Extracting Correlations

Yuval Ishai*
Computer Science Dept.
Technion and UCLA
yuvali@cs.technion.ac.il

Eyal Kushilevitz†
Computer Science Dept.
Technion and UCLA
eyalk@cs.technion.ac.il

Rafail Ostrovsky‡
CS and Math Dept.
UCLA
rafail@cs.ucla.edu

Amit Sahai§
Computer Science Dept.
UCLA
sahai@cs.ucla.edu

Abstract— Motivated by applications in cryptography, we consider a generalization of randomness extraction and the related notion of privacy amplification to the case of two correlated sources. We introduce the notion of *correlation extractors*, which extract nearly perfect independent instances of a given joint distribution from imperfect, or "leaky," instances of the same distribution.

More concretely, suppose that Alice holds a and Bob holds b, where (a, b) are obtained by taking n independent samples from a joint distribution (X, Y) and letting a include all X instances and b include all Y instances. An adversary Eve obtains partial information about (a, b) by choosing a function L with output length t and learning $L(a, b)$.

The goal is to design a protocol between Alice and Bob which may use additional fresh randomness, such that for every L as above the following holds. In the end of the interaction, Alice outputs a' and Bob outputs b' such that (a', b') are statistically indistinguishable from m independent instances of (X, Y) even when conditioned on Eve's view, and *even when conditioned on the joint view of Eve together with either Alice or Bob*.

The standard questions of privacy amplification and randomness extraction correspond to the case where X and Y are identical random bits. In this work we address this question for other types of correlations. A central special case is that of *OT extractors*, which are correlation extractors for the correlation (X, Y) corresponding to the cryptographic primitive of oblivious transfer.

Our main result is that for any finite joint distribution (X, Y) there is an explicit correlation extractor which extracts $m = \Omega(n)$ instances using $O(n)$ bits of communication, even when $t = \Omega(n)$ bits of information can be leaked to Eve.

We present several applications which motivate the concept of correlation extractors and our main result. These include:

- Protecting certain cryptographic protocols against side-channel attacks.
- A protocol which realizes m instances of oblivious transfer by communicating only $O(m)$ bits. The security of the protocol relies on a number-theoretic intractability assumption.
- A *constant-rate* unconditionally secure construction of oblivious transfer (for semi-honest parties) from *any nontrivial channel*. This establishes constant-rate equivalence of any two nontrivial finite channels.

Keywords-randomness extractors; secure computation; oblivious transfer; noisy channels; leakage-resilient cryptography;

1. INTRODUCTION

Randomness extractors [48] convert dirty sources of randomness into clean sources of randomness. Motivated by cryptographic applications, we study a natural extension of randomness extraction to the case of two correlated sources.

Background. The problem we consider is best explained as an extension of *privacy amplification* [8], [7], the first application of randomness extractors in cryptography. Suppose that Alice and Bob initially share an n-bit secret random key k. This key is partially compromised by an adversary Eve, who learns partial information about the key by applying some "leakage function" $L : \{0, 1\}^n \to \{0, 1\}^t$ to k and learning its output $z = L(k)$. (This may capture information leakage during the process of generating the common key, or alternatively measurements taken by Eve while the key is stored for later use.) Now Alice and Bob would like to engage in a public discussion which results in agreement on a shorter m-bit key k' on which Eve has essentially no information. (Ideally, $m \approx n - t$.) This problem can be solved by having Alice communicate a short random seed r for a (strong) randomness extractor Ext, and defining the new key as $k' = \mathsf{Ext}(k, r)$. The extractor Ext should guarantee that for every admissible leakage function L, the final key k' is almost uniformly distributed when conditioned on Eve's view (z, r).

How to clean noise. Standard privacy amplification can be thought of as the question of building a clean secure communication channel from a leaky secure channel. The question we ask is whether this can be generalized to other types of channels. This question is motivated by the usefulness of noisy channels and correlated randomness in cryptography [50], [18], [5], [17]. Similarly to the use of a common source of secret randomness as a resource for secure *communication*, correlated randomness is useful as a resource for secure *computation* [54], [31].

To illustrate the general question of cleaning channels, consider the simple case of a binary symmetric channel

Work partially done at the Institute for Pure and Applied Mathematics (IPAM).

* Supported in part by ISF grant 1310/06, BSF grants 2004361, 2008411, and NSF grants 0830803, 0716835, 0456717, 0627781.

† Supported in part by ISF grant 1310/06 and BSF grant 2008411.

‡ Supported in part by IBM Faculty Award, Xerox Innovation Group Award, Okawa Award, NSF grants 0830803, 0716835, 0716389, 0430254, BSF grant 2008411, and U.C. MICRO grant.

§ Supported in part by BSF grants 2004361, 2008411, NSF grants 0830803, 0716389, 0627781, 0456717, a Cyber-TA grant, an equipment grant from Intel, and an Okawa Research award.

(BSC). Here Alice initially receives[1] a random n-bit string a and Bob receives an n-bit string b defined by $b = a \oplus e$, where e is an n-bit noise vector in which each bit takes (independently) the value 1 with some fixed probability $0 < p < 1/2$ and the value 0 with probability $1 - p$. Can we build a clean BSC from a leaky BSC which may reveal arbitrary t bits of information about (a, b)?

The latter question requires further explanation. A natural solution that comes to mind is to have Alice and Bob first use their leaky BSC to agree on a common random string k on which Eve has essentially no information (this can be done efficiently using standard privacy amplification techniques), and then use k to generate a pair of local outputs (a', b') whose joint distribution is statistically close to that of an m-bit BSC, for some $m < n$. This solution is acceptable if we are only concerned about protecting Alice and Bob against an *external* Eve. In the classical privacy amplification scenario, this is indeed the only relevant concern: if Alice and Bob should end up sharing the same secret key, there is no need to protect the privacy of Alice from Bob or vice versa. The current case of a BSC is qualitatively different in that an ideal BSC keeps secrets from both parties. Since the above solution allows each of Alice and Bob to fully learn (a', b'), it falls short of faithfully emulating an ideal BSC. Instead, we would like the outputs to be distributed correctly not only from the point of view of an external Eve, but also from the point of view of Alice or Bob.

Extracting correlations. The above example leads us to our new notion of *correlation extractors*. Given a joint distribution (X, Y) which specifies an atomic correlation, or a "channel," the goal is to realize m clean independent instances of (X, Y) given n leaky instances of (X, Y). The original privacy amplification scenario corresponds to the case where X and Y are identical random bits. The above example of cleaning a BSC corresponds to the case where X and Y are random bits which differ with probability p.

More precisely, suppose that Alice holds a and Bob holds b, where (a, b) are obtained by taking n independent samples from (X, Y) and letting a include all X instances and b include all Y instances. An adversary Eve takes a partial measurement of Alice and Bob's inputs by applying a global leakage function L with output length t to (a, b) and learning its output $z = L(a, b)$.[2] The goal is to design an interactive protocol between Alice and Bob which uses a public communication channel and a small amount of fresh randomness, such that for every L as above the following holds. In the end of the protocol, Alice outputs a' and Bob outputs b' such that (a', b') are statistically indistinguishable from m independent instances of (X, Y) even when conditioned on Eve's view, and *even when conditioned on the joint view of Eve together with either Alice or Bob*. We refer to such a protocol as a correlation extractor for (X, Y).

