Single-Use Oblivious Transfer Combiners

Yuval Ishai[*] Hemanta K. Maji[†] Amit Sahai[‡] Jürg Wullschleger[§]

October 17, 2013

Abstract

An oblivious transfer (OT) protocol allows a receiver to obtain one of two bits held by a sender without revealing its selection. An *OT combiner* securely implements OT by using oracle access to n OT candidates of which at most t may be insecure. It is known that OT combiners exist when $t < n/2$. However, known constructions either invoke each candidate multiple times or alternatively require t to be a very small fraction of n, even in the semi-honest security model.

In this work we study the goal of maximizing the security level of *single-use* OT combiners in the semi-honest model, namely OT combiners in which each candidate can only be invoked once. This question is motivated by scenarios in which each OT instance is implemented via a separate physical process that may leak information independent of other instances.

Our main result is a statistically secure single-use OT combiner which tolerates $t = {}^{n}\!/_{2} - \tilde{O}(\log n)$ bad instances. We complement this by a negative result, showing that it is impossible to tolerate $t = {}^{n}\!/_{2} - O(1)$ bad instances in this setting. More generally, given n OT instances, we construct single-use OT combiners where an adversary can corrupt the sender and t_S OT instances, or it can corrupt the receiver and t_R OT instances, such that $n-(t_S+t_R) = \tilde{O}(\log n)$.

Finally, we apply our positive result and (re-prove) the semi-honest completeness of (p, q)-*Weak-OT* [DKS99] (i.e. an OT which reveals the receiver choice bit to a corrupt sender with probability p and reveals both sender bits to a corrupt receiver with probability q), where $p + q < 1$. We significantly reduce the total number of (p, q)-WOT copies needed to implement one copy of OT.

[*]Dept. of Computer Science, Technion, Haifa, Israel, yuvali@cs.technion.ac.il.
[†]University of California, Los Angeles, hemanta.maji@gmail.com.
[‡]University of California, Los Angeles, sahai@cs.ucla.edu.
[§]Google, Zurich, juerg@wulli.com.

Contents

1 Introduction

Most non-trivial cryptography is impossible against information theoretic adversaries in the plain model [IL89]; unless honest majority [BGW88, CCD88, RB89, DI06] or some trusted setup [IPS08, GIS+10] is guaranteed. Trusted setups and computational assumptions can, thus, be interpreted as denominations of non-trivial cryptography which can be morphed into other cryptographically useful primitives.

One such widely used universal denomination of non-trivial cryptography is: *2-choose-1 bit-Oblivious Transfer* (OT) [EGL82]. Oblivious transfer allows a receiver with input b to (oblivious to the sender) pick x_b from a sender's pair of inputs (x_0, x_1). This simple functionality is sufficiently sophisticated to enable interesting cryptography, for example, secure multi-party computation [Yao86, GMW87, IPS08]. Existence of OT is entailed by a) computational assumptions, like existence of enhanced/dense trapdoor permutations [Hai04] or hardness of factoring [RSA78], or b) physical setups, like precomputed non-trivial correlations [Kil00, WW06, MPR12] or hardware tokens [GIS+10].

Over time, security of each individual denomination can expire, for example computational assumptions might be falsified, correlations might be leaked or hardware units may fall under adversarial control. The security of cryptographic protocols should, therefore, be *robust*, i.e. immune to failure of components' security. Motivated by such concerns the formal study of *combiners* (and its more general variant, like correlation extractors [IKOS09]) was recently initiated by [HKN+05, Her05]. Intuitively, a combiner implements a non-trivial cryptographic task using cryptographically non-trivial components which remains secure even when some of these components fail. One of the earliest known examples of combiners is the construction of "universal" one-way function by [Lev87], i.e. a fixed function which is one-way if there exists a one-way function.

To motivate our problem statement consider the following two settings. In the first setting, each individual component implements arbitrarily many OTs; but failure of one component implies insecurity of every OT implemented by it. For example, OTs implemented based on the "hardness of factoring" computational assumption. In the second setting, each individual component implements only a simple cryptographic object, say, exactly one OT, and its failure is independent of other components. Thus, if a component gets compromised then only the OT implement by it becomes insecure. Such a setting is motivated by implementation of OT by physical processes, where some processes may fail independent of others with some probability. A combiner in the first setting shall be referred to as *multiple-use* combiner, while a combiner in the second setting shall be called *single-use* combiner.

Intuitively, the second setting provides more fine-grained corruption structure and, consequently, higher robustness should be more difficult to achieve. Hence, we expect a tradeoff between "achievable robustness" of combiners and "simplicity of components" it uses. In this work, we explore the following problem:

"What is the *maximal achievable robustness* for a secure (2-party) OT protocol where *each component implements only one OT?*"

As an additional optimization, we shall attempt to minimize the number of candidate OTs to implement one secure copy of OT.

1.1 Our Contribution

We construct a secure protocol in the clients-servers model, where there are two clients (sender and receiver) and n servers. Each server implements exactly one copy of OT. We consider semi-honest adversaries who can corrupt the sender and t_S servers, or the receiver and t_R servers. We show the following positive result (here security parameter is κ):

Informal Theorem 1 (Formally proven as Theorem 1)**.** *There exists a secure OT protocol (see Figure 1) in the clients-server model which is robust to (semi-honest) corruption of sender and t_S servers, or receiver and t_R servers, such that $t_R + t_S = n - \omega(\log \kappa)$.*

We emphasize that the techniques of Damgard et. al [DKS99] can be used to obtain the above mentioned result (see discussion in Section 1.3). But as discussed in Section 5 the number of candidate OTs needed by our construction will be significantly lower than the one obtained by "DKS-technique."

We exhibit a close relation between single-use OT combiners and public-information secret-sharing schemes with small share sizes. Using this relation, we show the following complementary negative result:

Informal Theorem 2 (Formally proven as Theorem 3)**.** *It is impossible to have single-use OT combiners which are robust to (semi-honest) corruption of sender and t_S servers, or receiver and t_R servers, such that $t_R + t_S = n - O(1)$.*

Due to these two results, our construction is (nearly) optimal.

Finally, we use our positive result an show the semi-honest completeness of (p, q)-weak OT (introduced in [DKS99]). We show that our construction is significantly efficient in the number of copies of (p, q)-weak OT needed to implement one OT.

Informal Theorem 3 (Formally proven as Theorem 4)**.** *Our combiner can be used to construct one OT from $n = \Theta(t)$ copies of (p, q)-weak OT, when $p + q < 1$, with simulation error 2^{-t}. On the other hand, the construction of [DKS99] needs $n = O(t^8)$ copies of (p, q)-weak OT to achieve the same simulation error.*

We believe that the significant efficiency improvement achieved in the setting of (p, q)-WOT illustrates how one can apply our one-time combiner result to obtain interesting new results in information theoretic cryptography with improved efficiency.

1.2 Related Work

Combiners have been implicit in several works, for example [AB81, EG83, Lev87, MM93, DK05, HL05]. Harnik et. al [HKN+05] construct multiple-use OT combiners. They also show that (transparent) black-box construction of such an OT combiner robust to corruption of half of the components is unlikely. The state of the art in multiple-use OT combiner is obtained by applying IPS-technique [IPS08] to the efficient OT-combiner of Harnik et. al [HIKN08]. Based on n components, each of which can be used to produce arbitrary number of OTs, this construction produces $\Theta(n)$ independent OT copies and is secure if a majority of these components remain secure. Combiners for private-information retrieval were proposed by Meier and Przydatek [MP06]; and cross-primitive combiners were also studied in [MPW07].

Single-use combiners are relatively new. Przydatek and Wullschleger [PW08] construct (error tolerant) single-use combiners which produce one OT but each component implements a significantly more complex cryptographic primitive: *oblivious linear function evaluation* over $\mathbb{F}_q$, where $q > 2$.

Recently, a significantly more general notion of *(correlation) extractors* was introduced by Ishai et. al [IKOS09]. They show that even if a (small) constant fraction of information is leaked about the whole system (i.e. considering all components collectively) it is possible to extract secure copies of OT. Note that the leakage need not be restricted to a small fraction of components.

1.3 Technical Overview

In the following paragraphs, we highlight the salient features of our technical contributions.

Combiner Construction. Before we proceed, let us provide an intuitive definition of combiners. An (n, t_S, t_R)-single use OT combiner is a protocol in the clients-servers model, where there are two clients sender and receiver and n servers (each implementing one OT), and is robust to semi-honest corruption of sender and t_S servers, or receiver and t_R servers.

First, let us provide an OT combiner which is based on techniques introduced by Damgård et. al [DKS99]. Suppose we are provided with n servers and each implements one copy of OT. Let $p = t_S/n$ and $q = t_R/n$; and suppose p and q are constants such that $p + q < 1$. Suppose an adversary chooses to corrupt the servers indexed by $\mathcal{S} \subseteq [n]$ and one of the clients. Parties use the servers to generated random OT correlations (ROT); and then generate a random permutation π to permute the servers. So, effectively, we now have n instances of (p, q)-weak ROT;[1] each server is randomly corrupted with probability p (if the adversary chooses to corrupt the sender) or corrupted with probability q (if the adversary chooses to corrupt the receiver). Now, we can use the protocol by Damgård et. al [DKS99] to construct an OT using these servers. But note that even if $p+q = 0.2$ their protocol uses $n = t^8$ servers to obtain one secure OT, with simulation error 2^{-t}.

Instead, we shall provide a significantly more efficient combiner. We intuitively explain one of the first steps towards our combiner construction (details are provided in Section 3). Suppose the sender picks a codeword $\mathbf{c_u} \equiv (c_{u,0}, \ldots, c_{u,n}) \xleftarrow{\$} \mathcal{C}$, where $\mathcal{C}$ is an (n, k, d) binary linear code. And the receiver picks a codeword $\mathbf{c_x} \equiv (c_{x,0}, \ldots, c_{x,n}) \xleftarrow{\$} \mathcal{C}^\perp$, where $\mathcal{C}^\perp$ is the $(n, n-k, d^\perp)$ binary linear code which is dual of $\mathcal{C}$. Note that the coordinate-wise dot-product of these two vectors have even parity, so it can correct one erasure. For $i \in [n]$, the receiver and sender feed $c_{u,i}$ and $c_{x,i}$ to server P_i and it outputs $c_{u,i} \cdot c_{x,i}$ to the receiver. The receiver can compute the erased value $c_{u,0} \cdot c_{x,0}$ from these.

Note that if the adversary corrupts the sender and a subset of servers indexed by $\mathcal{S}$ but the 0-th column of H (the generator matrix of $\mathcal{C}^\perp$) is not in the span of columns of H indexed by $\mathcal{S}$, then the value $c_{x,0}$ is perfectly hidden from the adversary. But if an adversary corrupts the receiver and

[1] A (p, q)-weak OT, represented as (p, q)-WOT is an OT which with probability p provides the receiver choice bit to the adversary (if the adversary has corrupted the sender of OT); or with probability q provides both the sender bits to the adversary (if the adversary has corrupted the receiver). Damgård et. al [DKS99] show that (p, q)-OT is semi-honest complete for every $p + q < 1$.

a subset of servers indexed by $\mathcal{S}$ then it is not evident whether $c_{u,0}$ remains hidden when $c_{x,0} = 0$.

So, we do the following modification. The sender picks $\mathbf{c_u} \xleftarrow{\$} \mathcal{C}$ and $\mathbf{c_v} \xleftarrow{\$} \mathcal{C}_{\mathsf{parity}}$;[2] and the sender picks $\mathbf{c_x} \xleftarrow{\$} \mathcal{C}^{\perp}$. Consider the new codeword $\mathbf{z}$, where $z_i = c_{u,i} \cdot c_{x,i} \oplus c_{v,i}$ for $i \in \{0\} \cup [n]$. It is a uniform distribution over $\mathcal{C}_{\mathsf{parity}}$ and can correct one erasure.

Now, we ask servers P_i to take $(c_{u,i}, c_{v,i})$ and $c_{x,i}$ as input from the clients and provide z_i as output to the receiver, for $i \in [n]$. The receiver can obtain z_0 from $(z_1, \ldots, z_n)$.

