1

Unconditionally Secure Multi-party Computation from Weak Primitives

Stefan Wolf and Jürg Wullschleger

Institut für Theoretische Informatik, ETH Zürich, Switzerland
{wolf,wullschj}@inf.ethz.ch

1.1 Introduction

Two old friends, who have both become millionaires, want to find out who is richer without, however, any one of them having to reveal any (additional) information about their wealth. Is it possible in principle to fulfill this task?

Alice and Bob, the two main characters of contemporary cryptography, speak on the telephone and would like to do a fair coin flip. Is this possible at all? Clearly, the obvious way of one party throwing a coin and telling the result to the other is useless by reasons that should be all too obvious: "I am sorry, but *you lost*...".

The world-wide cryptographers' union would like to elect a new president by e-mail in a fair way, i.e., such that nobody can influence the outcome of the vote beyond delivering her proper ballot.

Clearly, all these tasks are easy to fulfill as long as a trusted party is present. It is a natural goal to simulate such a party by a protocol between the involved players. This is called *multi-party computation (MPC)* and has been introduced by Yao [50] in 1982. The general setting is that a number of parties or *players* want to collaborate in a secure way in order to achieve a common goal, however they mutually distrust each other and do not want to make use of a *trusted third party*, i.e., a distinct player that would carry out the computation for them. Note that, in contrast to many other problem in cryptography, the players are not *a priori* good or bad.

In this chapter we will only look at special case of *secure function evaluation* , where every party holds an input to a function, and the output should be computed in a way such that no party has to reveal unnecessary information about her input. A complete solution for this problem with respect to computational security was given in [30]. In [5, 14], it was shown that in a model with only pairwise secure channels, MPC unconditionally secure against an active adversary is achievable if and only if $t < n/3$ players are corrupted. In [2, 42] it was shown that this bound can be improved to $t < n/2$, assuming

that global broadcast channels are additionally given—and this bound was shown tight. A protocol more efficient than those in [2, 42] was given in [16].

For the case where $n = 2$, there cannot exist a protocol that is unconditionally secure, as we will see later. However, if a primitive called *oblivious transfer* is assumed, then *any* function can be calculated unconditionally securely as has been shown by Kilian [34].

In this work we will present several protocols that implement the basic primitives of bit commitment, oblivious transfer, and broadcast based on noisy resources such as noisy channels or correlated randomness.

1.1.1 Basic Primitives

We will now present the basic functionalities in multi-party computation, where they are used and how they relate to each other.

Bit Commitment.

Bit commitment was introduced in [9]—together with distributed coin-flipping among two players. Bit commitment is a cryptographic primitive in which at some point Alice has to commit herself to a value of her choice that Bob does not get to know. Later, Alice can open her commitment to Bob. It is guaranteed that she cannot reveal any other value than the committed one. Bit commitments are used in identification schemes, zero-knowledge proofs, and general multi-party computation.

Bit commitment implies coin toss. A basic application of bit commitment is that it can be used to execute a distributed coin toss. This is not hard to achieve: Alice commits herself to a random bit, then Bob sends her a random bit, and finally Alice opens her commitment. It can easily be verified that the XOR of the two bits will always be random if one of the two players is honest.

Oblivious Transfer.

Oblivious transfer, or OT for short, was introduced by Wiesner [44] (under the name of "multiplexing") and Rabin [41]. We will be using the variant of OT introduced in [25], called *chosen one-out-of-two oblivious transfer*, $\binom{2}{1}$−OT for short, which is the most important version of OT today. Here, the sender sends two bits b_0 and b_1 and the receiver's input is a choice bit c; the latter then learns b_c but gets no information about the other bit b_{1-c}.

OT implies Bit Commitment. Bit commitment can easily be implemented using n instances of OT. To commit to a value v, the commiter sends random bits using the n instances of OT, where the XOR of each input pair equals v. The verifier chooses for each instance one of the two bits randomly. To open the commitment, the commiter sends all the bits he has sent before. The verifier checks whether the values match the values he has received before. It is easy to see that the verifier does not get any information about the value v

in the commit phase, as he only receives v xored with a random bit. On the other hand, the commiter cannot change the value he is committed to with a probability bigger than 2^{-n}, since he would need to be able to change the value the verifier did not receive, but since he does not know the values c_i he will fail to do so with probability $\frac{1}{2}$ in every round.

Pseudo-signature schemes (PSS).

Much less known than classical *digital signatures* [23, 35] are so-called *pseudo-signature schemes*, which guarantee, in contrast to the former, *unconditional* security. The inherent price for their higher security, however, is that a signature can only be transfered a limited number of times. After prior work in [42, 15], the first complete PSS was proposed in [40]. This scheme allows for any (constant) transferability λ and any number of corrupted players.

Broadcast.