The latter indistinguishability requirement can be formalized as follows. Suppose Eve corrupts Alice (the cases of a corrupted Bob or an external Eve are similar). Let the random variable V capture the view of Eve (including Alice's input a, the leakage z and messages exchanged between Alice and Bob), and (A', B') capture the outputs of Alice and Bob. Let $(\hat{A}', \hat{B}')$ denote the ideal output distribution $(X, Y)^m$ and let $\hat{B}'_{a'}$ denote a fresh sample from the conditional distribution $[\hat{B}' \mid \hat{A}' = a']$; that is, the distribution of Bob's ideal output conditioned on Alice's ideal output being a'. Using this notation, we say that the protocol ϵ-emulates $(X, Y)^m$ if the statistical distance between (V, B') and $(V, \hat{B}'_{A'})$ is at most ϵ.

The notion of correlation extractors can be elegantly captured using the standard simulation-based paradigm for defining secure computation [31], [30], [11], [12]. In the terminology of secure computation, an (n, m, t, ϵ) correlation extractor for (X, Y) is a two-party protocol which ϵ-securely realizes $(X, Y)^m$ given a single call to a "t-leaky oracle" for $(X, Y)^n$. This formulation is not only more compact, but it also facilitates (via suitable composition theorems) a modular construction of correlation extractors as well as their use as building blocks in other protocols.

We note that in contrast to standard randomness extraction, the problem of extracting general correlations with good parameters seems challenging also when allowing *non-explicit*, or even heuristic, constructions. The difficulty arises from the need to resolve the conflict between the structure of the correlated outputs and the secrecy requirements. This calls for the use of techniques from the domain of secure computation, which we indeed employ in our constructions. Another difference is that we will not be concerned with minimizing the amount of fresh randomness used by correlation extractors. The amount of randomness can be reduced by locally applying standard randomness extractors to portions of the sources.

[1] The standard notion of channels from information theory does not specify a distribution over Alice's inputs, and specifies only the conditional distributions $p_{y|x}$ of each received symbol y given each possible symbol x sent by Alice. Our formulation of a channel treats Alice and Bob symmetrically by specifying a joint distribution (X, Y) for their inputs. While the latter formulation is more convenient for the purposes of this work, it is possible to formulate our results using the former, more conventional, notion of a channel.

[2] Our main definition of correlation extractors follows the basic privacy amplification scenario by considering the correlation source to be perfectly distributed but leaky. However, our results apply (with the same asymptotic parameters) also to a stronger notion of correlation extractors which directly generalizes the traditional notion of strong randomness extractors. The stronger definition allows (a, b) to be taken from any joint distribution (A, B) such that $\Pr[(A, B) = (a, b)] \leq 2^t \cdot \Pr[(X, Y)^n = (a, b)]$ for all (a, b). The latter notion of an imperfect (X, Y)-source generalizes the standard notion of a weak n-bit source with min-entropy $n - t$ [48].

OT extractors. Our main applications of correlation extractors will use the random OT correlation, corresponding to a random instance of oblivious transfer [50], [25] in which the receiver Bob obtains one of two bits held by the sender Alice. That is, $X = (X_0, X_1)$ is uniformly random over $\{0,1\}^2$ and $Y = (b, X_b)$ for a random bit b. We refer to such a correlation extractor as an *OT extractor*. It is instructive to compare OT extractors with previous notions from the literature. When the leakage function L can only contain *physical bits* of its input, extracting OTs coincides with the previously studied goal of *combining* OTs [36], namely generating secure instances of OT from multiple candidates of which a bounded number may be faulty and leak information. Thus, OT extractors can be viewed as a common generalization of OT combiners and standard randomness extractors, which in turn both generalize the notion of extractors for bit-fixing sources [15].

1.1. Our Results

We initiate the study of correlation extractors, focusing on the case where the adversary Eve is "semi-honest," namely it can observe all information available to corrupted parties but cannot modify the messages they send to each other.

Our main result is a construction of a "constant-rate" correlation extractor for any distribution (X, Y) with constant-size support and rational probabilities. More precisely, for any such (X, Y) there exist constants $c_1, c_2, c_3 > 0$ and $c > 1$ for which there is an explicit constant-round (n, m, t, ϵ) correlation extractor with $m(n) = c_1 n$, $t(n) = c_2 n$, $\epsilon(n) = 2^{-c_3 n}$ and cn bits of communication.

Applications. We motivate the notion of correlation extractors and our main result by presenting several applications.

LEAKAGE-RESILIENT CRYPTOGRAPHY. As already noted, correlation extractors can be applied for obtaining clean versions of imperfect correlated random sources which can be useful for cryptographic applications. For instance, when basing unconditionally secure cryptographic protocols on a physical BSC, correlation extractors can be used to accommodate an imperfect or leaky implementation of the BSC. As another example, one can think of using an expensive process of generating many precomputed random OTs for the purpose of a very fast "non-cryptographic" secure computation which should be done in the future.[3] To eliminate the effect of leakage which may have occurred during the generation and storage of the precomputed OTs, an OT extractor can be applied shortly before the OTs are consumed. Similarly to the case of privacy amplification, it is crucial that correlation extraction be done *strictly after* leakage occurs, so that the fresh randomness used by the extractor is independent of the leakage.

CONSTANT-RATE SECURE COMPUTATION. Randomness extractors are useful not only in the realm of "information-theoretic" cryptography, but also in the context of complexity based cryptography. We show that this is also the case for correlation extractors by presenting a somewhat unexpected application of OT extractors to communication-efficient secure computation in the plain model. Here the leakage is not caused by an adversary or a physical process, but rather by a natural, computationally secure protocol. We first give some relevant background.

Almost all known techniques for secure two-party computation require poly(k) bits of communication for each gate of a boolean circuit being computed, where k is a cryptographic security parameter. One notable exception is a protocol based on Gentry's fully homomorphic encryption scheme [28], which can securely compute a polynomially large circuit with input and output of length n using only $n \cdot \text{poly}(k)$ communication. However, in the common scenario of evaluating a circuit of size $O(n)$ with n outputs, as in the case of performing n *finite* secure computation tasks, the overhead per gate is still poly(k). A general protocol with $O(1)$ (amortized) communication and computation per gate was proposed in [39]. This protocol relies on a pseudorandom generator with *polynomial stretch* in NC^0. While there are heuristic candidates for such pseudorandom generators, their existence is not known to follow from any well-studied assumption.

We apply a constant-rate OT extractor towards realizing n instances of OT with only $O(n)$ bits of communication. This implies a general secure two-party protocol for any boolean circuit of size n with $O(n)$ bits of communication, even in the case of security against *malicious* parties. The security of the protocol relies on a variant of the Φ-Hiding Assumption [10], [29], a fairly well-studied number theoretic intractability assumption.