When the server corrupts the sender and servers indexed by $\mathcal{S}$ the security of $c_{x,0}$ is maintained as before. Additionally, we also get that: If the adversary corrupts the receiver and servers indexed by $\mathcal{S}$ but the 0-th column of G (the generator matrix of $\mathcal{C}$) is not in the span of columns of G indexed by $\mathcal{S}$, then the value of $c_{u,0} \oplus c_{v,0}$ is perfectly hidden from the adversary.

So, if we use $\mathcal{C}$ with high distance and dual distance, then we can construct such robust combiners. But explicit construction of such binary linear codes is a great challenge in Coding Theory. Though we show that existence of such codes with (moderately) good distance and dual distance is guaranteed and they indeed help construct perfect combiners (see Lemma 1). To circumvent this problem, we use the following trick. We consider a random binary linear code. Though such a code might not have good distance and dual distance, it shall be guaranteed (with high probability) that 0-th column of G and 0-th column of H will be linearly independent of columns in G indexed by $\mathcal{S}$ and columns in H indexed by $\mathcal{S}$, respectively.

Our construction is provided in Figure 1 and our main result is stated as Theorem 1. We compare our efficiency with the "[DKS99]-based-combiner" in Section 5.

Negative Result. For our negative result, we show that an $(n = t_S + t_R + g, t_S, t_R)$-single-use combiner implies a public-information secret sharing schemes with short secrets.

A (n, t, r)-public information secret sharing scheme is a secret sharing scheme where parties $[n]$ receive shares $X_1, \ldots, X_n$ and additionally there is a public information X_0. The secret S is hidden given the shares of any t parties and the public information. But any group of r parties can use their respective shares and the public information to reconstruct the secret S.

Given a $(n = t_S + t_R + g, t_S, t_R)$-single-use combiner, we consider the following secret sharing scheme. Consider a random execution of the combiner where clients' inputs are chosen uniformly at random. Receiver's secret bit b is defined as the secret s. Shares of the i-th party X_i is the choice bit received by the i-th server. Finally, the complete view of the sender is defined to be the public-information X_0.

The privacy of this scheme follows from the fact that the sender cannot have any advantage in guessing the choice bit of the receiver even after corrupting t_S servers. The shares of t_S parties along with the public information is identical to the view of the sender who corrupts those t_S servers.

The reconstruction guarantee follows from the claim that the sender can figure out the receiver choice bit in the $(n = t_S + t_R + g, t_S, t_R)$-single-use combiner by corrupting any $(t_S + g)$ servers. Suppose for contradiction that there is some size $(t_S + g)$ subset of servers such that even after corrupting them the sender cannot find the receiver choice bit. Then consider a two party protocol where the sender simulates the $(t_S + g)$ servers and the receiver simulates the remaining t_R servers.

[2]Binary linear code of length $(n+1)$ strings with even parity.

By privacy of the combiner, the receiver cannot violate the security of the sender. Thus, it must be the case that the sender can find the the choice bit of the receiver with close to 1 probability (otherwise, at the end of the protocol the two clients end up with correlated randomness which is impossible in the information theoretic plain model).

Now, if we have a lower-bound on the threshold gap g of this secret sharing scheme in terms of share-sizes (which are all 2) then we will be done. But this secret-sharing scheme only has statistical privacy and reconstruction guarantees. So, first, we need to generalize the lower-bound on threshold gaps by [PCX13] to the statistical setting. And then we can argue that the gap g cannot be $O(1)$ if share sizes are 2.

Efficiency of Reduction. We consider the efficiency of reducing OT to (p,q)-WOT. This functionality was shown to be complete in [DKS99]; but the efficiency of the reduction was poor. We show that our combiner can be directly applied to significantly improve the efficiency of the reduction.

Suppose we are provided with n copies of (p,q)-WOT. Then the number of server corruptions when the adversary corrupts the sender is $t_S \approx pn$ and the number of server corruptions when the adversary corrupts the receiver is $t_R \approx qn$, with very high probability. To achieve 2^{-t} simulation error, we only need to use $n = \Theta(t)$ copies of (p,q)-WOT. This is significantly lower than the bound of $n = O(t^8)$ provided by [DKS99].

2 Preliminaries

Given $k \in \mathbb{N}$, we represent the set $\{1,\dots,k\}$ by $[k]$. We represent the range $[fa, a]$, for $f \in [0,1]$, as $a \times [f,1]$. If $b-\varepsilon \leq a \leq b+\varepsilon$, then we represent it as $a = b \pm \varepsilon$, for $\varepsilon \geq 0$. For any set S, the set of all size λ subsets of S is represented by $\binom{S}{\lambda}$; and the set of all subsets of S is represented by 2^S. A matrix M with m rows and n columns is said to be an $m \times n$ matrix. Given a set of indices $I \subseteq [n]$ and an $m \times n$ matrix M, the $m \times |I|$ matrix obtained by restricting M to the columns indexed by I is represented by $M^{(\mathsf{col},I)}$, read as "column-restriction of M to I." The binary-entropy function is represented by: $H_2(x) := -x\lg(x) - (1-x)\lg(1-x)$. A sample s drawn according the distribution $\mathcal{D}$ is represented by $s \sim \mathcal{D}$.

Next, we define the clients-servers model and a single-use OT combiner.

Definition 1 (Clients-Servers Model)**.** *The network consists of $n+2$ parties: two clients, namely S and R, and n servers $P_1,\dots,P_n$. There are secure channels between the clients and between the clients and the servers.*

Functionality: The functionality $\mathcal{F}$ takes inputs from the clients S and R; and gives output only to client R.

*Adversarial Corruptions: The adversary can corrupt at most t_S servers and the client S; or at most t_R servers and client R. We refer to a protocol secure against such an adversary as (t_S,t_R)-*secure *protocol in the clients-servers model. If $t_S = t_R$, then we simply refer to it as t-secure protocol. We shall consider semi-honest adaptive adversaries with unbounded computational power by default.*

Definition 2 (Single-use OT-Combiner)**.** *A protocol in the clients-servers model is called an* (n, t_S, t_R)*-*single-use OT-combiner *if:*

1. *It is a protocol between two clients and* n *servers,*
2. *Each server* P_i *implements a copy of OT; and is invoked exactly once, and*
3. *It is a* (t_S, t_R)*-secure protocol for* $\mathcal{F}_{\mathsf{OT}}$ *in the clients-servers model.*

An (n, t, t)*-single-use OT-combiner is referred to as* (n, t)*-single-use OT-combiner.*

3 Single-use OT Combiner

The construction of our (n, t_S, t_R)-single-use OT-combiner is provided in Figure 1, where $n - (t_S + t_R) = \omega(\log \kappa)$. As a warmup proof, we begin by proving the security of (n, t_S, t_R)-single-use OT-combiner against static corruption. Next, we modify the protocol (only in one step) to make is secure against adaptive corruption with erasures.

Below, we intuitively summarize our (n, t)-single-use OT-combiner, where $t = {}^{n}\!/_{2} - \omega(\log \kappa)$. Since, OLE and OT are local renamings of each other, we explain how OLE-combiner can be obtained where each server implements an OLE functionality. Client S has private inputs (u, v) and client R has private input x.

Client S picks a codeword $\mathbf{c_u} \equiv (c_{u,0} = u, c_{u,1}, \ldots, c_{u,n}) \xleftarrow{\$} \mathcal{C}$; and a random codeword $\mathbf{c_v} \equiv (c_{v,0} = v, c_{v,1}, \ldots, c_{v,n})$ with parity 0. And client R picks a codeword $\mathbf{c_x} \equiv (c_{x,0} = x, c_{x,1}, \ldots, c_{x,n}) \xleftarrow{\$} \mathcal{C}^{\perp}$. Note that component-wise dot-product of $\mathbf{c_u}$ and $\mathbf{c_x}$ has parity 0. Thus, we have: $\oplus_{0 \le i \le n}(c_{u,i} \cdot c_{x,i} \oplus c_{v,i}) = 0$.

To communicate the output of OLE functionality $z_0 = u \cdot x \oplus v$ to client R, it suffices to transfer the vector $(z_1, \ldots, z_n)$ to client R, where $z_i = c_{u,i} \cdot c_{x,i} \oplus c_{v,i}$. Note that each z_i is an OLE of client S input $(c_{u,i}, c_{v,i})$ and client R input $c_{x,i}$. Using each server once, client R obtains $(z_1, \ldots, z_n)$; and outputs $z = z_1 \oplus \cdots \oplus z_n$.

To argue receiver privacy, assume that the adversary corrupts client S and servers indexed by $\mathcal{S} \in \binom{[n]}{t}$. Let H be the generating matrix of the code $\mathcal{C}^{\perp}$. If the 0-th column of H is independent of the columns of H indexed by $\mathcal{S}$, then receiver privacy is maintained. Because, (if this is the case then) $c_{x,0}$ is a uniform bit even after conditioning on $\{c_{x,i} : i \in \mathcal{S}\}$.

Now suppose an adversary corrupts client R and t-servers indexed by $\mathcal{S}$. Similarly, with a slightly involved argument, it can be shown that sender privacy is ensured if G (the generator matrix of the code $\mathcal{C}$) has its 0-th column independent of the columns indexed by $\mathcal{S}$. This step additionally uses the property that $\mathbf{z}$ is uniformly distributed over length $(n+1)$ binary strings of even parity even after fixing $\mathbf{c_u}$ and $\mathbf{c_x}$.

We shall show that, for a randomly generated binary linear code $\mathcal{C}$ both these properties hold with $1 - \mathsf{negl}(\kappa)$ probability.

Theorem 1. *For any* n, t_S, t_R *such that* $n - (t_S + t_R) = \omega(\log \kappa)$*, the protocol presented in Figure 1 is an* (n, t_S, t_R)*-single-use OT-combiner secure against a semi-honest adversary who statically corrupts*

$\mathsf{Combiner}(n)$:
Inputs: The clients S and R have private inputs $(s_0, s_1) \in \{0,1\}^2$ and $c \in \{0,1\}$, respectively.

1. *Correlation Generation.* For $i \in [n]$, client S samples $s_{0,i}, s_{1,i} \xleftarrow{\$} \{0,1\}$ and sends them to server P_i. Client R picks $c_i \xleftarrow{\$} \{0,1\}$ and sends it to the server P_i; and receives $s_{c_i,i}$ from the server. To make this protocol secure against adaptive corruption with erasures, server P_i erases its state after delivering $s_{c_i,i}$ to client R.
2. *Random Code Generation.* The clients S and R use a coin tossing to generate a $k \times (n+1)$ binary matrix G each of whose elements are picked independently and uniformly at random, where k is any integer in the range $[t_R + \omega(\log\kappa), n - (t_S + \omega(\log\kappa))]$. If $\mathsf{rank}(G) < k$ then abort. Otherwise, let $\mathcal{C}$ be the code corresponding to the generating matrix G. Let H be the generating matrix corresponding to the code $\mathcal{C}^\perp$.
3. *Combiner.*
   (a) Client S sets $u = s_0 \oplus s_1$ and $v = s_0$. Client R sets $x = c$.
   (b) Client S picks $\mathbf{c_u} \equiv (c_{u,0}, c_{u,1}, \ldots, c_{u,n}) \xleftarrow{\$} \mathcal{C}$, such that $c_{u,0} = u$. Let $\mathcal{C}_{\mathsf{parity}} \subseteq \{0,1\}^{n+1}$ be the (linear) code consisting of every length $(n+1)$ strings of even parity. S picks $\mathbf{c_v} \equiv (c_{v,0}, c_{v,1}, \ldots, c_{v,n}) \xleftarrow{\$} \mathcal{C}_{\mathsf{parity}}$, such that $c_{v,0} = v$.
   (c) Let $\mathcal{C}^\perp$ be the dual code of $\mathcal{C}$. Client R picks $\mathbf{c_x} \equiv (c_{x,0}, c_{x,1}, \ldots, c_{x,n}) \xleftarrow{\$} \mathcal{C}^\perp$, such that $c_{x,0} = x$.
   (d) (In parallel) For each $i \in [n]$, the clients use the (pre-computed) ROT correlations to perform an OLE computation[a] where client R receives $z_i = c_{u,i} \cdot c_{x,i} \oplus c_{v,i}$ (see Appendix D.2.2). This takes two rounds.
   (e) Client R outputs $z = z_1 \oplus \cdots \oplus z_n$.

[a]Using the fact that OLE reduces to ROT correlations.