The *broadcast problem* was introduced by Lamport, Shostak, and Pease [36]. A protocol where one player can send a value and all players receive a value achieves broadcast, if all honest player receive the value sent by a honest sender, and receive the same value, if the sender is malicious. [36] showed that, in the model with secure channels between all pairs of players, but without the use of a signature scheme, broadcast is achievable if and only if the number t of cheaters satisfies $t < n/3$. Furthermore, it was shown that when additionally a signature scheme is given among the n players, then computationally secure broadcast is achievable for any number of corrupted players. The first efficient such protocol was given in [24]. In [40], an efficient protocol was given with unconditional security based on a pseudo-signature scheme with transferability $t + 1$.

1.1.2 Definition of Security

We will assume that the adversary is unlimited in computing power, and that his behavior may deviate from the protocol specification in an arbitrary way. The definition of security used today was given in [38] and [3], which is based on the so-called *real vs. ideal* paradigm. The idea behind the definition is that anything an adversary can achieve in the *real life protocol*, he could also achieve by an attack in an *ideal world*, i.e., where he only has black-box access to the functionality to be achieved. Such a protocol is secure if for any adversary in the real life, there exists an adversary in the ideal model such the both situations produce the same output distribution. A very important property of this definition is that it implies that the *sequential composition* of secure protocols is again a secure protocol, which was shown in [11]. This greatly simplifies proofs of protocols, as only small parts need to be shown to be secure.

1.1.3 Impossibility of Bit Commitment and Oblivious Transfer from Scratch

It would be preferable to have an implementation of unconditionally secure bit commitment and OT protocols only using (noiseless) communication. Unfortunately, such protocols cannot exist, as we will show now. Assume that there exists a bit commitment protocol, unconditionally secure for both players. It must be secure for the receiver, which means that after the commit phase, given all the communication between the two players, there can only exist *one* possible opening for the sender. But this means that the receiver is able, at least in principle, to calculate that value. Hence such a protocol would not be unconditionally secure for the sender. It follows that OT is also impossible to achieve from scratch, since OT implies bit commitment.

1.1.4 Models for Physical Resources

We will consider two different models for a physical system, distributed randomness and channels. Because they are easier to analyze, we will assume that the physical system *perfectly* implements these resources, i.e., without any error. In the last section, we will discuss some more realistic models.

Distributed Randomness. The two players receive values X and Y, distributed according to a certain distribution P_{XY}, from a resource outside their control. Such a source could for example be a satellite.

Channel. The two players are connected by a *noisy channel*, i.e., a resource where one player can choose an input X, and the other player receives an output Y, distributed according to a certain conditional distribution $P_{Y|X}$.

1.2 Preliminaries

In this section, we introduce some information-theoretic notions of central importance in the rest of this article.

Common Part

The *common part* of two random variables X and Y was first introduced in [29]. We denote it by $X \wedge Y$. It is the largest random variable that two players Alice and Bob knowing the random variables X and Y, respectively, can commonly extract, without any error, i.e., there exist functions f_X and f_Y with $X \wedge Y = f_X(X) = f_Y(Y)$.

Sufficient Statistics

The *sufficient statistics* of a random variable X with respect to Y is the part of X that is dependent on Y. We will use the notion $X \searrow Y$, as in [28, 46, 47], where it was called the *dependent part*. It was also used in [32, 45], where it was called *non-redundant*. $X \searrow Y$ can be calculated very easily in the following way: $X \searrow Y := f(X)$ for $f(x) = P_{Y|X=x}$. It is obtained by collapsing all values x_1 and x_2, for which Y has the same conditional distribution, to a single value.

The following lemma show an important property of the sufficient statistics. Roughly speaking, it shows that the sufficient statistics of X with respect to Y cannot be changed by a player that only knows X without changing the joint distribution with Y.

Lemma 1.1. [28] *Let X and Y be random variables, and let $K = X \searrow Y$. Let $\overline{X} = f(X)$ for a randomized function f. If $P_{KY} = P_{\overline{K}Y}$, then we have $\overline{K} = K$.*

This property will be very useful in our application, as it allows the player holding Y to verify whether the other player really sent him $X \searrow Y$ by doing a statistical test over many instances of X and Y.

Universal Hashing and Randomness Extraction

The *statistical distance* of two random variables X and Y over the same domain $\mathcal{V}$ is defined as $\mathsf{SD}[X, Y] := \frac{1}{2} \sum_{v \in \mathcal{V}} |P_X(v) - P_Y(v)|$. A function $h : \mathcal{R} \times \mathcal{X} \to \{0,1\}^m$ is called a *2-universal hash function* [12] if for all $x_0 \neq x_1 \in \mathcal{X}$, the probability that they are mapped to the same value is at most 2^{-m}; more precisely, are we have

$$\Pr[h(R, x_0) = h(R, x_1)] \leq 2^{-m}$$

if R is uniform over $\mathcal{R}$. We measure the *uncertainty* of a random variable X, given a random variable Y, as the *conditional min-entropy*, which is defined as

$$\mathsf{H}_{\min}(X \mid Y) = \min_{x,y} \log \frac{1}{P_{X|Y}(x \mid y)} \ .$$

Note that for *i.i.d.* random variables, the conditional min-entropy converges to the *conditional Shannon entropy*, which is defined as

$$\mathsf{H}(X \mid Y) = \sum_{x,y} P_{XY}(x, y) \log \frac{1}{P_{X|Y}(x \mid y)} \ ,$$

with an error that vanishes exponentially fast. See [31] for explicit bounds for this.

