Black-Box Constructions for Secure Computation*

(EXTENDED ABSTRACT)

Yuval Ishai† Eyal Kushilevitz† Yehuda Lindell‡ Erez Petrank†

ABSTRACT

It is well known that the secure computation of non-trivial functionalities in the setting of no honest majority requires computational assumptions. We study the way such computational assumptions are used. Specifically, we ask whether the secure protocol can use the underlying primitive (e.g., one-way trapdoor permutation) in a *black-box* way, or must it be *nonblack-box* (by referring to the code that computes this primitive)? Despite the fact that many general constructions of cryptographic schemes (e.g., CPA-secure encryption) refer to the underlying primitive in a black-box way only, there are some constructions that are inherently nonblack-box. Indeed, all known constructions of protocols for general secure computation that are secure in the presence of a malicious adversary and without an honest majority use the underlying primitive in a nonblack-box way (requiring to prove in zero-knowledge statements that relate to the primitive).

In this paper, we study whether such nonblack-box use is essential. We present protocols that use only *black-box* access to a family of (enhanced) trapdoor permutations or to a homomorphic public-key encryption scheme. The result is a protocol whose communication complexity is *independent* of the computational complexity of the underlying primitive (e.g., a trapdoor permutation) and whose computational complexity grows only *linearly* with that of the underlying primitive. This is the first protocol to exhibit these properties.

Categories and Subject Descriptors: F.1.2 [Theory of Computation]: Interactive and reactive computation

*Research supported by grant 36/03 from the Israel Science Foundation.

†Department of Computer Science, Technion, Israel. email: {yuvali,eyalk,erez}@cs.technion.ac.il

‡Department of Computer Science, Bar-Ilan University, Israel. email: lindell@cs.biu.ac.il. Much of this work was carried out while the author was visiting the Technion.

STOC'06, May21–23, 2006, Seattle, Washington, USA.

General Terms: Algorithms, Theory

Keywords: Theory of cryptography, secure computation, black-box reductions, oblivious transfer

1. INTRODUCTION

It is a known fact that most cryptographic tasks require the use of computational hardness assumptions. These assumptions typically come in two types: *specific assumptions* like the hardness of factoring, RSA, discrete log and others, and *general assumptions* like the existence of one-way functions, trapdoor permutations and others. In this paper, we refer to general assumptions and how they are used. Specifically, we consider an intriguing question regarding how secure protocols utilize a primitive that is assumed to carry some hardness property. Here again, there is a clear distinction between two types of uses:

1. **Black-box usage:** a protocol (or construction) uses a primitive in a black-box way if it refers only to the input/output behavior of the primitive.[1] For example, if the primitive is a trapdoor permutation, then the protocol may sample a permutation and its domain, and may compute the permutation and its inverse (if the trapdoor is given). Beyond this, no reference is made to the primitive. In particular, the *code* used to compute the permutation (or carry out any other task) is not referred to by the protocol. The vast majority of constructions in cryptography are black-box.

2. **Nonblack-box usage:** a protocol (or construction) uses a primitive in a nonblack-box way if it refers to the code for computing its functionality. A typical example of a nonblack-box construction is where a *Karp reduction* is applied to the circuit computing the function, say, in order to prove an $\mathcal{NP}$ zero-knowledge proof, as in [14].

A rich and fruitful body of work, initiated by [16], attempts to draw the borders between possibility and impossibility for black-box constructions in cryptography. While many of the relations between primitives are well understood, there are still some important tasks for which the *only* constructions that we have rely on nonblack-box access to the assumed primitive, yet the existence of a black-box construction is

[1] It is typically also required that the *security proof* of the construction is black-box in the sense that an adversary breaking the protocol can be used as an oracle in order to break the underlying primitive. See, e.g., [11, 12, 29] for a comprehensive treatment of black-box reductions in cryptography.

not ruled out. In particular, *all* known general constructions of multiparty protocols that are secure in the presence of malicious adversaries and without an honest majority, originating from [15], use nonblack-box access to the assumed primitive.[2] (We note that by "general constructions", we mean constructions that can be used to securely compute any functionality.) Another notable example of this phenomenon is the case of public-key encryption that is secure against chosen-ciphertext attacks [7, 30, 23]; here too, all known constructions are nonblack-box. The above phenomenon begs the following question:

> *Is it possible to construct general protocols for secure computation without an honest majority and with malicious adversaries, given only* black-box access *to a "low-level" primitive?*

Answering the above question is of interest for the following reasons. First, it is of theoretical interest to understand whether or not nonblack-box access to a primitive is necessary for these tasks. An answer to this question would enhance our understanding of how hardness assumptions can (or must) be used. Second, as we have mentioned, the nonblack-box use of the underlying primitive is typically utilized in order to apply a Karp reduction for the purpose of using a (general) zero-knowledge proof. Such reductions are highly inefficient and are unlikely to be very useful in practice. Furthermore, in these protocols the *communication complexity* depends on the complexity of computing the primitive and the *computational complexity* grows more than linearly with that of the primitive. (An exception to this rule is the communication-efficient compiler presented in [26], which relies on the communication-efficient arguments of [20, 25]. However, the *computational* complexity of the protocol of [26] is even worse than the GMW protocol [15].)

To illustrate the type of inefficiency resulting from current nonblack-box constructions, consider the following hypothetical scenario. Suppose that, due to major advances in cryptanalytic techniques, the security parameter must be large enough so that *all* basic cryptographic primitives require a full second of computation on a fast CPU. In such a case, would it still be possible to carry out a distributed task like oblivious transfer? Current nonblack-box techniques (e.g., the GMW protocol [15]) require parties to prove in zero-knowledge statements that involve the computation of the underlying primitive, say a trapdoor permutation. These zero-knowledge protocols, in turn, invoke cryptographic primitives for any gate of a circuit computing a trapdoor permutation. Since (by our assumption) a trapdoor permutation takes one second to compute, its circuit implementation contains trillions of gates, thereby requiring the protocol trillions of second to run. In contrast, a black-box construction of oblivious transfer from the trapdoor permutation primitive would make the number of invocations of the primitive independent of the complexity of implementing the primitive, thus making oblivious transfer feasible even in the hypothetical scenario described above.

We conclude that the current nonblack-box use of the underlying primitives constitutes an obstacle to efficiency. It is therefore of great interest to know whether or not it is possible to obtain solutions to these tasks that do not suffer from this obstacle. (We note that the inefficiency of nonblack-box constructions here is quite ironic because in many areas of cryptography, black-box constructions have been shown to have inherent computational limitations [21, 10].) Despite the above, we stress that the focus of this paper is not on efficiency, but rather on the *theoretical question* of whether or not it is possible to obtain the aforementioned black-box constructions. We believe this question to be interesting in its own right.

Our results. We show how to construct general secure multiparty computation (for the case of no honest majority and malicious adversaries), given black-box access to either homomorphic encryption schemes or enhanced trapdoor permutations (see [13, Appendix C.1] for the definition of *enhanced* trapdoor permutations). We note that all known general constructions for this task from "low-level" primitives rely on either enhanced trapdoor permutations or homomorphic encryption schemes. However, they all use them in an inherently nonblack-box way. This is the case even for protocols that implement very simple functionalities, such as oblivious transfer. We prove the following:

THEOREM 1.1. *There exist protocols for securely computing any multiparty functionality without an honest majority and in the presence of static malicious adversaries, that rely only on black-box access to a family of enhanced trapdoor permutations or to a homomorphic encryption scheme.*

We remark that nonblack-box access is not typically used when considering semi-honest adversaries [32, 15]. Rather, the nonblack-box access is utilized in known protocols in order to have the parties prove (in zero-knowledge) that they are correctly following the protocol specification. This is necessary for preventing a malicious adversary from effectively deviating from the protocol instructions. We note also that in the case of an honest majority, it is possible to securely compute any functionality information-theoretically, and without any hardness assumption [2, 5]. Thus, no primitive at all is needed. For this reason, we focus on the case of no honest majority (including the important two-party case) and malicious adversaries.