Figure 1: (n, t_S, t_R)-Single-Use OT-Combiner, where $n - (t_S + t_R) = \omega(\log\kappa)$.

the clients and the servers. Moreover, if servers erase their state after producing their output, then this combiner is semi-honest secure against adaptive corruption.

As a corollary, we obtain:

Corollary 2. *For any n and $t = n/2 - \omega(\log\kappa)$, the protocol presented in Figure 1 is an (n,t)-single-use OT-combiner secure against a semi-honest adversary who statically corrupts the clients and the servers. Moreover, if servers erase their state after producing their output, then this combiner is semi-honest secure against adaptive corruption.*

3.1 Static Corruption

Below we provide the security proof for static corruption case. The security against adaptive corruption with erasures is proven in Section 3.2. Let $\mathcal{S} \subseteq [n]$ be the indices of servers corrupted by the adversary. We shall show that conditioned on the the following events the protocol in Figure 1 perfectly securely realizes $\mathcal{F}_{\mathsf{OT}}$.

1. $G^{(\mathsf{col},\{0\})}$ does not lie in the span of $\left\{G^{(\mathsf{col},\{i\})} : i \in \mathcal{S}\right\}$, and
2. $H^{(\mathsf{col},\{0\})}$ does not lie in the span of $\left\{H^{(\mathsf{col},\{i\})} : i \in \mathcal{S}\right\}$, where H is a generator matrix for the code $\mathcal{C}^{\perp}$.

Finally, we shall show that these two events occur with negligible probability.

No Corruptions Case. This case considers the correctness of the protocol. Suppose the adversary does not corrupt any client or server. The probability that the construction outputs abort is at most: $\mathsf{negl}(\kappa)$, by Claim 1. Conditioned on the fact that the construction does not output abort, the output of client R is:

$$\begin{aligned} z &= \bigoplus_{i=1}^{n} z_i = \bigoplus_{i=1}^{n} (c_{u,i} \cdot c_{x,i} \oplus c_{v,i}) \\ &= (u \cdot x \oplus v) \oplus \left(\bigoplus_{i=0}^{n} c_{u,i} \cdot c_{x,i}\right) \oplus \left(\bigoplus_{i=0}^{n} c_{v,i}\right) \\ &= (u \cdot x \oplus v) = (s_0 \oplus s_1)c \oplus s_0 = s_c \end{aligned}$$

Thus, the protocol is perfectly correct conditioned on the fact that G has full rank, which happens with probability $1 - \mathsf{negl}(\kappa)$ (by Claim 1).

Corrupt Sender. Suppose the adversary corrupts client S and some t_S-servers indexed by $\mathcal{S} \subseteq [n]$. The simulator does the following: It samples $\tilde{c} \xleftarrow{\$} \{0,1\}$ and uniformly reverse samples a view of client S based on sender input being (s_0, s_1) and receiver input being $\tilde{c}$. The adversary additionally sees $\mathbf{c_x}^{(\mathsf{col},\mathcal{S})}$, i.e. $\{c_{x,i} : i \in \mathcal{S}\}$. It suffices to show that the distribution $(c_{x,0} | \mathbf{c_u}, \mathbf{c_v}, \mathbf{c_x}^{(\mathsf{col},\mathcal{S})})$ is a uniform bit. Suppose, for contradiction, it is not a uniform bit. This implies that given $(\mathbf{c_u}, \mathbf{c_v}, \mathbf{c_x}^{(\mathsf{col},\mathcal{S})})$, the value of $c_{x,0}$ is deterministically 0 or 1 (by properties of binary linear codes). Which is the case, if and only if $H^{(\mathsf{col},\{0\})}$ is in the span of $\left\{H^{(\mathsf{col},\{i\})} : i \in \mathcal{S}\right\}$. Let $\mathcal{S}^* = \mathcal{S} \cup \{0\}$. The probability of this event is at most the probability of the event that: $\mathsf{rank}(H^{(\mathsf{col},\mathcal{S}^*)}) < |\mathcal{S}^*| = t_S + 1$. This is $\mathsf{negl}(\kappa)$, by Corollary 5 (because, $n - k = t_S + \omega(\log \kappa)$).

Thus, the protocol is perfectly secure against corruption of client S and t_S-servers if G has full rank and $H^{(\mathsf{col},\{0\})}$ is not in the span of $\left\{H^{(\mathsf{col},\{i\})} : i \in \mathcal{S}\right\}$. This happens with probability $1 - \mathsf{negl}(\kappa)$ (by Claim 1 and Corollary 5). Note that the simulator, as explained above, seems to need unbounded computational power. But observe that it only needs to solve linear equations and, hence, is computationally efficient.

Corrupt Receiver. Suppose the adversary corrupts client R and t_R-servers indexed by $\mathcal{S} \subseteq [n]$. The simulator does the following: It samples $\tilde{s}_{1-c}$ and uniformly reverse samples a view of client R based on client S private input being $(s_c, \tilde{s}_{1-c})$ and private input of client S being c. The adversary additionally sees $\mathbf{c_u}^{(\mathsf{col},\mathcal{S})}$ and $\mathbf{c_v}^{(\mathsf{col},\mathcal{S})}$. Define $z_0 = z$. It suffices to show that $(s_0 \oplus s_1) = c_{u,0}$ remains perfectly hidden given $(\mathbf{c_x}, \mathbf{z}, \mathbf{c_u}^{(\mathsf{col},\mathcal{S})}, \mathbf{c_v}^{(\mathsf{col},\mathcal{S})})$.

First observe that $c_{v,i}$ is fixed given $c_{u,i}$, $c_{x,i}$ and z_i, for any i. Thus, it suffices to show that $c_{u,0}$ remains perfectly hidden given $(\mathbf{c_x}, \mathbf{z}, \mathbf{c_u}^{(\mathsf{col},\mathcal{S})})$. Consider the distribution $(\mathbf{c_u} | \mathbf{c_x}, \mathbf{z}, \mathbf{c_u}^{(\mathsf{col},\mathcal{S})})$.

$$\begin{aligned}
\mathrm{P}(\mathbf{c_u}|\mathbf{c_x}, \mathbf{z}, \mathbf{c_u}^{(\mathsf{col},\mathcal{S})}) &= \mathrm{P}(\mathbf{c_u}|\mathbf{c_u}^{(\mathsf{col},\mathcal{S})}) \times \frac{\mathrm{P}(\mathbf{c_x}, \mathbf{z}|\mathbf{c_u}, \mathbf{c_u}^{(\mathsf{col},\mathcal{S})})}{\mathrm{P}(\mathbf{c_x}, \mathbf{z}|\mathbf{c_u}^{(\mathsf{col},\mathcal{S})})} \\
&= \mathrm{P}(\mathbf{c_u}|\mathbf{c_u}^{(\mathsf{col},\mathcal{S})}) \times \frac{\mathrm{P}(\mathbf{c_x}, \mathbf{z}|\mathbf{c_u})}{\mathrm{P}(\mathbf{c_x}, \mathbf{z}|\mathbf{c_u}^{(\mathsf{col},\mathcal{S})})} \\
&= \mathrm{P}(\mathbf{c_u}|\mathbf{c_u}^{(\mathsf{col},\mathcal{S})}) \times \frac{\mathrm{P}(\mathbf{c_x}|\mathbf{c_u})\,\mathrm{P}(\mathbf{z}|\mathbf{c_x}, \mathbf{c_u})}{\mathrm{P}(\mathbf{c_x}|\mathbf{c_u}^{(\mathsf{col},\mathcal{S})})\,\mathrm{P}(\mathbf{z}|\mathbf{c_x}, \mathbf{c_u}^{(\mathsf{col},\mathcal{S})})} \\
&= \mathrm{P}(\mathbf{c_u}|\mathbf{c_u}^{(\mathsf{col},\mathcal{S})})
\end{aligned}$$

The last equality is due to the fact that $\mathbf{z}$ is uniform over $\mathcal{C}_{\mathsf{parity}}$ even if $\mathbf{c_x}$ and $\mathbf{c_u}$ are fixed; and from the fact that $\mathbf{c_x}$ is independent of $\mathbf{c_u}$. Thus, $c_{u,0}$ is perfectly hidden if and only if $G^{(\mathsf{col},\{0\})}$ is not in the span of $\left\{G^{(\mathsf{col},\{i\})} : i \in \mathcal{S}\right\}$ (this happens with $1 - \mathsf{negl}(\kappa)$ probability, by Corollary 6, if $k = t_R + \omega(\log \kappa)$). Similar to the previous case, the simulator mentioned above is efficient.

No Corrupt Client and Only Corrupt Servers. Note that the view of each server is a ROT correlation, so it can be simulated trivially. To argue privacy for this case, it is subsumed by the case when, additionally, one of the clients in also corrupted.

3.2 Adaptive Security with Erasure

We now modify the protocol in Figure 1 so that it is secure against adaptive corruption with erasures. The only additional step needed is for each server P_i to erase their internal state after it has delivered its output $s_{c_i,i}$ to client R in Correlation Generation phase. Thus, without loss of generality any server corruption can be assumed to happen before the Random Code Generation phase of the combiner.

Next, we argue the security of this protocol. The crucial ingredient for this is the fact that OLE protocol in ROT hybrid as described in Appendix D.2.2 is secure against adaptive corruption of parties. Note that, since all server corruptions take place before the Random Code Generation phase, with $1 - \mathsf{negl}(\kappa)$ we have $\mathsf{rank}(G) = k$, $G^{(\mathsf{col},0)}$ is not in the span of $\{G^{(\mathsf{col},i)} : i \in S\}$ and $H^{(\mathsf{col},0)}$ is not in the span of $\{H^{(\mathsf{col},i)} : i \in S\}$. We shall argue the security of the protocol conditioned on the above mentioned event.

So, all that remains is to show the simulations for various points in time where corruption of client S and R takes place.

Our simulator shall generate random ROT correlations as views of each server P_i, say $(s_{0,i}, s_{1,i})$ and $(c_i, s_{c_i,i})$. And update them, if necessary, only during the simulation of Combiner phase. If a server is corrupted prior to erasure of its state then its view is provided to the adversary; otherwise it is provided with an empty view. The simulation of Random Code Generation phase is trivial. Thus, it suffices to consider adaptive corruption of clients just at the completion of Correlation Generation phase and the completion of Combiner phase.

1. Suppose all client corruptions happen by the end of Correlation Generation phase. The simulator provides the set $\{(s_{0,i}, s_{1,i}) : i \in [n]\}$ as view of client S and $\{(c_i, s_{c_i}) : i \in [n]\}$

as view of client R. It obtains their respective private inputs and sends it to the external functionality.

2. Suppose all client corrupts happen at the end of the Combiner phase. Suppose $\mathcal{S}$ is the set of servers corrupted by the adversary. Note that at this point $\mathbf{c_u}^{(\mathsf{col},\mathcal{S})}$, $\mathbf{c_v}^{(\mathsf{col},\mathcal{S})}$ and $\mathbf{c_x}^{(\mathsf{col},\mathcal{S})}$ are all fixed.

   When client S is corrupted, obtain the inputs (s_0, s_1) and send it to the external functionality. Further, pick random codewords $\mathbf{c_u}$ and $\mathbf{c_v}$ consistent with $\mathbf{c_u}^{(\mathsf{col},\mathcal{S})}$ and $\mathbf{c_v}^{(\mathsf{col},\mathcal{S})}$, such that $c_{u,0} = u$ and $c_{v,0} = v$. This is possible because $c_{u,0}$ and $c_{v,0}$ are perfectly hidden given the adversary's view. Now resample $\{(s_{0,i}, s_{1,i}) : i \notin \mathcal{S}\}$ consistent with the adversary's view and the codewords $\mathbf{c_u}$ and $\mathbf{c_v}$. This resampling is possible because OLE reduction to ROT-correlation is secure against adaptive corruption of the parties [Lin09].

   When client R is corrupted, obtain the input c and send it to the external functionality. Further, pick random codewords $\mathbf{c_x}$ consistent with $\mathbf{c_x}^{(\mathsf{col},\mathcal{S})}$, such that $c_{x,0} = x$. This is possible because $c_{x,0}$ is perfectly hidden given the adversary's view. Now resample $\{(c_i, s_{c_i,i}) : i \notin \mathcal{S}\}$ consistent with the adversary's view and codeword $\mathbf{c_x}$. This resampling is possible because OLE reduction to ROT correlation is secure against adaptive corruption of the parties [Lin09].