The high level idea is the following. A 1-out-of-d PIR protocol [16], [43] is defined similarly to 1-out-of-d OT except that the receiver is not explicitly prevented from learning additional information about the sender's d-bit input. Instead, the communication complexity from the sender to the receiver is required to be smaller than d. The key observation is that PIR can be viewed as "leaky OT," where the amount of leakage to the receiver is bounded by the communication complexity. Let PIR_d^n denote a generalized version of PIR which supports n instances of 1-out-of-d selection. We apply our OT extractor to extract from a single invocation of *any* PIR_d^n protocol (with a sufficiently

[3] The OT generation can be done with unconditional security by using a trusted dealer, a multi-party protocol with an honest majority [6], [13], noisy channels [18], [17], [20], or bounded storage assumptions [45], [9], [22]. When using a *computationally secure* protocol for generating the OTs, one should guarantee that there is enough pseudo-entropy left in the secrets given the leakage. This is always the case if the parties erase everything except the outputs of the protocol before leakage occurs.

good[4] communication rate) $m = \Omega(n)$ instances of OT. In contrast to the general transformation from PIR to OT in [21], our procedure can be applied also with a *constant* d (with $2^{-\Omega(n)}$ error). Finally, we use a generalized version of the PIR protocols of [10], [29] to efficiently realize PIR_d^n. One formulation of our transformation from PIR to OT is:

Theorem 1.1: (informal) Let Π be any protocol realizing PIR_d^n with communication complexity $O(n \log d)$. There exists a protocol Π', which makes only black-box use of Π, that realizes m parallel bit OTs with communication $O(m)$.

Corollary 1.2: (informal) Assuming the General Decision Subgroup Problem / General Φ-Hiding Assumption (see full version), there exists a computationally secure protocol realizing m parallel bit OTs with communication $O(m)$.

CONSTANT-RATE REDUCTIONS BETWEEN CHANNELS. The above result implies that when settling for *computational* security, any finite channel can be realized *from scratch* with a constant communication rate. Using our OT extractor we show that any pair of such non-trivial channels (that imply OT) can be *unconditionally* reduced to each other via a *constant-rate* reduction. (Here security is in the semi-honest model.) The same argument also implies that any statistically-secure construction of OT from an arbitrary channel g (not necessarily of constant description size) can be turned into a constant-rate protocol which realizes n instances of OT using only $O(n)$ calls to g. This can be used to translate previous feasibility results on constructing OT from a variety of weak channels (including "unfair" channels [20] and Gaussian channels [52]) into constant-rate constructions.[5] The original constructions require $\mathrm{poly}(\kappa)$ invocations of g for each OT. A similar result for the case where g is a BSC was obtained in [35] using OT combiners, which do not seem sufficient for the more general goal.

1.2. Overview of Techniques

We now give a high level overview of our main result. For non-trivial instances of (X, Y) for which correlation extraction is not implied by standard privacy amplification, our construction proceeds roughly as follows. The first step is to convert the n leaky instances of (X, Y) into $n' = \Omega(n)$ leaky instances of δ-OT, where δ-OT is an "approximate" version of OT which may have a small error probability and leak a small amount of information. (The error comes from the imperfect implementation of the OT, and is in addition to the t bits of leakage we started with.) This step relies on previous results from the literature [42], [19]. The second and main step is to apply a (robust form of) constant-rate OT extractor to extract $m' = \Omega(n')$ clean OTs from the n' leaky δ-OTs. This constant-rate OT extractor is our main technical contribution. The final step is to apply OT-based secure computation protocols [41] for realizing $m = \Omega(m')$ good instances of (X, Y) from the m' OTs. This step requires the probabilities in (X, Y) to be rational.

The main building block, a constant-rate OT extractor, uses a combination of secure computation and randomness extraction techniques. First, using a symmetry property of OT [51], we can reduce the problem to protecting Bob against leakage to Alice. To this end, we introduce the following notion of *ε-biased secure computation*. Suppose π is a two-party protocol which securely computes a function f by making n calls to an OT oracle in which Bob acts as the receiver. We say that π is ε-biased if regardless of Bob's input, the distribution of the n selections Bob feeds into the OT oracle when conditioned on Alice's view is ε-biased in the sense of [46]. Using the standard reduction from OT to random OT [5], such an ε-biased protocol π can be transformed into a protocol π' for f which makes n calls to a *random OT* oracle and remains secure even if the oracle can leak $t = \Omega(\log(1/\varepsilon))$ bits to Alice.

The technical core of our OT extractor is an efficient implementation of an ε-biased[6] protocol π for the function f which realizes m instances of OT. This implementation relies on certain "nice" families of error-correcting codes [14] and can be seen as carefully instantiating and modifying a general approach from [35] for constructing OT *combiners* from secure computation protocols.

We note that OT extractors with poor asymptotic parameters could be obtained in the following easier way. First use blocks of random bit-OTs obtained from the leaky OT source to implement random string-OTs for strings of size κ, where κ is a statistical security parameter. Then make a local use of a standard randomness extractor to eliminate the receiver's partial information about each string which it should not learn. Finally, "reverse" the OTs [51] and repeat the above process. The rate of OT extractors obtained via this approach must deteriorate polynomially with κ, which makes them useless for most of our applications.

1.3. Related Work

Correlation extractors are very different from previous notions of distributed randomness extraction considered in the literature. Research on privacy amplification and related questions also considered a more general variant in which Alice and Bob initially hold correlated (but not necessarily identical) inputs [53], [8], [44], [24]. However, as in the standard case of privacy amplification, the goal of Alice and Bob is to agree on a common secret key, and so the only privacy concern is with respect to an external Eve. A recent

[4] Concretely, we need the communication complexity to be smaller than δdn, where δ is the fractional leakage tolerated by our OT extractor. For this reason we will need to use a *large* constant d.

[5] We stress that by "constant-rate OT" we refer to implementing a sequence of n independent instances of bit-OT using $O(n)$ communication. This is very different from the related (but easier) goal of obtaining a single instance of OT on n-bit strings with $O(n)$ communication [47].

[6] Actually we will not quite achieve this notion, but rather have the property that Bob's OT selections will be several copies of a sample drawn from an ε-biased distribution.

line of work on network extraction protocols [23], [34], [40] differs from our setting both in its typical goal (obtaining private and *independent* randomness) and the assumptions (parties are given imperfect but *independent* sources, but no perfect source of fresh randomness is available).

Finally, as discussed above, correlation extractors are related to and partially motivated by a recent line of work on protecting cryptographic protocols against side-channel attacks (see [1] and references therein). We believe that correlation extractors may serve as useful building blocks for future applications of this type.

Organization: Sections 2 and 3 contain some preliminaries and definitions. Section 4, the main technical section, describes the construction of a constant-rate OT extractor. Section 5 applies this OT extractor towards the construction of general correlation extractors. For lack of space, the application to constant-rate secure computation as well as many details and proofs are deferred to the full version.

2. Preliminaries

We use U_n to denote a random variable uniformly distributed over $\{0,1\}^n$. The *min-entropy* of a random variable Y is defined as $\mathrm{H}_\infty(Y) \overset{\text{def}}{=} \min_y \log(\frac{1}{\Pr[Y=y]})$. The *statistical distance* between distributions Y and Y', denoted $\mathrm{SD}(Y,Y')$, is defined as the maximum, over all functions A, of the *distinguishing advantage* $|\Pr[A(Y)=1]-\Pr[A(Y')=1]|$.