The following two facts (Lemmas 1.2 and 1.3) are often used in information-theoretic protocols in cryptography. The first one is called the *leftover hash*

lemma by Impagliazzo, Levin, and Luby [33]. It tells us that we can extract about $\mathsf{H}_{\min}(X \mid Y)$ bits from a random variable X that are uniformly distributed and independent of Y. In other words, an adversary who knows Y will have almost no knowledge about the extracted value. That is why this process is also called *privacy amplification* [8, 6].

Lemma 1.2. Leftover Hash Lemma [33] *Let X and Y be random variables over $\mathcal{X}$ and $\mathcal{Y}$, and let $m > 0$. Let $h : \mathcal{S} \times \mathcal{X} \to \{0,1\}^m$ be a 2-universal hash function. If $m \leq \mathsf{H}_{\min}(X|Y) - 2\log(1/\varepsilon)$, then for S uniform over $\mathcal{S}$, we have $\mathsf{SD}[(h(S,X),S,Y),(U,S,Y)] \leq \varepsilon$, where U is uniform over $\{0,1\}^m$ and independent of S and Y.*

Often, certain pieces of information are leaked to the adversary during a protocol. The following fact due to Maurer and Wolf [37] gives us a lower bound on the remaining min-entropy: Roughly speaking, it states that with high probability, the min-entropy decreases by at most the logarithm of the alphabet size of the leaked information.

Lemma 1.3. [37] *For all random variables X and Y, and for all $\varepsilon > 0$, we have $\Pr[\mathsf{H}_{\min}(X \mid Y) \geq \mathsf{H}_{\min}(XY) - \log|\mathcal{Y}| - \log(1/\varepsilon)] \geq 1 - \varepsilon$.*

Basic Reductions for OT

We will discuss two basic reductions for OT. The first implements OT from a randomized version of it. The protocol is due to Bennett, Brassard, Crépeau, and Skubiszewska [7].

Protocol 1 (OT from randomized OT) *The sender has inputs $b_0, b_1 \in \{0,1\}$, the receiver $c \in \{0,1\}$. Furthermore, the sender has the uniformly random value $\overline{B}_0, \overline{B}_1 \in \{0,1\}$, the uniformly random values $\overline{C} \in \{0,1\}$, and $\overline{Y} = \overline{B}_{\overline{C}}$.*

1. *The receiver sends $m = c \oplus \overline{C}$.*
2. *The sender sends $r_0 = b_0 \oplus \overline{B}_m$ and $r_1 = b_1 \oplus \overline{B}_{1-m}$*
3. *The receiver outputs $y = r_c \oplus \overline{Y}$.*

Intuitively, the protocol is secure because, first of all, the sender only gets the value c one-time padded by a random value $\overline{C}$, and hence will not get any information about c. Furthermore, the receiver will only receive either $\overline{B}_0$ or $\overline{B}_1$, but not both. Therefore he will only be able to decrypt one one-time pad, the one of his choice, while he will remain ignorant about the other value.

Note that one way to obtain the values $\overline{B}_0$ $\overline{B}_1$, $\overline{C}$, and $\overline{Y}$ is to give random inputs to an instance of OT. This gives us a method to *precompute OT* [4]. This can be very useful in our application, as implemented OT may not be available when the OT is needed.

The following protocol is due to Crépeau and Kilian [18]. It shows that many instances of OT, all of which are secure for the sender but only one of them is secure for the receiver, can be combined to achieve a secure implementation of OT.

Protocol 2 (R-Combiner) *The sender has inputs $b_0, b_1 \in \{0,1\}$, the receiver $c \in \{0,1\}$. Furthermore, they have k implementations of OT.*

1. *Alice chooses the values $b_{01}, \ldots, b_{0k-1}$ uniformly at random and sets $b_{0k} := \bigoplus_{i=1}^{k-1} b_{0i} \oplus b_0$ and $b_{1i} := b_{0i} \oplus b_0 \oplus b_1$, for $i \in [k]$.*
2. *Bob chooses the values $c_1, \ldots, c_{k-1}$ uniformly at random and sets $c_k := \bigoplus_{i=1}^{k-1} c_i \oplus c$.*
3. *They execute the k implementations of OT, using b_{0i}, b_{1i} and c_i as input in the i-th execution. Bob receives y_i.*
4. *Bob outputs $y := \bigoplus_{i=1}^{k} y_i$.*

It is easy to verify that the output is correct. If the sender gets the receiver's input in $k-1$ instances, he will not learn c, as it is still one-time padded with a random value. On the other hand, if all instances of OT are secure for the sender, then the receiver will not get any information about $b_0 \oplus b_1$, a fact that implies that the resulting OT is secure for the sender.