3. Client S is corrupted at the end of Correlation Generation phase and client R is corrupted at the end of Combiner phase. The view of client S is generated as in Step 1 and view of client R is generated as in Step 2.

4. Client R is corrupted at the end of Correlation Generation phase and client S is corrupted at the end of Combiner phase. The view of client R is generated as in Step 1 and view of client S is generated as in Step 2.

3.3 Perfect Security

Note that if the distances d and $d^{\perp}$, respectively, of the codes $\mathcal{C}$ and $\mathcal{C}^{\perp}$ are both $> t$, then an adversary cannot corrupt any t servers and one of the clients to violate the security of the other client. By Gilbert-Varshamov bound for linear matrices, we know that there exists a code (for sufficiently large n and with probability $1 - \mathsf{negl}(\kappa)$) which has $H_2(d/n) \geq 1 - (k/(n+1)) - \varepsilon$, for every constant $\varepsilon > 0$ and $k = \lfloor n/2 \rfloor$. By union bound, with probability $1 - \mathsf{negl}(\kappa)$, both $\mathcal{C}$ and $\mathcal{C}^{\perp}$ satisfy $H_2(\min\{d, d^{\perp}\}/n) \geq \frac{1}{2} - \frac{1}{n+1} - \varepsilon$. In particular, there exists a binary linear code $\mathcal{C}$ with $\min\{d, d^{\perp}\} \geq 0.11n$. So, our combiner in Figure 1 when instantiated with this code shall be robust to semi-honest corruption of one client and at most $0.11n$ servers.

Lemma 1. *There exists a binary $(n, \lfloor n/2 \rfloor, d)$-linear code $\mathcal{C}$ such that $\min\{d, d^{\perp}\} \geq 0.11n$; and there exists a $(n, 0.11n)$-single use OT combiner with perfect security against semi-honest adversaries.*

4 Tightness Result for Single-use OT-Combiners

In this section we shall show that existence of $(n = t_S + t_R + g, t_S, t_R)$-single-use OT-combiner implies that $g = \omega(1)$. Assume, without loss of generality, that $t_S \leq t_R$. We construct a *public-information secret-sharing scheme* with binary shares which has t_S-privacy and $(t_S + g)$-reconstruction from any

$(n = t_S + t_R + g, t_S, t_R)$-single-use OT-combiner. The impossibility is shown by exhibiting lower bounds on the threshold gap g of secret sharing schemes in terms of share-sizes.

Note that when privacy and reconstruction in secret sharing scheme is perfect, then a lower bound on the threshold gap as shown by [PCX13] suffices. But, since we shall consider combiners which have statistical security, the public-information secret sharing scheme generated above cannot be guaranteed to have perfect privacy or reconstruction. Therefore, we need to generalize the lower bound on threshold gaps of secret sharing schemes to the case where privacy and reconstruction guarantees hold statistically.

So, in Appendix B, we prove a lower bound (see Lemma 2) on the threshold gap of statistical public-information secret sharing schemes. Finally, in Section 4.2 we construct a public-information secret-sharing scheme from any single-use OT combiner. And conclude that it is impossible for to have $(n = t_S + t_R + g, t_S, t_R)$-single-use OT-combiner if $g = O(1)$. Further, for $g = O(1)$, $1 - o(1)$ fraction of the servers need to be invoked at least $\omega(\log t_R)$ times.

4.1 Definition

First we define secret sharing schemes.

Definition 3 ((n, t, r)-Secret-sharing scheme with (ε, δ)-security)**.** *Consider alphabet sets* $[\Lambda], [\Lambda_1], \ldots, [\Lambda_n]$ *and a joint distribution* $\mathfrak{S}{:=}(\mathbf{S}, \mathbf{X}_1, \ldots, \mathbf{X}_n)$ *over the space* $[\Lambda] \times [\Lambda_1] \cdots \times [\Lambda_n]$*. The random variable* $\mathbf{S}$ *represents the* secret *being shared and* $\mathbf{X}_i$*, for* $i \in [n]$*, represents the* i*-th* share*. For* $s \in [\Lambda]$ *and a set* $T = \{i_1, \ldots, i_k\} \subseteq [n]$*, the conditional distribution* $(\mathbf{X}_T | \mathbf{S} = s)$ *is defined as the conditional distribution over* $(\mathbf{X}_{i_1}, \ldots, \mathbf{X}_{i_j})$ *as induced by* $\mathfrak{S}$ *when* $\mathbf{S} = s$*. The joint distribution* $\mathfrak{S}$ *is an* (n, t, r)*-Secret-sharing scheme with* (ε, δ)*-security if the following conditions are satisfied:*

1. $(1 - \varepsilon)$ *t-privacy: For any* $s, s' \in [\Lambda]$ *and* $T \subseteq [n]$*, such that* $|T| \leq t$*, we have:*

$$\mathbf{\Delta}\left((\mathbf{X}_T | \mathbf{S} = s), (\mathbf{X}_T | \mathbf{S} = s')\right) \leq \varepsilon$$

2. $(1 - \delta)$ *r-reconstruction: For any distinct* $s, s' \in [\Lambda]$ *and* $T \subseteq [n]$*, such that* $|T| = r$*, we have:*

$$\mathbf{\Delta}\left((\mathbf{X}_T | \mathbf{S} = s), (\mathbf{X}_T | \mathbf{S} = s')\right) \geq 1 - \delta$$

Definition 4 ((n, t, r)-Public-Information Secret-sharing scheme with (ε, δ)-security)**.** *Consider a joint distribution* $\mathfrak{S}{:=}(\mathbf{S}, \mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_n)$ *over the space* $[\Lambda] \times [\Lambda_0] \times [\Lambda_1] \cdots \times [\Lambda_n]$*. The random variable* $\mathbf{S}$ *represents the* secret *being shared and* $\mathbf{X}_i$*, for* $i \in [n]$*, represents the* i*-th* share*; and* $\mathbf{X}_0$ *represents the public information. The joint distribution* $\mathfrak{S}$ *is an* (n, t, r)*-Public-information Secret-sharing scheme with* (ε, δ)*-security if the following conditions are satisfied:*

1. $(1 - \varepsilon)$ *t-privacy: For any* $s, s' \in [\Lambda]$ *and* $T \subseteq \{0\} \cup [n]$*, such that* $|T \cap [n]| \leq t$*, we have:*

$$\mathbf{\Delta}\left((\mathbf{X}_T | \mathbf{S} = s), (\mathbf{X}_T | \mathbf{S} = s')\right) \leq \varepsilon$$

2. $(1 - \delta)$ *r-reconstruction: For any distinct* $s, s' \in [\Lambda]$ *and* $T \subseteq \{0\} \cup [n]$*, such that* $0 \in T$ *and* $|T \cap [n]| = r$*, we have:*

$$\mathbf{\Delta}\left((\mathbf{X}_T | \mathbf{S} = s), (\mathbf{X}_T | \mathbf{S} = s')\right) \geq 1 - \delta$$

We shall show the following lower bound on the threshold gap of (n,t,r)-public-information secret-sharing schemes:

Lemma 2 (Lower Bound for (n,t,r)-Public-information Secret Sharing schemes with (ε,δ)-security)**.** *Let $\mathfrak{S} = (\mathbf{S},\mathbf{X}_1,\ldots,\mathbf{X}_n)$ be a (n,t,r)-Secret-sharing scheme with (ε,δ)-security. If $\varepsilon+\delta = o\left(\Lambda^{-4}\binom{n-t+1}{r-t+1}^{-2}\right)$, then*

$$(r-t) \geq \max_{\substack{I\subseteq\{0\}\cup[n] \\ \wedge\ |I\cap[n]|=t-1 \\ \wedge\ 0\in I}} \left(\sum_{i\in[n]\setminus I}\frac{1}{\Lambda_i}\right) - (n-t+1)\left(\frac{\varepsilon}{2}+\binom{n-t+1}{r-t+1}\sqrt{\varepsilon+\delta}\right)$$

The proof of this lemma in included in Appendix B.3.

4.2 Lower Bound on (n,t_S,t_R)-single-use OT-combiners

Given a $(n = t_S+t_R+g, t_S, t_R)$-single-use OT-combiner we shall construct a (n, t_S, t_S+g)-public-information secret-sharing scheme with $(\nu(t),\nu(t))$-security, where $\nu(\cdot)$ is a negligible functions. Assume that $t_S \leq t_R$; otherwise swap the roles of client S and client R. To construct the secret sharing scheme, we execute the combiner with random private inputs for the clients (with uniformly chosen, respective, local random tapes for the clients). The secret s is the choice bit of client R. The public-information is the complete view of client S. And the share X_i, where $i\in[n]$, is the choice bit sent to server P_i by client R.

We shall leverage security of the combiner to claim that the public-information along with the choice bits of t_S servers is insufficient to predict s with any advantage. Because the choice bit of client R is hidden from any adversary who corrupts client S and any t_S servers.

Next, we show that the public-information along with any (t_S+g) shares suffice to reconstruct the secret. Fix any (t_S+g) subset of servers. Consider a two-party protocol in the plain model where party A simulates the execution of party S and these (t_S+g)-servers; and party B simulates the execution of party R and the remaining t_R-servers. By the security of the combiner, party B has no advantage in predicting $s_0\oplus s_1$, where (s_0,s_1) are private inputs for client S. Since, there is no two-party protocol for $\mathcal{F}_{\mathsf{OT}}$ in the plain model, party A must be able to predict the choice bit of party B with (near) certainty. Thus, the public-information and any (t_S+g) shares are sufficient to predict the secret. Formally,

Theorem 3. *Suppose π is a $(n = t_S+t_R+g, t_S, t_R)$-single-use OT-combiner. Then, $g=\omega(1)$. Further, if $g=O(1)$ then it is impossible to have $\omega(1)$ servers which are invoked only $O(1)$ times.*

Proof. Suppose π is a $(n = t_S+t_R+g, t_S, t_R)$-single-use OT-combiner. The generation of joint distribution $\mathfrak{S}$ is defined in Figure 2.

The remaining proof is provided in Appendix B.4. □

1. If $t_R < t_S$, swap the roles of client S and client R. Now the servers implement the TOR-correlation (see Appendix D.2.1) which can be locally renamed to ROT correlation. The received choice bit of a server P_i is the received choice bit in this renamed ROT correlation.
2. Execute the protocol π starting with client S local randomness $r_S \xleftarrow{\$} \mathbf{U}$ and private input $(s_0, s_1) \xleftarrow{\$} \{0,1\}^2$; and client R local randomness $r_R \xleftarrow{\$} \mathbf{U}$ and private input $c \xleftarrow{\$} \{0,1\}$. Define $s = c$. Define $X_0 = V_S$, where V_S is the complete view of client S. For all $i \in [n]$, define $X_i = c_i$, where c_i is the choice bit sent to the server P_i by client R.
3. Output $(s, X_0, \ldots, X_n)$.

Figure 2: Construction of Public-information Secret Sharing schemes from Single-use Combiners.

5 Completeness of Weak Oblivious Transfer

Definition 5 ((p,q) Weak Oblivious Transfer). *A (p,q) Weak Oblivious Transfer (written as (p,q)-WOT) is the following functionality. It takes two inputs (s_0, s_1) from the sender and a choice bit c from the receiver and provides s_c to the receiver. If the adversary has corrupted the sender then with probability p it sends c to the adversary. If the adversary has corrupted the receiver then with probability q it sends s_{1-c} to the adversary.*

A (p,q)-WROT is a (p,q)-WOT with random inputs.

Damgård et. al [DKS99] show that for any constant $p, q \in (0,1)$ such that $p + q < 1$, the functionality (p,q)-WOT is semi-honest complete. We shall show the same result but with significant efficiency improvement. The number of copies of (p,q)-WOT needed to semi-honest securely implement one copy of OT with simulation error 2^{-t} decreases significantly if we use the combiner provided in Figure 1 (each server shall implement one copy of (p,q)-WOT). These efficiency improvements are summarized below:

Theorem 4 (Efficiency). *Using the combiner in Figure 1, we can construct a secure OT protocol with simulation error 2^{-t} where every server implements a copy of (p,q)-WOT for a suitable choice of n. The choice of n is explained below:*

1. *If $p + q \leq 0.2$, then $n = \Theta(t)$ suffices for our combiner. Using the technique of [DKS99] we need $n = O(t^8)$ to implement one OT with simulation error 2^{-t}.*
2. *If $p + q = 1 - 1/k$, then $n = \Theta(kt)$ suffices for our combiner. Using the technique of [DKS99] we need $n \gg O(k^{7.6} t^8)$ to implement one OT with simulation error 2^{-t}.*

The proof is provided in Appendix C.