Techniques. In order to prove Theorem 1.1, we begin by constructing oblivious transfer protocols that use only black-box access to enhanced trapdoor permutations or homomorphic encryption schemes, but provide rather weak security guarantees. We then "boost" the security of these protocols in order to obtain protocols that are secure in the presence of malicious adversaries. Constructions until today that have followed this paradigm work by first obtaining protocols that are secure in the presence of semi-honest adversaries, and then boosting them so that they are secure in the presence of malicious adversaries. However, it is not known how to carry out this "boosting" in a black-box way (and, indeed, it has been conjectured that malicious oblivious transfer *cannot* be constructed from semi-honest oblivious transfer in a black-box way [24]). Since we wish to make our construction black-box, we take a different route.

[2] We stress that the above discussion is only true when considering general assumptions. Furthermore, it is only true when considering "low-level primitives" like trapdoor permutations. Specifically, there *do* exist constructions of secure multiparty protocols that use only black-box access to an oblivious transfer primitive [18]. However, since it is not known how to construct oblivious transfer using only black-box access to, say trapdoor permutations, the overall construction obtained does not use its "low-level" primitive in a black-box way.

Protocol number	Security for corrupted sender	Security for corrupted receiver
3.1, 3.3	Private for defensible sender	Private for defensible receiver
4.1	Private for defensible sender	Secure for malicious receiver
5.1	Secure for malicious sender	Private for defensible receiver
In Theorem 6.1	Secure for malicious sender	Secure for malicious receiver

Table 1: The progression of our constructions: each protocol uses the previous one as a subprotocol.

Specifically, we begin by introducing the notion of a defensible adversary. In order to describe this notion, we describe what a defense is: a defense is an input and random-tape that is provided by the adversary *after* the protocol execution concludes. A defense is good if the honest party upon that input and random-tape would have sent the same messages as the adversary sent. Such a defense is a supposed "proof" of honest behavior. However, the adversary need not actually behave honestly and can construct its defense retroactively (after the execution concludes). A protocol is said to be private in the presence of defensible adversaries if privacy is preserved in the event that an adversary provides a good defense. However, in the case that the adversary doesn't provide a good defense, *nothing is guaranteed*, and the entire honest party's input may be learned. This notion is therefore rather weak. We note that the oblivious transfer protocol of [8] is not secure under this notion. However, it can be efficiently modified into one that is secure under this notion. It is also possible to efficiently construct such an oblivious transfer protocol from homomorphic encryption. Importantly, we show that it is possible to construct oblivious transfer that is secure in the presence of malicious adversaries from oblivious transfer that is private in the presence of defensible adversaries. Furthermore, this construction is *black-box*.

As we have mentioned, we start by constructing oblivious transfer protocols that are private in the presence of *defensible adversaries*. We present two such protocols: one that uses black-box access to a family of enhanced trapdoor permutations, and one that uses black-box access to a homomorphic public-key encryption scheme. Next, we construct from the above oblivious transfer protocol a new oblivious transfer protocol that is still private in the presence of defensible senders, but is *secure* in the presence of malicious receivers (where security is "full security" according to the ideal/real simulation paradigm). This is achieved using the so-called cut-and-choose technique. That is, many oblivious transfer executions (using random inputs) are run, and the receiver is asked to present a defense for its behavior in half of them. If it indeed presents a good defense, then we are guaranteed that it behaved somewhat honestly in most of the executions.

We stress that this step is novel, because the requirements on a protocol that is secure according to the ideal/real simulation paradigm are much stricter than when only privacy is guaranteed. Indeed, some efficient protocols for oblivious transfer from the literature [27, 1, 17] are private for both (malicious) parties, but are not fully secure for either party. Nevertheless, we are able to boost both the resilience of the protocol (from a defensible to a malicious adversary) and its security guarantee (from privacy to full simulation-based security). Next, we "reverse" the oblivious transfer protocol (i.e., by switching the sender and receiver roles) in order to obtain a protocol with reversed security properties. Specifically, this next protocol is secure in the presence of malicious senders and private in the presence of defensible receivers. At this point, we reapply our security boosting technique in order to obtain a protocol that is "fully secure"; that is, a protocol that is secure in the presence of malicious senders and receivers. See Table 1 for the series of oblivious transfer protocols that we construct. Needless to say, each protocol uses its subprotocol in a black-box way.

Finally, having constructed secure oblivious transfer protocols using only black-box access to primitives, it suffices to apply the well-known result of Kilian [18, 19] that shows that any functionality can be securely computed using black-box access to a secure oblivious transfer protocol. This therefore yields Theorem 1.1, as desired.

Related work. Recently, in [6], it was shown that it is possible to construct constant-round protocols for the setting of an *honest majority*, that use only black-box access to the assumed primitive. As we have mentioned, in the setting of an honest majority, it is possible to construct information-theoretically secure protocols (which are, by triviality, black-box). Nevertheless, there are no known (general) constant-round protocols for the information-theoretic setting, and so [6] relates to this issue. We remark that the techniques used in [6] and here are vastly different, due to the inherent differences between the setting of an honest majority and that of no honest majority.

Organization. Due to lack of space in this abstract, we present only brief sketches of the definitions and proofs. Complete details appear in the full version of the paper. We often write OT as shorthand for oblivious transfer.

2. DEFINITIONS

2.1 Preliminaries

We denote by $\langle P_1(1^n, x_1, \rho_1), P_2(1^n, x_2, \rho_2)\rangle$ the transcript of an execution between parties P_1 and P_2 with a security parameter n, where P_i has input x_i and random-tape ρ_i. For brevity, we will sometimes omit the security parameter 1^n. The message sent by party P_i (on the above inputs) after having received the series of incoming messages α is denoted by $P_i(x_i, \rho_i; \alpha)$. Stated otherwise, $P_i(x_i, \rho_i; \cdot)$ denotes the next message function of P_i. Let $t = \langle P_1(x_1, \rho_1), P_2(x_2, \rho_2)\rangle$. Then, denote the ℓ^{th} message sent by P_i in t by $\mathsf{sent}_\ell^{P_i}(t)$ and the first ℓ messages received by P_i in t by $\mathsf{received}_{1,\ldots,\ell}^{P_i}(t)$. We also denote the output of P_i in an execution by $\mathsf{output}_{P_i}\langle P_1(x_1, \rho_1), P_2(x_2, \rho_2)\rangle$.

In our presentation, we assume familiarity with the standard definitions of secure computation; see [13, Chapter 7] for a full treatment. In this work, we consider *malicious adversaries* (i.e., adversaries that may arbitrarily deviate from the protocol specification), and *static corruptions* (meaning that the set of corrupted parties is fixed before the protocol execution begins).

We use a non-uniform formulation of adversaries here and therefore, without loss of generality, assume that they are deterministic. However, this is not essential and all of our proofs hold for the uniform model of computation.

Black-box access to primitives. In this paper, we consider constructions of protocols that use only black-box access to an underlying primitive. This can be easily formalized by defining oracles that provide the functionality of the primitive. For example, a trapdoor permutation can be defined by an oracle that samples a function description along with a trapdoor, an oracle that is given the function description and samples a random value from the domain, an oracle that is given the function description and a point in the domain and computes the permutation, and an oracle that is given the trapdoor and a point in the domain and computes the permutation inverse. It is easy to see that our protocols rely on the underlying primitive in a black-box way. We will therefore not burden the presentation by formally defining these oracles. We remark that we also construct protocols that use subprotocols in a black-box way. This can be formalized by just looking at the input/output behavior of the protocol. We will not formalize this. It suffices for our result to note that if the subprotocol uses the underlying primitive in a black-box way, then the protocol (that uses the subprotocol) also uses the underlying primitive in a black-box way. Again, this is easy to verify for *all* of our protocols. In addition to using the underlying primitive in a black-box way, our proofs of security are also black-box. Therefore, our reductions are what are typically called "fully black-box" [29].

