Constant-Round Oblivious Transfer in the Bounded Storage Model

Yan Zong Ding[1], Danny Harnik[2], Alon Rosen[3] and Ronen Shaltiel[2]

[1] College of Computing, Georgia Institute of Technology. 801 Atlantic Drive, Atlanta, GA 30332-0280. Email: ding@cc.gatech.edu. Research supported by NSF Grant CCR-0205423.

[2] Dept. of Computer Science and Applied Math., Weizmann Institute of Science. Rehovot 76100, Israel. Email: {harnik, ronens}@wisdom.weizmann.ac.il. Research supported in part by a grant from the Israel Science Foundation.

[3] Laboratory for Computer Science, Massachusetts Institute of Technology. 200 Technology Square, Cambridge, MA 02139.† Email: alon@lcs.mit.edu

Abstract. We present a constant round protocol for Oblivious Transfer in Maurer's *bounded storage model*. In this model, a long random string $\mathcal{R}$ is initially transmitted and each of the parties interacts based on a small portion of $\mathcal{R}$. Even though the portions stored by the honest parties are small, security is guaranteed against any malicious party that remembers almost all of the string $\mathcal{R}$.

Previous constructions for Oblivious Transfer in the bounded storage model required polynomially many rounds of interaction. Our protocol has only 5 messages. We also improve other parameters, such as the number of bits transferred and the probability of immaturely aborting the protocol due to failure.

Our techniques utilize explicit constructions from the theory of derandomization. In particular, we use constructions of almost t-wise independent permutations, randomness extractors and averaging samplers.

1 Introduction

Oblivious transfer (OT) is one of the fundamental building blocks of modern cryptography. First introduced by Rabin [Rab81], oblivious transfer can serve as a basis to a wide range of cryptographic tasks. Most notably, any multi-party secure computation can be based on the security of OT. This was shown for various models in several works (cf. [Yao86, GMW87, Kil88]).

Oblivious transfer has been studied in several variants, all of which were eventually shown to be equivalent. In this paper we consider the one-out-of-two variant of OT by Even, Goldreich ad Lempel [EGL85], which was shown to be equivalent to Rabin's variant by Crépeau [Cré87].

One-out-of-two OT is a protocol between two players, Alice holding two secrets s_0 and s_1, and Bob holding a choice bit c. At the end of the protocol Bob should learn the secret of his choice (i.e., s_c) but learn nothing about the other secret. Alice, on the other hand, should learn nothing about Bob's choice c.

† Part of this work done while at the Weizmann Institute of Science, Israel.

Traditionally, constructions for OT have been based on strong computational assumptions. Either specific assumptions such as factoring or Diffie Hellman (cf. [Rab81, BM89, NP01]) or generic assumption such as the existence of enhanced trapdoor permutations (cf. [EGL85, Gol03, GKM$^+$00]). In contrast, OT cannot be reduced in a black box manner to presumably weaker primitives such as one-way functions [IR89].

This state of affairs motivates the construction of OT in other types of setups. Indeed, protocols for OT were suggested in different models such as under the existence of noisy channels [CK88] or quantum channels [BBCS92]. In this work we follow a direction initiated by Cachin, Crépeau and Marcil [CCM98] and construct OT in the *Bounded Storage model*.

1.1 The bounded storage model

In contrast to the usual approach in modern Cryptography, Maurer's *bounded storage model* [Mau92, Mau93] bounds the *space* (memory size) of dishonest players rather than their running time.

In a typical protocol in the bounded storage model a long random string $\mathcal{R}$ of length N is initially broadcast and the interaction between the polynomial-time participants is conducted based on a short portion of $\mathcal{R}$.[5] What makes such protocols interesting is that, even though the honest players store only a small fraction $k << N$ of the string $\mathcal{R}$, security is guaranteed even against dishonest players with space K where $k << K < N$. Moreover, dishonest players are not restricted to be computationally bounded (This is formalized by allowing dishonest players to choose an arbitrary *memory function* $g^* : \{0,1\}^N \rightarrow \{0,1\}^K$, and store $g^*(\mathcal{R})$. From that moment on, they are not bounded in any way). Naturally, we'd like to maximize K and minimize k. In this paper we have $K = \nu N$ for an arbitrary constant $\nu < 1$ and k will be about $K^{1/2}$.

The bounded storage model has two appealing properties: (1) The security obtained is information theoretic and thus everlasting in the sense that security is guaranteed even if adversaries acquire infinite space after the protocol is executed. (2) Protocols in the bounded storage model need not rely on any assumption except the limitation on the storage capabilities of the adversary.

The latter property should be contrasted with traditional works in Cryptography in which, besides bounding the adversary's computational capabilities, it is also required to rely on unproven hardness assumptions (such as the existence of enhanced trapdoor permutations, or the hardness of factoring large integers).

We mention that most of the previous work on the bounded storage model concentrated on private key encryption [Mau92, CM97, AR99, ADR02, DR02, DM02, Lu02, Vad03] and key agreement [Mau93, CM97].

[5] One possible implementation is that $\mathcal{R}$ is broadcast at a very high rate by a trusted party. Another possibility is to have $\mathcal{R}$ transmitted from a satellite. We remark that in our protocol (as in many previous ones) one of the parties can transmit these bits. Furthermore, the assumption that $\mathcal{R}$ is uniformly distributed can be relaxed and it is sufficient that $\mathcal{R}$ has high min-entropy.

1.2 Oblivious transfer in the bounded storage model

A protocol for OT in the bounded storage model was given in [CCM98]. This protocol requires $k \approx K^{2/3}$ and allows $K = \nu N$ for an arbitrary constant $\nu < 1$. The *error* ϵ in this protocol is rather large $\epsilon = k^{-O(1)}$. (Loosely speaking the error ϵ measures the probability that a dishonest receiver with storage bound K learns both secrets.)

A modified protocol with smaller error ϵ and smaller space k was given in [Din01]. For every constant $c > 0$, it achieves $k = K^{1/2+c}$ and $\epsilon = 2^{-k^{c'}}$ where $c' > 0$ is a constant that depends on c. We mention that the security of [Din01] is proven in a slightly different (and weaker) model, where it is assumed that two random strings $\mathcal{R}_1, \mathcal{R}_2$ of length $6K$ are transmitted one after the other and the bounded receiver chooses what to remember about $\mathcal{R}_2$ as a function of what he remembers about $\mathcal{R}_1$. The work of [Din01] was subsequently extended to deal with one-out-of-k OT for any small constant $k \geq 2$ in [HCR02].[6]

All protocols mentioned above require a lot of interaction. Specifically, for $\epsilon = 2^{-k^{O(1)}}$, they require the exchange of $k^{\Omega(1)}$ messages between the two players.

1.3 Our results

We give a *constant round* OT protocol in the bounded storage model. Our protocol uses 5 messages following the transmission of the random string $\mathcal{R}$. We achieve parameters k and ϵ similar to that of [Din01] (that is, for every $c > 0$ there exist $c' > 0$ such that our protocol has $k = K^{1/2+c}$ and $\epsilon = 2^{-k^{c'}}$) while working in the stronger model of [CCM98]. Similar to [CCM98] we can achieve $K = \nu N$ for an arbitrary constant $\nu < 1$.

In addition to being constant round our protocol also achieves the following improvements over [CCM98, Din01]:

- The previous protocols are designed to transfer secrets in $\{0, 1\}$. Thus, transferring long secrets requires many messages. Our protocol can handle secrets of length $k^{\Omega(1)}$ in one execution.
- The previous protocols abort unsuccessfully with probability $1/2$ even if both players are honest. Our protocol aborts only with probability $2^{-k^{\Omega(1)}}$.
- For error $\epsilon = 2^{-k^{\Omega(1)}}$, the number of bits communicated in the two previous protocols is at least $K^{1/2}$. In contrast, for error $\epsilon = 2^{-k^c}$ our protocol communicates only $O(k^c)$ bits.

We also give a precise definition for the security of oblivious transfer in the bounded storage model, and point out difficulties arising when trying to consider the more standard notion of a "simulation based" definition.

[6] We note that a similar extension can be easily applied to our work.

1.4 Interactive Hashing

An important building block in the OT protocol is a construction of a constant round 2-to-1 *interactive hashing* protocol for unbounded parties. Loosely speaking, in such a protocol Bob holds an input $W \in \{0,1\}^m$, and Alice and Bob want to agree on a pair W_0, W_1 such that $W_d = W$ for some $d \in \{0,1\}$, yet Alice does not know d. It is also required that a dishonest Bob cannot "control" both W_0 and W_1. (See Section 5 for a precise definition.)

As observed in [CCM98], the protocol of Naor, Ostrovsky, Venkatesan and Yung [NOVY98] (originally used in the context of perfectly-hiding commitments) achieves 2-to-1 interactive hashing. One major drawback of the NOVY protocol, however, is that it requires m rounds of interaction. In this paper we give a new 4-message protocol for 2-to-1 interactive hashing that can be used to replace the NOVY protocol in the context of oblivious transfer in the bounded-storage model. Our protocol relies on a construction of almost t-wise independent permutations, such as the construction presented by Gowers in [Gow96].

Organization. Due to space limitation, some of the details and proofs have been omitted from this version. In Section 2 we present an overview of the techniques that were utilized to achieve our results. Some preliminary definitions are given in Section 3. Section 4 provides a definition of OT in the bounded storage model. In Section 5 we define and state our theorem regarding interactive hashing. The OT protocol is presented in Section 6. Sections 7, 8 and 9 give a high level analysis of the protocol. Conclusions and open problems are in Section 10.

2 Overview of the technique

As motivation for our protocol, we begin by suggesting a simple protocol for OT in the bounded storage model which is bad in the sense that it requires large storage from the honest parties: Alice is required to store *all* of the string $\mathcal{R}$ and Bob is required to store half this string. We partition the N bit long string $\mathcal{R}$ into two equally long parts $\mathcal{R}_0, \mathcal{R}_1$ of length $N/2$. Recall that Alice has two secrets s_0, s_1 and Bob has a "choice bit" c and wants to obtain s_c. Bob will choose which of the two parts $\mathcal{R}_0, \mathcal{R}_1$ to store depending on his "choice bit" c.

Input of Alice: Secrets s_0, s_1.
Input of Bob: Choice bit: $c \in \{0,1\}$.
A random string $\mathcal{R} = (\mathcal{R}_0, \mathcal{R}_1)$ is transmitted.
Alice: Store all of $\mathcal{R}$.
Bob: Store $\mathcal{R}_c$.
Alice: For $i \in \{0,1\}$, send a uniformly chosen seed Y_i, compute $V_i = \mathrm{Ext}(\mathcal{R}_i, Y_i)$ and $Z_i = V_i \oplus s_i$. Send Y_i, Z_i.
Bob: Compute $V_c = \mathrm{Ext}(\mathcal{R}_c, Y_c)$ and obtain $s_c = V_c \oplus Z_c$.