6 Conclusions and Open Problems

In this work we construct single-use OT combiners which exhibit optimal robustness. But there are some interesting efficiency issues left open. Potential optimizations include: a) Reduction of communication complexity between the clients to $O(n)$ (or even $o(n)$); and b) Generation of larger number of (parallel and independent) OTs. Finally, it will be interesting to explore whether such combiners can also be made error-tolerant.

One of the biggest open problems in the field of combiners is to explore the construction of an OT protocol based on two different computational assumptions (each of which individually imply secure OT protocols), which remains secure even if one of these two assumptions is falsified.

References

[AB81] C. A. Asmuth and G. R. Blakley. An efficient algorithm for constructing a cryptosystem which is harder to break than two other cryptosystems. *Comp. and Maths. with Appls.*, 7:447–450, 1981. 2

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In Janos Simon, editor, *STOC*, pages 1–10. ACM, 1988. 1

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In Janos Simon, editor, *STOC*, pages 11–19. ACM, 1988. 1

[DI06] Ivan Damgård and Yuval Ishai. Scalable secure multiparty computation. In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 501–520. Springer, 2006. 1

[DK05] Yevgeniy Dodis and Jonathan Katz. Chosen-ciphertext security of multiple encryption. In Joe Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 188–209. Springer, 2005. 2

[DKS99] Ivan Damgård, Joe Kilian, and Louis Salvail. On the (im)possibility of basing oblivious transfer and bit commitment on weakened security assumptions. In Jacques Stern, editor, *EUROCRYPT*, volume 1592 of *Lecture Notes in Computer Science*, pages 56–73. Springer, 1999. 1, 2, 3, 4, 5, 13, 23, 25

[EG83] Shimon Even and Oded Goldreich. On the power of cascade ciphers. In David Chaum, editor, *CRYPTO*, pages 43–50. Plenum Press, New York, 1983. 2

[EGL82] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO*, pages 205–210. Plenum Press, New York, 1982. 1

[GIS$^+$10] Vipul Goyal, Yuval Ishai, Amit Sahai, Ramarathnam Venkatesan, and Akshay Wadia. Founding cryptography on tamper-proof hardware tokens. In Daniele Micciancio, editor, *TCC*, volume 5978 of *Lecture Notes in Computer Science*, pages 308–326. Springer, 2010. 1

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In Alfred V. Aho, editor, *STOC*, pages 218–229. ACM, 1987. 1

[Hai04] Iftach Haitner. Implementing oblivious transfer using collection of dense trapdoor permutations. In Moni Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 394–409. Springer, 2004. 1

[Her05] Amir Herzberg. On tolerant cryptographic constructions. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 172–190. Springer, 2005. 1

[HIKN08] Danny Harnik, Yuval Ishai, Eyal Kushilevitz, and Jesper Buus Nielsen. Ot-combiners via secure computation. In Ran Canetti, editor, *TCC*, volume 4948 of *Lecture Notes in Computer Science*, pages 393–411. Springer, 2008. 2

[HKN$^+$05] Danny Harnik, Joe Kilian, Moni Naor, Omer Reingold, and Alon Rosen. On robust combiners for oblivious transfer and other primitives. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 96–113. Springer, 2005. 1, 2

[HL05] Susan Hohenberger and Anna Lysyanskaya. How to securely outsource cryptographic computations. In Joe Kilian, editor, *TCC*, volume 3378 of *Lecture Notes in Computer Science*, pages 264–282. Springer, 2005. 2

[IKOS09] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Extracting correlations. In *FOCS*, pages 261–270. IEEE Computer Society, 2009. 1, 3

[IL89] Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *FOCS*, pages 230–235. IEEE, 1989. 1

[IPS08] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Founding cryptography on oblivious transfer - efficiently. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 572–591. Springer, 2008. 1, 2

[Kil00] Joe Kilian. More general completeness theorems for secure two-party computation. In *STOC*, pages 316–324, 2000. 1

[Lev87] Leonid A. Levin. One-way functions and pseudorandom generators. *Combinatorica*, 7(4):357–363, 1987. 1, 2

[Lin09] Yehuda Lindell. Adaptively secure two-party computation with erasures. *IACR Cryptology ePrint Archive*, 2009:31, 2009. 10, 28

[MM93] Ueli M. Maurer and James L. Massey. Cascade ciphers: The importance of being first. *J. Cryptology*, 6(1):55–61, 1993. 2

[MP06] Remo Meier and Bartosz Przydatek. On robust combiners for private information retrieval and other primitives. In Cynthia Dwork, editor, *CRYPTO*, volume 4117 of *Lecture Notes in Computer Science*, pages 555–569. Springer, 2006. 3

[MPR12] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. A unified characterization of completeness and triviality for secure function evaluation. In Steven D. Galbraith and Mridul Nandi, editors, *INDOCRYPT*, volume 7668 of *Lecture Notes in Computer Science*, pages 40–59. Springer, 2012. 1

[MPW07] Remo Meier, Bartosz Przydatek, and Jürg Wullschleger. Robuster combiners for oblivious transfer. In Salil P. Vadhan, editor, *TCC*, volume 4392 of *Lecture Notes in Computer Science*, pages 404–418. Springer, 2007. 3

[PCX13] Ignacio Cascudo Pueyo, Ronald Cramer, and Chaoping Xing. Bounds on the threshold gap in secret sharing and its applications. *IEEE Transactions on Information Theory*, 59(9):5600–5612, 2013. 5, 11

[PW08] Bartosz Przydatek and Jürg Wullschleger. Error-tolerant combiners for oblivious primitives. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP (2)*, volume 5126 of *Lecture Notes in Computer Science*, pages 461–472. Springer, 2008. 3

[RB89] Tal Rabin and Michael Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In David S. Johnson, editor, *STOC*, pages 73–85. ACM, 1989. 1

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978. 1

[WW06] Stefan Wolf and Jürg Wullschleger. Oblivious transfer is symmetric. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 222–232. Springer, 2006. 1

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, pages 162–167. IEEE Computer Society, 1986. 1

A Semi-honest Combiner

In this section we prove two results which prove the security of the combiner provided in Figure 1.

Corrupt Sender Experiment. Consider the following two experiments:

$\mathsf{Experiment}_1(n, k, \lambda, \Lambda)$:
Precondition: $0 < n$, $1 \le k < n$, $1 \le \lambda < n - k$ and $\Lambda = \{1\} \cup \Lambda'$, where $\Lambda' \in \binom{[n] \setminus \{1\}}{\lambda - 1}$.

1. Pick a random $k \times n$ binary matrix $G \xleftarrow{\$} \{0,1\}^{k \times n}$.
2. If $\mathsf{rank}(G) < k$ then output abort_1.
3. Else, i.e. $\mathsf{rank}(G) = k$, let $\mathcal{C}$ be the $[n, k]$ code corresponding to the generator matrix G; and its dual code be $\mathcal{C}^\perp$. Pick H as a random $(n-k) \times n$ generator matrix for the code $\mathcal{C}^\perp$.
4. Let $H^{(\mathsf{col},\Lambda)}$ be the $(n-k) \times \lambda$ binary sub-matrix of H formed by the *columns* of H indexed by Λ. Output $(H, \mathsf{rank}(H^{(\mathsf{col},\Lambda)}))$.

$\mathsf{Experiment}_2(n, k, \lambda, \Lambda)$:
Precondition: $0 < n$, $1 \le k < n$, $1 \le \lambda < n - k$ and $\Lambda = \{1\} \cup \Lambda'$, where $\Lambda' \in \binom{[n] \setminus \{1\}}{\lambda - 1}$.

1. Pick a random $(n-k) \times \lambda$ binary matrix $\tilde{H}$.
2. Pick a random $(n-k) \times n$ binary matrix H such that $H^{(\mathsf{col},\Lambda)} = \tilde{H}$.
3. If $\mathsf{rank}(H) < n - k$ output abort_2.
4. Else, i.e. $\mathsf{rank}(H) = n - k$, output $(H, \mathsf{rank}(\tilde{H}))$.

We shall show that these two experiments produce outputs which are identical with probability close to 1 for suitable choices of parameters k and λ, for every choice of Λ. When $k = \omega(\log \kappa)$, $n - k = \omega(\log \kappa)$ and $\lambda = (n-k) - \omega(\log \kappa)$, we have:

1. Probability of abort_1 is *small.* The probability of abort_1 is equal to the probability that k random binary vectors of length n (which form the rows of G) are not all linearly independent.

   Claim 1. *The probability that k random binary vectors of length n are linearly dependent is* $< \frac{1}{2^{n-k}}$.

   Proof. This probability is at most: $\frac{1}{2^n}\left(1 + 2 + \cdots + 2^{k-1}\right) < 2^{-(n-k)}$ □

   Therefore, probability of abort_1 is $\mathsf{negl}(\kappa)$.

2. Conditioned on $\neg\mathsf{abort}_1$, the first coordinate of output of $\mathsf{Experiment}_1$ is *identical* to the distribution of random $(n-k) \times n$ binary matrices with rank $(n-k)$. This is trivial to see.

3. Probability of abort_2 is *small*. The experiment simply samples a random H (by first sampling $\tilde{H} = H^{(\mathsf{col},\Lambda)}$; and then extending it). We know that the probability of abort_2, thus, is at most $2^{-(n-(n-k))} = \mathsf{negl}(\kappa)$ (by Claim 1).

4. Conditioned on $\neg\mathsf{abort}_2$, the first coordinate of output of $\mathsf{Experiment}_2$ is *identical* to the distribution of random $(n-k) \times n$ binary matrices with rank $(n-k)$. This is also trivial to see.

5. Outputs of $\mathsf{Experiment}_1$ conditioned on $\neg\mathsf{abort}_1$ and $\mathsf{Experiment}_2$ conditioned on $\neg\mathsf{abort}_2$ are identically distributed. Alternately, $(\mathsf{Experiment}_1|\neg\mathsf{abort}_1) \equiv (\mathsf{Experiment}_2|\neg\mathsf{abort}_2)$. This follows from the fact that the second output coordinate is the same function of the first output coordinate in both experiments.

We want to claim the following:

Claim 2. *If $k = \omega(\log\kappa)$, $n-k = \omega(\log\kappa)$ and $(n-k)-\lambda = \omega(\log\kappa)$, then:*

$$\mathrm{P}\left[\mathsf{rank}\left(H^{(\mathsf{col},\Lambda)}\right) = \lambda\right] = 1 - \mathsf{negl}(\kappa)$$

Proof. Note that, in $\mathsf{Experiment}_2$ the probability of $\mathsf{rank}(\tilde{H}) < \lambda$ is $2^{-((n-k)-\lambda)} = \mathsf{negl}(\kappa)$, by Claim 1. Consider the following manipulation:

$$\begin{aligned}
& & \mathrm{P}\left[\mathsf{rank}(\tilde{H}) = \lambda\right] &= 1 - \mathsf{negl}(\kappa) & & \\
\implies & & \mathrm{P}\left[\mathsf{rank}(\tilde{H}) = \lambda \ \wedge\ \neg\mathsf{abort}_2\right] &= 1 - \mathsf{negl}(\kappa), & & \because \mathrm{P}[\mathsf{abort}_2] = \mathsf{negl}(\kappa) \\
\implies & & \mathrm{P}\left[\mathsf{rank}(\tilde{H}) = \lambda|\neg\mathsf{abort}_2\right] &= 1 - \mathsf{negl}(\kappa), & & \because \mathrm{P}[\mathsf{abort}_2] = \mathsf{negl}(\kappa) \\
\implies & & \mathrm{P}\left[\mathsf{rank}(H^{(\mathsf{col},\Lambda)}) = \lambda|\neg\mathsf{abort}_1\right] &= 1 - \mathsf{negl}(\kappa), & & \because (\mathsf{Experiment}_1|\neg\mathsf{abort}_1) \\
& & & & & \equiv (\mathsf{Experiment}_2|\neg\mathsf{abort}_2) \\
\implies & & \mathrm{P}\left[\mathsf{rank}(H^{(\mathsf{col},\Lambda)}) = \lambda \ \wedge\ \neg\mathsf{abort}_1\right] &= 1 - \mathsf{negl}(\kappa), & & \because \mathrm{P}[\mathsf{abort}_1] = \mathsf{negl}(\kappa) \\
\implies & & \mathrm{P}\left[\mathsf{rank}(H^{(\mathsf{col},\Lambda)}) = \lambda\right] &= 1 - \mathsf{negl}(\kappa) & &
\end{aligned}$$

We clarify that, in the above mentioned expressions, whenever events $H^{(\mathsf{col},\Lambda)}$ and abort_1 occur in the probability expression, it is implicit that the underlying distribution over the sample space is the one induced by the output of $\mathsf{Experiment}_1$. Similarly, when the events $\tilde{H}$ and abort_2 occur in the probability expression, it is implicit that the underlying distribution over the sample space is the one induced by the output of $\mathsf{Experiment}_2$. □

In particular:

Corollary 5. *For $k \in [t_R + \omega(\log\kappa), n - (t_S + \omega(\log\kappa))]$, $|\Lambda| = \lambda = t_S + 1$, the probability that $\mathsf{rank}(H^{(\mathsf{col},\Lambda)}) = \lambda$ is at least $1 - \mathsf{negl}(\kappa)$.*

Corrupt Receiver Experiment. We consider two experiments:

$\mathsf{Experiment}_1(n, k, \lambda, \Lambda)$: *Precondition*: $0 < n$, $1 \leq k < n$, $1 \leq \lambda < n - k$ and $\Lambda = \{1\} \cup \Lambda'$, where $\Lambda' \in \binom{[n] \setminus \{1\}}{\lambda - 1}$.