2.2 Defensible Adversarial Behavior

We introduce the notion of defensible adversarial behavior. Loosely speaking, an adversary that exhibits defensible behavior may arbitrarily deviate from the protocol specification. However, at the *conclusion* of the protocol execution, the adversary must be able to justify or *defend* its behavior by presenting an input and a random-tape such that the honest party (with this input and random-tape) would behave in the same way as the adversary did. A protocol is "private" under defensible adversarial behavior if it is "private" in the presence of such adversaries. We stress that if an adversary behaves maliciously and cannot provide a good defense, then no security guarantees are given.

We now define the notion of a *good defense*. Intuitively, a defense is an "explanation" of an adversary's behavior during the protocol execution. Such an explanation consists of an input and random-tape, and the defense is "good" if an honest party, given that input and random-tape, would have sent the same messages as the adversary did during the protocol execution. The formal definition follows.

DEFINITION 2.1. (good defense for t): *Let t be the transcript of an execution of a protocol $\pi = (P_1, P_2)$ between an adversary $\mathcal{A}$ (say, controlling P_1) and the honest party (say P_2). Then, we say that the pair (x_1, ρ_1) constitutes a* good defense by $\mathcal{A}$ for t in π, *denoted* $(x_1, \rho_1) = \mathsf{defense}^{\pi}_{\mathcal{A}}(t)$, *if for every ℓ it holds that* $\mathsf{sent}^{\mathcal{A}}_{\ell}(t) = P_1(x_1, \rho_1; \mathsf{received}^{\mathcal{A}}_{1,\ldots,\ell-1}(t))$.

In other words, every message sent by $\mathcal{A}$ in the execution is such that the honest party P_1 with input (x_1, ρ_1) would have sent the same message.

2.3 Security of OT Protocols

The starting point of our constructions is an oblivious transfer protocol [28, 8] that is private in the presence of a *defensible* receiver or sender. Recall that an oblivious transfer protocol involves a sender S with two input strings s_0 and s_1, and a receiver R with an input bit $r \in \{0, 1\}$. Very informally, an oblivious transfer protocol has the property that the sender learns nothing about the receiver's bit r and the receiver obtains s_r, but learns nothing about s_{1-r}. (The variant of oblivious-transfer that we use here is usually referred to as "1-out-of-2 OT".) We begin by presenting the formal definition of *oblivious transfer* that is *private* in the presence of a defensible receiver and then proceed to define privacy in the presence of a defensible sender.

Non-trivial protocols. One technicality that must be dealt with is that a protocol that does nothing is trivially "private" in that it does not reveal anything about the parties' inputs. Of course, such a protocol is also useless. In order to make sure that the oblivious transfer protocols that we construct are "useful", we define the notion of a non-trivial oblivious transfer protocol. Such a protocol has the property that if both the sender and receiver are honest, then the receiver will receive its output as designated by the oblivious transfer functionality $f((s_0, s_1), r) = (\lambda, s_r)$ (where λ denotes the empty output).

Privacy for random inputs in the presence of a defensible receiver. We now define privacy for defensible receivers. Recall that the receiver in an oblivious transfer protocol is supposed to obtain one of the pair (s_0, s_1) in the execution. However, the other value must remain secret. When considering defensible adversaries, the requirement is that, *as long as the adversary can provide a good defense,* it can only learn one of the values. Recall that, by Definition 2.1, a party's defense includes its input (in this case, the bit r of the receiver, meaning that it wishes to obtain the value s_r). We therefore require that a defensible receiver can learn nothing about s_{1-r} when its defense contains the input value r. Due to technical reasons in our proofs later on, we define privacy only for the case that the sender's inputs are *uniformly distributed bits.* Fortunately, this will suffice for our constructions.

We define an experiment for a protocol π and an adversary $\mathcal{A}$ modelled by a polynomial-size family of circuits $\{\mathcal{A}_n\}_{n \in \mathsf{N}}$. Informally, the experiment begins by choosing a random pair of bits (s_0, s_1) to be used for the sender's input. The adversary's aim is to guess the value of the input that it doesn't receive as output.

Experiment $\mathsf{Expt}^{\mathrm{rec}}_{\pi}(\mathcal{A}_n)$**:**

1. *Choose $s_0, s_1 \in_R \{0, 1\}$ uniformly at random.*
2. *Let ρ_S be a uniformly distributed random tape for S and let $t = \langle S(1^n, s_0, s_1, \rho_S), \mathcal{A}_n \rangle$.*
3. *Let $((r, \rho_r), (\tau))$ be the output of $\mathcal{A}_n(t)$. (The pair (r, ρ_r) constitute $\mathcal{A}_n$'s defense and τ is its guess for s_{1-r}.)*
4. *Output 1 if and only if (r, ρ_r) is a good defense by $\mathcal{A}_n$ for t in π, and $\tau = s_{1-r}$.*

Notice that by $\mathcal{A}$'s defense, it should have received s_r. The challenge of the adversary is therefore to guess the value of s_{1-r}; if it cannot do this, then the sender's privacy is preserved.

DEFINITION 2.2. (privacy for random inputs in the presence of a defensible receiver): *Let $\pi = (S, R)$ be a non-trivial oblivious transfer protocol. We say that π is* private for random inputs in the presence of a defensible receiver *if for every polynomial-size family of circuits $\mathcal{A} = \{\mathcal{A}_n\}_{n\in\mathsf{N}}$ controlling R, for every polynomial $p(\cdot)$ and for all sufficiently large n's*

$$\Pr\left[\mathsf{Expt}_{\pi}^{\mathrm{rec}}(\mathcal{A}_n) = 1\right] < \frac{1}{2} + \frac{1}{p(n)} .$$

Remark. The definition of $\mathsf{Expt}_{\pi}^{\mathrm{rec}}$ only considers the case that the inputs of the sender are uniformly distributed. We stress that this is a very weak definition. However, the reasons that we make this restriction are because **(a)** it suffices for our construction of "fully secure" oblivious transfer (see Protocol 4.1), and more importantly, **(b)** without this restriction we were unable to prove the privacy of Protocol 3.3 for defensible receivers (see Section 3.2). We stress that this restriction is not made when considering security in the presence of malicious parties.

Privacy in the presence of a defensible sender. In an oblivious transfer protocol, the sender is not supposed to learn anything about the receiver's input. When considering a defensible sender, this means that the sender should not be able to simultaneously present a good defense of its behavior and make a correct guess as to the value of the receiver's input. We stress that this privacy requirement only needs to hold when the sender outputs a good defense; in all other cases, there may be no privacy whatsoever. The exact definition is formulated in a similar way as above.

Security. The definitions above refer only to "privacy", meaning that the adversary can learn nothing more about the honest party's input than what is revealed by the output. However, these definitions say nothing about the *simulatability* of the protocols in question. In particular, a protocol that is private by one of the above definitions may not be secure according to the *real/ideal simulation paradigm* (see [13, Chapter 7] for these definitions). When we mention *security* in this paper, we refer to security according to the ideal/real model paradigm.

3. PRIVACY FOR DEFENSIBLE SENDERS AND DEFENSIBLE RECEIVERS

In this section we show how to construct oblivious transfer protocols that are private for defensible senders and receivers. We present two protocols: one based on homomorphic encryption and one based on enhanced trapdoor permutations. Importantly, both protocols access the underlying primitive in a black-box way only.

3.1 Bit OT from Homomorphic Encryption

We assume the existence of a public-key encryption scheme (G, E, D) that is indistinguishable under chosen-plaintext attacks and has the following homomorphic property:

1. The plaintext is taken from a finite Abelian group determined by the public key. For notational convenience, we assume here that the group is an "additive" group Z_q; however, the same construction works for "multiplicative" groups as well.
2. Given *any* public-key pk generated by the key generation algorithm G and *any* two ciphertexts $c_1 = E_{pk}(m_1)$ and $c_2 = E_{pk}(m_2)$, it is possible to efficiently compute a *random* encryption of the sum $E_{pk}(m_1 + m_2)$. Consequently, it is also possible to efficiently compute $E_{pk}(\alpha \cdot m_1)$ for any known integer α.