Fig. 1. A naïve protocol for OT

Intuitively, even if Bob is dishonest and has storage bound νN then there is an $I \in \{0,1\}$ such that Bob "does not remember" $(1-\nu)N/2$ bits of information about $\mathcal{R}_I$. This can be formalized by saying that the conditional entropy of $\mathcal{R}_I$ given the memory content of Bob is roughly $(1-\nu)N/2$. (Actually, in this paper, as in [CCM98, Din01], we work with a variant of entropy called *min-entropy*).

Let $\mathrm{Ext}(X, Y)$ (Ext for *extractor*) denote a function such that whenever X has sufficiently high min-entropy and Y is uniformly distributed then $\mathrm{Ext}(X, Y)$ is close to being uniformly distributed. (The reader is referred to [Nis96, Sha02] for surveys on extractors). To complete the protocol, Alice sends $Z_i = s_i \oplus \mathrm{Ext}(\mathcal{R}_i, Y_i)$ for both $i = 0$ and $i = 1$.

Note that an honest Bob can compute $\mathrm{Ext}(\mathcal{R}_c, Y_c) \oplus Z_c$ and obtain s_c. However, if Bob is dishonest then Z_I is close to uniform from Bob's point of view and reveals no information about s_I.[7] It is easy to prove that even an unbounded dishonest Alice does not learn c.

Using a setup stage before the naïve protocol. The naïve protocol above requires very large storage bounds from the honest parties. In order to instantiate it in a more efficient manner we will first apply a carefully designed *setup stage*. Our goal is that at the end of the setup stage the two players will agree on two small subsets $C_0, C_1 \subseteq [N]$ of size $\ell << N$, such that Alice stores $\mathcal{R}_0 = \mathcal{R}_{C_0}$ and $\mathcal{R}_1 = \mathcal{R}_{C_1}$. (We use $\mathcal{R}_C$ to denote the $|C|$ bit long string obtained by restricting $\mathcal{R}$ to the indices in C.) Bob remembers only one of $\mathcal{R}_0, \mathcal{R}_1$ and cannot remember too much information about the other string. Furthermore, Alice does not know which of the two strings is not known to Bob. Following the setup stage, the two parties can perform OT by using the naïve protocol. We call this second stage the *transfer stage*. As the sets C_0, C_1 are of size $\ell << N$ the storage required by the honest parties at the transfer stage is much smaller than before, and honest players can follow the naïve protocol with space $O(\ell) << N$.

Implementing the setup stage. An implementation for such a setup stage was suggested in [CCM98]: Alice and Bob each choose a random subset of $[N]$ of size $n = \sqrt{2N\ell}$. We denote them by A and B respectively. When the string $\mathcal{R}$ is transmitted Alice and Bob store $\mathcal{R}_A$ and $\mathcal{R}_B$ respectively. Alice then sends A to Bob. By the birthday paradox, with high probability $C = A \cap B$ is of size roughly ℓ. Note that Bob remembers $\mathcal{R}_C$, and Alice does not know C.

To complete the setup stage, Alice and Bob play an interactive hashing protocol with $W = C$. They obtain sets $C_0, C_1 \subseteq A$ such that $C = C_d$ for some $d \in \{0,1\}$ and such that Alice does not know d. The security requirement of the interactive hashing can be then used to guarantee that Bob "does not remember a lot of information" about one of the strings $\mathcal{R}_{C_0}, \mathcal{R}_{C_1}$. Thus, the two sets C_0, C_1 satisfy the properties required above and the parties can complete the

[7] We mention that the argument above is imprecise. Given the memory content of Bob, the strings Z_0, Z_1 are no longer independent. Thus, to prove security it is not sufficient to prove that Z_I is uniformly distributed given the memory content of Bob. In the technical proof we prove that Z_I is uniformly distributed given the memory content of Bob, Z_{1-I} and Y_0, Y_1.

A long random string $\mathcal{R}$ of length N is transmitted.
Alice: Choose random $A \subseteq [N]$ of size n and store $\mathcal{R}_A$.
Bob: Choose random $B \subseteq [N]$ of size n and store $\mathcal{R}_B$.
Alice: Send A to Bob.
Bob: Verify that $C = A \cap B$ is of size at least $\ell = n^2/2N$.
Alice and Bob: Play an interactive hashing protocol where Bob's input is C. Both Alice and Bob obtain $C_0, C_1 \subseteq A$ such that $C \in \{C_0, C_1\}$.
At this point, Alice and Bob use the naïve protocol with $\mathcal{R}_0 = \mathcal{R}_{C_0}$ and $\mathcal{R}_1 = \mathcal{R}_{C_1}$.

Fig. 2. The protocol for the setup stage

OT protocol by using the naïve protocol.[8] Note that the setup stage requires the honest parties to store only $k = n = \sqrt{N\ell}$ bits. In this presentation, we did not discuss the security of Bob, however it is easy to show that even an unbounded Alice, which remembers all of $\mathcal{R}$, cannot learn any information about c.

Previous protocols. The protocols of [CCM98, Din01] both use the setup stage described above. They implement interactive hashing using the NOVY-protocol from [NOVY98] which takes $\ell = k^{\Omega(1)}$-rounds. Following the setup stage they perform what can be seen in retrospect as variants of our naïve protocol. (Both papers do not use extractors explicitly, however their strategies can be viewed as some (weak) implementations of extractors.)

Our improvements. Our main improvement comes from replacing the NOVY-protocol for interactive hashing by a new 4-message protocol. This protocol is based on explicit constructions of almost t-wise independent permutations. Some of the additional improvements are given by using competitive explicit constructions of extractors for the naïve protocol above. Another source of improvement comes from allowing Alice to choose the set A using an *averaging sampler* (The reader is referred to [Gol97] for a survey on samplers). Choosing the set A using a competitive averaging sampler reduces the memory requirements of Alice and Bob, as well as the overall communication.[9] We remark that the usefulness of extractors in the bounded storage model was demonstrated in [Lu02], and that of averaging samplers was demonstrated in [Vad03].[10] Our paper can be seen as another example of the usefulness of ideas from the theory of derandomization when designing protocols for the bounded storage model.

[8] A subtlety is that Bob has no control whether $C = C_0$ or $C = C_1$. In the actual protocol we allow Bob to ask Alice to "switch" between the roles of C_0, C_1 in order to receive the desired secret.

[9] Note that using a samplers to choose the set B as well, we can further improves the total communication and memory requirements.

[10] It should be noted that the seminal paper of Nisan and Zuckerman [NZ96] which defined extractors, already used them in a very related context to construct pseudo-random generators against bounded space machines.

2.1 The improved interactive hashing protocol

In an interactive hashing protocol Bob holds an input $W \in \{0,1\}^m$ and at the end of the protocol both parties should agree on W_0, W_1. It is required that there is a $d \in \{0,1\}$ such that $W = W_d$ and that a dishonest Alice cannot learn d. The main requirement is that a dishonest Bob cannot "control" both W_0, W_1. This is captured by the following condition: For every strategy of Bob and every set S of size 2^s (where s is a parameter), If Alice is honest then with high probability Bob cannot force that both W_0 and W_1 are in S.

A naïve solution. A naïve solution to this problem is that Alice sends a random 2-to-1 "hash function" $h : \{0,1\}^m \to \{0,1\}^{m-1}$ and Bob replies with $z = h(W)$. Then the two parties compute the two preimages W_0, W_1 of z under h. Note that for $s > m/2$ this protocol fails even if Alice sends a completely random function $h : \{0,1\}^m \to \{0,1\}^{m-1}$ (By the birthday paradox, for every S of size $2^s > 2^{m/2}$ with high probability over h there are $W_0, W_1 \in S$ such that $h(W_1) = h(W_2)$).

The NOVY-protocol. The NOVY-protocol [NOVY98] for interactive hashing can be thought of as a variant of the naïve solution described above in which Alice does not send "all" of the hash function at once. Alice chooses a random $m \times m$ matrix A with entries in $\{0,1\}$ subject to the restriction that A is invertible. Every such A can be seen as defining a function $h_A(x) = A \cdot x$. It is easy to see that the function h_A is a pairwise independent permutation. In particular, the function $h'_A(x) = (A \cdot x)_{1,\ldots,m-1}$ is 2-to-1. The protocol consists of $m-1$ rounds. In round i, Alice sends A_i (the i'th row of A), and Bob replies with the $z_i = \langle A_i, W \rangle = h_A(W)_i$. Intuitively, revealing h_A slowly in return to bits z_i restricts Bob in the sense that he has to "choose at least part of his input" before seeing all of h_A.

The new protocol. Viewing the NOVY-protocol this way suggests the following improvement: We replace the family $\{h_A\}_A$ by a family of permutations with stronger independence properties. Namely, we will let π be randomly chosen from a family of m-wise independent permutations. In the new protocol, Alice sends π to Bob and in exchange Bob sends at once $z_1, \cdots, z_v$ where $z_i = \pi(W)_i$ for v close to m. We can show that the independence properties of π "protect Alice" and allow the parties to engage in a new interactive hashing protocol for sending the remaining few $m-v$ bits. By choosing the parameters appropriately, the two parties can use the naïve solution (with a pairwise independent hash function $g : \{0,1\}^{m-v} \to \{0,1\}^{m-v-1}$) after the first round. As a result of that we obtain a 2-round (4-messages) protocol (see Section 5.4).

Unfortunately, we are not aware of any explicit construction of a small sample space of t-wise independent permutations for $t > 3$. Nevertheless, in [Gow96] (see also [NR99] and the references therein) it was shown how to construct a sample space of permutations in which every t elements are close to being independent, and we can carry out the argument with this weaker property.

3 Preliminaries

We use $[N]$ to denote the set $\{1,\ldots,N\}$. We use $X \overset{\mathrm{R}}{\leftarrow} S$ to denote uniformly choosing X from S. For a set $A \subseteq [N]$ and a string $\mathcal{R} \in \{0,1\}^N$ we let $\mathcal{R}_A$ denote the substring of $\mathcal{R}$ consisting of the bits indexed by A. For a set S and $\ell \leq |S|$, we use $\binom{S}{\ell}$ to denote the set of all subsets $T \subseteq S$ with $|T| = \ell$.