1. Pick a random $k \times n$ binary matrix $G \xleftarrow{\$} \{0,1\}^{k \times n}$.
2. If $\mathsf{rank}(G) < k$ then output abort_1.
3. Else, i.e. rank (G)=k, output $(G, \mathsf{rank}(G^{(\mathsf{col},\Lambda)}))$.

$\mathsf{Experiment}_2(n, k, \lambda, \Lambda)$: *Precondition*: $0 < n$, $1 \leq k < n$, $1 \leq \lambda < n - k$ and $\Lambda = \{1\} \cup \Lambda'$, where $\Lambda' \in \binom{[n] \setminus \{1\}}{\lambda - 1}$.

1. Pick a random $k \times \lambda$ binary matrix $\tilde{G}$.
2. Pick a random $k \times n$ binary matrix G such that $G^{(\mathsf{col},\Lambda)} = \tilde{G}$.
3. If $\mathsf{rank}(G) < k$ output abort_2.
4. Else, i.e. $\mathsf{rank}(G) = k$, output $(G, \mathsf{rank}(\tilde{G}))$.

We shall show that these two experiments produce outputs which are identical with probability $1 - \mathsf{negl}(\kappa)$, when $k = \omega(\log \kappa)$ and $n - k = \omega(\log \kappa)$. The argument is similar to the argument presented in previous section.

1. Probability of abort_1 is $1 - \mathsf{negl}(\kappa)$, by Claim 1.
2. Probability of abort_2 is $1 - \mathsf{negl}(\kappa)$, by Claim 1.
3. Conditioned on $\neg\mathsf{abort}_1$, the first coordinate of output of $\mathsf{Experiment}_1$ is identical to the distribution of random (k, n) binary matrices with rank k. And, conditioned on $\neg\mathsf{abort}_2$, the first coordinate of output of $\mathsf{Experiment}_2$ is identical to the distribution of random (k, n) binary matrices with rank k. Thus, $(\mathsf{Experiment}_1 | \neg\mathsf{abort}_1) \equiv (\mathsf{Experiment}_2 | \neg\mathsf{abort}_2)$ because the second output coordinate is identical function of the first coordinate in both these experiments.

We want to claim the following:

Claim 3. *If $k = \omega(\log \kappa)$, $n - k = \omega(\log \kappa)$ and $\lambda = k - \omega(\log \kappa)$, then:*

$$\mathrm{P}\left[\mathsf{rank}\left(G^{(\mathsf{col},\Lambda)} = \lambda\right)\right] = 1 - \mathsf{negl}(\kappa)$$

Proof. Note that $\mathrm{P}[\mathsf{rank}(\tilde{G}) = \lambda] = 1 - \mathsf{negl}(\kappa)$, by Claim 1. Consider the following manipulation:

$$\begin{aligned}
& & \mathrm{P}\left[\mathsf{rank}\left(\tilde{G}\right) = \lambda\right] &= 1 - \mathsf{negl}(\kappa) & & \\
&\implies & \mathrm{P}\left[\mathsf{rank}\left(\tilde{G}\right) = \lambda \ \wedge \ \neg\mathsf{abort}_2\right] &= 1 - \mathsf{negl}(\kappa), & & \because \mathrm{P}[\mathsf{abort}_2] = \mathsf{negl}(\kappa) \\
&\implies & \mathrm{P}\left[\mathsf{rank}\left(\tilde{G}\right) = \lambda | \neg\mathsf{abort}_2\right] &= 1 - \mathsf{negl}(\kappa), & & \because \mathrm{P}[\mathsf{abort}_2] = \mathsf{negl}(\kappa) \\
&\implies & \mathrm{P}\left[\mathsf{rank}\left(G^{(\mathsf{col},\Lambda)}\right) = \lambda | \neg\mathsf{abort}_1\right] &= 1 - \mathsf{negl}(\kappa), & & \because (\mathsf{Experiment}_1 | \neg\mathsf{abort}_1) \\
& & & & & \equiv (\mathsf{Experiment}_2 | \neg\mathsf{abort}_2) \\
&\implies & \mathrm{P}\left[\mathsf{rank}\left(G^{(\mathsf{col},\Lambda)}\right) = \lambda \ \wedge \ \neg\mathsf{abort}_1\right] &= 1 - \mathsf{negl}(\kappa), & & \because \mathrm{P}[\mathsf{abort}_1] = \mathsf{negl}(\kappa) \\
&\implies & \mathrm{P}\left[\mathsf{rank}\left(G^{(\mathsf{col},\Lambda)}\right) = \lambda\right] &= 1 - \mathsf{negl}(\kappa) & & \square
\end{aligned}$$

In particular:

Corollary 6. *For $k \in [t_R + \omega(\log \kappa), n - (t_S + \omega(\log \kappa))]$, $|\Lambda| = \lambda = t_R + 1$, the probability that $\mathsf{rank}(G^{(\mathsf{col},\Lambda)}) = \lambda$ is at least $1 - \mathsf{negl}(\kappa)$.*

B Threshold Gap in Secret-sharing Schemes

In this section we shall prove some lower bounds on threshold gap of secret sharing schemes. First, we shall show a lower bound on threshold gap of $(n, 1, r)$-secret-sharing schemes in Lemma 3. Next, we reduce the lower bound on threshold gap of (n, t, r)-secret sharing schemes to Lemma 3 in Lemma 4. Finally, we prove Lemma 2 by using the same proof as Lemma 4 but using some special set of indices.

B.1 Lower bound on $(n, 1, r)$-Secret-sharing schemes

Lemma 3 (Lower Bound for $(n, 1, r)$-Secret-sharing schemes with (ε, δ)-security)**.** *Let $\mathfrak{S}{:=}(\mathbf{S}, \mathbf{X}_1, \ldots, \mathbf{X}_n)$ be a $(n, 1, r)$-Secret-sharing scheme with (ε, δ)-security. Then,*

$$r \geq \left(\sum_{i \in [n]} \frac{1}{\Lambda_i}\right) + 1 - n\left(\frac{\varepsilon^2}{2} + \binom{n}{r}\delta\right)$$

Proof. Recall that, the joint distribution over the shares of all parties conditioned on $S = s$ is represented by $(\mathbf{X}_{[n]} | \mathbf{S} = s)$. For brevity, we shall denote $(\mathbf{X}_{\{i\}} | \mathbf{S} = s)$ by $(\mathbf{X}_i | \mathbf{S} = s)$.

Since the secret sharing scheme is $(1 - \varepsilon)$ 1-private, we get that:

$$\mathbf{\Delta}\left((\mathbf{X}_i | \mathbf{S} = s), (\mathbf{X}_i | \mathbf{S} = s')\right) \leq \varepsilon$$

Therefore, by Lemma 6,

$$\mathop{\mathrm{P}}_{(u,u') \sim (\mathbf{X}_i | \mathbf{S}=s) \times (\mathbf{X}_i | \mathbf{S}=s')}[u = u'] \geq \frac{1}{\Lambda_i} - \frac{\varepsilon^2}{2}$$

Let $\mathbf{u}, \mathbf{v} \in [\Lambda_1] \times \cdots \times [\Lambda_n]$. Define $\mathsf{Intersect}(\mathbf{u}, \mathbf{v}) = \{i : i \in [n] \text{ and } u_i = v_i\}$; and $\mathsf{Size}(\mathbf{u}, \mathbf{v}) = |\mathsf{Intersect}(\mathbf{u}, \mathbf{v})|$. Consider the following expectation:

$$E := \mathop{\mathbb{E}}_{(\mathbf{u},\mathbf{v})\sim(\mathbf{X}_{[n]}|\mathbf{S}=s)\times(\mathbf{X}_{[n]}|\mathbf{S}=s')}[\mathsf{Size}(\mathbf{u}, \mathbf{v})] \geq \left(\sum_{i\in[n]} \frac{1}{\Lambda_i}\right) - \frac{n\varepsilon^2}{2} \tag{1}$$

Let $I \subseteq [n]$ be any set of size r. Recall that $\mathbf{\Delta}((\mathbf{X}_I|\mathbf{S} = s), (\mathbf{X}_I|\mathbf{S} = s')) \geq 1 - \delta$, because of r-reconstruction. For $\mathbf{u}, \mathbf{v} \in [\Lambda_1] \times \cdots \times [\Lambda_n]$, consider the following event: $\mathsf{ID_r}(I, \mathsf{Intersect}(\mathbf{u}, \mathbf{v}))$ is true if and only if $I \subseteq \mathsf{Intersect}(\mathbf{u}, \mathbf{v})$. Note that if $\mathsf{ID_r}(I, \mathsf{Intersect}(\mathbf{u}, \mathbf{v}))$ holds then the shares of all r parties indexed by I are identical in $\mathbf{u}$ and $\mathbf{v}$. Using Lemma 7, we have the following bound for any r-subset I:

$$\mathop{\mathrm{P}}_{(\mathbf{u},\mathbf{v})\sim(\mathbf{X}_{[n]}|\mathbf{S}=s)\times(\mathbf{X}_{[n]}|\mathbf{S}=s')}[\mathsf{ID_r}(I, \mathsf{Intersect}(\mathbf{u}, \mathbf{v}))] \leq \delta$$

Now, we claim that:

$$\mathop{\mathrm{P}}_{(\mathbf{u},\mathbf{v})\sim(\mathbf{X}_{[n]}|\mathbf{S}=s)\times(\mathbf{X}_{[n]}|\mathbf{S}=s')}[f(\mathbf{u}, \mathbf{v}) \geq r] \leq \delta' = \binom{n}{r}\delta \tag{2}$$

Otherwise, there exists an r-subset $I \subseteq [n]$ such that the probability of $R(I, I(\mathbf{u}, \mathbf{v}))$ is $> \delta$, which is a contradiction. Let:

$$\begin{aligned} E_{\mathsf{thresh}} &= E - n\delta', && \text{and} \\ p_{\mathsf{thresh}} &= \mathop{\mathrm{P}}_{(\mathbf{u},\mathbf{v})\sim(\mathbf{X}_{[n]}|\mathbf{S}=s)\times(\mathbf{X}_{[n]}|\mathbf{S}=s')}[f(\mathbf{u}, \mathbf{v}) \geq E_{\mathsf{thresh}}] \end{aligned}$$

Note that $f(\mathbf{u}, \mathbf{v}) < E_{\mathsf{thresh}}$ with probability $(1 - p_{\mathsf{thresh}})$ and in the range $[E_{\mathsf{thresh}}, n]$ with probability p_{thresh}. So, we have:

$$\begin{aligned} E &< (1 - p_{\mathsf{thresh}})E_{\mathsf{thresh}} + p_{\mathsf{thresh}}n \\ &< 1 \cdot (E - n\delta') + p_{\mathsf{thresh}}n \\ &= E + n(p_{\mathsf{thresh}} - \delta') \\ \implies p_{\mathsf{thresh}} &> \delta' \end{aligned}$$

If $p_{\mathsf{thresh}} > \delta'$ then it must be the case that: $E_{\mathsf{thresh}} \leq r - 1$; otherwise it will contradict with Equation 2. □

B.2 Lower bound on (n, t, r)-Secret-sharing schemes

Lemma 4 (Lower Bound for (n, t, r)-Secret Sharing schemes with (ε, δ)-security)**.** *Let $\mathfrak{S} = (\mathbf{S}, \mathbf{X}_1, \ldots, \mathbf{X}_n)$ be a (n, t, r)-Secret-sharing scheme with (ε, δ)-security. If $\varepsilon + \delta = o\left(\Lambda^{-4}\binom{n-t+1}{r-t+1}^{-2}\right)$, then*

$$(r - t) \geq \max_{I\subseteq[n]\ \wedge\ |I|=t-1} \left(\sum_{i\in[n]\setminus I} \frac{1}{\Lambda_i}\right) - (n - t + 1)\left(\frac{\varepsilon}{2} + \binom{n-t+1}{r-t+1}\sqrt{\varepsilon + \delta}\right)$$

Proof. Pick any $t-1$ subset I of $[n]$.