1.3 Monotones

It was shown in [47] that the following three quantities are *monotones* for two party computation, i.e., cannot increase during the execution of any protocol based on (noiseless) communication and (lossless) processing (where X and Y are the random variables summarizing the entire information accessible to A and B, respectively):

$$\mathsf{H}(Y \searrow X \mid X) \ , \qquad \mathsf{H}(X \searrow Y \mid Y) \ , \qquad \mathbf{I}(X;Y \mid X \wedge Y) \ .$$

These monotones can now be used to show the impossibility of certain reduction protocols, or at least to derive lower bounds on the efficiency of such protocols.

Since OT is equivalent to a randomized version OT (see Protocol 1), which is simply distributed randomness, we can directly apply these monotones: For $\binom{2}{1}-$OT, all three monotones are equal to 1. However, if the players do not have any distributed randomness to start with, all three monotones have the value 0 at the beginning. Since they cannot be increased, we get another proof for the impossibility of OT from scratch! But we can now also easily conclude that OT cannot be duplicated, i.e., it is impossible to make $n+1$ instances of OT out of n instances, or we can derive lower bounds on the number of instances needed for a certain reduction. For example, to produce one instance of $\binom{2}{1}-$OT out of distributed randomness that is distributed according to P_{XY}, we need at least

$$\max\left(\frac{1}{\mathsf{H}(Y \searrow X \mid X)} \ , \ \frac{1}{\mathsf{H}(X \searrow Y \mid Y)} \ , \ \frac{1}{\mathbf{I}(X;Y \mid X \wedge Y)}\right)$$

instances of this randomness.

1.4 Bit Commitment from Noise

Bit commitment based on common randomness was introduced in [43]. In [45], the *commitment capacity* of two correlated random variables is defined. It is the supremum of all rates (bits per instance) that can be achieved with an arbitrarily small error. Note that in that model, all bits are committed to and opened *simultaneously* and cannot be opened separately. It was proved in [45] that the commitment capacity of X and Y, where the committer holds X and the verifier holds Y, is

$$\mathsf{H}(X \searrow Y \mid Y) \ .$$

The algorithm of [32] is based on a code that has been introduced by Wyner [49] for the *wire-tap channel*, but is inefficient. An efficient (and also simpler) protocol was given in [46]; it only relies on universal hashing [12] and Lemma 1.2, and works as follows.

Let Alice have $X^n = X_1, \ldots, X_n$ and Bob have $Y^n = Y_1, \ldots, Y_n$. Assume that Alice wants to commit to a value $d \in \{0,1\}^\ell$. Let $h : \{0,1\}^* \times \mathcal{K}^n \to \{0,1\}^m$ and $\mathrm{ext} : \{0,1\}^* \times \mathcal{K}^n \to \{0,1\}^\ell$ be 2-universal hash functions.

Protocol 3 *Commit to a value $d \in \{0,1\}^l$.*

1. *Bob chooses $r_1 \in \{0,1\}^*$ and sends it to Alice.*
2. *Alice calculates $K^n := (X_1 \searrow Y_1, \ldots, X_n \searrow Y_n)$. She chooses $r_0 \in \{0,1\}^*$ and sends $c := (r_0, h(r_1, K^n), d \oplus ext(r_0, K^n))$ to Bob.*

Protocol 4 *Open the commitment.*

1. *Alice sends (d, K^n) to Bob.*
2. *Bob accepts d if (K^n, Y^n) is distributed according to $P_{X \searrow Y, Y}$ and $c = (r_0, h(r_1, K^n), d \oplus ext(r_0, K^n))$, but rejects otherwise.*

Since Alice has to send the sufficient statistics K^n to Bob, we know by Lemma 1.1 that she cannot change the value K^n without changing the joint statistics of K^n with Y^n. Therefore, she can only change $s = \mathcal{O}(\sqrt{n})$ values in K^n without being detected. Therefore, Alice can only choose from

$$\binom{n}{s} |\mathcal{K}|^s \leq (n|\mathcal{K}|)^s$$

different values. Let us choose $m = s \log(n|\mathcal{K}|) + \log(1/\varepsilon)$. Now, the probability that among the values Alice can choose from, there exist two values with the same hash value is at most

$$2^{-s \log(n|\mathcal{K}|) - \log(1/\varepsilon)} (n|\mathcal{K}|)^s = (n|\mathcal{K}|)^{-s} \varepsilon (n|\mathcal{K}|)^s = \varepsilon \ .$$

Therefore, with probability of at least $1 - \varepsilon$ Alice will not be able to change her commitment, which implies that the commitment is binding. To ensure that Bob has no information about Alice's secret, she applies the function ext

on K^n, where $\ell \leq \mathsf{H}_{\min}(K^n \mid Y^n) - m - 3\log(1/\varepsilon)$. Lemma 1.3 implies that the min-entropy of Bob about K^n after he has received $h(r_1, K^n)$ is at least $\mathsf{H}_{\min}(K^n \mid Y^n) - m - \log(1/\varepsilon)$ with probability at least $1-\varepsilon$. Lemma 1.2 states now that given Bobs information, the extracted string is ε-close to uniform. Therefore, with a probability of at least $1-2\varepsilon$, Bob has no information about d in the commitment phase. Since $m = \mathcal{O}(\sqrt{n}\log(n))$ is sub-linear in n, and $\mathsf{H}_{\min}(K^n \mid Y^n)$ converges to $n\mathsf{H}(X \searrow Y \mid Y)$, it follows that our scheme achieves the commitment rate $\mathsf{H}(X \searrow Y \mid Y)$.