We also assume that (G, E, D) has *no decryption errors.* Such encryption schemes can be constructed under the quadratic-residuosity, decisional Diffie-Hellman and other assumptions; see [1, 17] for some references. The following protocol is implicit in [22].

PROTOCOL 3.1.

- **Inputs:** *The sender S has a pair of bits (s_0, s_1); the receiver R has a bit r.*
- **The protocol:**
  1. *The receiver R chooses a pair of keys $(pk, sk) \leftarrow G(1^n)$, computes $c = E_{pk}(r)$ and sends c and p_k to S.*
  2. *The sender S uses the homomorphic property and its knowledge of s_0 and s_1 to compute a* random *encryption $c' = E_{pk}((1-r)s_0 + rs_1)$.*
  3. *R computes and outputs $s_r = D_{sk}(c')$.*

Before proving security, note that if S and R are both honest, then R receives the correct output. For example, if $r = 0$, then $c' = E_{pk}(1 \cdot s_0 + 0 \cdot s_1) = E_{pk}(s_0)$ and so R receives the correct value after decryption.

CLAIM 3.2. *Assume that the encryption scheme (G, E, D) is indistinguishable under chosen-plaintext attacks and has no decryption errors. Then, Protocol 3.1 is a non-trivial oblivious transfer protocol that is private in the presence of defensible senders and private for random inputs in the presence of defensible receivers.*

Privacy in the presence of a defensible (or even malicious) sender follows from the fact that the sender's view consists only of a single encryption under E, and this encryption is secure. Privacy with respect to a defensible receiver follows since the existence of a proper defense implies that c is indeed an encryption of 0 or 1. This, in turn, guarantees that c' is a random encryption of s_r. Hence, again, privacy follows from the security of E.

3.2 Bit OT from Enhanced Trapdoor Permutations

The following protocol is a modified version of [8] that is private in the presence of defensible adversaries. We stress that the original protocol of [8] is completely insecure in the presence of defensible adversaries. The construction uses any family of enhanced trapdoor permutations. Informally speaking, a family of trapdoor permutations is comprised of a function-sampling algorithm I, a domain-sampling algorithm D_f, an algorithm F for computing the permutation and an algorithm F^{-1} for inverting the permutation (given the trapdoor). Such a family is called *enhanced* if it is hard to invert a random value y even when given the coins used by the domain-sampling algorithm to sample y. See [13, Appendix C.1 and Section 7.3] for a full definition. In the sequel, we will abuse notation and refer to the random coins used by D_f as its *input.* We note that the enhanced property is used in all constructions of oblivious transfer from trapdoor permutations. Indeed it has been shown that black-box constructions of oblivious transfer from plain trapdoor permutations is impossible [9].

We will require that I is *errorless*, meaning that for every series of random coins provided to I, the description of the function output is indeed a *permutation*. We call this errorless function sampling, or just errorless sampling.

The protocol uses a perfectly binding commitment scheme C. We denote a commitment to a using randomness ρ by $C(a;\rho)$. For simplicity, we assume that in order to commit to a string a of length n, it suffices to use a random string that is also of length n. Such a commitment scheme can be obtained using black-box access to any trapdoor permutation or homomorphic encryption scheme.

PROTOCOL 3.3.

- **Inputs:** *The sender S has a pair of random bits (s_0, s_1); the receiver R has a bit r.*
- **Auxiliary information:** *The description of a family of (enhanced) trapdoor permutations (I, D_f, F, F^{-1}) and a hard-core bit B for the family.*
- **The protocol:**
  1. *The receiver R chooses $\rho_1, \rho \in_R \{0,1\}^n$ and sends $c = C(\rho_1; \rho)$ to the sender S.*
  2. *S chooses a trapdoor permutation pair $(i,t) \leftarrow I(1^n)$ and a random $\rho_2 \in_R \{0,1\}^n$, and sends i and ρ_2 to R.*
  3. *R computes $y_{1-r} = D_f(\rho_1 \oplus \rho_2)$; i.e., y_{1-r} is obtained by running the domain sampling algorithm with coins $\rho_1 \oplus \rho_2$. In addition, R chooses $\rho' \in_R \{0,1\}^n$, obtains $x_r = D_f(\rho')$ and computes $y_r = f_i(x_r)$. Finally, R sends (y_0, y_1) to S.*
  4. *S uses t to compute $\sigma_0 = B(f_i^{-1}(y_0)) \oplus s_0$ and $\sigma_1 = B(f_i^{-1}(y_1)) \oplus s_1$. S sends (σ_0, σ_1) to R.*
  5. *R computes and outputs $s_r = B(x_r) \oplus \sigma_r$.*

Note that the only difference between Protocol 3.3 and the protocol of [8] is that in [8], the value y_{1-r} is chosen singlehandedly by the receiver, whereas here the value is chosen mutually using a (weak non-simulatable) coin-tossing protocol. (Indeed, in the protocol of [8] a cheating receiver can just choose a value y_{1-r} for which it knows the preimage. The receiver will then learn both s_0 and s_1. Note also that a defensible receiver can also easily cheat in the protocol of [8] because it can send any value y_{1-r} and not the value that equals $D_f(\rho_1 \oplus \rho_2)$. In particular, it can send a value y_{1-r} for which it knows its preimage x_{1-r} under f_i, and can still claim in its defense that its coins are such that y_{1-r} was sampled directly.)

CLAIM 3.4. *Assume that (I, D_f, F, F^{-1}) is a family of enhanced one-way trapdoor permutations and that the scheme C is perfectly binding and computationally hiding. Then, Protocol* 3.3 *is a non-trivial oblivious transfer protocol that is private in the presence of defensible receivers and private for random inputs in the presence of defensible senders.*

Intuitively, a corrupted sender cannot guess the value of r from (y_0, y_1) because these values are identically distributed. This actually only holds as long as the function f_i chosen by the sender is really a permutation from the family. (Otherwise, it may be possible to distinguish y_r which is generated by computing $f_i(x_r)$ from y_{1-r} which is randomly chosen from the domain.) The fact that the function is really a permutation is "proven" in the defense, and so if a good defense is provided, y_r and y_{1-r} are identically distributed. We therefore have that the only way a defensible sender can learn the value of r is from the commitments. However, this involves distinguishing between $c = C(D_f^{-1}(y_0) \oplus \rho_2)$ and $c = C(D_f^{-1}(y_1) \oplus \rho_2)$ which is hard due to the hiding property of commitments. (Notice that $y_{1-r} = D_f(\rho_1 \oplus \rho_2)$ and so $c = C(\rho_1) = C(D_f^{-1}(y_{1-r}) \oplus \rho_2)$. Therefore, the problem of guessing r reduces to the problem of distinguishing such commitments.) As for privacy in the presence of a defensible receiver R^*: intuitively, if R^* behaves so that it can present a good defense, then it is unable to compute $B(f^{-1}(y_{1-r}))$ because it has no freedom in choosing y_{1-r}. That is, R^* must choose $y_{1-r} = \rho_1 \oplus \rho_2$ and so it cannot know the preimage $f^{-1}(y_{1-r})$. This implies that it can only learn the sender's bit s_r.

4. ACHIEVING SECURITY AGAINST A MALICIOUS RECEIVER

In this section we construct a bit oblivious transfer protocol that is secure in the presence of a malicious receiver and private in the presence of a defensible sender. We stress that the security achieved for malicious receivers is according to the ideal/real model definition of security for secure computation. Our construction uses black-box access to an oblivious transfer protocol that is private for defensible receivers and senders (like those constructed in the previous section). Thus, in this section we show how to boost the security guarantee from *privacy in the presence of a defensible receiver* to *security in the presence of a malicious receiver.* The guarantee regarding a corrupted sender remains unchanged.