Consider the following sampling procedure $\mathfrak{R}(i)$: Sample $x \sim (\mathbf{X}_I|\mathbf{S}=1)$ and $y \sim (\mathbf{X}_{[n]\setminus I}|\mathbf{S}=i, \mathbf{X}_I = x)$. Output $(x \circ y) \equiv (\tilde{X}_1, \dots, \tilde{X}_n)$. We can use this sampling procedure to define a secret sharing scheme $\tilde{\mathfrak{S}}$ as the joint distribution of $(s, \mathbf{x})$ sampled as: $s \sim \mathbf{S}$ and $\mathbf{x} \sim \mathfrak{R}(s)$. The new secret-sharing scheme's joint distribution shall be represented by $(\mathbf{S}, \tilde{\mathbf{X}}_1, \dots, \tilde{\mathbf{X}}_n)$.

Since $\mathfrak{S}$ is a (n,t,r) secret-sharing scheme with (ε, δ) privacy, and $|I| = t-1$, we have: $\boldsymbol{\Delta}\left((\tilde{\mathbf{X}}_I|\mathbf{S}=1), (\tilde{\mathbf{X}}_I|\mathbf{S}=i)\right) \leq \varepsilon$, for all $i \in [n]$. Therefore, the new secret sharing scheme $\tilde{\mathfrak{S}}$ is a (n,t,r) secret-sharing scheme with $(2\varepsilon, \varepsilon+\delta)$ privacy.

Part One. Let T be any t subset of $[n]$ such that $I \subseteq T$. Therefore, we have: $\boldsymbol{\Delta}\left((\tilde{\mathbf{X}}_T|\mathbf{S}=i), (\tilde{\mathbf{X}}_T|\mathbf{S}=j)\right) \leq 2\varepsilon$.

Note that $\boldsymbol{\Delta}\left((\tilde{\mathbf{X}}_T|\mathbf{S}=i), (\tilde{\mathbf{X}}_T|\mathbf{S}=j)\right)$ is expectation of $\boldsymbol{\Delta}\left((\tilde{\mathbf{X}}_T|\mathbf{S}=i, \tilde{\mathbf{X}}_I = x), (\tilde{\mathbf{X}}_T|\mathbf{S}=j, \tilde{\mathbf{X}}_I = x)\right)$ when $x \sim (\tilde{\mathbf{X}}_I|\mathbf{S}=1)$. Thus, $\boldsymbol{\Delta}\left((\tilde{\mathbf{X}}_T|\mathbf{S}=i, \tilde{\mathbf{X}}_I = x), (\tilde{\mathbf{X}}_T|\mathbf{S}=j, \tilde{\mathbf{X}}_I = x)\right) \leq \sqrt{\varepsilon}$ with probability at least $1-2\sqrt{\varepsilon}$. By union bound, for every $i,j \in [\Lambda]$ and $\ell \in [n] \setminus I$,

$$\boldsymbol{\Delta}\left((\tilde{\mathbf{X}}_T|\mathbf{S}=i, \tilde{\mathbf{X}}_I = x), (\tilde{\mathbf{X}}_T|\mathbf{S}=j, \tilde{\mathbf{X}}_I = x)\right) \leq \sqrt{\varepsilon}, \tag{3}$$

with probability at least $1-2(n-t+1)\Lambda^2\sqrt{\varepsilon}$, where $T = I \cup \{\ell\}$.

Part Two. Let T be any r subset of $[n]$ such that $I \subseteq T$. Therefore, we have: $\boldsymbol{\Delta}\left((\tilde{\mathbf{X}}_T|\mathbf{S}=i), (\tilde{\mathbf{X}}_T|\mathbf{S}=j)\right) \geq 1-(\varepsilon+\delta)$.

By averaging argument, we can conclude that: $\boldsymbol{\Delta}\left((\tilde{\mathbf{X}}_T|\mathbf{S}=i, \tilde{\mathbf{X}}_I = x), (\tilde{\mathbf{X}}_T|\mathbf{S}=j, \tilde{\mathbf{X}}_I = x)\right) \geq 1-\sqrt{\varepsilon+\delta}$ with probability at least $1-\sqrt{\varepsilon+\delta}$.

There are $\mu = \binom{n-t+1}{r-t+1}$ r-subsets T which contain I. By union bound, for every $i,j \in [\Lambda]$ and r-subset T which contains I,

$$\boldsymbol{\Delta}\left((\tilde{\mathbf{X}}_T|\mathbf{S}=i, \tilde{\mathbf{X}}_I = x), (\tilde{\mathbf{X}}_T|\mathbf{S}=j, \tilde{\mathbf{X}}_I = x)\right) \geq 1-\sqrt{\varepsilon+\delta}, \tag{4}$$

with probability at least $1-\mu\Lambda^2\sqrt{\varepsilon+\delta}$.

Putting Things Together. If $\varepsilon+\delta = o(1/\mu^2\Lambda^4)$, then with positive probability both Equation 3 and Equation 4 are satisfied. Thus, there exists a fixing $\tilde{\mathbf{X}}_I = x^*$ such that both conditions are satisfied. Now consider the new $(n-t+1, 1, r-t+1)$ secret-sharing scheme with $(\sqrt{\varepsilon}, \sqrt{\varepsilon+\delta})$-security. Consider the joint distribution $(s, \mathbf{x})$ obtained by sampling $s \sim \mathbf{S}$ and then $x \sim (\tilde{\mathbf{X}}_{[n]\setminus I}|\mathcal{S} = s, \tilde{\mathbf{X}}_I = x^*)$.

By using Lemma 3, we have:

$$r - t + 1 \geq \left(\sum_{i \in [n] \setminus I} \frac{1}{\Lambda_i} \right) + 1 - (n-t+1) \left(\frac{\varepsilon}{2} + \binom{n-t+1}{r-t+1} \sqrt{\varepsilon + \delta} \right)$$

$$\implies \quad (r-t) \geq \left(\sum_{i \in [n] \setminus I} \frac{1}{\Lambda_i} \right) - (n-t+1) \left(\frac{\varepsilon}{2} - \binom{n-t+1}{r-t+1} \sqrt{\varepsilon + \delta} \right)$$

Note that the same argument also holds for every $(t-1)$ subset $I \subseteq [n]$; and hence the result follows. $\square$

B.3 Proof of Lemma 2

The result follows immediately by using the proof of Lemma 4 with an index set I such that $0 \in I$.

B.4 Proof of Theorem 3

We shall show that $\mathfrak{S}$ is a $(n = t_S + t_R + g, t_S, t_S + g)$-public-information secret-sharing scheme with $(\nu(t), \nu(t))$-security, where $\nu(\cdot)$ is a negligible function.

t_S-Privacy. For the privacy guarantee, assume that $0 \in \mathcal{S}$ without loss of generality. Consider the security of π against an adversary who corrupts the client S and the servers indexed by $\mathcal{S} \cap [n]$. Note that the view of the adversary is identical to the random variable $\mathbf{X}_{\mathcal{S}}$. By security guarantee of the combiner, we have:

$$\mathop{\mathbb{E}}_{X_{\mathcal{S}} \sim \mathbf{X}_{\mathcal{S}}} \left[\mathbf{\Delta} \left((\mathbf{S} | X_{\mathcal{S}}), \mathbf{U}_{\{0,1\}} \right) \right] \leq \mathsf{negl}(n)$$

Using the contrapositive of Lemma 5, the privacy guarantee follows.

$(t_S + g)$-Reconstruction. Now, for the reconstruction guarantee, consider the following two-party protocol π' in the plain model between party A and party B:

1. Party A simulates client S and the servers indexed by $\mathcal{S} \cap [n]$; and party B simulates client R and the servers indexed by $[n] \setminus \mathcal{S}$.

By the security of the combiner π it is clear that party B cannot predict $s_0 \oplus s_1$ with non-negligible advantage.

Note that the view of party A is identical to the distribution $\mathbf{X}_{\mathcal{S}}$. If there exists a polynomial $p(\cdot)$ such that:

$$\mathop{\mathbb{E}}_{X_{\mathcal{S}} \sim \mathbf{X}_{\mathcal{S}}} \left[\mathbf{\Delta} \left((\mathbf{S} | X_{\mathcal{S}}), \mathbf{U}_{\{0,1\}} \right) \right] \in \left[\frac{1}{p(n)}, \frac{1}{2} - \frac{1}{p(n)} \right],$$

then we can amplify this "weak-OT" into OT using techniques in [DKS99]. So, either the above mentioned quantity is $< \frac{1}{p(n)}$ or $> \frac{1}{2} - \frac{1}{p(n)}$, for every polynomial $p(\cdot)$.

Claim 4.

$$\mathop{\mathbb{E}}_{X_{\mathcal{S}} \sim \mathbf{X}_{\mathcal{S}}}\left[\boldsymbol{\Delta}\left((\mathbf{S}|X_{\mathcal{S}}), \mathbf{U}_{\{0,1\}}\right)\right] \geq 1/4 - \mathsf{negl}(n)$$

Proof. Let $(\mathbf{T}'|(s_0, s_1), c)$ be the transcript distribution of protocol π' when party A's private input is (s_0, s_1) and party B's private input is c. We know that the distribution $(\mathbf{T}'|(0,0),1)$ is $\mathsf{negl}(n)$ close to the distribution $(\mathbf{T}'|(1,0),1)$ (because of the privacy guarantee of combiner π proven above). But, by correctness of the protocol π', the distribution $(\mathbf{T}'|(0,0),0)$ is at least $1 - \mathsf{negl}(n)$ far from the distribution $(\mathbf{T}'|(1,0),0)$.

Thus, by triangle inequality, we can conclude that:

$$\boldsymbol{\Delta}\left((\mathbf{T}'|(0,0),0), (\mathbf{T}'|(0,0),1)\right) + \boldsymbol{\Delta}\left((\mathbf{T}'|(1,0),0), (\mathbf{T}'|(1,0),1)\right) \geq 1 - 2\mathsf{negl}(n)$$

Similarly, we also have: $\boldsymbol{\Delta}\left((\mathbf{T}'|(0,1),0), (\mathbf{T}'|(0,1),1)\right) + \boldsymbol{\Delta}\left((\mathbf{T}'|(1,1),0), (\mathbf{T}'|(1,1),1)\right) \geq 1 - 2\mathsf{negl}(n)$. Consequently, we have:

$$\boldsymbol{\Delta}\left((\mathbf{X}_{\mathcal{S}}|S=0), (\mathbf{X}_{\mathcal{S}}|S=1)\right) \geq 1/2 - \mathsf{negl}(n)$$

The claim follows from Lemma 5. □

Therefore, it must me the case that:

$$\mathop{\mathbb{E}}_{X_{\mathcal{S}} \sim \mathbf{X}_{\mathcal{S}}}\left[\boldsymbol{\Delta}\left((\mathbf{S}|X_{\mathcal{S}}), \mathbf{U}_{\{0,1\}}\right)\right] > \frac{1}{2} - \mathsf{negl}(n)$$

The reconstruction property follows due to the contrapositive of Lemma 5.