Channels.

The same protocol can also be applied to the model where the two players are connected by a channel: we let the sender give random input, according to a specified distribution, to the channel, and then apply the above protocol. Additionally, we have to make sure that the receiver can verify whether the sender has used the specified input distribution [45]. It turns out that bit commitment is possible for any non-trivial channel, the same condition as for OT.

1.5 Oblivious Transfer from Noise

We will describe the algorithm of [19] showing that OT can be achieved from *any non-trivial channel* $P_{Y|X}$, which generalizes the results of [17]. A channel is non-trivial if there exist two inputs x_1 and x_2 such that $P_{Y|X=x_1} \neq P_{Y|X=x_2}$ holds and there exists a $y \in \mathcal{Y}$ such that $P_{Y|X=x_1}(y) > 0$ and $P_{Y|X=x_2} > 0$ both hold. Furthermore, we require that x_1 and x_2 are *extremal neighbors*, meaning, in particular, that the distributions $P_{Y|X=x_1}$ and $P_{Y|X=x_2}$ cannot be produced by any linear combination of other inputs. OT can now be achieved in several steps. First, the channel is used to implement a *binary-symmetric erasure channel* (BSEC). Let (y_1, y_2) be the *most informative pair*, i.e., the values which give the best estimate over the input, given the input is uniform over x_1 and x_2.

Protocol 5 (BSEC) *The sender has an input $r \in \{0,1\}$.*

1. *The sender sends x_1x_2 if $r = 0$, and x_2x_1 if $r = 1$ over $P_{Y|X}$.*
2. *The receiver returns O if he receives y_1y_2, and 1 if he receives y_2y_1. Otherwise, Δ.*

In this protocol, the sender that correctly chooses the inputs will get no information about whether the receiver outputs Δ or not. The receiver will get some information about the input of the sender even if he receives Δ, however this information will be smaller than in the case when he receives a value in $\{0,1\}$. Also note that the channel might make some errors, i.e., the receiver may get an output 1, even if the sender has sent a value 0.

We can now use this protocol to construct an OT that is secure if the sender follows the protocol. The protocol makes use of *one-way key agreement* [20, 1].

Protocol 6 (Passively Secure OT) *The sender has inputs $b_0, b_1 \in \{0,1\}$ and the receiver $c \in \{0,1\}$.*

1. *The sender picks $2n$ random bits r_i and sends them to the receiver using the BSEC; the receiver gets r'_i.*
2. *The receiver picks and sends two disjoint sets I_0 and I_1 of the same size n', such that $r_i \neq \Delta$ for all $i \in I_c$.*
3. *The sender and the receiver execute twice a one-way key-agreement protocol, once for the values in I_0 and once for the values in I_1. The sender gets two key bits, k_0 and k_1. The receiver gets k_c.*
4. *The sender sends $m_0 = b_0 \oplus k_0$ and $m_1 \oplus k_1$. The receiver outputs $m_c \oplus k_c$.*

The parameter n' has to be as large as possible, but such that the honest player can construct one set I_c which does not contain any $r'_i = \Delta$ with high probability. Then, a malicious player does not receive enough values to construct two such sets, and therefore has some disadvantage against the honest player in at least one of the two sets. The key-agreement protocol then ensures that the honest player gets the key, while the dishonest does not get any information about the key for the string about which he has some disadvantage.

Now we have to ensure that the sender cannot actively cheat the protocol by not choosing the input values to the channel as he is supposed to. To do so, we apply the above protocol $k = \lceil n^{1+\varepsilon} \rceil$ times on random input, and use Protocol 2 to combine them to one instance of OT. In order to be able to cheat in this randomized OT, the sender would have to cheat in every instance of the k OTs at least once, which means he would have to choose a different input in at least $n^{1+\varepsilon}$ instances BSEC of the total $n^{2+\varepsilon}$ instances. But this would bias the statistical distribution of the outputs of the BSECs since $n^{1+\varepsilon} = (\sqrt{n^{2+\varepsilon}})^{1+\Theta(1)}$. We let the sender apply a statistical test to prevent this. Then, we only have to apply Protocol 1 to implement OT from randomized OT to end up with a secure implementation of OT.

Distributed Randomness.