Encoding subsets. We use a method of encoding sets in $\binom{[n]}{\ell}$ into binary strings. The following method was used in [CCM98]:

Theorem 3.1 ([Cov73]) *For every integers $\ell \leq n$ there is a one to one mapping $F : \binom{[n]}{\ell} \to [\binom{n}{\ell}]$ such that both F and F^{-1} can be computed in time polynomial in n and space $O(\log\binom{n}{\ell})$.*

Using Theorem 3.1 we can encode $\binom{[n]}{\ell}$ by binary strings of length $\lceil \log\binom{n}{\ell} \rceil$. However, it could be the case that images of subsets constitute only slightly more than half of the strings above. This is exactly what causes the protocols of [CCM98, Din01] to unsuccessfully abort with probability 1/2 (and is solved by repeating the protocol until the execution succeeds). Since in this work we are aiming for low round complexity, it would be beneficial to have the probability of unsuccessful abort to be significantly smaller than 1/2. To achieve this, we will use a more redundant encoding. This encoding is more "dense" than the original one and thus guarantees that most strings can be decoded.

Definition 3.2 (Dense encoding of subsets) *For every integers $\ell \leq n$ let F be the mapping from Theorem 3.1. Given an integer $m \geq \lceil \log\binom{n}{\ell} \rceil$ we set $t_m = \lfloor 2^m / \binom{n}{\ell} \rfloor$. Define the mapping $F_m : \binom{[n]}{\ell} \times [t_m] \to \{0,1\}^m$ as $F_m(S,i) = (i-1)\binom{n}{\ell} + F(S)$ (every subset S is mapped to t_m different m bit strings).*

We now have the following Lemma (proof omitted).

Lemma 3.3 *For every $\ell \leq n$ and $m \geq \lceil \log\binom{n}{\ell} \rceil$, the encoding F_m is a one-to-one mapping. Furthermore: (1) F_m and F_m^{-1} are computable in time $\mathrm{poly}(n, \log m)$ and space $O(\log\binom{n}{\ell}) + \log m$. (2) Let D be the image of F_m (D contains all m bit strings that are legal encodings of subsets), then $\frac{|D|}{2^m} > 1 - \binom{n}{\ell}/2^m$.*

Min-entropy and Extractors. Min-entropy is a variant of Shannon's entropy that measures information on the *worst case*.

Definition 3.4 (Min-entropy) *For a distribution X over a probability space Ω the* min-entropy *of X is defined by: $H_\infty(X) = \min_{x \in \Omega} \log(1/\Pr[X = x])$. We say that X is a k-source if $H_\infty(X) \geq k$.*

Definition 3.5 (Statistical distance) *Two distributions P and Q over Ω are ϵ-close (also denoted $P \overset{\epsilon}{\equiv} Q$) if for every $A \subseteq \Omega$, $|\Pr_{x \leftarrow P}(A) - \Pr_{x \leftarrow Q}(A)| \leq \epsilon$.*

An extractor is a function that "extracts" randomness from arbitrary distributions which "contain" sufficient (min)-entropy[NZ96].

Definition 3.6 (Strong extractor) *A function* $\mathrm{Ext} : \{0,1\}^{n_E} \times \{0,1\}^{d_E} \to \{0,1\}^{m_E}$ *is a* (k_E, ϵ_E)*-strong extractor if for every* k_E*-source* X *over* $\{0,1\}^{n_E}$ *the distribution* $(\mathrm{Ext}(X,Y), Y)$ *where* Y *is uniform over* $\{0,1\}^{d_E}$ *is* ϵ_E*-close to* (U_{m_E}, Y) *where* U_{m_E} *is uniform over* $\{0,1\}^{m_E}$*.*

We remark that a *regular* (non-strong) extractor is defined in a similar way, replacing the random variable $(\mathrm{Ext}(X,Y), Y)$ by $\mathrm{Ext}(X,Y)$.

Averaging Samplers and Min-Entropy Samplers. A fundamental lemma by Nisan and Zuckerman [NZ96] asserts that given a δv-source X on $\{0,1\}^v$, with high probability over choosing $T \subseteq [v]$ of size t, X_T is roughly a δt-source.

In [CCM98] this lemma is used to assert that if a bounded storage adversary has memory bound νv for $\nu \approx 1-\delta$ then for a random T he "remembers at most νt bits about X_T". This approach is also used in [Vad03] which constructs private key encryption in the bounded storage model. As shown in [RSW00, Vad03] the lemma does not require a uniformly chosen subset. It is sufficient that T is chosen using a "good averaging sampler"[11](such samplers have been a subject of a line of studies starting with [BR94], see survey of [Gol97]).

Definition 3.7 (Averaging sampler) *A function* $\mathrm{Samp} : [L] \to [v]^t$ *is a* (μ, θ, γ)-averaging sampler *if for every function* $f : [v] \to [0,1]$ *with average value* $\frac{1}{v}\sum_i f(i) \geq \mu$*,*

$$\Pr_{p \in [L]}\left[\frac{1}{t}\sum_{1 \leq i \leq t} f(\mathrm{Samp}(p)_i) < \mu - \theta\right] \leq \gamma$$

The function Samp *is said to have* distinct samples *if for every* $p \in [L]$*, the* t *outputs of* $\mathrm{Samp}(p)$ *are distinct.*

A *min-entropy sampler* has the property that for most choices of p, the variable $X_{\mathrm{Samp}(p)}$ is close to having high min-entropy. As shown in [Vad03], every averaging sampler yields a min-entropy sampler.

Definition 3.8 (Min-entropy sampler) *A function* $\mathrm{Samp} : [L] \to [v]^t$ *with distinct samples is an* $(\delta, \delta', \phi, \epsilon)$-min-entropy sampler *if for every* δv*-source* X *over* $\{0,1\}^v$ *there is a set* $G \subseteq [L]$ *of density* $1-\phi$ *such that for every* $p \in G$ *the distribution* $X_{\mathrm{Samp}(p)}$ *is* ϵ*-close to a* $\delta' t$*-source.*

Lemma 3.9 ([Vad03] restated) *Let* $\mathrm{Samp} : [L] \to [v]^t$ *be a* (μ, θ, γ)*-averaging sampler with distinct samples for* $\mu = (\delta - 2\tau)/\log(1/\tau)$ *and* $\theta = \tau/\log(1/\tau)$*. Then there is a constant* $c > 0$ *such that for every* $0 < \alpha < 1$*,* Samp *is a* $(\delta, \delta - 3\tau, (\gamma + 2^{-c\tau v})^{1-\alpha}, (\gamma + 2^{-c\tau v})^{\alpha})$*-min-entropy sampler.*

---

[11] We remark that most constructions of averaging samplers do not depend on μ and work for every $0 \leq \mu \leq 1$.

4 Oblivious Transfer in the Bounded Storage Model

We now turn to formally define oblivious transfer in the bounded storage model. The following definitions characterize malicious strategies for Alice and Bob. Note that in the definitions below the malicious strategies are asymmetric. We restrict malicious strategies for Bob to have bounded storage while no bounds are placed on malicious strategies for Alice. Clearly, if a protocol is secure against unbounded strategies for Alice, it is also secure against bounded strategies. Thus, the security defined here is even stronger than that explained in the introduction.

Definition 4.1 (Malicious Strategy for Alice) *A (malicious) strategy A^* for Alice is an unbounded interactive machine with inputs $\mathcal{R} \in \{0,1\}^N$ and $s_0, s_1 \in \{0,1\}^u$. That is, A^* receives $\mathcal{R}$ and s_0, s_1 and interacts with B, in each stage, it may compute the next message as any function of its inputs, its randomness and the messages it received thus far. The view of A^* when interacting with B that holds input c (denoted $\mathrm{view}_{\mathrm{A}^*}^{\langle \mathrm{A}^*,\mathrm{B}\rangle}(\mathrm{s}_0, \mathrm{s}_1; \mathrm{c})$) consists of its local output.* [12]

The following definition captures a bounded storage strategy with storage bound K. Loosely speaking, the only restriction made on a bounded storage strategy B^* is that it has some *memory function* $g^* : \{0,1\}^N \to \{0,1\}^K$ and its actions depend on $\mathcal{R}$ only through $g^*(\mathcal{R})$. This formally captures that B^* remembers only K bits about $\mathcal{R}$.

Definition 4.2 (Bounded storage strategy for Bob) *A bounded storage strategy B^* for Bob with memory bound K is a pair $(g^*, \widehat{B}^*)$ where:*

- *$g^* : \{0,1\} \times \{0,1\}^N \to \{0,1\}^K$ is an arbitrary (not necessarily efficiently computable) function with input c and $\mathcal{R}$.*
- *$\widehat{B}^*$ is an unbounded interactive machine with inputs $c \in \{0,1\}$ and $b^* \in \{0,1\}^K$.*

The behavior described by a strategy B^* with input c is the following: When given the string $\mathcal{R} \in \{0,1\}^N$, B^* computes $b^* = g^*(c, R)$. B^* then interacts with A using the interactive machine $\widehat{B}^*$ receiving inputs c and b^*. The view of B^* with input c when interacting with A with inputs s_0, s_1 (denoted $\mathrm{view}_{\mathrm{B}^*}^{\langle \mathrm{A},\mathrm{B}^*\rangle}(\mathrm{s}_0, \mathrm{s}_1; \mathrm{c})$) is defined as the view of $\widehat{B}^*$ when interacting with A.

We now turn to the definition of oblivious transfer in the bounded storage model. The security of Bob asks that for any malicious strategy for Alice, its view is identically distributed whether Bob inputs $c = 0$ or $c = 1$. The definition of Alice's security is a bit more complex because one of her secrets is passed to Bob. For this definition, we partition *every* protocol that implements OT into two stages. The first stage called the *Setup Stage* and includes the transmission of the long string $\mathcal{R}$ and all additional messages sent by Alice and Bob until the point where Alice first makes use of her input s_0, s_1. The remaining steps in the protocol are called the *Transfer Stage*. Next define consistent pairs of secrets.

[12] The view of A may be thought of as also containing the party's randomness, inputs and outputs, as well as the messages received from B. This more intuitive "view" is possible since w.l.o.g. the malicious party may copy this view to his output.

Definition 4.3 *Two pairs* $\bar{s}=(s_0,s_1)$ *and* $\bar{s}'=(s'_0,s'_1)$ *are c-consistent if* $s_c=s'_c$.

The security of Alice asks that following the setup stage (which does not depend on the secrets), there is an index C (possibly a random variable which depends on $\mathcal{R}$ and the messages sent by the two parties in the setup stage) such that Bob's view is (close to) identically distributed for every two C-consistent pairs. In other words, Bob's view is (almost) independent of one of the secrets (defined by $1-C$). We next present the actual definition.