Putting things together. Next, by Lemma 2:

$$\begin{aligned} r - t = g &\geq \frac{t_R + 1 + g}{2} - (t_R + g)^g \mathsf{negl}(\kappa) \\ \iff \quad g &\geq (t_R + 1) - (t_R + g)^g \mathsf{negl}(\kappa) \end{aligned}$$

If $g = O(1)$, this results in a contradiction, and hence $(n = t_S + t_R + g, t_S, t_R)$-*single*-use-OT combiner is impossible. If $g = O(1)$, it is impossible to have $\omega(1)$ servers with $O(1)$ share size.

C Proof of Theorem 4

First Efficiency Calculation. Even when $p + q \leq 0.2$, then need 4^4 copies of $(p,q) - WOT$ to obtain one copy of (p', q')-WOT, where $p' + q' = (p+q)^2$. So, starting with $p + q = 0.2$ and applying their transformation N times, they reach simulation error:

$$\begin{aligned} (0.2)^{2^N} &= 2^{-t} \\ \iff \quad 2^N &= \frac{t}{\lg 5} \end{aligned}$$

Note that applying the DKS-operation N times implies that 4^{4N} copies of $(p,q) - WOT$ are needed. This implies that to obtain simulation error of 2^{-t}, the number of servers needed is $O(t^8)$.

On the other hand, our construction uses only $n = \Theta(t)$ servers implementing (p,q)-ROT. This follows from a direct observation that: if $n = \Theta(t)$ servers are used then with $2^{-t}/2$ probability we shall have $t_S + t_R \leq n \times (1 - p - q - \delta)$, for a suitable constant δ. By choosing n suitably, we can insure that: $t_S + t_R \leq n - \Theta(t)$ and the overall simulation error in our combiner is $2^{-t}/2$ (by Theorem 1). Thus, we use $\Theta(t)$ instances of (p,q)-WROT and achieve 2^{-t} (overall) simulation error.

Second Efficiency Calculation. Now suppose we have $p + q = 1 - {}^1\!/_k$, for some parameter k; and we are interested in implementing one OT with simulation error 2^{-t}.

According to the technique in [DKS99], their reduction first converts a (p,q)-WOT into a (p',q')-WOT where $1 - (p' + q') \geq 1.2 \times (1 - (p+q))$. And each step uses 2^2 copies of (p,q)-WOT. So, the number of iterations N needed to reach $(p' + q') = 0.2$.

$$\begin{aligned} (1.2)^N \times \frac{1}{k} &= 0.8 \\ \iff \quad 2^N &= (0.8k)^{1/\lg 1.2} \end{aligned}$$

So, the number of (p,q)-WOT needed to build (p',q')-WOT (with $p' + q' \leq 0.2$) is $2^{2N} = (0.8k)^{2/\lg 1.2}$.

From previous calculation, we get that: $\Theta(k^{2/\lg 1.2} \times t^8) \gg \Theta(k^{7.6}t^8)$ copies of (p,q)-WOT are needed to implement one copy of OT with simulation error 2^{-t}, where $p + q = 1 - {}^1\!/_k$.

In our case, by using $n = \Theta(kt)$ we get overall simulation error 2^{-t}.

D Technical Results

D.1 Probability Basics

Lemma 5. *Over a sample space $[n]$, the distribution $\mathcal{D}_0$ is defined by the probabilities $\{p_1, \dots, p_n\}$; and the distribution $\mathcal{D}_1$ is defined by the probabilities $\{q_1, \dots, q_n\}$. Consider the following experiment: Pick $b \xleftarrow{\$} \{0,1\}$ and output $i \sim \mathcal{D}_b$.*

If $\mathbf{\Delta}(\mathcal{D}_0, \mathcal{D}_1) = \delta$ then the advantage of guessing b is ${}^\delta\!/_2$.

Proof. Suppose the sample is i and suppose that $p_i > q_i$. Thus, the best strategy to guess b given i is to output $\tilde{b} = 0$. The advantage of this guess is: $\frac{p_i}{p_i+q_i} - \frac{1}{2} = \frac{1}{2}\frac{|p_i - q_i|}{p_i+q_i}$.

Thus, the overall advantage of guessing b is: $\sum_{i\in[n]} \frac{1}{2}\frac{|p_i-q_i|}{p_i+q_i} \times \frac{p_i+q_i}{2} = \delta/2$. □

Lemma 6 (Collision probability for Similar distributions). *Let $\mathbf{a} = \{a_i, \dots, a_n\}$ and $\mathbf{b} = \{b_1, \dots, b_n\}$ be two probability distributions over sample space $[n]$ such that $\mathbf{\Delta}(\mathbf{a}, \mathbf{b}) = \varepsilon$. Then,*

$$\sum_{i\in[n]} a_i b_i \geq \frac{1}{n} - \frac{\varepsilon^2}{2}$$

Proof. Let $A \subseteq [n]$ be the set of all indices $i \in [n]$ such that $b_i \geq a_i$; and define $B = [n] \setminus A$, i.e. all $i \in [n]$ such that $b_i < a_i$. Suppose $b_i = a_i + \varepsilon_i$ for all $i \in A$; and $b_i = a_i - \varepsilon_i$ for all $i \in B$. It is clear that $\sum_{i\in A} \varepsilon_i = \varepsilon = \sum_{i \in B} \varepsilon_i$.

Consider the following manipulation:

$$\begin{aligned}
\sum_{i\in[n]} a_i b_i &= \sum_{i\in A} a_i(a_i+\varepsilon_i) + \sum_{i\in B} a_i(a_i-\varepsilon_i) \\
&= \sum_{i\in A}\left(a_i+\frac{\varepsilon_i}{2}\right)^2 + \sum_{i\in B}\left(a_i-\frac{\varepsilon_i}{2}\right)^2 - \frac{1}{4}\sum_{i\in[n]}\varepsilon_i^2 \\
&\geq \frac{\sum_{i\in A}\left(a_i+\frac{\varepsilon_i}{2}\right)+\sum_{i\in B}\left(a_i-\frac{\varepsilon_i}{2}\right)}{n} - \frac{1}{4}\sum_{i\in[n]}\varepsilon_i^2, && \text{by Chauchy-Schwarz} \\
&= \frac{1}{n} - \frac{(\sum_{i\in A}\varepsilon_i)^2}{4} - \frac{(\sum_{i\in B}\varepsilon_i)^2}{4} = \frac{1}{n} - \frac{\varepsilon^2}{2}, && \text{by: } \sum_{i\in[n]} a_i = 1 \text{ and } \sum_{i\in A}\varepsilon_i = \sum_{i\in B}\varepsilon_i = \varepsilon
\end{aligned}$$

And one can easily show that this inequality is tight for every $n \in \mathbb{N}$. $\square$

Lemma 7 (Collision probability for Separate distributions)**.** *Let* $\mathbf{a} = \{a_i, \ldots, a_n\}$ *and* $\mathbf{b} = \{b_1, \ldots, b_n\}$ *be two probability distributions over sample space* $[n]$ *such that* $\boldsymbol{\Delta}(\mathbf{a}, \mathbf{b}) = 1 - \delta$*. Then,*

$$\sum_{i\in[n]} a_i b_i \leq \delta$$

Proof. Let $A \subseteq [n]$ be the set of all indices such that $b_i \geq a_i$ and $B = [n] \setminus A$. If $\sum_{i\in A} a_i = u \in [0, \delta]$, then:

$$\sum_{u\in B} a_i = (1-u), \quad \sum_{i\in A} b_i = (1-\delta+u), \text{ and } \quad \sum_{i\in B} b_i = (\delta - u)$$

Given these, consider the following manipulation:

$$\begin{aligned}
\sum_{i\in[n]} a_i b_i &= \sum_{i\in A} a_i b_i + \sum_{i\in B} a_i b_i \\
&\leq \left(\sum_{i\in A} a_i\right)\left(\sum_{i\in A} b_i\right) + \left(\sum_{i\in B} a_i\right)\left(\sum_{i\in B} b_i\right) \\
&= u(1-\delta+u) + (1-u)(\delta-u) \\
&= 2u(u-\delta) + \delta \quad \leq \delta
\end{aligned}$$

And it is easy to see that this inequality is tight for $n = 2$ and the left-hand side could be arbitrarily close to (but less than) δ for $n > 2$. $\square$

D.2 Some Protocols

In this section we present some simple reduction among primitives.

D.2.1 Some Local Renaming Schemes

Some Representative Correlations. We present some representative correlated randomness which shall be useful in our paper.

1. Random OLE-Correlation: Party A receives $(u, v) \xleftarrow{\$} \{0,1\}^2$ and party B receives $(x, ux \oplus v)$, for $x \xleftarrow{\$} \{0,1\}$.
2. Random ELO-Correlation: This is identical to OLE-correlation except that the roles of party A and party B are reversed.
3. ROT-Correlation: Party A receives $(s_0, s_1) \xleftarrow{\$} \{0,1\}^2$ and party B receives (c, s_c), for $c \xleftarrow{\$} \{0,1\}$.
4. TOR-Correlation: This is identical to ROT-correlation except that the roles of party A and party B are reversed.

Reduction of ROT to Random OLE correlation. Suppose parties have the following correlations: (u, v) with party A and (x, y) with party B such that $y = ux \oplus v$, where $u, v, x, y \in \{0,1\}$ and $u, v, x \xleftarrow{\$} \{0,1\}$.

Consider the following renaming of variables: $(s_0, s_1) \equiv (v, u \oplus v)$ by party A and $(c, z) \equiv (x, y)$ by party B. Note that: $z = s_c$ if $y = ux \oplus v$. Hence, under this local renaming of variables Random OLE-correlation can be converted into ROT correlations.

Reduction of Random OLE to ROT correlation. Suppose parties have the following correlations: (s_0, s_1) with party A and $(c, z = s_c)$ with party B, where $s_0, s_1, c \xleftarrow{\$} \{0,1\}$. Consider the following renaming of variables: $(u, v) \equiv (s_0 \oplus s_1, s_0)$ by party A and $(x, y) \equiv (c, z)$. Note that: $z = ux \oplus v$ if $z = s_c$. Hence, under this local renaming of variables ROT correlations can be converted into Random OLEL correlations.

Reduction between Random OLE and Random ELO correlations. Suppose party A has (u, v) and party B has $(x, y = ux \oplus v)$, where $u, v, x \xleftarrow{\$} \{0,1\}$. Consider the following renaming scheme: party B defines $(a', b') \equiv (x, y)$ and party B defines $(x', y') \equiv (u, v)$. Note that $y' = a'x' \oplus b'$ and $a', b', x' \xleftarrow{\$} \{0,1\}$. Thus, OLE and ELO generate identical correlations.

Between ROT and TOR. Consider the following sequence of local renaming schemes: ROT $\rightarrow$ OLE $\rightarrow$ ELO $\rightarrow$ TOR.

More concretely, suppose party A has (s_0, s_1) and party B has (c, s_c), where $s_0, s_1, c \xleftarrow{\$} \{0,1\}$. Then we define: $(c', z') \equiv (s_0 \oplus s_1, s_0)$ for party A and $(s'_0, s'_1) \equiv (s_c, c \oplus s_c)$ for party B. It is easy to verify that $z' = s'_{c'}$, for all $s'_0, s'_1, c' \in \{0,1\}$.

D.2.2 Some useful Protocols

OLE reduction to ROT correlation. Suppose party A has (s_0, s_1) and party B has $(c, z = s_c)$, where $s_0, s_1, c \xleftarrow{\$} \{0,1\}$. Suppose they want to implement an OLE where party A has inputs (u^*, v^*)

and party B has input x^* and receives $y^* = u^*x^* \oplus v^*$.

First they locally rename ROT correlation into Random OLE correlation. Now party A has (u, v) and party B has $(x, y = ux \oplus v)$. Now party A sends $\tilde{u} = u^* \oplus u$ and $\tilde{v} = v^* \oplus v$. Party B outputs: $y \oplus (\tilde{u}x \oplus \tilde{v})$.

This protocol is perfectly UC-secure implementation of OLE in ROT-hybrid. Correctness trivially follows. The security trivially follows from: Note that u is perfectly hidden from party B. And, thus, u^* is perfectly hidden from party B. Further this is secure against adaptive corruption of parties as well [Lin09].

OLE reduction to TOR correlation. Locally rename TOR correlation into ROT correlation and use the previous reduction.