We will use the simple technique of [46] to show how OT can be reduced to distributed randomness. A more efficient way can be found in [39].

Let Alice have $X_1, X_2, \ldots, X_n$ and Bob $Y_1, Y_2, \ldots, Y_n$. Alice and Bob simulate a channel in the following way: For all i, Alice erases the values X_i with a certain probability, such that all $x \in \mathcal{X}$ occur with the probability of the least probable x_0. On input value x, Alice sends Bob the index i of the first value X_i with $X_i = x$. Bob outputs the value Y_i. It can be shown that this channel satisfies the condition to achieve OT if the distributed randomness satisfies $\mathsf{H}(X \searrow Y \mid Y) > 0$. Clearly, this bound is tight as for $\mathsf{H}(X \searrow Y \mid Y) = 0$, we cannot even achieve bit commitment.

1.6 Pseudo-Signatures and Broadcast from Noise

It was shown in [40] how to set up a PSS using global broadcast channels, using the *dining-cryptographers protocol* [13, 10]. Obtaining a PSS from a common random source was considered in [26, 27], but only with respect to three players and *one particular* probability distribution.

In [28] an implementation of a PSS among three players is presented, assuming that the players have access to some distributed randomness, that we will present now. Let the signer have X^n, the intermediate player have Y^n, and the receiver have Z^n. These random variables are *i.i.d.* according to P_{XYZ}. Furthermore, let us assume that $\mathbf{I}(X \searrow Y; Z \mid Y) > 0$.

Protocol 7 *Let $v \in \{0,1\}$ be the value P_1 wants to sign.*

1. *P_1 calculates $K_i := X_i \searrow Y_i$ and sends $(v, K_{1+(n/2)v}, \ldots, K_{n/2+(n/2)v})$ to P_2.*
2. *P_2 checks whether the received K_i and the corresponding Y_i are have the correct joint distribution $P_{X\searrow Y,Y}$. If so, he accepts, calculates $L_i := (K_i, Y_i) \searrow Z_i$, and sends $(v, L_{1+(n/2)v}, \ldots, L_{n/2+(n/2)v})$ to P_3.*
3. *P_3 checks whether the received L_i and the corresponding Z_i have the correct joint distribution $P_{(X\searrow Y,Y)\searrow Z,Z}$. If so, he accepts.*

First of all, the signed bit from a correct sender P_1 is accepted by P_2 except with exponentially small probability. Because of Lemma 1.1, even a malicious signer has to send the correct values K_i if he wants the intermediate player to accept. If an honest intermediate player accepts, then only $\mathcal{O}(\sqrt{n})$ values of K^n may be false. Therefore, he can be sure that the receiver accepts his message. (Note that the receiver has to be somewhat more tolerant about the errors he accepts.) If the intermediate player tries to transfer $1 - v$, he needs to calculate the values L_i only using his values Y_i. It can now be shown that he will not be able to do that if $\mathbf{I}(X \searrow Y; Z \mid Y) > 0$.

1.6.1 Broadcast from Pseudo-Signatures

We will now present a protocol for broadcast among the three players P_1, P_2, and P_3, where P_1 is the sender, using the above pseudo-signature scheme where P_1 is the signer. We assume that signatures can be transfered along the path $P_1 \to P_2 \to P_3$. The following protocol is adapted from [27].

Protocol 8 *Let $v \in \{0,1\}$ be the value P_1 wants to send.*

1. *P_1 signs v and sends it to P_2. He also sends v (unsigned) to P_3.*
2. *P_2 receives v' and checks the signature. If it is correct, he transfers it to P_3 and outputs v'. If the signature is not correct, he sends $\perp$ to P_3.*
3. *P_3 receives a value v'' from P_2 and u from P_1. If the signature of v'' is correct, then he outputs v''. Otherwise, he outputs u and sends u to P_2.*
4. *If P_2 has not yet output a value, he outputs what P_3 sends to him.*

If P_1 is honest, then an honest P_2 will identify the signature as correct and output v. P_2 can now correctly transfer v and its signature to P_3, who will be able to verify the signature and output v. However, he cannot send any other value to P_3 that will be accepted, since he is not able to forge signatures. Hence, an honest P_3 will also output v, and the protocol is correct if the sender is honest. Let now P_2 and P_3 be honest. If P_2 has received a correctly signed value v', then he will output v', but he will also be able to transfer the signature to P_3 who will also output v'. If P_2 did not receive a correctly signed value v', then he will finally output what P_3 sent him, which is the same value as P_3 outputs. Hence, the protocol is also correct if the sender is malicious.

Using our PSS, we can achieve broadcast among three players having X, Y, and Z, where the sender holds X, if

$$\mathbf{I}(X \searrow Y; Z \mid Y) + \mathbf{I}(X \searrow Z; Y \mid Z) > 0 \; .$$

It was shown in [28] that this bound is tight, i.e., no such protocol can exist if $\mathbf{I}(X \searrow Y; Z \mid Y) + \mathbf{I}(X \searrow Z; Y \mid Z) = 0$.