Definition 4.4 (Oblivious Transfer) *A protocol* $\langle A,B\rangle$ *is said to implement* $(1-\epsilon)$-oblivious transfer (OT) *if it is a protocol in which Alice inputs two (secrets)* $s_0,s_1 \in \{0,1\}^u$*, Bob inputs a choice bit* $c\in\{0,1\}$*, and that satisfies:*

Functionality : *If Alice and Bob follow the protocol then for any* s_0,s_1 *and* c*,*

1. *The protocol does not abort with probability* $1-\epsilon$.
2. *If the protocol ends then Bob outputs* s_c*, whereas Alice outputs nothing.*

Security for Bob*: The view of any strategy* A^* *is independent of* c*. Namely, for every* s_0,s_1*:*

$$\left\{\mathrm{view}_{A^*}^{\langle A^*,B\rangle}(s_0,s_1;c)\mid c=0\right\} \;\equiv\; \left\{\mathrm{view}_{A^*}^{\langle A^*,B\rangle}(s_0,s_1;c)\mid c=1\right\}$$

(K,ϵ)**-Security for Alice***: for every bounded storage strategy* B^* *for Bob with memory bound* K *and input* c *there is a random variable* C *defined by the end of the setup stage such that for every two pairs* $\bar{s}$ *and* $\bar{s}'$ *that are* C*-consistent:*

$$\left\{\mathrm{view}_{B^*}^{\langle A,B^*\rangle}(\bar{s};c)\right\} \;\stackrel{\epsilon}{\equiv}\; \left\{\mathrm{view}_{B^*}^{\langle A,B^*\rangle}(\bar{s}';c)\right\}$$

If Bob is semi-honest then $C=c$,[13] however, a dishonest receiver can always choose to ignore c and play with an input c' which depends on $\mathcal{R}$ and the messages in the setup stage. Thus, letting C depend on $\mathcal{R}$ and the messages in the setup stage is unavoidable. We remark that the definition would be meaningless if C was allowed to depend on the secrets s_0,s_1, and this is the reason we require a partitioning of a protocol into a setup stage and transfer stage. We stress that the security achieved in this definition is *information theoretic.*

Remark. We mention that it does not immediately follow that all the "standard" applications of OT can be performed in the bounded storage model (this is also the case for the previous protocols in this model [CCM98, Din01]). Nevertheless, we now explain how this protocol can be used as a sub-protocol to perform other cryptographic tasks. For this we note that the above definition implies security by a *simulation argument* (although the simulator is not necessarily efficient).[14]

[13] A semi-honest receiver is one that follows the protocol but *remembers more than required about* $\mathcal{R}$ and attempts to use this information to learn both secrets.

[14] Loosely speaking, the simulation paradigm requires that any attack of a malicious party can be simulated in an ideal setting where the parties interact only through a trusted party. This insures that the protocol is as secure as an interaction in the ideal setting.

Thus, for example, our OT protocol can be used as in the construction of Kilian [Kil88], to give a protocol for secure two-party computation in the bounded storage model. The security achieved guarantees that an unbounded party learns nothing about the input of the other party. We stress that typically one requires that the simulators should run with essentially the same efficiency as the attack being simulated, and that this provides a stronger notion of security.

We now give a sketch of the simulator for the receiver's strategy B^*. The simulator plays the roles of both B^* and A in the protocol up to the transfer stage. At this point the simulator computes the random variable C and calls the trusted party asking for secret C. It continues by simulating A with inputs s_C as received from the trusted party and a random s_{1-C}. By the definition this turns out to be a valid simulation, however, computing C is not necessarily efficient and therefore the simulation is unbounded.

5 Interactive Hashing

One of the main tools we use in this paper is the interactive hashing protocol. While useful in the bounded storage model, it is important to note that interactive hashing is not necessarily related to this model. As a matter of fact, the definitions and protocols given here achieve security against all powerful adversaries with no storage bounds at all.

5.1 Preliminaries: Permutations and Hash Functions

Definition 5.1 (2^k-to-1 Hash Functions) *A hash function $h : \{0,1\}^m \rightarrow \{0,1\}^{m-k}$ is 2^k-to-1 if for every output of h there are exactly 2^k pre-images. That is, $|h^{-1}(z)\}| = 2^k$ for every $z \in \{0,1\}^{m-k}$.*

One simple method of constructing a 2^k-to-1 hash function is to take a permutation on m-bit strings and omit the last k bits of its output. Clearly every output of the resulting function can be extended to 2^k different strings and therefore has 2^k pre-images. Examples of useful permutations follow.

Almost t-wise Independent Permutations. In our discussion we would like to use a random permutation on m bit strings. However, a description of such a permutation would be exponentially long since there are $(2^m)!$ such permutations. The solution is to use a permutation that falls short of being truly random but still has enough randomness to it. Specifically we want to efficiently sample a permutation π out of a small space of permutations such that when looking at π applied on any t points in $\{0,1\}^m$ then π behaves like a truly random permutation. Such a space is called a t-wise independent permutation space.

Unlike in the case of functions, where there are extremely randomness efficient constructions of t-wise independent functions, we are unaware of such constructions for permutations. Instead we further relax our demands and ask the construction to be *almost* t-wise independent, that is, the distribution induced by the permutation π on any t points is statistically close to the distribution induced on these points by a truly random permutation. Formally:

Definition 5.2 *An η-almost t-wise independent permutation space is a procedure that takes as input a seed of l bits and outputs a description of an efficiently computable permutation in S_{2^m}.*[15] *A uniformly chosen seed induces a distribution $\Pi_{t,\eta}$ on permutations such that for any t strings $x_1, \ldots x_t \in \{0,1\}^m$:*

$$\{\pi(x_1), \ldots \pi(x_t)\}_{\pi \xleftarrow{R} \Pi_{t,\eta}} \stackrel{\eta}{\equiv} \{\pi(x_1), \ldots \pi(x_t)\}_{\pi \xleftarrow{R} S_{2^m}}$$

We use the construction presented by Gowers in [Gow96].

Theorem 5.3 ([Gow96]) *There exists an η-almost t-wise independent permutation space $\Pi_{t,\eta}$ with $t = m$, $\eta = \left(\frac{1}{2^m}\right)^t$ and seed length $l = m^C$ for some constant C. Furthermore, $\Pi_{t,\eta}$ runs in time and space polynomial in the seed length.*

We note that the main Theorem of Gowers requires some special properties from the value of m. However, this is only needed to improve parameters, and the weaker results presented in the middle of the paper (Lemma 3) are satisfactory and put no limitation on the value of m. The constant in the exponent of the above Theorem is around $C = 10$, which is high but acceptable.

Other constructions of almost t-wise independent permutations were discussed in [NR99] and other references therein.

Pairwise Independent Permutations. A widely used tool is a pairwise independent permutation of strings of m bits. This is simply a 2-wise independent permutation as defined above (i.e., a 0-almost 2-wise independent permutation).

The construction that we use identifies $\{0,1\}^m$ with the field $GF(2^m)$. A permutation is sampled by randomly choosing two elements $a, b \in GF(2^m)$ with the restriction that $a \neq 0$. The permutation is then defined by $g_{a,b}(x) = ax+b$ (where all operations are in the field). Generating a pairwise independent permutation therefore requires $2m$ random bits.

Note: To construct a pairwise independent 2-to-1 hash function simply take a pairwise independent permutation and omit the last bit of its output.

5.2 Definition: Interactive Hashing

Interactive hashing is a protocol between Alice with no input and Bob with an input string. At the end of the protocol Alice and Bob should agree on two strings: One should be Bob's input and intuitively the other should be random. Moreover, Alice should not be able to distinguish which of the two is Bob's input and which is the random string.

Definition 5.4 (Interactive Hashing) *A protocol $\langle A, B\rangle$ is called an* interactive hashing protocol *if it is an efficient protocol between Alice with no input and Bob with input string $W \in \{0,1\}^m$. At the end of the protocol both Alice and Bob output a (succinct representation of a) 2-to-1 function $h : \{0,1\}^m \rightarrow$*

[15] S_{2^m} denotes the family of all permutations on m bit strings

$\{0,1\}^{m-1}$ *and two values* $W_0, W_1 \in \{0,1\}^m$ *(in lexicographic order) so that* $h(W_0) = h(W_1) = h(W)$.

Let $d \in \{0,1\}$ *be such that* $W_d = W$. *Furthermore, if the distribution of the string* W_{1-d} *over the randomness of the two parties is* η*-close to uniform, then the protocol is called* η-uniform interactive hashing *(or simply uniform interactive hashing if* $\eta = 0$*).*

Definition 5.5 (Security of Interactive Hashing) *An interactive hashing protocol is* **secure for B** *if for every unbounded deterministic strategy* A^**, and every* W*, if* h, W_0, W_1 *are the outputs of the protocol between an honest Bob with input* W *and* A^**. Then*

$$\left\{\mathrm{view}_{\mathrm{A}^*}^{\langle \mathrm{A}^*,\mathrm{B}\rangle}(\mathrm{W}) \mid \mathrm{W} = \mathrm{W}_0\right\} \equiv \left\{\mathrm{view}_{\mathrm{A}^*}^{\langle \mathrm{A}^*,\mathrm{B}\rangle}(\mathrm{W}) \mid \mathrm{W} = \mathrm{W}_1\right\}$$

An interactive hashing protocol is (s,ρ)**-secure for A** *if for every* $S \subseteq \{0,1\}^m$ *of size at most* 2^s *and every unbounded strategy* B^**, if* W_0, W_1 *are the outputs of the protocol, then:*

$$\Pr[W_0, W_1 \in S] < \rho$$

where the probability is taken over the coin tosses of A *and* B^**.*

An interactive hashing protocol is (s,ρ)**-secure** *if it is secure for* B *and* (s,ρ)*-secure for* A*.*

Remark. The definition above does not deal with the case that dishonest players abort before the end of the execution. Intuitively, such a definition is sufficient for our purposes since in our OT protocol, the interactive hashing is used before the players send any message that depends on their secrets, and thus their secrets are not compromised.

5.3 Partial Result: A Two Message Interactive Hashing

We start by showing that when the bad set S is small enough then the following naïve protocol is sufficiently good. In this 2 message protocol called 2M-IH, Alice sends a random 2-to-1 hash function $h : \{0,1\}^m \to \{0,1\}^{m-1}$ and Bob replies with $z = h(W)$.