PROTOCOL 4.1.

- **Inputs:** *The sender S has a pair of bits (s_0, s_1); the receiver R has a bit r.*
- **The protocol:**
  1. *The receiver R chooses $2n$ uniformly distributed bits $r_1, \ldots, r_{2n} \in_R \{0,1\}$.*
  2. *The sender S chooses $2n$ pairs of random bits $s_i^0, s_i^1 \in_R \{0,1\}$ for $i = 1, \ldots, 2n$.*
  3. *S and R run $2n$ parallel executions of a bit oblivious transfer protocol π that is private in the presence of defensible receivers and defensible senders. In the i^{th} execution, S inputs (s_i^0, s_i^1) and R inputs r_i. Let $t_1, \ldots, t_{2n}$ be the transcripts that result from these executions.*
  4. *S and R run a secure two-party coin-tossing protocol (that accesses a one-way function in a black-box way) for generating a random string of length n: $q = q_1, \ldots, q_n$.*[3] *The string q is used to define a set of indices $Q \subset \{1, \ldots, 2n\}$ of size n in the following way: $Q = \{2i - q_i\}_{i=1}^n$. (Thus, for $n = 3$ and $q = 010$ we have that $Q = \{2,3,6\}$.)*

[3]Sequential executions of the coin-tossing protocol of [3] can be used. The security of this has been proven formally in [13].

5. *For every $i \in Q$, the receiver R provides a defense (r_i, ρ_r^i).*
6. *S checks that for every $i \in Q$, the pair (r_i, ρ_r^i) constitutes a good defense by R for t_i. If not, then S aborts and halts. Otherwise, it continues to the next step.*
7. *For every $j \notin Q$, the receiver R computes $\alpha_j = r \oplus r_j$ (where r is R's initial input) and sends $\{\alpha_j\}_{j\notin Q}$ to S.*
8. *S computes $\sigma_0 = s_0 \oplus \left(\bigoplus_{j\notin Q} s_j^{\alpha_j}\right)$ and $\sigma_1 = s_1 \oplus \left(\bigoplus_{j\notin Q} s_j^{1-\alpha_j}\right)$, and sends (σ_0, σ_1) to R.*
9. *R computes and outputs $s_r = \sigma_r \oplus \left(\bigoplus_{j\notin Q} s_j^{r_j}\right)$.*

We note that the sender's inputs to the executions of the oblivious transfer subprotocol π in Protocol 4.1 are uniformly distributed. Therefore, it suffices to use Protocol 3.3, even though it has only been proven "private" for the case of uniformly distributed sender inputs.

We stress that our proof below of Protocol 4.1 relies on the fact that the sender's inputs are single bits.[4]

CLAIM 4.2. *Assume that π is a non-trivial oblivious transfer protocol that is private for random inputs in the presence of defensible senders and receivers. Then, Protocol* 4.1 *is a non-trivial oblivious transfer protocol that is* secure in the presence of malicious receivers *and private in the presence of defensible senders.*

PROOF SKETCH: We first demonstrate the non-triviality property; that is, we show that if S and R are honest, then R receives s_r, as required. To see this, first note that by the non-triviality of π, the receiver R obtains all of the bits $s_j^{r_j}$, and in particular all $s_j^{r_j}$ for $j \notin Q$. Now, if $r = 0$, then R sets $\alpha_j = r_j$ for every $j \notin Q$. Therefore, R will compute $s_0 = \sigma_0 \oplus \left(\bigoplus_{j\notin Q} s_j^{r_j}\right) = \sigma_0 \oplus \left(\bigoplus_{j\notin Q} s_j^{\alpha_j}\right)$. This computation is correct because S computed $\sigma_0 = s_0 \oplus \left(\bigoplus_{j\notin Q} s_j^{\alpha_j}\right)$. In contrast, if $r = 1$, then $\alpha_j = 1 \oplus r_j$ for every j, which is equivalent to $r_j = 1-\alpha_j$. Thus, once again, R's computation of $\bigoplus_{j\notin Q} s_j^{r_j}$ when computing s_1 equals S's computation of $\bigoplus_{j\notin Q} s_j^{1-\alpha_j}$ when computing σ_1, and R will obtain σ_1.

Privacy in the presence of defensible senders. We present only the idea behind the proof that Protocol 4.1 is private in the presence of a defensible sender $\mathcal{A}$. Intuitively, if protocol π is private in the presence of a defensible sender, then a defensible adversary here cannot learn any of the r_i values in the execution (apart from those explicitly revealed by R when it provides its defenses). Therefore, the $\alpha_j = r_j \oplus r$ values that it receives reveal nothing of the receiver's input r, because for all $j \notin Q$, the value r_j is not learned.

Security in the presence of malicious receivers. We present an almost full proof that Protocol 4.1 is secure in the presence of malicious receivers. The intuition behind this proof is that the cut-and-choose technique forces an adversarial receiver $\mathcal{A}$ to be able to provide a good defense for most of the oblivious transfer executions (or be caught with high probability). In particular, there must be at least one $j \notin Q$ for which $\mathcal{A}$ *could* have provided a good defense. This implies that there exists some j for which $\mathcal{A}$ cannot predict the value of $s_j^{1-r_j}$ with any non-negligible advantage. Since s_{1-r} is masked by $s_j^{1-r_j}$, it follows that $\mathcal{A}$ also learns nothing about s_{1-r}. We stress that the above intuition shows that a malicious $\mathcal{A}$ cannot learn anything about s_{1-r}. However, we actually need to prove a much stronger claim in that the protocol is *secure* for a malicious R^*, as defined via the ideal/real model simulation paradigm. We present our analysis in the so-called "hybrid model", where the honest parties use a trusted party to compute the coin-tossing functionality for them.

We now describe the simulator Sim for $\mathcal{A} = \{\mathcal{A}_n\}$:

1. For each $i = 1, \ldots, 2n$, simulator Sim chooses random pairs $s_i^0, s_i^1 \in_R \{0,1\}$ and plays the honest sender in π with these inputs, where $\mathcal{A}_n$ plays the receiver.
2. Sim chooses a random string $q \in_R \{0,1\}^n$ and hands it to $\mathcal{A}_n$ as if it is the output of the coin-tossing functionality, as sent by the trusted party. Let Q be the index set derived from q. Upon receiving back pairs (r_i, ρ_r^i) for $i \in Q$, simulator Sim checks that they all constitute good defenses, respectively. If not, then it aborts (just like the honest sender).
3. Sim rewinds $\mathcal{A}_n$ to the beginning of the previous step and chooses a new random string q' with associated index set Q'. (We stress that q' is independent of q.) Sim hands q' to $\mathcal{A}_n$ and sees if it replies with pairs (r_i, ρ_r^i) that are good defenses, for all $i \in Q'$. Sim repeats this process with a new q' until $\mathcal{A}_n$ indeed replies with pairs (r_i, ρ_r^i) that are good defenses, for all $i \in Q'$. If $Q' = Q$, then Sim outputs fail. Otherwise it proceeds to the next step.
4. Given that $Q' \neq Q$ (and $|Q'| = |Q|$), there exists at least one index j such that $j \notin Q'$ but $j \in Q$. For such a j, Sim computes $r = r_j \oplus \alpha_j$ and sends r to the trusted party. (Note that r_j is obtained from the defense (r_j, ρ_r^j) that was received from $\mathcal{A}_n$ after it was sent the query set Q. In contrast, α_j is the value received from $\mathcal{A}_n$ after rewinding; i.e., when the query set was Q'.)
5. Upon receiving back a bit s_r from the trusted party, Sim computes σ_0 and σ_1 as follows:
   (a) If $r = 0$, then $\sigma_0 = s_0 \oplus \left(\bigoplus_{j\notin Q'} s_j^{\alpha_j}\right)$ and $\sigma_1 \in_R \{0,1\}$.
   (b) If $r = 1$, then $\sigma_0 \in_R \{0,1\}$ and $\sigma_1 = s_1 \oplus \left(\bigoplus_{j\notin Q'} s_j^{1-\alpha_j}\right)$.

