. In Ruoheng Liu and Wade Trappe, editors, *Securing Wireless Communications at the Physical Layer*, pages 1–18. Springer US, 2010.

[Mau92] Ueli M. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992.

[Mau93] Ueli M. Maurer. Secret key agreement by public discussion from common information. *Information Theory, IEEE Transactions on*, 39(3):733–742, May 1993.

[MDH$^+$15] David Meyer, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, and Friedrich Leisch. *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*, 2015. R package version 1.6-7.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, 1991.

[NBSI08] Anderson C. A. Nascimento, João Barros, Stefan Skludarek, and Hideki Imai. The commitment capacity of the gaussian channel is infinite. *Information Theory, IEEE Transactions on*, 54(6):2785–2789, June 2008.

[NJ01] Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In *Advances in Neural Information Processing Systems 14, Proceedings of the 2001 NIPS conference*, pages 841–848. MIT Press, 2001.

[NMQO$^+$03] Anderson C. A. Nascimento, Jörn Müller-Quade, Akira Otsuka, Goichiro Hanaoka, and Hideki Imai. Unconditionally secure homomorphic pre-distributed bit commitment and secure two-party computations. In Colin Boyd and Wenbo Mao, editors, *ISC 2003: 6th International Conference on Information Security*, volume 2851 of *Lecture Notes in Computer Science*, pages 151–164, Bristol, UK, October 1–3, 2003. Springer, Heidelberg, Germany.

[NMQO$^+$04] Anderson C. A. Nascimento, Jörn Müller-Quade, Akira Otsuka, Goichiro Hanaoka, and Hideki Imai. Unconditionally non-interactive verifiable secret sharing secure against faulty majorities in the commodity based model. In Markus Jakobsson, Moti Yung, and Jianying Zhou, editors, *ACNS 04: 2nd International Conference on Applied Cryptography and Network Security*, volume 3089 of *Lecture Notes in Computer Science*, pages 355–368, Yellow Mountain, China, June 8–11, 2004. Springer, Heidelberg, Germany.

[NNOB12] Jesper Buus Nielsen, Peter Sebastian Nordholt, Claudio Orlandi, and Sai Sheshank Burra. A new approach to practical active-secure two-party computation. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 681–700, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Heidelberg, Germany.

[NOVY98] Moni Naor, Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Perfect zero-knowledge arguments for NP using any one-way permutation. *Journal of Cryptology*, 11(2):87–108, 1998.

[NW08] Anderson C. A. Nascimento and Andreas Winter. On the oblivious-transfer capacity of noisy resources. *Information Theory, IEEE Transactions on*, 54(6):2572–2581, June 2008.

[NWI$^+$13] Valeria Nikolaenko, Udi Weinsberg, Stratis Ioannidis, Marc Joye, Dan Boneh, and Nina Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *2013 IEEE Symposium on Security and Privacy*, pages 334–348, Berkeley, CA, USA, May 19–22, 2013. IEEE Computer Society Press.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, February 1996.

[OH11] Frédérique Oggier and Babak Hassibi. The secrecy capacity of the mimo wiretap channel. *Information Theory, IEEE Transactions on*, 57(8):4961–4972, August 2011.

[OVY93] Rafail Ostrovsky, Ramarathnam Venkatesan, and Moti Yung. Fair games against an all-powerful adversary. In Renato Capocelli, Alfredo De Santis, and Ugo Vaccaro, editors, *Sequences II*, pages 418–429. Springer New York, 1993.

[OW85] Lawrence H. Ozarow and Aaron D. Wyner. Wire-tap channel II. In Thomas Beth, Norbert Cot, and Ingemar Ingemarsson, editors, *Advances in Cryptology – EUROCRYPT'84*, volume 209 of *Lecture Notes in Computer Science*, pages 33–50, Paris, France, April 9–11, 1985. Springer, Heidelberg, Germany.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238, Prague, Czech Republic, May 2–6, 1999. Springer, Heidelberg, Germany.

[PB05] Patricio Parada and Richard Blahut. Secrecy capacity of simo and slow fading channels. In *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*, pages 2152–2155, Adelaide, Australia, September 4–9, 2005.