Extraction via ε-biased masking. Let $\mathbb{F}_2$ denote the binary field. We say that a distribution X on $\mathbb{F}_2^n$ is *ε-biased* [46] if for every non-constant linear function $D:\mathbb{F}_2^n \to \mathbb{F}_2$ it holds that $|\Pr[D(X)=1]-\frac{1}{2}| \leq \varepsilon$. Such ε-biased distributions can be used to extract randomness:

Lemma 2.1 ([2], [33]): Let X be an ε-biased distribution over $\mathbb{F}_2^n$, and let Y be an independent distribution over $\mathbb{F}_2^n$ such that $H_\infty(Y) \geq n-t$. Then $\mathrm{SD}((X\oplus Y),U_n) \leq \varepsilon \cdot 2^{(t-1)/2}$.

Secure computation. Our main definition of correlation extractors uses the standard terminology of secure computation (cf., [11], [12], [30]). We briefly explain this terminology for self-containment. In this work we restrict the attention to the two-party case, and mostly to the case of *unconditionally* secure protocols against a *semi-honest* (or "honest-but-curious") adversary Eve, who can learn all information available to corrupted parties but does not modify messages they send. A secure computation task is specified by a *functionality* f, a (possibly randomized) mapping from a pair of inputs (a,b) to a pair of outputs (c,d). To capture leakage, we use a standard extension of the notion of a functionality by allowing the functionality to also take an input from and deliver an output to the adversary.

We say that a protocol π realizes f with (statistical) ϵ-security if for any adversary Eve who passively corrupts at most[7] one party there exists a simulator Eve' such that for any choice of inputs (a,b) for Alice and Bob, the statistical distance between the following distributions is at most ϵ:

- $\mathsf{Real}(a,b)$ includes the view of Eve when π is run on inputs (a,b), together with the *output* of the uncorrupted parties.
- $\mathsf{Ideal}(a,b)$ is defined similarly to $\mathsf{Real}(a,b)$, except that instead of invoking π the parties (and possibly also Eve') send their inputs to a trusted party who evaluates f and correctly delivers the outputs. The output of $\mathsf{Ideal}(a,b)$ includes the output of Eve' together with the output of the uncorrupted parties.

We do not require the simulator Eve' to be efficient. However, in the semi-honest model, the existence of an inefficient simulator implies the existence of an efficient simulator for all "efficiently invertible" functionalities, and in particular the ones considered here.

Secure reductions. We will make use of the following standard notion of secure reductions. For any two-party functionalities f and g, we say that π is an *ϵ-secure reduction from f to g* if π realizes f with ϵ-security in the *g-hybrid model*, in which π may instruct the parties to invoke a trusted party (or oracle) computing g. This affects $\mathsf{Real}(a,b)$, and requires the simulator (which still only has access to f) to simulate the messages in $\mathsf{Real}(a,b)$ received from the g-oracle. We will typically consider reductions which only make a single call to g. Secure reductions can be used for the modular design of secure protocols via the following composition theorem [11], [30]: if π is a secure reduction from f to g and π_g securely realizes g, then a protocol π_f which securely realizes f without a g-oracle can be obtained from π by using π_g to emulate each oracle call to g in π.

Oblivious Transfer (OT). We will use OT to denote the standard bit-OT functionality, which takes a pair of bits (x_0,x_1) from Alice (the "sender") and a selection bit b from Bob (the "receiver"), and returns x_b to Bob. We denote by ROT the random OT functionality which outputs to the receiver a pair of random bits (x_0,x_1) and to the sender (b,x_b) for a random bit b. These OT variants are reducible to each other via simple and efficient perfectly secure reductions. For instance, given a single "precomputed" instance of ROT with sender outputs (x_0,x_1) and receiver outputs (b,x_b) one can realize a single instance of OT with sender inputs (s_0,s_1) and receiver input r by letting the receiver send $d=r\oplus b$ and the sender send $(s_0\oplus x_d, s_1\oplus x_{1-d})$ [5], [4]. Another useful reduction is between ROT and ROT where the roles of the sender and the receiver are reversed. Such a reduction requires only local computation on the output of ROT [51].

[7] Even if Eve does not corrupt any party, her view includes messages exchanged between Alice and Bob.

3. Leaky Oracles and Correlation Extractors

A formal definition of correlation extractors in terms of conditional distributions is outlined in the introduction. Towards a modular construction of correlation extractors and their use in cryptographic applications, it will be convenient to define and analyze them within the general framework of secure computation. In these terms, a correlation extractor for (X, Y) is an ϵ-secure protocol for realizing $(X, Y)^m$ given a "t-leaky oracle" for $(X, Y)^n$.

We start by defining a notion of a leaky functionality. Informally, a t-leaky version of a functionality f is the functionality obtained from f by allowing the adversary to specify a leakage function L with output length t, and learn the output of L applied to the inputs and randomness of f. While the following definition refers to general functionalities (which may take inputs from the parties), we will typically use it in the case of randomized functionalities that generate correlated randomness without taking any inputs.

Definition 3.1 (Leaky functionality): Let f be a (possibly randomized) two-party functionality, and let t be a leakage parameter. The t-leaky version of f, denoted $f^{[t]}$, is the functionality defined by the following process:

- Receive a from Alice and b from Bob. Pick randomness r for f and let $(x, y) = f((a, b), r)$.
- Receive from the adversary a description of a (possibly randomized) circuit representing a leakage function $L : \{0,1\}^* \to \{0,1\}^t$ and locally compute $z = L(a, b, r)$.
- Deliver x to Alice, y to Bob, and z to the adversary.

Note that since the leakage depends on both inputs as well as the randomness for f, our basic notion is equivalent to the seemingly stronger notion which allows the adversary to pick the leakage function only after observing an output from f. The following standard lemma says that except with negligible probability, t bits of leakage cannot decrease the entropy of a source by much more than t.

Lemma 3.2: For any distribution P, a (possibly randomized) leakage function $L : \{0,1\}^* \to \{0,1\}^t$, and $\kappa > 0$ the following holds. Except with at most $2^{-\kappa}$ probability over the choice of p from P, we have

$$H_\infty[P \mid L(P) = L(p)] \geq H_\infty(P) - t - \kappa.$$

The following lemma will be useful to argue that if some protocol π securely realizes a functionality f using an ideal implementation of g, then the same protocol will still realize a t-leaky relaxation of f if the perfect oracle to g is replaced by a t-leaky oracle.

Lemma 3.3: Let π be a perfectly secure (resp., statistically ϵ-secure) reduction from f to g which makes a single call to g. Then, π is also a perfectly secure (resp., statistically ϵ-secure) reduction from $f^{[t]}$ to $g^{[t]}$.