Claim 5.6 *For all* u*, the 2M-IH protocol is a* $(s, 2^{-(m-2s+1)})$*-secure uniform interactive hashing.*

Proof: The 2M-IH is clearly an interactive hashing protocol, and since h is pairwise independent, then it is also uniform (W_{1-d} is uniformly distributed). The 2M-IH is also secure for B since all that Bob sends to Alice is $h(W)$, which is the exact same view whether Bob has input $W = W_1$ or $W = W_0$. On the other hand, since h is a pairwise independent hash function, then the probability over the choice of h for any two strings W_0, W_1 to be mapped to a certain cell $z \in \{0,1\}^{m-1}$ is perfectly random, that is:

$$\Pr_h[h(W_0) = h(W_1) = z] = 2 \cdot \frac{1}{2^m} \cdot \frac{1}{2^m - 1}$$

Denote $X_z = 1$ if both strings mapped to cell z are from the set S and $X_z = 0$ otherwise. Then:

$$\Pr_h[X_z = 1] \leq \binom{2^s}{2} \Pr_h[h(W_0) = h(W_1) = z] \leq \frac{2^s}{2^m} \cdot \frac{2^s - 1}{2^m - 1} \leq \frac{2^{2s}}{2^{2m}}$$

Denote by X the number of cells z such that both values mapped into z are from the set S, then:

$$E(X) = E\big(\sum_z X_z\big) = \sum_z E\left(X_z\right) \leq 2^{m-1} \cdot \frac{2^{2s}}{2^{2m}} \leq 2^{-(m-2s+1)}$$

The protocol is insecure only if Bob finds a cell z with two bad values, that is only if $X \geq 1$. But using Markov's inequality we have that $\Pr[X \geq 1] \leq E(x) \leq 2^{-(m-2s+1)}$. Thus this protocol is $(s, 2^{-(m-2s+1)})$-secure for Alice. ■

5.4 A Four Message Protocol for Interactive Hashing

The two message protocol is useful when the bad set S is very small. However, if S is large (for example, if $|S| = 2^s$ and $s = \gamma m$ for any constant γ) then this protocol does not suffice. While the interactive hashing protocol of [NOVY98] takes m round of communication to overcome this, the following protocol achieves this using an interaction of just four messages.

4M-IH (4 Message Interactive Hashing)

Common Input: Parameters m and s.

Let $v = s - \log m$.

A family Π of η-almost t-wise independent permutations $\pi : \{0,1\}^m \rightarrow \{0,1\}^m$

Take $t = m$ and $\eta = \left(\frac{1}{2^v}\right)^t$.

A family G of 2-wise independent 2-1 hash functions $g\colon \{0,1\}^{m-v} \rightarrow \{0,1\}^{m-v-1}$

A family H (induced by Π, G) of 2-1 hash functions $h : \{0,1\}^m \rightarrow \{0,1\}^{m-1}$ defined as:

$$h(x) \stackrel{\text{def}}{=} \pi(x)_1, \ldots, \pi(x)_v, g\left(\pi(x)_{v+1} \ldots, \pi(x)_m\right)$$

where $\pi(x)_i$ denotes the i^{th} bit of $\pi(x)$.

Input of Alice: $\perp$.

Input of Bob: $W \in \{0,1\}^m$.

- **Alice:** Choose $\pi \stackrel{\text{R}}{\leftarrow} \Pi$. Send π to Bob.
- **Bob:** Compute $z_1, \ldots z_m = \pi(W)$. Send $\pi'(W) = z_1, \ldots, z_v$ to Alice (let π' denote π when truncated to its first v bits).
- **Alice:** Choose $g \stackrel{\text{R}}{\leftarrow} G$. Send g to Bob.
- **Bob:** Send $g(z_{v+1}, \ldots, z_m)$ to Alice.
- **Alice and Bob:** Output W_0, W_1 s.t. $h(W_0) = h(W_1) = h(W)$.

Fig. 3. The four message protocol for interactive hashing.

Theorem 5.7 *For all s, the 4M-IH protocol is an $(s, 2^{-(m-s+O(\log m))})$-secure η-uniform interactive hashing protocol for $\eta = \left(\frac{1}{2^{s-\log m}}\right)^m < 2^{-m}$.*

Proof: We start by noting that the protocol is efficient for both parties due to the efficiency of the permutations used. Furthermore, they can run in small space. This is an η-uniform interactive hashing protocol since h is η close to pairwise independent and therefore the distribution of W_{1-d} is η close to uniform.

The 4M-IH protocol is secure for B since no matter what strategy A^* Alice uses, the messages that Bob sends are identical whether his input is $W = W_0$ or $W = W_1$ (recall that $h(W_0) = h(W_1)$).

This protocol has two stages of question and answer (4 messages), and in order to prove the security for A we view each of these two parts separately. In the first part, all strings $W \in \{0,1\}^m$ are divided by π' into 2^v cells (according to the value of $\pi'(W)$). Our goal is to show that no cell $z' \in \{0,1\}^v$ has too many strings from the bad set S mapped to it. The second part of the protocol can then be viewed as implementing the 2M-IH protocol on strings in the cell z', yielding the security of the combined protocol (the portion of bad strings in the cell z' is reduced to less than a square root of the strings in the cell). We start by bounding the probability that a specific set of t strings are mapped by π' to the same cell z.

Claim 5.8 *For every $z \in \{0,1\}^v$ and all $x_1, \ldots, x_t \in \{0,1\}^m$ we have that:*

$$\rho = \Pr_{\pi \in \Pi}[\pi'(x_1) = \pi'(x_2) = \ldots = \pi'(x_t) = z] \leq \left(\frac{1}{2^v}\right)^t + \eta$$

Proof: Suppose that π was a t-wise independent function (and not permutation), then for every $x_i \in \{0,1\}^m$ we have that the probability that $\pi'(x_i) = z$ is exactly $\frac{1}{2^v}$ and the probability that this is the case for t different values is exactly $\left(\frac{1}{2^v}\right)^t$. But since π is a permutation, this probability is smaller since for every i we have $\Pr[\pi'(x_i) = z | \pi'(x_1) = \pi'(x_2) = \ldots \pi'(x_{i-1}) = z] \leq \frac{1}{2^v}$. But π is actually an almost t-wise independent permutation, the probability on t elements may deviate by be up to η from the truly random permutation and therefore $\rho \leq \left(\frac{1}{2^v}\right)^t + \eta$ ■

Let us focus on a specific cell $z \in \{0,1\}^v$. For every set of t elements $x_1, \ldots, x_t \in S$ denote the $Y_z^\pi(x_1, \ldots, x_t)$ the indicator if all x_i is mapped to z or not. That is:

$$Y_z^\pi(x_1, \ldots, x_t) = \begin{cases} 1 & \pi'(x_1) = \pi'(x_2) = \ldots = \pi'(x_t) = z \\ 0 & \text{otherwise} \end{cases}$$

Let Y_z^π denote the number of strings from S mapped to cell z by π'. Let $E = \frac{2^s}{2^v}$, which is the expected number of strings from S in each cell, if they were divided uniformly at random. We claim that with high probability, Y_z^π does not deviate much from E.

Lemma 5.9 *For all* $z \in \{0,1\}^v$,

$$\Pr_{\pi\in\Pi}\left[Y_z^\pi \geq 4E\right] \leq 2^{-(t-1)}$$

Proof: Consider the table of all possible $Y_z^\pi(x_1,\ldots,x_t)$, where each row stands for a specific set $x_1,\ldots,x_t$ and each column stands for a choice of π. By Claim 5.8, the fraction of ones in each row and hence the fraction of ones in the whole table is at most $\left(\frac{1}{2^v}\right)^t + \eta$. On the other hand, for each π such that $Y_z^\pi \geq 4E$ there are at least $\binom{4E}{t}$ sets of t elements for which $Y_z^\pi(x_1,\ldots,x_t) = 1$, therefore the fraction of ones is at least $\Pr_{\pi\in\Pi}[Y_z^\pi \geq 4E]\cdot\binom{4E}{t}/\binom{2^s}{t}$. Therefore we get that:

$$\Pr_{\pi\in\Pi}\left[Y_z^\pi \geq 4E\right] \leq \frac{\binom{2^s}{t}}{\binom{4E}{t}}\left(\left(\frac{1}{2^v}\right)^t + \eta\right)$$

Recall that $\eta = \left(\frac{1}{2^v}\right)^t$ and using the fact that $\binom{a}{c}/\binom{b}{c} \leq \left(\frac{a}{b-c+1}\right)^c$ we get:

$$\Pr_{\pi\in\Pi}\left[Y_z^\pi \geq 4E\right] \leq \left(\frac{2^s}{4E-t+1}\right)^t \cdot 2 \cdot \left(\frac{1}{2^v}\right)^t$$

We take $t+1 \leq 2E$ and recall that $E = \frac{2^s}{2^v}$:

$$\begin{aligned}\Pr_{\pi\in\Pi}\left[Y_z^\pi \geq 4E\right] &\leq 2\cdot\left(\frac{2^s}{2E2^v}\right)^t \\ &\leq 2\cdot\left(\frac{2^s}{2\frac{2^s}{2^v}2^v}\right)^t = 2\cdot 2^{-t}\end{aligned}$$

This completes the proof of Lemma 5.9. ∎

As a corollary of Lemma 5.9 we get that with high probability there is no cell that contains a large number of bad elements. Applying a union bound gives:

$$\Pr_{\pi\in\Pi}\left[\exists z \text{ s.t. } Y_z^\pi \geq 4E\right] \leq 2^{-(t-1-v)}$$

Recall that $t = m$ and $v = s - \log m$ so the probability of error here is $2^{-(m-s)+\log m-1}$.

Assuming that indeed for all cells z we have $Y_z^\pi < 4E$ then the second part of the protocol is actually running the 2M-IH on the strings in a specific cell z'. This cell contains all the possible extensions of z' into an m bit string. Therefore, the 2M-IH is run on strings of length $m' = m - v$. There are no more than $2^{s'} = 4\cdot 2^{s-v}$ strings that belong to the bad set S. According to Claim 5.6 the second part of the protocol is an $(s', 2^{-(m'-2s'+1)})$-interactive hashing protocol. The probability that Bob can choose a cell with two string from the bad set is therefore $2^{-(m'-2s'+1)} = 2^{-(m-v-2(s-v+2)+1)} = 2^{-(m-s)+\log m+3}$. Combined with the probability that there exist a z with $Y_z^\pi \geq 4E$ we get that the probability that any strategy B^* that Bob plays succeeds in choosing both W_0 and W_1 in the set S is at most $2^{-(m-s+O(\log m))}$. ∎

6 The Oblivious Transfer protocol

Our BS-OT protocol is presented in figure 4. The protocol relies on three ingredients: An extractor, a min-entropy sampler, and an interactive hashing protocol. The precise requirements from the ingredients are presented in figure 5.