[PDMN11] Adriana C. B. Pinto, Rafael Dowsley, Kirill Morozov, and Anderson C. A. Nascimento. Achieving oblivious transfer capacity of generalized erasure channels in the malicious model. *Information Theory, IEEE Transactions on*, 57(8):5566–5571, August 2011.

[Ped92] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *Advances in Cryptology – CRYPTO'91*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140, Santa Barbara, CA, USA, August 11–15, 1992. Springer, Heidelberg, Germany.

[Pul13] Pille Pullonen. Actively secure two-party computation: Efficient beaver triple generation. Master's thesis, University of Tartu, Tartu, Estonia, May 2013.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 554–571, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Heidelberg, Germany.

[PW96] Birgit Pfitzmann and Michael Waidner. Information-theoretic pseudosignatures and byzantine agreement for t >= n/3. Technical report, Research Report RZ 2882 (#90830) 18/11/96, IBM Research Division, Zürich, 1996.

[R C15] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015.

[Rab81] Michael O. Rabin. How to exchange secrets by oblivious transfer. Technical Report Technical Memo TR-81, Aiken Computation Laboratory, Harvard University, 1981.

[RBO89] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In *21st Annual ACM Symposium on Theory of Computing*, pages 73–85, Seattle, WA, USA, May 15–17, 1989. ACM Press.

[Riv99] Ronald L. Rivest. Unconditionally secure commitment and oblivious transfer schemes using private channels and a trusted initializer. Preprint available at http://people.csail.mit.edu/rivest/Rivest-commitment.pdf, 1999.

[Rom90] John Taylor Rompel. *Techniques for Computing with Low-independence Randomness*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1990.

[RTS00] Jaikumar Radhakrishnan and Amnon Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM J. Discret. Math.*, 13(1):2–24, January 2000.

[RW04] Renato Renner and Stefan Wolf. The exact price for unconditionally secure asymmetric cryptography. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 109–125, Interlaken, Switzerland, May 2–6, 2004. Springer, Heidelberg, Germany.

[RW05] Renato Renner and Stefan Wolf. Simple and tight bounds for information reconciliation and privacy amplification. In Bimal K. Roy, editor, *Advances in Cryptology – ASIACRYPT 2005*, volume 3788 of *Lecture Notes in Computer Science*, pages 199–216, Chennai, India, December 4–8, 2005. Springer, Heidelberg, Germany.

[Sav07] George Savvides. *Interactive Hashing and Reductions Between Oblivious Transfer Variants*. PhD thesis, McGill University, Montreal, Quebec, Canada, 2007.

[SKLR04] Ashish P. Sanil, Alan F. Karr, Xiaodong Lin, and Jerome P Reiter. Privacy preserving regression modelling via distributed computation. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 677–682, Seattle, WA, USA, August 22–25, 2004. ACM, New York, NY, USA.

[SSBD14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[SSW10] Ahmad-Reza Sadeghi, Thomas Schneider, and Immo Wehrenberg. Efficient privacy-preserving face recognition. In Donghoon Lee and Seokhie Hong, editors, *ICISC 09: 12th International Conference on Information Security and Cryptology*, volume 5984 of *Lecture Notes in Computer Science*, pages 229–244, Seoul, Korea, December 2–4, 2010. Springer, Heidelberg, Germany.

[ST06] Berry Schoenmakers and Pim Tuyls. Efficient binary conversion for Paillier encrypted values. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 522–537, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Heidelberg, Germany.

[SW02] Douglas Stebila and Stefan Wolf. Efficient oblivious transfer from any non-trivial binary-symmetric channel. In *Information Theory, 2002. Proceedings. 2002 IEEE International Symposium on*, page 293, Lausanne, Switzerland, June 30 – July 5, 2002.

[SY11] Junji Shikata and Daisuke Yamanaka. Bit commitment in the bounded storage model: Tight bound and simple optimal construction. In Liqun Chen, editor, *13th IMA International Conference on Cryptography and Coding*, volume 7089 of *Lecture Notes in Computer Science*, pages 112–131, Oxford, UK, December 12–15, 2011. Springer, Heidelberg, Germany.