We note that Lemma 3.3 does *not* hold for computationally secure reductions. Also, the leakage function used by the simulator in the reduction from $f^{[t]}$ to $g^{[t]}$ will not generally be efficient. However, it is guaranteed to be efficient whenever π satisfies a certain "inverse sampling" property which our protocols satisfy. See full version for further discussion.

We are now ready to define correlation extractors and the special case of OT extractors. We naturally view a joint distribution (X, Y) as a randomized functionality (with no input) and let $(X, Y)^n$ denote the functionality which delivers n independent instances of (X, Y).

Definition 3.4 (Correlation extractor): Let (X, Y) be a finite probability distribution. An (n, m, t, ϵ) *correlation extractor for* (X, Y) is an ϵ-secure realization of $(X, Y)^m$ in a hybrid model that allows a single call to $((X, Y)^n)^{[t]}$.

Definition 3.5 (OT extractor): An *OT extractor* is a correlation extractor for the concrete distribution (X, Y) defined by the outputs of the random OT functionality ROT. That is, $X = (x_0, x_1)$ and $Y = (b, x_b)$ for uniformly random and independent bits x_0, x_1, b.

We will also refer to a more liberal notion of leaky oracles, which only restricts the entropy loss given the leakage as opposed to the number of bits being leaked. This corresponds to the stronger notion of correlation extractors discussed in Footnote 2. Finally, we will sometimes also allow the adversary to modify a bounded fraction of the bits delivered by the oracle to uncorrupted parties. This generalizes the notion of error-tolerant combiners [49]. Our constructions of correlation extractors are robust to both of these relaxations.

4. Constant-Rate OT Extractor

The key ingredient in our main result is a construction of an (n, m, t, ϵ) OT extractor with $m = \Omega(n)$, $t = \Omega(n)$, $\epsilon(n) = 2^{-\Omega(n)}$ and $O(n)$ bits of communication. We refer the reader to Section 1.2 for a high level overview of our technical approach.

4.1. Building Blocks

Our construction relies on a family of linear error correcting codes with the following special properties.

Claim 4.1 (Implicit in [14]): There exists a finite field $\mathbb{F}$ of characteristic 2 and an efficiently constructible family of linear error-correcting codes $C_K : \mathbb{F}^K \to \mathbb{F}^{N_K}$ with the following properties: (1) $N_K = O(K)$; (2) The dual distance of C_K is $\delta_K = \Omega(K)$; (3) The linear code C'_K spanned by all pointwise-products of pairs of codewords in C_K has minimal distance $\Delta_K = \Omega(K)$ and supports efficient decoding of up to $\mu_K = \Omega(K)$ errors. (The pointwise product of $(c_1, \ldots, c_N)$ and $(c'_1, \ldots, c'_N)$ is $(c_1c'_1, \ldots, c_Nc'_N)$.)

The last property implies that C_K also has minimal distance $\Omega(K)$. The efficient decoding property of C'_K will not be used in the current section.

The above properties of algebraic-geometric codes were used by Chen and Cramer [14] to construct strongly multiplicative secret sharing schemes over constant-size fields.

A family of codes C_K as above can be obtained from the construction of Garcia and Stichtenoth [27].

It is instructive to note that Reed-Solomon codes satisfy all the properties stated in Claim 4.1 except for requiring the size of the field $\mathbb{F}$ to grow linearly with K. One could use Reed-Solomon codes in our constructions (instead of algebraic geometric codes) at the cost of a polylogarithmic deterioration of the parameters.

Randomized encodings. A second building block is an unconditionally secure two-party computation protocol which only involves parallel invocations of an OT oracle and no additional interaction. For instance, we can use an information-theoretic variant of Yao's garbled circuit construction [54], [37] or Kilian's protocol [41]. More generally, such protocols can be obtained from the following variant of the notion of randomized encoding of functions from [3].

Definition 4.2 (Decomposable randomized encoding): Let $f : \{0,1\}^n \to \{0,1\}^m$ be a function. We say that $\hat{f}(x,r)$ is a *randomized encoding* of $f(x)$ if the following requirements hold. (1) Given $\hat{f}(x,r)$ it is possible to efficiently decode $f(x)$. (2) Given $f(x)$, it is possible to efficiently sample from the distribution of $\hat{f}(x,r)$ induced by a uniform choice of r. We say that $\hat{f}$ is *decomposable* if every output bit of $\hat{f}$ depends on at most a single bit of x (but can depend arbitrarily on r). Thus we can write $\hat{f}(x,r) = (\hat{f}_1(x_1,r), \ldots, \hat{f}_n(x_n,r))$.

Several constructions of decomposable randomized encodings, with incomparable efficiency features, are implicit in the secure computation literature (e.g., [41], [26], [37]). These yield the following:

Lemma 4.3: Let $f : \{0,1\}^n \to \{0,1\}^m$ be a function in which each output bit depends on at most c input bits. Then, f admits a decomposable randomized encoding $\hat{f}(x,r)$ with output length $2^{O(c)} \cdot m$. Furthermore, if each input bit of f influences at most d output bits, then each input bit x_i of $\hat{f}$ influences at most $d \cdot 2^{O(c)}$ output bits and $\hat{f}$ has a decomposition $\hat{f}(x,r) = (\hat{f}_1(x_1,r), \ldots, \hat{f}_n(x_n,r))$ such that the output length of each $\hat{f}_i(x_i,r)$ is at most $d \cdot 2^{O(c)}$.

We note that we will only apply the above lemma to "structured" functions f for which the $2^{O(c)}$ term can be made polynomial in c. However, this will only affect the parameters of our correlation extractors by constant factors, which we do not attempt to optimize in this work.

4.2. Main Construction

Recall that an OT extractor securely realizes ROT^m (namely, m independent instances of the randomized bit-OT functionality) using a single call to a t-leaky oracle for ROT^n. Note that it suffices to realize the deterministic functionality OT^m; the ROT^m functionality can then be obtained by running the OT^m protocol on random inputs, and letting each party in the ROT^m protocol output its input and output in the OT^m protocol.

We start by describing a protocol which realizes OT^m from a leaky ROT^n oracle, where we assume that the ROT oracle leaks t bits of information to the sender only. The case of symmetric leakage will be addressed in Section 4.3.

Following the invocation of the leaky ROT^n oracle, the parties have the following information.

RECEIVER'S INFORMATION. This includes selection bits $(r_1, \ldots, r_m)$ for the OT^m protocol, random precomputed selection bits $(p_1, \ldots, p_n)$, and corresponding sender bits $(q_{1,p_1}, \ldots, q_{n,p_n})$ obtained from the ROT^n oracle.

SENDER'S INFORMATION. This includes pairs of sender bits $((s_{1,0}, s_{1,1}), \ldots, (s_{m,0}, s_{m,1}))$ for the OT^m protocol, random pairs of precomputed bits $((q_{1,0}, q_{1,1}), \ldots, (q_{n,0}, q_{n,1}))$ obtained from the ROT^n oracle, and the output of a leakage function L on the outputs of the ROT^n oracle.