1.7 More Realistic Models

The results of the previous section seem very promising: almost *any* noise can be used to achieve tasks which would otherwise be impossible. Unfortunately, the assumptions used are not as weak as one might think.

Let us for example look at the *binary noisy channel*. This is a model that is often used in communication, because *there*, it *is* a very weak primitive. It is only required that the receiver obtains a bit that is enough correlated with the bit the sender has sent. However, in cryptography this condition does not suffice, because the receiver (or the sender) must not receive too much information. This means that the error has to be *exactly* as specified, and the sender or receiver must not get *any* information about whether an error occurred or not. It seems to be very hard to come up with a physical implementations that satisfies these specifications.

If it is possible to physically implement a primitive such as binary noisy channel, it will probably not be perfect, but only approximate the binary noisy channel with a small error that cannot be chosen arbitrarily small, but is constant. But then, the protocols we have seen will give us an implementation of OT with a constant error which, depending on the application, might not be sufficiently small. One possibility to reduce the error is to *amplify* the quality of our implementation. For oblivious transfer, it was shown in [22] that a weak version of oblivious transfer can be amplified for certain parameters. These results were corrected and improved in [48].

To get a more realistic model of a physical noisy channel, the notion of so-called *unfair noisy channels* was introduced in [22]. Such a channel is binary

symmetric with error probability of δ if both players are honest. However, if one of the player cheats, he may reduce the error to $\gamma < \delta$ and use the additional knowledge to his advantage. It has been proved in [22] and [21] that for a certain range, oblivious transfer is still possible.

References

1. R. Ahlswede and I. Csiszár. Common randomness in information theory and cryptography – part I: Secret sharing. *IEEE Transactions on Information Theory*, 39(4):1121–1132, 1993.
2. D. Beaver. Multiparty protocols tolerating half faulty processors. In *Advances in Cryptology—CRYPTO 1989*, volume 435 of *Lecture Notes in Computer Science*, pages 560–572. Springer-Verlag, 1989.
3. D. Beaver. Foundations of secure interactive computing. In *Advances in Cryptology—CRYPTO 1991*, pages 377–391. Springer-Verlag, 1992.
4. D. Beaver. Precomputing oblivious transfer. In *Advances in Cryptology—EUROCRYPT 1995*, Lecture Notes in Computer Science, pages 97–109. Springer-Verlag, 1995.
5. M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC '88)*, pages 1–10. Springer-Verlag, 1988.
6. C. Bennett, G. Brassard, C. Crépeau, and U. Maurer. Generalized privacy amplification. *IEEE Transactions on Information Theory*, 41, 1995.
7. C. H. Bennett, G. Brassard, C. Crépeau, and H. Skubiszewska. Practical quantum oblivious transfer. In *Advances in Cryptology—CRYPTO 1991*, volume 576 of *Lecture Notes in Computer Science*, pages 351–366. Springer, 1992.
8. C. H. Bennett, G. Brassard, and J.-M. Robert. Privacy amplification by public discussion. *SIAM Journal on Computing*, 17(2):210–229, 1988.
9. M. Blum. Coin flipping by telephone: A protocol for solving impossible problems. In *Proceedings of the 24th IEEE Computer Conference*, pages 133–137, 1982.
10. J. Bos and B. den Boer. Detection of disrupters in the DC protocol. In *Advances in Cryptology—EUROCRYPT 1989*, volume 434 of *Lecture Notes in Computer Science*, pages 320–327. Springer-Verlag, 1990.
11. R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.
12. J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18:143–154, 1979.
13. D. Chaum. The Dining Cryptographers Problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1(1):65–75, 1988.
14. D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols (extended abstract). In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC '88)*, pages 11–19. ACM Press, 1988.
15. D. Chaum and S. Roijakkers. Unconditionally-secure digital signatures. In *Advances in Cryptology—CRYPTO 1990*, volume 537 of *Lecture Notes in Computer Science*, pages 206–214. Springer-Verlag, 1990.

