Extending Oblivious Transfer Efficiently

or - How to Get Active Security with Constant Cryptographic Overhead

Enrique Larraia(✉)

Department of Computer Science, University of Bristol, Bristol, UK
cseldv@bristol.ac.uk

Abstract. On top of the passively secure extension protocol of [IKNP03] we build a new construction secure against active adversaries. We can replace the invocation of the hash function that is used to check the receiver is well-behaved with the XOR of bit strings. This is possible by applying a cut-and-choose technique on the length of the bit strings that the receiver sends in the reversed OT. We also improve on the number of seeds required for the extension, both asymptotically and practically. Moreover, the protocol used to test receiver's behaviour enjoys unconditional security.

1 Introduction

Oblivious Transfer (OT), concurrently introduced by Rabin [Rab81] and Wiesner [Wie83] (the latter under the name of multiplexing) is a two-party protocol between a sender Alice and a receiver Bob. In its most useful version the sender has two secret bit strings, and the receiver wants to obtain one of the secrets at his choosing. After the interaction the receiver has not learnt anything about the secret string he has not chosen, and the sender has not learnt anything about the receiver's choice. Several flavours have been considered and they turn out to be equivalent [EGL85, BCR86a, BCR86b, Cré87].

In the *Universally Composable Framework* [Can01], OT has been rigorously formalized and proved secure [CLOS02] under the assumption of trapdoor permutations (static adversaries) and non-committing encryption (adaptive adversaries). It was further realized [PVW08] under several hard assumptions (DDH, QR or worst-case lattice problems).

OT is a powerful cryptographic primitive that may be used to implement a wide range of other cryptographic primitives [Kil88, IPS08, Yao82, GMW86, GV87, EGL85]. Unfortunately, the results of Impagliazzo and Rudich [IR89] make it very unlikely that one can base OT on one-way functions (as a black-box).

As a second best solution, Beaver showed in its seminal paper [Bea96] that one can implement a large number of oblivious transfers assuming that only a small number of OTs are available. This problem is known as *Extended Oblivious Transfer*. The OTs that one starts with are sometimes called the *seeds* of

D.F. Aranha and A. Menezes (Eds.): LATINCRYPT 2014, LNCS 8895, pp. 368–386, 2015.
DOI: 10.1007/978-3-319-16295-9_20

the extension. Beaver showed that if one starts with say n seeds, it is possible to obtain any polynomial number (in n) of extended OTs. His solution is very elegant and concerns feasibility, but it is inherently non-efficient. Later, Ishai et al. [IKNP03] showed a very efficient reduction for semi-honest adversaries. Since then other works have focused on extensions with active adversaries [IKNP03,HIKN08,IPS08,NNOB12]. This paper continues this line of research.

State of the Art. The approach initiated in [IKNP03] runs at his core a reversed OT to implement the extension. As already noted in [IKNP03], proving security against a cheating receiver Bob^* is not trivial, as nothing refrains him from inputting whatever he likes in the reversed OT, allowing him to recover both secrets on Alice's side.

In terms of efficiency, the passive version of [IKNP03] needs $O(s)$ OT seeds, where s is a security parameter, with cut-and-choose techniques and the combiner of [CK88] active security comes at the cost of using $\Omega(s)$ seed OTs[1]. In [HIKN08] active security is achieved at no extra cost in terms of seed expansion (and communication), they apply OT-combiners worsening the computational cost. In [NNOB12] the expansion factor is $\frac{8}{3} \approx 2.66$, which is already quite good. Recently, it has been shown [LZ13] that meaningful extensions only exist if one starts with $\omega(\log s)$ seeds, (for $\log s$ seeds one would have to construct an OT protocol from the scratch). The constructions of [Bea96,IKNP03] can be instantiated with superlogarithmic seeds, so are optimal in this respect.

The communication cost is not really an issue, due to known almost-free reductions of $\mathcal{OT}^{n}_{poly(n)}$ to $\mathcal{OT}^{n}_{n}$, using a *pseudo random generator*, and running the small OT on the seeds. The computational cost of [IKNP03] is extremely efficient (passive version), it needs $O(s)$ work, i.e. constant work per extended OT (precisely it needs three invocations of the cryptographic primitive). All active extensions need at least *amortized* $\Omega(s)$ work.