Informal description. We now give a high level overview of the main protocol. The receiver will use the code family C_K promised by Claim 4.1 with $K = O(m)$ to locally secret-share its input bits $(r_1, \ldots, r_m)$ for the OT^m protocol. The large dual distance of C_K and the assumption that $\mathbb{F}$ has characteristic 2 guarantee that the bits of the shares are $\Omega(K)$-wise independent. However, to eliminate the effect of the leakage, we will need to make these bits ε-biased with $\varepsilon = 2^{-\Omega(K)}$. This is done by using an additional step of non-linear local randomization, where the receiver replaces each bit in each share by a random triple of bits whose majority is equal to the original bit. The encoded shares can be shown to satisfy the required ε-bias property. Next, the sender uses the code C_K to locally encode each of his two input vectors $(s_{1,0}, \ldots, s_{m,0})$ and $(s_{1,1}, \ldots, s_{m,1})$. For the purpose of preventing the receiver from obtaining additional information, the sender will also need to secret-share a zero-vector using the code C'_K.

The large minimal distance of C'_K allows us to define an NC^0 function f on the local vectors generated by the two parties (in fact, f is a disjoint union of N_K functions on constant-size inputs), such that the output of f reveals the correct receiver's outputs s_{i,r_i}, $1 \le i \le m$, but reveals no information about the unselected bits $s_{i,1-r_i}$. To reveal the output of f to the receiver while hiding the receiver's inputs from the sender, we use a non-interactive OT-based protocol for f obtained via a decomposable randomized encoding $\hat{f}$ (see Lemma 4.3). To realize the OTs consumed by this sub-protocol, we use the precomputed outputs of the leaky ROT oracle. The information revealed to the sender in this sub-protocol is (essentially) the exclusive-or of the encoded input of the receiver and the ROT outputs $p = (p_1, \ldots, p_n)$. Since the former is ε-biased and by Lemma 3.2 the latter has high min-entropy even when conditioned on the leakage, Lemma 2.1 guarantees that the exclusive-or of the two will keep the receiver's selections r_i hidden from the sender. The analysis will be slightly complicated by the fact that we will actually need to replicate the receiver's encoded input a constant number of times before masking it with p. However, the same conclusion applies in this case as well.

Formal description. We now formally specify the protocol up to the setting of the parameters.

Protocol 1: Protecting the receiver against leakage.

1. LOCAL ENCODING. Let $N = N_K$. The receiver secret-shares its input $r = (r_1, \ldots, r_m)$ by picking a random codeword in C_K subject to the restriction that the first m coordinates agree with r, and discarding the first m coordinates of this codeword. Let $\hat{r} \in \mathbb{F}^{N-m}$ denote the resulting share-vector. Now the receiver applies a second level of randomized encoding to $\hat{r}$ by replacing each bit b in the binary representation of $\hat{r}$ with a random triple of bits whose majority is equal to b. Let $\tilde{r}$ be the resulting $(N-m)$-tuple of binary strings of length $3\log_2|\mathbb{F}|$.

The sender parses its inputs as $s^0 = (s_{1,0}, \ldots, s_{m,0})$ and $s^1 = (s_{1,1}, \ldots, s_{m,1})$. For $b = 0, 1$ it picks an arbitrary (not necessarily random) codeword in C_K whose first m coordinates agree with s^b, and lets $\hat{s}^b$ be the length $(N-m)$ vector obtained by discarding the first m coordinates. In addition, the sender picks a random codeword in C'_K whose first m coordinates are 0, and lets $\hat{z}$ be the vector obtained by discarding the first m coordinates.

2. INTERACTION. The receiver and the sender engage in a two-message interaction which reveals to the receiver the function $f(\tilde{r}, \hat{s}^0, \hat{s}^1, \hat{z}) = (1-\hat{r})\hat{s}^0 + \hat{r}\hat{s}^1 + \hat{z}$ (with point-wise multiplications and additions). That is, the function f outputs $N-m$ field elements where the i-th output of f is $(1-\hat{r}_i)\hat{s}^0_i + \hat{r}_i\hat{s}^1_i + \hat{z}_i$, and where $\hat{r}_i$ is obtained from $\tilde{r}_i$ by applying the majority function to each 3-bit block in $\hat{r}_i$. Note that since $\mathbb{F}$ is of constant size, f is in NC^0.

This step uses an OT-based protocol for secure two-party computation, which can be obtained from a decomposable randomized encoding. Let $\hat{f}((\tilde{r}, \hat{s}^0, \hat{s}^1, \hat{z}), \rho) = (\hat{f}_1(\tilde{r}_1, \rho), \hat{f}_2(\tilde{r}_2, \rho), \ldots, \hat{f}_\ell(\tilde{r}_\ell, \rho), \hat{f}_S((\hat{s}^0, \hat{s}^1, \hat{z}), \rho))$ be a decomposable randomized encoding of f (with randomness ρ), as promised by Lemma 4.3, where ℓ denotes the bit-length of $\tilde{r}$ and $\hat{f}_S$ denotes the concatenation of all $\hat{f}_i$ where i is an input of the sender. The sender uniformly picks the randomness ρ for $\hat{f}$ and lets $\sigma^{i,b} = \hat{f}_i(b, \rho)$ for $1 \leq i \leq \ell$ and $b = 0, 1$. Since f also has constant *input locality*, the size of all $\sigma^{i,b}$ can be bounded by some constant u. Using padding we may assume that $|\sigma^{i,b}| = u$ for all i, b.

The sender and the receiver use the precomputed random OTs to transmit to the receiver the output of $\hat{f}$ while only revealing to the sender the exclusive-or of a string containing u repetitions of $\tilde{r}$ with the precomputed selections p. Concretely:

(a) The receiver sends a binary string α of length $u\ell$ obtained by taking the exclusive-or of the precomputed receiver's selections p with the string containing u repetitions of each bit of $\tilde{r}$. That is, $\alpha_{(i-1)u+j} = \tilde{r}_i \oplus p_{(i-1)u+j}$ for $1 \leq i \leq \ell$ and $1 \leq j \leq u$. Note that here we assume that $n = u\ell = O(m)$.

(b) The sender views each bit α_k as a specification of whether to swap the bits in the precomputed pair $(q_{k,0}, q_{k,1})$ before masking them with corresponding bits from $(\sigma^{i,0}, \sigma^{i,1})$. Concretely, it sends to the receiver 2ℓ strings $\beta^{i,b}$ of length u, where for each $1 \leq i \leq \ell$, $b = 0, 1$, and $1 \leq j \leq u$ it lets $\beta^{i,b}_j = \sigma^{i,b}_j \oplus q_{(i-1)u+j\,,\, b\oplus\alpha_{(i-1)u+j}}$. The sender additionally sends a separate message $\beta_S = \hat{f}_S((\hat{s}^0, \hat{s}^1, \hat{z}), \rho)$.

3. DECODING THE OUTPUTS. The receiver uses the messages $\beta^{i,b}$ and β_S together with its precomputed received bits $(q_{1,p_1}, \ldots, q_{n,p_n})$ to recover its output $(s_{1,r_1}, \ldots, s_{m,r_m})$. This is done via the following decoding steps:

(a) For each $1 \leq i \leq \ell$, the receiver recovers $\sigma^{i,\tilde{r}_i} = \hat{f}_i(\tilde{r}_i, \rho)$ by taking the exclusive-or of $\beta^{i,\tilde{r}_i}$ with the corresponding block of received precomputed bits. That is, for each $1 \leq j \leq u$ we have $\sigma^{i,\tilde{r}_i}_j = \beta^{i,\tilde{r}_i}_j \oplus q_{(i-1)u+j,p_i}$.