Input of Alice: Secret bits $s_0, s_1 \in \{0,1\}^u$.
Input of Bob: Choice bit $c \in \{0,1\}$.
Setup Stage:

Subsets Stage: Alice and Bob store subsets of the string $\mathcal{R} \in \{0,1\}^N$.

- **Alice:** Choose $P \xleftarrow{R} L_A$. Compute $A \subset [N]$ of size n by $A = \mathrm{Samp}_A(P)$ and store the bits $\mathcal{R}_A$.
- **Bob:** Choose random $B \subset [N]$ of size n and store the bits $\mathcal{R}_B$.
- **Alice:** Send A to Bob by sending P.
- **Bob:** Determine $C = A \cap B$. If $|C| < \ell$ abort. If $|C| > \ell$, randomly truncate it to be of size ℓ.
- **Bob:** Compute h_m as in Definition 3.2. Choose $Q \xleftarrow{R} [h_m]$ and compute $W = F_m(C,Q)$.*

Interactive Hashing Stage: Interactively hash W.

- **Bob:** Input W into the interactive hashing protocol.
- **Alice and Bob:** Interactively obtain h and W_0, W_1 s.t. $h(W_0) = h(W_1) = h(W)$. Compute the subsets C_0, C_1 encoded by W_0, W_1. If W_0 or W_1 isn't a valid encoding then abort.

Choice Stage:

- **Bob:** Let $d \in \{0,1\}$ be such that $W_d = W$. Send $e = c \oplus d$.
- **Alice:** For $i \in \{0,1\}$ send $Y_i \xleftarrow{R} \{0,1\}^{d_E}$.

Transfer Stage:

- **Alice:** Set $X_0 = \mathcal{R}_{C_0}$ and $X_1 = \mathcal{R}_{C_1}$.
- **Alice:** Send "encrypted" values of s_0 and s_1: For $i \in \{0,1\}$, Send $Z_i = s_{i\oplus e} \oplus E(X_i, Y_i)$.
- **Bob:** Compute $X = \mathcal{R}_C$. Bob's output is given by $\mathrm{Ext}(X, Y_{c\oplus e}) \oplus Z_{c\oplus e}$

* The range of F_m is $[n]$ and not $A = \mathrm{Samp}_A(P)$. For simplicity, we treat C as a subset of A.

Fig. 4. Protocol BS-OT for 1-2 OT in the bounded storage model.

In our suggested implementation of BS-OT we choose Samp_A to be the sampler from [Vad03], Ext to be an extractor from [RRV99] and use the $4M - IH$ interactive hashing protocol from the previous section. The precise choices of parameters for these ingredients appear in Section 8. These choices meet the requirements of figure 5 with $\epsilon = 2^{-\Omega(\ell)}$.The main theorem of this paper asserts that this implementation of BS-OT is a constant round protocol for oblivious transfer in the bounded storage model.

At first reading, the reader may safely ignore the sampler and assume that the set A is chosen uniformly at random. That is assume that Samp_A is the identity mapping on $\binom{[N]}{n}$.[16]

[16] Using different samplers allows choosing a "random" set A which has a shorter description. Specifically, using the sampler from Section 8 reduces the description size of A from $\log\binom{N}{n} = \Theta(n \log n)$ to $O(\ell)$.

Parameters:
- N - the length of the long random string $\mathcal{R}$.
- n - the number of bits honest players remember about $\mathcal{R}$.
- u - the length of the secrets.
- $\ell = n^2/2N$ - the size of the intersection set.
- ν - the dishonest receiver remembers at most νN bits about $\mathcal{R}$.
- ϵ - the error of the protocol. We can only achieve $\epsilon \geq 2^{-c\delta'_A \ell / \log(1/\delta'_A)}$ where δ'_A is defined below and $c > 0$ is some constant which may depend on the constant c_{IH} defined below. We therefore require that ϵ satisfy this condition.

Ingredients:
- A $(\delta_A, \delta'_A, \phi_A, \epsilon_A)$-min-entropy sampler $\mathrm{Samp}_A : [L_A] \to [N]^n$ with:
  - $\delta_A \leq (1-\nu)/2$.
  - $\delta'_A = \delta_A/8$.
  - $\phi_A \leq \epsilon/20$.
  - $\epsilon_A \leq \epsilon/20$.
  - L_A determines the length of the first message sent by Alice.
- A (k_E, ϵ_E)-strong extractor $\mathrm{Ext} : \{0,1\}^{n_E} \times \{0,1\}^{d_E} \to \{0,1\}^{m_E}$ with:
  - $n_E = \ell$
  - $d_E \leq \delta'_A \ell/12$
  - $m_E = u \leq \delta'_A \ell/12$.
  - $k_E \geq \delta'_A \ell/6$.
  - $\epsilon_E \leq (\epsilon/20)^2$.
- An (s, ρ)-secure (2^{-m})-uniform interactive hashing protocol for strings of length $m = 10\ell \log n$ with:
  - $s \leq m - c_{IH}\delta'_A \ell / \log \delta'_A + 1$ ($c_{IH} > 0$ is a constant chosen in the proof).
  - $\rho \leq \epsilon/20$.*

* Note that ρ depends on c_{IH} and this is why we allow the constant c in the requirement on ϵ to depend on c_{IH}. The order of quantifiers is as follows: There is some constant $c_{IH} > 0$ chosen in the proof. The constant c depends on this constant.

Fig. 5. Ingredients and requirements for Protocol BS-OT.

Theorem 6.1 *There is a constant $\alpha > 0$ such that if N, n and ℓ satisfy $\log n \leq \ell \leq n^\alpha$ then for every constant $\nu < 1$ let protocol BS-OT use the ingredients described in Section 8. Protocol BS-OT is a $(1-\epsilon)$-oblivious transfer protocol for $\epsilon = 2^{-\Omega(\ell)}$. Furthermore:*

- *The protocol has 5 messages.*
- *The strategies for Alice and Bob runs in time $poly(n)$ and space $k = O(n \log n)$.*
- *The protocol passes secrets of length $u = \Omega(\ell)$.*
- *The overall number of bits exchanged is $TC = O(\ell^{O(1)})$.*

The constants hidden in ϵ, s, u and TC above depend on ν.[17]

[17] Tracing this dependency gives that for $\delta = (1-\nu)$: $\epsilon = 2^{-\Omega(\delta\ell/\log(1/\delta))}$, $s = m - O(\delta\ell/\log(1/\delta))$, and $u = \Omega(\delta\ell)$. This holds even when ν isn't a constant as long as $n \geq \ell/\delta^4$. That is, the Theorem holds even for $\nu \approx 1 - (\ell/n)^4$.

The results mentioned in the introduction can be obtained by choosing $n = N^{1/2+a}/\log N$ for some small constant $a > 0$. Note that if a is sufficiently small then the space of honest players satisfies $k = O(n \log n) = O(N^{1/2+a}) \leq O(K^{1/2+a})$, where the last inequality follows assuming $\nu > 1/2$ which we can assume w.l.o.g. As $\ell = n^2/2N$ we have that $\ell = n^{2a}/2\log N \geq k^a$ for large enough n, and we have that $\epsilon = 2^{-\Omega(\ell)} = 2^{-\Omega(k^a)}$.

7 The Functionality and Security of the OT Protocol

The proof of Theorem 6.1 follows from the combination of several lemmas stated below. The first Lemma asserts that protocol BS-OT indeed implements oblivious transfer.

Lemma 7.1 *For every choice of ingredients for BS-OT and every s_0, s_1, c, If Alice and Bob follow protocol BS-OT then*

- *With probability $1 - 2^{-\Omega(\ell)}$ the protocol does not abort.*
- *If the protocol does not abort then Bob's output is indeed s_c.*

Proof: We first show that with high probability $|A \cap B| \geq \ell$. This is because for every fixed A, as B is a random set the expected size of $A \cap B$ is $n^2/N \geq 2\ell$. A standard Lemma (see for example Corollary 3 in [Din01]) can be used to show that there exists a constant $0 < d < 1$ such that probability that $|A \cap B| < \ell$ is at most $2e^{-d\ell}$.

We now show that the probability that one of W_0, W_1 is not a valid encoding of a subset is small. W_d was chosen by Bob and is certainly a valid encoding. By the definition of Interactive Hashing, the other string W_{1-d} is η-close to uniformly distributed in $\{0,1\}^m$, for $\eta < 2^{-m}$. By Lemma 3.3 the probability that a random string $W \in \{0,1\}^m$ is not a valid encoding is at most $\binom{n}{\ell}2^{-m} \leq 2^{\ell \log n - m} \leq 2^{-\ell-1}$ as $m = 10\ell \log n$. It follows that the probability of abort is bounded by $2^{-m} + 2^{-\ell-1} \leq 2^{-\ell}$.

To see that whenever the protocol does not abort Bob indeed outputs s_c, we observe that $X = \mathcal{R}_C$ is known to Bob (since $C = A \cap B \subseteq B$ and Bob has stored all the bits $\mathcal{R}_B$). In particular, Bob is always able to compute $E(X, Y_{c \oplus e})$ and subsequently use it in order to "decrypt" the value $Z_{c \oplus e}$. By the definition of the protocol we then have:

$$\begin{aligned} E(X, Y_{c\oplus e}) \oplus Z_{c\oplus e} &= E(X, Y_d) \oplus (s_c \oplus E(X_d, Y_d)) \\ &= E(X, Y_d) \oplus (s_c \oplus E(X, Y_d)) \qquad (1) \\ &= s_c \end{aligned}$$

where Eq. (1) follows from the fact that X_d equals $\mathcal{R}_C$ $(= X)$, which in turns follows from the fact that $C_d = C$ (since $W_d = W$ and the encoding F_m is one-to-one). The lemma follows. ∎

Theorem 7.2 *For every choice of ingredients of BS-OT, the protocol is secure for Bob.*

Proof: We show that for any strategy A^*, the view of A^* is independent of the bit c. This is shown by the following argument: Fix the randomness of A^* and $\mathcal{R}$. We show a perfect bijection between possible pairs of B's randomness r_B and input c. That is, for each pair (r_B, c) that is consistent with the view V of A^*, there exists a unique pair $(r'_B, 1-c)$ such that r'_B and $1-c$ are consistent with the same view V. There are two possible options for a $V = \text{view}_{A^*}^{\langle A^*, B\rangle}$:

- The protocol aborts before the choice stage where Bob sends Alice the value $e = c \oplus d$. In such a case, the view V is totaly independent of c and we map every consistent r_B to itself ($r'_B = r_B$). Clearly r_B is consistent with both $c = 0$ and $c = 1$.
- V includes the message $e = c \oplus d$ sent by Bob. In such a case, suppose that (r_B, c) is consistent V. That is, r_B is the randomness that chooses the random set B so that $C = A \cap B$ is encoded by the string W_d. By the fact that the protocol did not abort, we are assured that also W_{1-d} encodes a legal set C'. Then we choose r'_B to be the randomness that chooses $B' = B \setminus C \cup C'$ and encodes C' by W_{1-d}. This perfectly defines $(r'_B, 1-c)$ that is consistent with the view V. Furthermore, $(r'_B, 1-c)$ is mapped by the same process back to (r_B, c), hence we get a perfect bijection.