16. R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin. Efficient multiparty computations secure against an adaptive adversary. In *Advances in Cryptology—EUROCRYPT 1999*, volume 1592 of *Lecture Notes in Computer Science*, pages 311–326. Springer-Verlag, 1999.
17. C. Crépeau. Efficient cryptographic protocols based on noisy channels. In *Advances in Cryptology—CRYPTO 1997*, volume 1233 of *Lecture Notes in Computer Science*, pages 306–317. Springer-Verlag, 1997.
18. C. Crépeau and J. Kilian. Achieving oblivious transfer using weakened security assumptions (extended abstract). In *IEEE Symposium on Foundations of Computer Science*, pages 42–52, 1988.
19. C. Crépeau, K. Morozov, and S. Wolf. Efficient unconditional oblivious transfer from almost any noisy channel. In *SCN*, pages 47–59, 2004.
20. I. Csiszár and J. Körner. Broadcast channels with confidential messages. *IEEE Transactions on Information Theory*, 24:339–348, 1978.
21. I. Damgård, S. Fehr, K. Morozov, and L. Salvail. Unfair noisy channels and oblivious transfer. In *Theory of Cryptography Conference—TCC 2004*, pages 355–373, 2004.
22. I. Damgård, J. Kilian, and L. Salvail. On the (im)possibility of basing oblivious transfer and bit commitment on weakened security assumptions. In *Advances in Cryptology—CRYPTO 1999*, volume 1592 of *Lecture Notes in Computer Science*, pages 56–73. Springer-Verlag, 1999.
23. W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.
24. D. Dolev and H. R. Strong. Polynomial algorithms for multiple processor agreement. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC '82)*, pages 401–407, 1982.
25. S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Commun. ACM*, 28(6):637–647, 1985.
26. M. Fitzi, N. Gisin, and U. Maurer. Quantum solution to the Byzantine agreement problem. *Physical Review Letters*, 87(21):7901–1–7901–4, 2001.
27. M. Fitzi, N. Gisin, U. Maurer, and O. von Rotz. Unconditional Byzantine agreement and multi-party computation secure against dishonest minorities from scratch. In *Advances in Cryptology—EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 482–501. Springer-Verlag, 2002.
28. M. Fitzi, S. Wolf, and J. Wullschleger. Pseudo-signatures, broadcast, and multiparty computation from correlated randomness. In *Advances in Cryptology—CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 562–578. Springer-Verlag, 2004.
29. P. Gacs and J. Körner. Common information is far less than mutual information. *Probl. Contr. Inform. Theory*, 2:149–162, 1973.
30. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC '87)*, pages 218–229. ACM Press, 1987.
31. T. Holenstein and R. Renner. On the randomness of independent experiments. http://arxiv.org/, cs.IT/0608007, 2006.
32. H. Imai, J. Müller-Quade, A. Nascimento, and A. Winter. Rates for bit commitment and coin tossing from noisy correlation. In *Proceedings of the IEEE International Symposium on Information Theory (ISIT '04)*, 2004.

33. R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC '89)*, pages 12–24. ACM Press, 1989.
34. J. Kilian. Founding cryptography on oblivious transfer. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, pages 20–31, 1988.
35. L. Lamport. Constructing digital signatures from a one-way function. Technical Report SRI-CSL-98, SRI International Computer Science Laboratory, 1979.
36. L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.
37. U. Maurer and S. Wolf. Privacy amplification secure against active adversaries. In *Advances in Cryptology—CRYPTO 1997*, volume 1294 of *Lecture Notes in Computer Science*, pages 307–321. Springer-Verlag, 1997.
38. S. Micali and P. Rogaway. Secure computation (abstract). In *Advances in Cryptology—CRYPTO 1991*, pages 392–404. Springer-Verlag, 1992.
39. A. Nascimento and A. Winter. On the oblivious transfer capacity of noisy correlations. In *Proceedings of the IEEE International Symposium on Information Theory (ISIT '06)*, 2006.
40. B. Pfitzmann and M. Waidner. Information-theoretic pseudosignatures and Byzantine agreement for $t >= n/3$. Technical Report RZ 2882 (#90830), IBM Research, 1996.
41. M. O. Rabin. How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard Aiken Computation Laboratory, 1981.
42. T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC '89)*, pages 73–85, 1989.
43. R. L. Rivest. Unconditionally secure commitment and oblivious transfer schemes using private channels and a trusted initializer. Unpublished, 1999.
44. S. Wiesner. Conjugate coding. *SIGACT News*, 15(1):78–88, 1983.
45. A. Winter, A. C. A. Nascimento, and H. Imai. Commitment capacity of discrete memoryless channels. In *IMA Int. Conf.*, pages 35–51, 2003.
46. S. Wolf and J. Wullschleger. Zero-error information and applications in cryptography. In *Proceedings of 2004 IEEE Information Theory Workshop (ITW 2004)*, 2004.
47. S. Wolf and J. Wullschleger. New monotones and lower bounds in unconditional two-party computation. In *Advances in Cryptology—CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 467–477, 2005.
48. J. Wullschleger. Oblivious transfer amplification. http://arxiv.org/, cs.CR/0608076, 2006.
49. A. D. Wyner. The wire-tap channel. *Bell System Tech. Journal*, 54:1355–1387, 1975.
50. A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science (FOCS '82)*, pages 160–164, 1982.

Index

binary-symmetric erasure channel, 9
bit commitment, 2, 8
broadcast, 3, 11

coin toss, 2
commitment capacity, 8
common part, 4

extremal neighbors, 9

leftover hash lemma, 6

monotones, 7
most informative pair, 9
multi-party computation, 1

non-trivial channel, 9

oblivious transfer, 2, 9
one-way key agreement, 9

pseudo-signature schemes, 3, 11

real vs. ideal paradigm, 3

secure function evaluation, 1
sequential composition, 3
sufficient statistics, 5

unfair noisy channel, 12

wire-tap channel, 8