(b) The receiver applies the decoder of $\hat{f}$ to the strings $\sigma^{i,\tilde{r}_i}$ and β_S, obtaining $v = f(\tilde{r}, \hat{s}^0, \hat{s}^1, \hat{z})$, where $v \in \mathbb{F}^{N-m}$.

(c) The receiver now finds a codeword v' of C'_K which agrees with v in its last $N-m$ coordinates, and outputs the first m coordinates of v'.

Proposition 4.4: There exist constants $c_1, c_2, c_3 > 0, c > 1$, and a corresponding choice of parameters for Protocol 1 such that the following holds for all sufficiently large n. The protocol correctly realizes $m = c_1 n$ instances of OT. It is perfectly secure against a corrupted (semi-honest) receiver when given a perfect ROT^n oracle. It is $2^{-c_2 n}$-secure against a corrupted (semi-honest) sender even when given a leaky $(ROT^n)^{[t]}$ oracle, where $t = c_3 n$. The protocol communicates at most cn bits.

4.3. The Final OT Extractor

Recall that so far we only addressed the case where leakage is available to the sender. To handle an ROT^n oracle which can also leak to the receiver, we exploit the fact that OTs can be easily "reversed" [51]. (A similar use of OT reversal was made in [38].) For the purpose of the following description, we refer to the two parties as Alice and Bob. The final protocol proceeds as follows.

Protocol 2: Constant-rate OT Extractor with sender Alice and receiver Bob.

- Apply Protocol 1 on random inputs with Alice as the sender (with the parameters guaranteed by Proposition 4.4).
- Apply the local OT reversal procedure, relabeling the outputs of the previous step and viewing the relabeled outputs as outputs of an ROT^m oracle in which Bob is the sender.

- Use the outputs of the previous step to run Protocol 1 again, on random and independent inputs (r, s), with Bob as the sender.
- Apply again the local OT reversal procedure to the $m' = \Omega(m) = \Omega(n)$ outputs of the previous step, and output the result.

Let ROT denote the random OT functionality with Alice as the sender and TOR denote the same functionality with Bob as the sender. By Proposition 4.4, the combination of the first two steps of Protocol 2 securely realize TOR^m (given an $(ROT^n)^{[t]}$ oracle) against a corrupted Alice and, using Lemma 3.3, these two steps also securely realize $(TOR^m)^{[t]}$ (given an $(ROT^n)^{[t]}$ oracle) against a corrupted Bob. Applying Proposition 4.4 again, the full protocol securely realizes $ROT^{m'}$ (given an $(ROT^n)^{[t]}$ oracle) against both a corrupted Alice and a corrupted Bob. Thus, Protocol 2 establishes the following theorem.

Theorem 4.5 (Constant-rate OT extractor): There exists an (n, m, t, ϵ) OT extractor with $m(n) = \Omega(n)$, $t(n) = \Omega(n)$, $\epsilon(n) = 2^{-\Omega(n)}$, and communication complexity $O(n)$.

5. General Correlation Extractors

In this section, we apply a constant-rate OT extractor to obtain constant-rate correlation extractors for an arbitrary finite distribution (X, Y). We assume that (X, Y) is a distribution which has a finite support size (independently of n). We also assume that each pair (x, y) in the support of (X, Y) occurs with a *rational* probability; the rate of our correlation extractors will deteriorate (poly-logarithmically) with the size of the maximal denominator of these probabilities.

We rely on the fact that every finite distribution (X, Y) is either "trivial," in the sense that it can be securely realized using only secure communication and private randomness, or it is "complete" in the sense that multiple independent instances of (X, Y) can be used to realize a statistically secure OT. Such a 0-1 law can be derived from a similar 0-1 law obtained in [42], [19]. For trivial distributions (X, Y) it is fairly easy to obtain correlation extractors directly by using standard randomness extractors (see full version). For any nontrivial distribution (X, Y), a constant-rate correlation extractor which securely realizes $(X, Y)^m$ given a leaky $((X, Y)^n)^{[t]}$ oracle is obtained via the following steps:

Step 1. Let $\pi(\kappa)$ be a statistically secure reduction of OT to (X, Y) with security parameter κ, whose existence is guaranteed by the non-triviality of (X, Y). We choose a sufficiently large constant value κ_0 so that $\pi_0 = \pi(\kappa_0)$ yields a sufficiently good "approximation" of OT for the purpose of applying our OT extractor. Let c_0 be the number of instances of (X, Y) consumed by π_0. Using $n' = n/c_0$ independent invocations of π_0 on disjoint portions of $(X, Y)^n$, we obtain a protocol for $ROT^{n'}$ which may have two types of imperfection: (1) each party can learn a small amount of information about the other party's secrets (beyond what it should learn); (2) a small fraction of the n' instances will have inconsistent outputs.

Step 2. Applying a robust version of Protocol 2 to the output of the previous step, we get a statistically secure realization of $ROT^{m'}$, for $m' = \Omega(n') = \Omega(n)$. The robust version differs from the original version only in Step 3(c) of Protocol 1: before finding v', the vector v is error-corrected to the nearest codeword of C'_K.

Step 3. Using general protocols for OT-based secure computation [31], [32], [30], [41], we use $ROT^{m'}$ to obtain a statistically secure realization of $(X, Y)^m$ with $m = \Omega(m') = \Omega(n)$. This step is immediate if the denominators of all probabilities in (X, Y) are powers of two, but can also be performed with constant (amortized) overhead in the case of general rational probabilities.

This yields the following main theorem:

Theorem 5.1 (Constant-rate correlation extractors): Let (X, Y) be a finite distribution such that $\Pr[(X, Y) = (x, y)]$ is a rational number for every (x, y). Then, there exists an explicit (n, m, t, ϵ) correlation extractor for (X, Y) with $m(n) = \Omega(n)$, $t(n) = \Omega(n)$, $\epsilon(n) = 2^{-\Omega(n)}$, and $O(n)$ bits of communication.

The first two steps of the above construction actually yield the following independently useful corollary:

Theorem 5.2 (Constant-rate OT from non-trivial channel): Let g be any functionality such that there is a (possibly inefficient) statistically secure reduction π of OT to g. Then, there is a statistically ϵ-secure reduction π' of OT^m to g (in the semi-honest model) which only makes $n = O(m)$ calls to g and has statistical error $\epsilon(m) = 2^{-\Omega(m)}$. Furthermore, π' remains secure even if the n calls to g jointly leak t bits of information for some $t = \Omega(m)$.

Acknowledgments

We thank Ronald Cramer, Venkat Guruswami, and Salil Vadhan for useful discussions and pointers. We also thank the anonymous FOCS reviewers for their helpful comments.

References

[1] A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous Hardcore Bits and Cryptography against Memory Attacks. In *TCC 2009*, pages 474-495.