Theorem 7.2 follows. ■

The following theorem (which is technically the most challenging theorem of this paper) guarantees Alice's security against bounded storage receivers. This theorem refers to a list of requirements on the parameters of the ingredients which appears in figure 5.

Theorem 7.3 *For every $\nu < 1$ (not necessarily constant), if all the requirements in figure 5 are met then protocol BS-OT is $(\nu N, \epsilon)$-secure for Alice.*

The proof of this theorem is long and technical and appears in the full version of this paper. Section 9 is dedicated to giving an outline of this proof.
As we show in section 8, Ext and Samp_A and *4M–IH* satisfy all the requirements in figure 5 for $\epsilon = 2^{-\Omega(\ell)}$. Theorem 7.3 thus implies the following corollary.

Corollary 7.4 *Let* Ext, Samp_A *and IH be chosen as in Theorem 6.1. Protocol BS-OT is $(\nu N, \epsilon)$-secure for Alice, for $\epsilon = 2^{-a\ell}$ where $a > 0$ is a constant that depends on ν.*

Lemma 7.5 *Let* Ext, Samp_A *and IH be chosen as in Theorem 6.1. The statements in the itemized list in Theorem 6.1 hold.*

Proof: It is easy to verify that the protocol has 5 messages (not including the transmission of $\mathcal{R}$). By section 8 the extractor and sampler run in time polynomial in n and space $\ell^{O(1)} + O(n)$. Protocol 4M-IH runs in time and space polynomial in $m = 10\ell \log n$. Thus, both parties run in time polynomial in n. Both parties require space n to store $\mathcal{R}_A$ and $\mathcal{R}_B$ and space $m^{O(1)}$ to play 4M-IH. Alice's set A is chosen by a sampler with $\log L_A = O(\ell)$, thus it can be stored in space $O(\ell)$. Overall, Alice's space is bounded by $O(n) + poly(\ell)$. Bob's set B is a random set, and thus takes $O(n \log n)$ bits to store. We conclude that both players can run their strategies in space $O(n \log n) + poly(\ell)$ which is bounded by $O(n)$ for sufficiently small α as required. The protocol passes secrets of length m_E where $m_E = \Omega(\ell)$. Finally, the longest message sent in the protocol is the description of the permutation π in the interactive which is of length at most $\ell^{O(1)}$. ■

8 Choosing the ingredients

We now turn to choose the ingredients for BS-OT to get the parameters guaranteed in Theorem 6.1. Given n, N, u, ν, we shoot for $\epsilon = 2^{-\Omega(\ell)}$. We need to show an extractor and sampler that satisfy the conditions specified in figure 5.

The extractor. In [RRV99] it was shown how to construct a (k_E, ϵ_E)-strong extractor, $\mathrm{Ext}: \{0,1\}^\ell \times \{0,1\}^{d_E} \to \{0,1\}^u$, for every k_E, $u = k_E - 2\log(1/\epsilon_E) - O(1)$ and $d_E = c\log(1/\epsilon_E)$ for some constant c as long as $\log(1/\epsilon) > \log^4 \ell$.

Setting $k_E = \delta'_A \ell/6$, we can get $u = \delta'_A \ell/12$ for $d_E \leq \delta'_A \ell/6$ and $\epsilon_E = 2^{-c'\delta'_A \ell}$ for some constant $c' > 0$ (which depends on c). This choice satisfies the requirements in figure 5. We note that the above extractor can be computed in time and space polynomial in ℓ.

The sampler. In [Vad03] it was shown how to construct a (μ, θ, γ)-averaging sampler $\mathrm{Samp} : [L] \to [v]^t$ with distinct samples for every $\mu > \theta > 0$ and $\gamma > 0$ as long as $t \geq \Omega(\log(1/\gamma)/\theta^2)$. This sampler has $\log L \leq \log(v/t) + \log(1/\gamma)(1/\theta)^{O(1)}$. By Lemma 3.9, for every δ, γ such that $\log(1/\gamma)/\delta^4 \leq n$ this sampler yields a $(\delta, \delta/2, (\gamma + 2^{-\Omega(\delta n)})^{1/2}, (\gamma + 2^{-\Omega(\delta n)})^{1/2})$-min-entropy sampler $\mathrm{Samp}_A : [L_A] \to [N]^n$. Setting $\gamma = 2^{-\ell}$ we have that as long as $n \geq \ell/\delta^4$ this sampler has $\phi = \epsilon = 2^{-\Omega(\delta\ell))}$, and $\log L_A \leq \log n + \ell(1/\delta)^{O(1)}$.

Note that the condition $n \geq \ell/\delta^4$ is satisfied when ν is a constant (as in this case $\delta = \delta_A$ is also a constant).[18] We also note that the above sampler can be computed in time polynomial in n and space $O(n)$.

The interactive hashing protocol. We need to show that protocol 4M-IH satisfies the requirements of figure 5. It is required there that 4M-IH is $(s, 2^{-\Omega(\delta'_A \ell/\log \delta'_A)})$-secure for $s \leq m - c_{IH}\delta'_A \ell/\log \delta'_A + 1$ where $c_{IH} > 0$ is some constant and $\delta'_A = \alpha(1-\nu)$ for some $\alpha > 0$. By Theorem 5.7, we have that

$$\rho \leq 2^{-(m-s+\log m)} \leq 2^{-c_{IH}\delta'_A \ell/\log \delta'_A + O(\log m)} \leq 2^{-\Omega(\delta'_A \ell/\log \delta'_A)}$$

as $m = 10\ell \log n$ and $\ell \geq \log n$. When ν is a constant, δ'_A is also a constant and we have that $\rho = 2^{-\Omega(\ell)}$ as required. We note that Protocol 4M-IH requires requires time and space polynomial in ℓ.

[18] We remark that we don't have to require that ν is a constant. Our protocol also works for $\nu = 1 - o(1)$ as long as the condition above ($n \geq \ell/\delta^4$) is satisfied.

9 Overview of Proof of Security for Alice

Theorem 7.3 regarding Alice's security is somewhat technical and involves many parameters. Due to lack of space, we will only give a sketch of the proof while ignoring the precise parameters.

Fix some bounded storage strategy B^* with storage bound νN for some $\nu < 1$, and an input c. We need to show that there exists a random variable C determined in the setup stage such that for every two pairs of secrets s, s' which are C-consistent the view of B^* is distributed roughly the same way no matter whether Alice's input is s or s'.

Recall that in the protocol, the secrets s_0, s_1 are only involved in the transfer stage where $z_i = Ext(X_i, Y_i) \oplus s_i$ for $i \in \{0,1\}$. Our goal is to show that there exists a random variable I determined in the setup stage such that for every choice of secrets s_0, s_1, the string Z_I is close to uniformly distributed from B^*'s point of view. More precisely, for every $i \in \{0,1\}$ we split Alice's messages into Z_i and all the rest of the messages which we denote by MSG_i. For every fixing of r of $\mathcal{R}$ and msg_i of MSG_i, B^*'s point of view on Z_i is captured by considering the distribution $Z'_i = (Z_i | g^*(\mathcal{R}) = g^*(r), MSG_i = msg_i)$. We show that for most fixings r and msg_I, the random variable Z'_I is close to uniformly distributed.

We now explain how we achieve this goal. It is instructive to first consider a simplified scenario in which B^* chooses to remember the content of $\mathcal{R}$ at νN indices. We call these indices "bad" indices, and the remaining $(1-\nu)N$ indices "good" indices. Let $\delta = (1-\nu)$. The proof proceeds as follows:

1. We note that B^* does not remember the δN good indices.
2. When Alice uses a sampler to choose A, with high probability she hits a large fraction (say $\delta n/2$) of the good indices.
3. We have that the set A contains many good indices. If we were to choose a random subset of A with ℓ indices, then with high probability we will hit many (say $\delta \ell/4$) good indices. Let S be the set of all such subsets which hit less indices. By the above argument S is a small set.
4. It follows that when Alice and Bob use interactive hashing to determine the subsets C_0 and C_1, at least one of the subsets is not in S. We define the random variable I to point to this subset. It follows that C_I contains many good indices.
5. We now consider $X_I = \mathcal{R}_{C_I}$ given MSG_I. As it contains many good indices, it has high min-entropy. It follows that with high probability over the choice of Y_I, $E(X_I, Y_I)$ is close to uniformly distributed even given MSG_I. Thus, Z_I is close to uniformly distributed as required.

We now sketch how to make this argument work when B^* is allowed to remember an arbitrary function $g^* : \{0,1\}^N \to \{0,1\}^{\nu N}$ of $\mathcal{R}$. Intuitively, the notion of "min-entropy" replaces that of "good bits" in this case.

1. It is easy to see that for most fixings r of $\mathcal{R}$, the random variable $(\mathcal{R}|g^*(\mathcal{R}) = g^*(r))$ has high min-entropy (say $\Omega(\delta N)$).

2. When Alice uses a min-entropy sampler for most fixings p of P she obtains a set A such that $(\mathcal{R}_A | g^*(\mathcal{R}) = g^*(r), P = p)$ has high min-entropy.
3. Choosing a random subset is a min-entropy sampler, and thus for most choices of a subset of C of size ℓ, $(\mathcal{R}_C | g^*(\mathcal{R}) = g^*(r), P = p)$ has high min-entropy.
4. As before it follows that following the interactive hashing with high probability there exists an I such that $(\mathcal{R}_{C_I} | g^*(\mathcal{R}) = g^*(r), P = p)$ has high min-entropy.
5. Here we have to be a little more careful than before. It is no longer the case that $\mathcal{R}_{C_0}$ and $\mathcal{R}_{C_1}$ are independent given the conditioning. Thus, it may be the case that Z_{1-I} gives information about $\mathcal{R}_{C_I}$. However, we set the parameters so that $\mathcal{R}_{C_I}$ has min-entropy much larger than the length of the pair (Z_{1-I}, Y_{1-I}). As a consequence we can argue that for most fixings z_{1-I} and y_{1-I}, $(\mathcal{R}_{C_I} | g^*(\mathcal{R}) = g^*(r), P = p, Z_{1-I} = z_{1-I}, Y_{1-I} = y_{1-I})$ has high min-entropy. Thus, running an extractor, with high probability over Y_I we obtain a distribution which is close to uniform given MSG_I just as before.