   Sim sends (σ_0, σ_1) to $\mathcal{A}_n$ and output whatever $\mathcal{A}_n$ does.

We proceed to prove that the joint output of Sim and the honest sender S in the ideal model is computationally indistinguishable from the joint output of $\mathcal{A}_n$ and S in the real model. Actually, since the honest S has no output from the protocol, it suffices here to show that the output of Sim in the ideal model is computationally indistinguishable from the output of $\mathcal{A}_n$ in the real model. We first claim that apart from the pair (σ_0, σ_1), the view of $\mathcal{A}_n$ in the simulation with

[4] This is due to our definition of "oblivious transfer that is private for defensible adversaries". It is possible to define a stronger notion of defensible adversaries that *is* sufficient for proving that Protocol 4.1 is secure even when the sender's inputs are strings of an arbitrary length. However, we were not able to prove that Protocol 3.3 is private for defensible adversaries under this stronger notion (in contrast to Protocol 3.1 that can be proven secure under the stronger notion).

Sim is *statistically close* to its view in a real execution with S; the only difference being in the case that Sim outputs fail. This can be seen as follows: if $\mathcal{A}_n$ does not send good defenses after receiving q, then Sim aborts, just as the honest S would (and in this case the simulation is perfect). If $\mathcal{A}_n$ does send good defenses, then Sim continues until it finds another (independent) q' for which $\mathcal{A}_n$ also replies with good defenses. It is not hard to see that this yields a distribution that is the same as in a real execution, except when $q' = q$, in which case Sim outputs fail. However, this event (that it provides good defenses on q and then the next time that it provides good defenses is again on q) can happen with probability only 2^{-n}.

We therefore have that in the simulation by Sim, the adversary $\mathcal{A}_n$'s partial view up until the point that it receives (σ_0, σ_1) is statistically close to its view in a real execution with S. We now show that $\mathcal{A}_n$'s full view is computationally indistinguishable. To do this, we consider a modified ideal-model simulator Sim' who receives the sender S's input pair (s_0, s_1). Simulator Sim' works in exactly the same way as Sim, except that it computes σ_{1-r} as an honest sender would instead of choosing it uniformly. By the above argument, it follows that the distribution generated by Sim' in the ideal model is *statistically close* to the distribution generated by a real execution between S and $\mathcal{A}_n$. (Recall that Sim already generates σ_r in the same way as an honest S, and therefore so does Sim'.) It remains to show that the distribution generated by Sim' is computationally indistinguishable to that generated by Sim.

The only difference between Sim' and Sim is in the generation of σ_{1-r}: simulator Sim' generates it "honestly", whereas Sim chooses it uniformly. As mentioned above, intuitively, indistinguishability follows from the fact that at least one $s_j^{1-r_j}$ masks the value of s_{1-r}. Formally, we show that if this "fake" σ_{1-r} can be distinguished from a real one, then we can construct a defensible receiver $\tilde{\mathcal{A}}_n$ that can break the oblivious transfer protocol π.

That is, we show that if the output generated by Sim and Sim' can be distinguished with non-negligible probability, then it is possible for a *defensible adversary* $\tilde{\mathcal{A}}_n$ to succeed in the experiment of Definition 2.2 with non-negligible advantage, with respect to the subprotocol π. Assume by contradiction that there exists a distinguisher D, a polynomial $p(\cdot)$ and infinitely many n's such that

$$|\Pr[D(\mathsf{output}_{\mathsf{Sim}}) = 1] - \Pr[D(\mathsf{output}_{\mathsf{Sim}'}) = 1]| \geq \frac{1}{p(n)} .$$

Without loss of generality, assume that

$$\Pr[D(\mathsf{output}_{\mathsf{Sim}}) = 1] - \Pr[D(\mathsf{output}_{\mathsf{Sim}'}) = 1] \geq \frac{1}{p(n)} . \quad (1)$$

We now use the above to construct a defensible adversary $\tilde{\mathcal{A}} = \{\tilde{\mathcal{A}}_n\}$. Adversary $\tilde{\mathcal{A}}_n$ begins its attack by starting the simulation of Protocol 4.1, according to Sim's strategy. Specifically, $\tilde{\mathcal{A}}_n$ chooses $s_0, s_1 \in_R \{0,1\}$ and runs the simulation strategy of Sim with $\mathcal{A}_n$ up until the point where σ_0 and σ_1 are sent. The simulation is the same as Sim, except for the following difference: $\tilde{\mathcal{A}}_n$ begins by choosing $j \in_R \{1, \ldots, 2n\}$ and internally invokes $\mathcal{A}_n$, simulating an execution of Protocol 4.1. Then, all of the oblivious transfers subexecutions of π, *except* for the j^{th} one, are run internally with $\tilde{\mathcal{A}}_n$ playing the honest sender ($\tilde{\mathcal{A}}_n$ also chooses the s_i^0 and s_i^1 values as S would); in contrast, the messages of the j^{th} execution of the oblivious transfer protocol π are forwarded between $\tilde{\mathcal{A}}_n$'s external sender and the internal $\mathcal{A}_n$ playing the receiver. Following the oblivious transfer executions, $\tilde{\mathcal{A}}_n$ runs the honest sender in the coin-tossing protocol to generate q and thus Q as required. If $j \notin Q$, then $\tilde{\mathcal{A}}_n$ outputs fail and halts. Otherwise, $\tilde{\mathcal{A}}_n$ receives back the defenses; since $j \in Q$, the j^{th} defense is included. If (r_j, ρ_r^j) is not a good defense, then $\tilde{\mathcal{A}}_n$ outputs fail and halts. Otherwise, it stores (r_j, ρ_r^j) and continues like Sim by rewinding $\mathcal{A}_n$ and generating a new q' and Q'. If $j \in Q'$, then once again $\tilde{\mathcal{A}}_n$ outputs fail and halts. Otherwise, it continues like Sim (using the j chosen above for which it is given that $j \in Q$ and $j \notin Q'$). $\tilde{\mathcal{A}}_n$ continues in the same way that Sim does up until (but not including) the point at which (σ_0, σ_1) must be sent. Now, $\tilde{\mathcal{A}}_n$ computes (σ_0, σ_1) as follows. First, note that $\tilde{\mathcal{A}}_n$ knows the values (s_0, s_1) and s_i^0, s_i^1 for all $i \neq j$ (because it chose them). However, the values s_j^0 and s_j^1 are not known to $\tilde{\mathcal{A}}_n$ because these are the values used by the external sender with whom it interacts. Nevertheless, the (good) defense provided by $\mathcal{A}_n$ is enough to obtain the value $s_j^{r_j}$. This holds because given the transcript of the j^{th} oblivious transfer execution and the input and random-tape of the receiver, it is possible to derive $s_j^{r_j}$. The only value unknown to $\tilde{\mathcal{A}}_n$ is therefore $s_j^{1-r_j}$. Therefore, $\tilde{\mathcal{A}}_n$ is able to compute σ_r like the honest sender. In contrast, it cannot honestly compute σ_{1-r}. Rather, $\tilde{\mathcal{A}}_n$ guesses the value of $s_j^{1-r_j} \in_R \{0,1\}$ randomly, and then computes σ_{1-r} using s_{1-r}, all of the s_i values that it knows (i.e., all apart from $s_j^{1-r_j}$), and the uniformly chosen $s_j^{1-r_j}$. In order to determine its output, $\tilde{\mathcal{A}}_n$ obtains the output of $\mathcal{A}_n$ and runs the distinguisher D (from Eq. (1)) on this output; let b be the bit output by D. Then, $\tilde{\mathcal{A}}_n$ sets $\tau = s_j^{1-r_j} \oplus b$. (Recall that τ is $\tilde{\mathcal{A}}_n$'s guess for the "not-received" bit used by the honest sender. The motivation for this guess is that by Eq. (1), D outputs 1 with higher probability on Sim (when the bit is random) than on Sim' (when the bit is correct). Thus, when D outputs 1, we flip $\tilde{\mathcal{A}}_n$'s guess for $s_j^{1-r_j}$.) Finally, $\tilde{\mathcal{A}}_n$ outputs the defense (r_j, ρ_r^j) from above and the bit τ.