[2] N. Alon and Y. Roichman. Random cayley graphs and expanders. *Random Struct. Algorithms*, 5(2):271–285, 1994.

[3] B. Applebaum, Y. Ishai, and E. Kushilevitz. Cryptography in NC^0. *SIAM J. Comput.* 36(4), pages 845–888, 2006.

[4] D. Beaver. Precomputing oblivious transfer. In *CRYPTO '95*, pages 97–109.

[5] D. Beaver and S. Goldwasser. Multiparty Computation with Faulty Majority. In *FOCS 1989*, pages 468-473.

[6] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *STOC '88*, pages 1–10.

[7] C. H. Bennett, G. Brassard, C. Crépeau, and U. Maurer. Generalized privacy amplification. *IEEE Transactions on Information Theory*, 41:1915-1923, 1995.

[8] C. H. Bennett, G. Brassard, and J.-M. Robert. Privacy Amplification by Public Discussion. *SIAM J. Comput.* 17(2): 210-229, 1988.

[9] C. Cachin, C. Crépeau, and J. Marcil. Oblivious Transfer with a Memory-Bounded Receiver. In *FOCS '98*, pages 493-502.

[10] C. Cachin, S.Micali, and M. Stadler. Computationally private information retrieval with polylogarithmic communication. In *EUROCRYPT '99*, pages 402-414.

[11] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[12] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *FOCS '01*, pages 136-145.

[13] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols. In *STOC '88*, pages 11–19.

[14] H. Chen and R. Cramer. Algebraic geometric secret sharing schemes and secure multi-party computations over small fields. In *CRYPTO '06*, pages 521–536.

[15] B. Chor, O. Goldreich, J. Hastad, J. Friedman, S. Rudich, and R. Smolensky. The Bit Extraction Problem of t-Resilient Functions. In *FOCS '85*, pages 396-407.

[16] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. *J. ACM*, 45:965–981, 1998.

[17] C. Crépeau. Efficient cryptographic protocols based on noisy channels. In *EUROCRYPT '97*, pages 306–317.

[18] C. Crépeau and J. Kilian. Achieving oblivious transfer using weakened security assumptions. In *FOCS '88*, pages 42–52.

[19] C. Crépeau, K. Morozov, and S. Wolf. Efficient Unconditional Oblivious Transfer from Almost Any Noisy Channel. In *SCN '04*, pages 47-59.

[20] I. Damgård, S. Fehr, K. Morozov, and L. Salvail. Unfair Noisy Channels and Oblivious Transfer. In *TCC '04*, pages 355-373.

[21] G. Di Crescenzo, T. Malkin, and R. Ostrovsky. Single Database Private Information Retrieval Implies Oblivious Transfer. In *EUROCRYPT '00*, pages 122-138.

[22] Y. Ding, D. Harnik, R. Shaltiel and A. Rosen. Constant-Round Oblivious Transfer in the Bounded Storage Model. *J. Cryptology* 20(2): 165-202, 2007.

[23] Y. Dodis and R. Oliveira. On extracting private randomness over a public channel. In *RANDOM '03*, pages 252-263.

[24] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. *SIAM J. Comput.* 38(1): 97-139, 2008.

[25] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28(6):637–647, 1985.

[26] U. Feige, J. Kilian, and M. Naor. A minimal model for secure computation. In *STOC '94*, pages 554–563.

[27] A. Garcia and H. Stichtenoth. On the asymptotic behavior of some towers of function fields over finite fields. *Journal of Number Theory*, 61(2):248-273, 1996.

[28] C. Gentry. On Homomorphic Encryption over Circuits of Arbitrary Depth. In *STOC '09*, pages 169-178.

[29] C. Gentry and Z. Ramzan. Single-Database Private Information Retrieval with Constant Communication Rate. In *ICALP '05*, pages 803-815.

[30] O. Goldreich. **Foundations of Cryptography** - *Volume 2*. Cambridge University Press, 2004.

[31] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *STOC '87*, pages 218–229.

[32] O. Goldreich and R. Vainish. How to solve any protocol problem - an efficiency improvement. In *CRYPTO '87*, pages 73–86.

[33] O. Goldreich and A. Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Struct. Algorithms*, 11(4):315–343, 1997.

[34] S. Goldwasser, M. Sudan, and V. Vaikuntanathan. Distributed computing with imperfect randomness. In *DISC '05*, pages 288–302.

[35] D. Harnik, Y. Ishai, E. Kushilevitz, and J. B. Nielsen. OT-Combiners via Secure Computation. In *TCC '08*, pages 393-411.

[36] D. Harnik, J. Kilian, M. Naor, O. Reingold, and A. Rosen. On tolerant combiners for oblivious transfer and other primitives. In *EUROCRYPT '05*, pages 96–113.

[37] Y. Ishai and E. Kushilevitz. Perfect Constant-Round Secure Computation via Perfect Randomizing Polynomials. In *ICALP '02*, pages 244-256.

[38] Y. Ishai, E. Kushilevitz, Y. Lindell, and E. Petrank. Black-Box Constructions for Secure Computation. In *STOC '06*, pages 99-108.

[39] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Cryptography with constant computational overhead. In *STOC '08*, pages 433-442.

[40] Y. Tauman Kalai, X. Li, A. Rao, and D. Zuckerman. Network Extractor Protocols. In *FOCS '08*, pages 654-663.

[41] J. Kilian. Founding cryptography on oblivious transfer. In *STOC '88*, pages 20–31.

[42] J. Kilian. More general completeness theorems for secure two-party computation. In *STOC '00*, pages 316-324.

[43] E. Kushilevitz and R. Ostrovsky. Replication is not needed: single database, computationally-private information retrieval. In *FOCS '97*, pages 364-273.

[44] U. Maurer. Perfect Cryptographic Security from Partially Independent Channels. In *STOC '91*, pages 561-571.

[45] U. Maurer. Conditionally-Perfect Secrecy and a Provably Secure Randomized Cipher. *J. Cryptology* 5(1):53-66, 1992.

[46] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993.

[47] A. Nascimento and A. Winter. On the Oblivious Transfer Capacity of Noisy Correlations. *ISIT 06*, pages 1871-1875.

[48] N. Nisan and D. Zuckerman. Randomness is linear in space. *J. Computer and System Sciences*, 52(1):4352, 1996.

[49] B. Przydatek and J. Wullschleger. Error-Tolerant Combiners for Oblivious Primitives. In *ICALP '08*, pages 461-472.

[50] M.O. Rabin. How to exchange secrets by oblivious transfer. TR-81, Harvard, 1981.

[51] S. Wolf and J. Wullschleger. Oblivious Transfer Is Symmetric. In *EUROCRYPT '06*, pages 222-232.

[52] J. Wullschleger. Oblivious Transfer from Weak Noisy Channels. In *TCC '09*, pages 332-349.

[53] A. D. Wyner. The wire-tap channel. *Bell Cyst. Tech. J.*, vol. 54, pp. 1355-1387, 1975.

[54] A. C. Yao. How to generate and exchange secrets. In *FOCS '86*, pages 162–167.