10 Conclusions and open problems

We have shown a 5-message protocol for oblivious transfer in the bounded storage model. As mentioned before, this protocol has some additional concrete improvements over previous work [CCM98, Din01].

Our protocol achieves k very close to $\sqrt{K} \approx \sqrt{N}$. In words, the space of the honest parties is about a square root of the space allowed for the malicious parties. It is not clear whether there exist protocols that allow $k = N^{\delta}$ for every constant $\delta > 0$. We remark that to achieve $\delta < 2$ it is required to break the "birthday paradox barrier". A typical first step of a bounded storage protocol instructs both parties to store random subsets of the $\mathcal{R}$. When $k << \sqrt{N}$ these sets are not likely to overlap. It seems that breaking this barrier requires introducing some new ideas. We mention that to the best of our knowledge, this barrier is also present in protocols for Key-Agreement in the bounded storage model [Mau93, CM97].

We give a new constant round protocol for interactive hashing. This protocol can replace the NOVY-protocol of [NOVY98] in our setting. A similar phenomena was observed also in the context of Zero-Knowledge. Damgård [Dam93] used the NOVY-protocol to give certain transformations of "honest verifier" Zero-Knowledge protocols into general Zero-Knowledge protocols. Later works [DGOW95, GSV98] replaced the NOVY-protocol with a constant round protocol. This raises the question whether the NOVY-protocol can be replaced by a constant round protocol for the application in [NOVY98]. That is, for constructing perfectly hiding bit commitment schemes from arbitrary one-way permutations. We remark that constant round perfectly hiding bit commitment schemes are known only using seemingly stronger assumptions [NY89, DPP93, GK96].

The NOVY-protocol achieves a stronger security for interactive hashing than the one defined here. This stronger security allows its use in the application of

[NOVY98]. Loosely speaking, it is shown in [NOVY98] that their protocol is secure in the following sense: For every polynomial time malicious strategy B^* for Bob there is a polynomial time "simulator" $A_{B^*}(W')$ such that for most $W' \in \{0,1\}^m$, the simulator can run B^* playing Alice's role and generate random transcripts in which one of the outputs is W'. (Intuitively, this is a stronger and *computational* form of the intuition that Bob does not "control" the two outputs.) Obtaining this property with fewer rounds seems hard. A very related open problem was raised in [DGW95] in the context of Zero-Knowledge.

Acknowledgements. We thank Yuval Ishai for bringing [CCM98] to our attention and Oded Goldreich for helpful discussions and for pointing out some relevant work. We are also grateful to Moni Naor for insights on the NOVY protocol. Thanks also to Claude Crépeau, Salil Vadhan and the reviewers for their helpful remarks.

References

[ADR02] Y. Aumann, Y.Z. Ding, and M. O. Rabin. Everlasting security in the bounded storage model. *IEEE Transactions on Information Theory*, 48, 2002.

[AR99] Y. Aumann and M. O. Rabin. Information theoretically secure communication in the limited storage space model. In *Advances in Crypology – CRYPTO '99*, volume 1666, pages 65–79, 1999.

[BBCS92] C.H. Bennett, G. Brassard, C. Crépeau, and M.H. Skubiszewska. Practical quantum oblivious transfer protocols. In *Advances in Cryptology - CRYPTO '91, Lecture Notes in Computer Science*, volume 576, pages 351–366. Springer, 1992.

[BM89] M. Bellare and S. Micali. Non-interactive oblivious transfer and applications. In *Advances in Cryptology - CRYPTO '89, Lecture Notes in Computer Science*, volume 435, pages 547–557. Springer, 1989.

[BR94] M. Bellare and J. Rompel. Randomness-efficient oblivious sampling. In *35th IEEE Symposium on Foundations of Computer Science*, pages 276–287, 1994.

[CCM98] C. Cachin, C. Crépeau, and J. Marcil. Oblivious transfer with a memory-bound receiver. In *39th IEEE Symposium on Foundations of Computer Science*, pages 493–502, 1998.

[CK88] C. Crépeau and J. Kilian. Achieving oblivious transfer using weakened security assumptions. In *29th IEEE Symposium on Foundations of Computer Science*, pages 42–52, 1988.

[CM97] C. Cachin and U. Maurer. Unconditional security against memory-bound adversaries. In *Advances in Cryptology – CRYPTO '97*, pages 292–306, 1997.

[Cov73] T.M. Cover. Enumerative source encoding. *IEEE Transaction on Information Theory*, 19(1):73–77, 1973.

[Cré87] C. Crépeau. Equivalence between two flavours of oblivious transfers. In *Advances in Cryptology - CRYPTO '87, Lecture Notes in Computer Science*, volume 293, pages 350–354. Springer-Verlag, 1987.

[Dam93] I. Damgård. Interactive hashing can simplify zero-knowledge protocol design without computational assumptions. In *Advances in Cryptology - CRYPTO '93, Lecture Notes in Computer Science*, volume 773, pages 100–109. Springer, 1993.

[DGOW95] I. Damgård, O. Goldreich, T. Okamoto, and A. Wigderson. Honest verifier vs dishonest verifier in public cain zero-knowledge proofs. In *Advances in Cryptology - CRYPTO '95, Lecture Notes in Computer Science*, volume 963, pages 325–338. Springer, 1995.

[DGW95] I. Damgård, O. Goldreich, and A. Wigderson. Information theory versus complexity theory: Another test case, 1995.

[Din01] Y.Z. Ding. Oblivious transfer in the bounded storage model. In *Advances in Cryptology - CRYPTO '01, Lecture Notes in Computer Science*, volume 2139, pages 155–170. Springer, 2001.

[DM02] S. Dziembowski and U. Maurer. Tight security proofs for the bounded-storage model. In *34th ACM Symposium on the Theory of Computing*, pages 341–350, 2002.

[DPP93] I. Damgård, T. Pedersen, and B. Pfitzmann. On the existence of statistically hiding bit commitment schemes and fail-stop signatures. In *Advances in Cryptology - CRYPTO '93, Lecture Notes in Computer Science*, volume 773, pages 250–265. Springer, 1993.

[DR02] Y.Z. Ding and M.O. Rabin. Hyper-encryption and everlasting security. In *Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 1–26, 2002.

[EGL85] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. *Communications of the ACM*, 28(6):637–647, 1985.

[GK96] O. Goldreich and A. Kahan. How to construct constant-round zero-knowledge proof systems for np. *Journal of Cryptology*, 9(2):167–189, 1996.

[GKM+00] Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. The relationship between public key encryption and oblivious transfer. In *41st IEEE Symposium on Foundations of Computer Science*, pages 325–335, 2000.

[GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game - a completeness theorem for protocols with honest majority. In *19th ACM Symposium on the Theory of Computing*, pages 218–229, 1987.

[Gol97] O. Goldreich. A sample of samplers - a computational perspective on sampling (survey). In *Electronic Colloquium on Computational Complexity (ECCC) (20)*, volume 4, 1997.

[Gol03] O. Goldreich. Foundations of cryptography - volume 2. Working Draft, available at www.wisdom.weizmann.ac.il/oded/foc-vol2.html, 2003.

[Gow96] W.T. Gowers. An almost m-wise independent random permutation of the cube. *Combinatorics, Probability and Computing*, 5:119–130, 1996.

[GSV98] O. Goldreich, A. Sahai, and S. Vadhan. Honest-verifier statistical zero-knowledge equals general statistical zero-knowledge. In *30th ACM Symposium on the Theory of Computing*, pages 399–408, 1998.

[HCR02] Dowon Hong, Ku-Young Chang, and Heuisu Ryu. Efficient oblivious transfer in the bounded-storage model. In *Advances in Cryptology – ASIACRYPT '02, Lecture Notes in Computer Science*, pages 143–159. Springer-Verlag, December 2002.

[IR89] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *21st ACM Symposium on the Theory of Computing*, pages 44–61, 1989.

[Kil88] J. Kilian. Founding cryptography on oblivious transfer. In *20th ACM Symposium on the Theory of Computing*, pages 20–31, 1988.

[Lu02] C. Lu. Hyper-encryption against space-bounded adversaries from on-line strong extractors. In *Advances in Cryptology - CRYPTO '02*, volume 2442, pages 257–271. Springer, 2002.

[Mau92] U. Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992.

[Mau93] U. Maurer. Secret key agreement by public discussion. *IEEE Transaction on Information Theory*, 39(3):733–742, 1993.

[Nis96] N. Nisan. Extracting randomness: How and why, a survey. *IEEE Conference on Computational Complexity*, pages 44–58, 1996.

[NOVY98] M. Naor, R. Ostrovsky, R. Venkatesan, and M. Yung. Perfect zero-knowledge arguments for np using any one-way permutation. *Journal of Cryptology*, 11(2):87–108, 1998. preliminary version in CRYPTO 92.

[NP01] M. Naor and B. Pinkas. Efficient oblivious transfer protocols. In *SIAM Symposium on Discrete Algorithms (SODA 2001)*, pages 448–457, 2001.

[NR99] M. Naor and O. Reingold. On the construction of pseudorandom permutations: Luby-rackoff revisited. *Journal of Cryptology*, 12(1):29–66, 1999.

[NY89] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *21st ACM Symposium on the Theory of Computing*, pages 33–43, 1989.

[NZ96] N. Nisan and D. Zuckerman. Randomness is linear in space. *JCSS*, 52(1):43–52, 1996.

[Rab81] M. O. Rabin. How to exchange secrets by oblivious transfer. TR-81, Harvard, 1981.

[RRV99] R. Raz, O. Reingold, and S. Vadhan. Error reduction for extractor. In *40th IEEE Symposium on Foundations of Computer Science*, pages 191–201, 1999.

[RSW00] O. Reingold, R. Shaltiel, and A. Wigderson. Extracting randomness via repeated condensing. In *41st IEEE Symposium on Foundations of Computer Science*, pages 22–31, 2000.

[Sha02] R. Shaltiel. Recent developments in explicit constructions of extractors. *Bulletin of the EATCS*, 77:67–95, 2002.

[Vad03] S.P. Vadhan. On constructing locally computable extractors and cryptosystems in the bounded storage model. In *Advances in Cryptology - CRYPTO '03*. Springer, 2003.

[Yao86] A. C. Yao. How to generate and exchange secrets. In *27th IEEE Symposium on Foundations of Computer Science*, pages 162–167, 1986.

This article was processed using the LaTeX macro package with LLNCS style