[SZ13] Thomas Schneider and Michael Zohner. GMW vs. Yao? Efficient secure two-party computation with low depth circuits. In Ahmad-Reza Sadeghi, editor, *FC 2013: 17th International Conference on Financial Cryptography and Data Security*, volume 7859 of *Lecture Notes in Computer Science*, pages 275–292, Okinawa, Japan, April 1–5, 2013. Springer, Heidelberg, Germany.

[TAR15] Terry Therneau, Beth Atkinson, and Brian Ripley. *rpart: Recursive Partitioning and Regression Trees*, 2015. R package version 4.1-10.

[TND$^+$15] Rafael Tonicelli, Anderson C. A. Nascimento, Rafael Dowsley, Jörn Müller-Quade, Hideki Imai, Goichiro Hanaoka, and Akira Otsuka. Information-theoretically secure oblivious polynomial evaluation in the commodity-based model. *International Journal of Information Security*, 14(1):73–84, 2015.

[Tof07] Tomas Toft. *Primitives and Applications for Multi-party Computation.* PhD thesis, Aarhus University, 2007.

[Tof09a] Tomas Toft. Constant-rounds, almost-linear bit-decomposition of secret shared values. In Marc Fischlin, editor, *Topics in Cryptology – CT-RSA 2009*, volume 5473 of *Lecture Notes in Computer Science*, pages 357–371, San Francisco, CA, USA, April 20–24, 2009. Springer, Heidelberg, Germany.

[Tof09b] Tomas Toft. Solving linear programs using multiparty computation. In Roger Dingledine and Philippe Golle, editors, *FC 2009: 13th International Conference on Financial Cryptography and Data Security*, volume 5628 of *Lecture Notes in Computer Science*, pages 90–107, Accra Beach, Barbados, February 23–26, 2009. Springer, Heidelberg, Germany.

[Vad04] Salil P. Vadhan. Constructing locally computable extractors and cryptosystems in the bounded-storage model. *Journal of Cryptology*, 17(1):43–77, January 2004.

[Veu15] Thijs Veugen. Linear round bit-decomposition of secret-shared values. *Information Forensics and Security, IEEE Transactions on*, 10(3):498–506, March 2015.

[WFNL15] David J Wu, Tony Feng, Michael Naehrig, and Kristin E Lauter. Privately evaluating decision trees and random forests. Cryptology ePrint Archive, Report 2015/386, 2015. `http://eprint.iacr.org/`.

[WHC$^+$14] Xiao Shaun Wang, Yan Huang, T.-H. Hubert Chan, Abhi Shelat, and Elaine Shi. SCORAM: Oblivious RAM for secure computation. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 14: 21st Conference on Computer and Communications Security*, pages 191–202, Scottsdale, AZ, USA, November 3–7, 2014. ACM Press.

[WNI03] Andreas Winter, Anderson C. A. Nascimento, and Hideki Imai. Commitment capacity of discrete memoryless channels. In Kenneth G. Paterson, editor, *Cryptography and Coding, 9th IMA International Conference*, volume 2898 of *Lecture Notes in Computer Science*, pages 35–51, Cirencester, UK, December 16–18, 2003. Springer, Heidelberg, Germany.

[Wul09] Jürg Wullschleger. Oblivious transfer from weak noisy channels. In Omer Reingold, editor, *TCC 2009: 6th Theory of Cryptography Conference*, volume 5444 of *Lecture Notes in Computer Science*, pages 332–349. Springer, Heidelberg, Germany, March 15–17, 2009.

[WW10] Severin Winkler and Jürg Wullschleger. On the efficiency of classical and quantum oblivious transfer reductions. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture*

Notes in Computer Science, pages 707–723, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Heidelberg, Germany.

[Wyn75] Aaron D. Wyner. The wire-tap channel. *Bell System Technical Journal*, 54(8):1355–1387, 1975.

[Yao82] Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science*, pages 160–164, Chicago, Illinois, November 3–5, 1982. IEEE Computer Society Press.

[Zuc97] David Zuckerman. Randomness-optimal oblivious sampling. *Random Structures & Algorithms*, 11(4):345–367, 1997.