We proceed to analyze the probability that $\tilde{\mathcal{A}}_n$ succeeds in $\mathsf{Expt}_\pi^{\text{rec}}$. First, note that unless $\tilde{\mathcal{A}}_n$ outputs fail, the view of $\mathcal{A}_n$ when interacting with $\tilde{\mathcal{A}}_n$ above is *identical* to its view in the simulation by Sim. This is due to the fact that $\tilde{\mathcal{A}}_n$ follows Sim's strategy, except for two differences. The first difference is that in the j^{th} execution of the oblivious transfer protocol π is run externally. However, since Sim plays the role of an honest receiver in all of the executions, this makes no difference to $\mathcal{A}_n$'s view. The second difference is in how σ_{1-r} is computed: Sim chooses it uniformly, whereas $\tilde{\mathcal{A}}_n$ computes it as described above. Clearly, the distribution generated is the same because $\tilde{\mathcal{A}}_n$ uses a uniformly distributed $s_j^{1-r_j}$, and thus σ_{1-r} is also uniformly distributed.

Now, denote the inputs of the honest sender that $\tilde{\mathcal{A}}_n$ interacts with by $(\tilde{s}_0, \tilde{s}_1)$. Using the facts that **(a)** $\tilde{\mathcal{A}}_n$ generates the exact same distribution as Sim, **(b)** $\tilde{\mathcal{A}}_n$ sets $\tau = s_j^{1-r_j} \oplus b$ (where b is D's output bit), and **(c)** $\tilde{\mathcal{A}}_n$ presents a good defense every time that it does not output fail, we have that

$$\begin{aligned} &\Pr\left[\mathsf{Expt}_\pi^{\text{rec}}(\tilde{\mathcal{A}}_n) = 1 \mid \mathsf{output}_{\tilde{\mathcal{A}}_n} \neq \mathsf{fail}\right] \quad (2) \\ &\qquad = \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] . \end{aligned}$$

(Recall that $\mathsf{Expt}^{\rm rec}_\pi(\tilde{\mathcal{A}}_n) = 1$ if $\tilde{\mathcal{A}}_n$ presents a good defense and $\tau = \tilde{s}_{1-r_j}$.)

In contrast to the above, conditioned on the event that $s_j^{1-r_j} = \tilde{s}_{1-r_j}$ (i.e., the event that $\tilde{\mathcal{A}}_n$ guessed correctly), the result is an execution that is distributed exactly according to Sim'. (Recall that the only difference between Sim and Sim' is with respect to the computation of σ_{1-r}.) That is,

$$\begin{aligned}
&\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j} \mid s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \\
&\quad = \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j} \mid s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \\
&\quad = \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 0\right]
\end{aligned}$$

where the last equality is just due to the fact that $s_j^{1-r_j} = \tilde{s}_{1-r_j}$. Now, recalling that $s_j^{1-r_j}$ is chosen uniformly by $\tilde{\mathcal{A}}_n$ (and so equals $\tilde{s}_{1-r_j}$ with probability exactly $1/2$), we have:

$$\begin{aligned}
&\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \\
&= \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j} \mid s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \\
&\quad + \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j} \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] \\
&= \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 0\right] \\
&\quad + \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] \\
&= \frac{1}{2}\cdot\left(1 - \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 1\right]\right) \\
&\quad + \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] \\
&= \frac{1}{2} + \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] \\
&\quad - \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 1\right] .
\end{aligned}$$

Recalling again that when $s_j^{1-r_j} = \tilde{s}_{1-r_j}$ the output of Sim is the same as Sim', we have that

$$\begin{aligned}
&\frac{1}{2} + \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] \\
&\qquad - \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 1\right] \\
&= \frac{1}{2} + \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] \\
&\quad + \frac{1}{2}\cdot\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \\
&\quad - \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 1\right] \\
&= \frac{1}{2} + \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1\right] - \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 1\right] .
\end{aligned}$$

Combining the above with Equations (1) and (2), we have that for infinitely many n's

$$\begin{aligned}
&\Pr\left[\mathsf{Expt}^{\rm rec}_\pi(\tilde{\mathcal{A}}_n) = 1 \mid \mathsf{output}_{\tilde{\mathcal{A}}_n} \neq \mathsf{fail}\right] \\
&\quad = \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \geq \frac{1}{2} + \frac{1}{p(n)} .
\end{aligned}$$

Recall now that $\tilde{\mathcal{A}}_n$ outputs fail if $\mathcal{A}_n$ does not output a good defense, if $j \notin Q$, or if $j \in Q'$. We first claim that $\mathcal{A}_n$ must output a good defense with non-negligible probability. This follows simply from the fact that when $\mathcal{A}_n$ does not output a good defense, the execution is truncated and the distributions generated by Sim and Sim' are *identical*. Therefore, Eq. (1) implies that for infinitely many n's, $\mathcal{A}_n$ outputs a good defense with probability at least $1/p(n)$. Next, recall that $\tilde{\mathcal{A}}_n$ chooses the sets Q and Q' randomly (under the constraints prescribed in the protocol). Thus, with probability exactly $1/4$, $j \in Q$ and $j \notin Q'$ (because the probability that a given j is in a specified set is exactly $1/2$). We conclude that with non-negligible probability, $\tilde{\mathcal{A}}_n$ does not output fail, and thus $\Pr[\mathsf{Expt}^{\rm rec}_\pi(\tilde{\mathcal{A}}_n) = 1]$ is non-negligible.

It remains to show that Sim runs in expected polynomial-time. Aside from the rewinding stage, all work takes a fixed polynomial amount of time. Regarding the rewinding stage, we have the following. Let p denote the probability that $\mathcal{A}_n$ replies correctly upon a random set of indices Q of size n, as specified in the protocol. Then, given that $\mathcal{A}_n$ replied correctly to the initial query set Q, the expected number of rewinding attempts with independent Q' made by Sim equals $1/p$. Since these rewinding attempts are only made if $\mathcal{A}_n$ replied correctly to the initial query set Q, we have that the expected number of attempts overall equals $p \cdot 1/p = 1$. This completes the proof. ∎

5. MALICIOUS SENDERS AND DEFENSIBLE RECEIVERS

In this section, we *reverse* the oblivious transfer protocol of Protocol 4.1 to obtain a protocol that is secure in the presence of a malicious sender and private for random inputs in the presence of a defensible receiver. We use the construction of [31] for reversing Protocol 4.1. The protocol is as follows:

PROTOCOL 5.1. (reversing oblivious transfer):

- **Inputs:** *The sender S has a pair of bits (s_0, s_1) for input and the receiver R has a bit r.*
- **The protocol:**
  1. *The sender and receiver run an oblivious transfer protocol π that is secure in the presence of a malicious receiver and private in the presence of a defensible sender:*
     (a) *The sender S, playing the receiver in π, inputs $\tilde{r} = s_0 \oplus s_1$*
     (b) *The receiver R, playing the sender in π, chooses a random bit $\rho \in_R \{0,1\}$ and inputs $\tilde{s}_0 = \rho$ and $\tilde{s}_1 = \rho \oplus r$.*

     Denote S's output from π by a.
  2. *S sends R the bit $\alpha = s_0 \oplus a$.*
  3. *R outputs $s_r = \rho \oplus \alpha$.*

The security of Protocol 5.1 can be easily proven as an information-theoretic reduction, or when the original oblivious transfer protocol is *fully secure*. In contrast, it is far more subtle in the setting where only privacy in the presence of a defensible sender is assumed. Nevertheless, we do obtain the following claim:

CLAIM 5.2. *If π is a non-trivial oblivious transfer protocol that is secure in the presence of a malicious receiver and private in the presence of a defensible sender, then Protocol 5.1 is a non-trivial oblivious transfer protocol that is secure in the presence of a malicious sender and private for random inputs in the presence of a defensible receiver.*

6. FULLY-SECURE BIT OT

In this section, we use the construction of Protocol 4.1 again in order to boost the security of Protocol 5.1 so that it is secure in the presence of both a malicious sender and a malicious receiver; we call such a protocol fully secure to stress that it is *secure* in the face of any corruption.

By Claim 4.2, we have that Protocol 4.1 boosts the security of *any* oblivious transfer protocol that is private for defensible receivers into one that is secure in the presence of malicious receivers. We can therefore use Protocol 4.1 to boost the security of Protocol 5.1 so that the result is a protocol that is secure in the presence of malicious receivers. This does not suffice, however, because we must show that if the subprotocol used in Protocol 4.1 is *secure* in the presence of malicious senders, then the result is still secure in the presence of malicious senders. (Claim 4.1 considers only privacy for defensible senders.) This is actually easy to show, and is omitted here due to lack of space.

THEOREM 6.1. *Assume that there exists a non-trivial bit oblivious transfer protocol π that is secure in the presence of malicious senders and private for random inputs in the presence of defensible receivers. Then, Protocol 4.1 that is instantiated using this π, is a non-trivial bit oblivious transfer protocol that is secure in the presence of malicious receivers and senders.*

Black-box construction of oblivious transfer. Noting that perfectly-binding commitment schemes (as used in Protocol 3.3) can be constructed using black-box access to homomorphic encryption or enhanced trapdoor permutations, and combining Protocols 3.1 and 3.3 with Protocol 4.1, followed by Protocol 5.1 and the construction in Theorem 6.1, we obtain secure bit oblivious transfer with *black-box access* to a homomorphic encryption scheme or a family of enhanced trapdoor permutations.

7. BLACK-BOX SECURE COMPUTATION

Kilian [18] showed that any function can be securely computed given black-box access to a bit oblivious transfer functionality. We therefore have the following theorem, that constitutes our main result:

THEOREM 7.1. *Assume that there exist homomorphic encryption schemes with errorless decryption or families of enhanced trapdoor permutations. Then, for any probabilistic polynomial-time functionality f there exists a protocol that uses only black-box access to a homomorphic encryption scheme or to a family of enhanced trapdoor permutations, and securely computes f with any number of corrupted parties and in the presence of a static malicious adversary.*

We remark that as is standard for the setting of no honest majority, the security guarantee achieved here is that of "security with abort"; see [13, Chapter 7] for formal definitions.

8. REFERENCES

[1] W. Aiello, Y. Ishai and O. Reingold. Priced Oblivious Transfer: How to Sell Digital Goods. In *EUROCRYPT 2001*, Springer-Verlag (LNCS 2045), pages 119–135, 2001.

[2] M. Ben-Or, S. Goldwasser and A. Wigderson. Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation. In 20*th STOC,* pages 1–10, 1988.

[3] M. Blum. Coin Flipping by Phone. In *IEEE Spring COMPCOM*, pages 133–137, 1982.

[4] R. Canetti. Security and Composition of Multiparty Cryptographic Protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[5] D. Chaum, C. Crépeau and I. Damgård. Multi-party Unconditionally Secure Protocols. In 20*th STOC*, pages 11–19, 1988.

[6] I. Damgård and Y. Ishai. Constant-Round Multiparty Computation Using a Black-Box Pseudorandom Generator. In *CRYPTO 2005*, Springer-Verlag (LNCS 3621), pages 378–394, 2005.

[7] D. Dolev, C. Dwork and M. Naor. Non-Malleable Cryptography. *SIAM Journal on Computing,* 30(2):391–437, 2000.

[8] S. Even, O. Goldreich and A. Lempel. A Randomized Protocol for Signing Contracts. In *Communications of the ACM,* 28(6):637–647, 1985.

[9] R. Gennaro, Y. Lindell and T. Malkin. Enhanced versus Plain Trapdoor Permutations for Non-Interactive Zero-Knowledge and Oblivious Transfer. Manuscript in preparation, 2006.

[10] R. Gennaro and L. Trevisan. Lower Bounds on the Efficiency of Generic Cryptographic Constructions. In 41*st FOCS,* pages 305–314, 2000.

[11] Y. Gertner, S. Kannan, T. Malkin, O. Reingold and M. Viswanathan. The Relationship between Public Key Encryption and Oblivious Transfer. In 41*st FOCS,* pages 325–334, 2000.

[12] Y. Gertner, T. Malkin and O. Reingold. On the Impossibility of Basing Trapdoor Functions on Trapdoor Predicates. In 42*nd FOCS,* pages 126–135, 2001.

[13] O. Goldreich. *Foundations of Cryptography: Volume 2 – Basic Applications.* Cambridge University Press, 2004.

[14] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing but their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *Journal of the ACM*, 38(1):691–729, 1991.

[15] O. Goldreich, S. Micali and A. Wigderson. How to Play any Mental Game – A Completeness Theorem for Protocols with Honest Majority. In 19*th STOC,* pages 218–229, 1987.

[16] R. Impagliazzo and S. Rudich. Limits on the Provable Consequences of One-way Permutations. In *CRYPTO'88*, Springer-Verlag (LNCS 403), pages 8–26, 1988.

[17] Y.T. Kalai. Smooth Projective Hashing and Two-Message Oblivious Transfer. In *EUROCRYPT 2005*, Springer-Verlag (LNCS 3494) pages 78–95, 2005.

[18] J. Kilian. Founding Cryptograph on Oblivious Transfer. In 20*th STOC*, pages 20–31, 1988.

[19] J. Kilian. *Uses of Randomness In Algorithms and Protocols.* MIT Press, 1990.

[20] J. Kilian. Improved Efficient Arguments. In *CRYPTO'95*, Springer-Verlag (LNCS 963), pages 311–324, 1995.

[21] J.H. Kim, D.R. Simon and P. Tetali. Limits on the Efficiency of One-Way Permutation-Based Hash Functions. In *40th FOCS*, pages 535–542, 1999.

[22] E. Kushilevitz and R. Ostrovsky. Replication Is Not Needed: Single Database, Computationally-Private Information Retrieval. In 38*th FOCS*, pages 364–373, 1997.

[23] Y. Lindell. A Simpler Construction of CCA2-Secure Public-Key Encryption Under General Assumptions. In *EUROCRYPT 2003,* Springer-Verlag (LNCS 2656), pages 241–254, 2003.

[24] T. Malkin and O. Reingold. Personal communication, 2006.

[25] S. Micali. Computationally Sound Proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2000.

[26] M. Naor and K. Nissim. Communication Preserving Protocols for Secure Function Evaluation. In 33*rd STOC*, pages 590–599, 2001.

[27] M. Naor and B. Pinkas. Efficient Oblivious Transfer Protocols. In 12*th SODA*, pages 458–457, 2001.

[28] M. Rabin. How to Exchange Secrets by Oblivious Transfer. Tech. Memo TR-81, Harvard University, 1981.

[29] O. Reingold, L. Trevisan, and S. Vadhan. Notions of Reducibility between Cryptographic Primitives. In 1*st TCC*, pages 1–20, 2004.

[30] A. Sahai. Non-Malleable Non-Interactive Zero-Knowledge and Adaptive Chosen-Ciphertext Security. In 40*th FOCS*, pages 543–553, 1999.

[31] S. Wolf and J. Wullschleger. Oblivious Transfer Is Symmetric. To appear in *EUROCRYPT 2006.* Appears at *Cryptology ePrint Archive*, Report 2004/336, 2004.

[32] A. Yao. How to Generate and Exchange Secrets. In 27*th FOCS*, pages 162–167, 1986.