Our Contributions. A technique that has proven to be quite useful [Nie07] is to split the extension protocol in two: an outer protocol ρ, and an inner protocol π. The former implements the actual extended transfers, whereas the latter wraps the reversed OT, ensuring at the same time that the receiver Bob is well-behaved in some sense. We follow the same idea, the novelty of our construction being in how the inner protocol π is realized. More concretely, for a fixed security level s we give a family of protocols $\pi_{m,n,t}$, where n is the number of seeds, m is the number of extended transfers, and $t \in [\frac{1}{n}, 1)$. Values of t close to $\frac{1}{n}$ render less OT seeds, and values close to 1 less computational and communication cost. We obtain

- The overall construction has *amortized constant cost in terms of cryptographic computation*. Active security is obtained at the cost of XORing $O((1-t)n^2)$ bits. The construction has similar communication complexity. The previous best [NNOB12] need to hash $O(n)$ bits per extended transfer.

[1] The hidden constant is quite big.

- The seed expansion factor of the reduction, with respect to the passive version of [IKNP03] is asymptotically close to 2, and this convergence is quite fast, for example for security level $s = 128$ one needs about $n = 323$ seeds to produce about $1, 00, 000$ extended OTs. This means that our construction essentially suffers an overhead factor of 2 in the security parameter, with respect to the passive protocol of [IKNP03].
- The reduction of π to the inner OT is *information-theoretic*. Other constructions either required computational assumptions e.g. [IKNP03,HIKN08, IPS08], or were in the random oracle [Nie07,NNOB12]. The outer protocol ρ is the standard protocol of [IKNP03], thus it uses a *correlation robust function*.

Our proof technique is, to some extent, similar to those of [Nie07,NNOB12] in the sense that it is combinatorial. Instead of working with permutations, we are able to connect security with set partitions. In [NNOB12] adversarial behaviour was quantified through what the authors called *leakage functions*. We take a different approach, and measure adversarial behaviour with the *thickness* of a partition. Details are in Sect. 4.3.

Paper Organization. Notation and basic background is introduced in Sect. 2. Section 3 discusses the approach of [IKNP03] and fits it in our context. In Sect. 4 we present the inner protocol π and prove it secure. In Sect. 5 the final construction is concluded, we discuss complexity and further directions.

2 Preliminaries

2.1 Notation

We denote with $[n]$ the set of natural number less or equal than n. Let $\mathbb{F}_2$ be the field of telements, binary vectors $\mathbf{x}$ are written in bold lowercase and binary matrices $\mathbf{M}$ in bold uppercase. When $\mathbf{M}$ is understood from the context, its rows will be denoted with subindices $\mathbf{m}_i$, and its columns with superindices $\mathbf{m}^j$. The entry at position (i, j) is denoted with m_i^j. Accordingly, the jth bit of a row vector $\mathbf{r} \in \mathbb{F}_2^n$ will be denoted with r^j, and the ith bit of a column vector $\mathbf{c} \in \mathbb{F}_2^m$ with c_i. For any two matrices $\mathbf{M}$, $\mathbf{N}$, of dimension $m \times n$, we let $[\mathbf{M}, \mathbf{N}]$ be the $m \times 2n$ matrix whose first n columns are $\mathbf{m}^j$ and last n columns are $\mathbf{n}^j$. The symbol $\mathbf{a}_{|J}$ stands for the vector obtained by restricting $\mathbf{a}$ at positions indexed by J.

2.2 Set Partitions

Given a finite set X of n objects, for any $p \leq n$, a *partition* $\mathcal{P}$ of X is a collection of p pairwise disjoint subsets $\{P_k\}_{k=1}^p$ of X whose union is X. Each P_k is a *part* of X. We say that part P_k is maximal if its size is the largest one. Let $\mathcal{ER}(X)$ denote the set of all possible *equivalence relations* in X. There is a one-to-one correspondence between partitions of X and equivalence relations in X, given by the mapping $\mathcal{P} \mapsto \mathcal{R}$, where $x\mathcal{R}y$ iff $x \in P_k$ and $y \in P_k$. We write $\mathcal{P}^X$ to denote the set of all partitions of X. In this work we will be concerned with partitions of the set $[n]$, where n is the number of OT seeds.

2.3 Universally Composable Framework

Due to lack of space we assume the reader is familiar with the UC Framework [Can01], especially with the notions of environment, ideal and real adversaries, indistinguishability, protocol emulation, and the composition theorem. Functionalities will be denoted with calligraphic $\mathcal{F}$. As an example $\mathcal{OT}_n^m$ denotes the OT functionality, in which the sender inputs m pairs of secret strings $(\mathbf{l}_i, \mathbf{r}_i)_{i\in[m]}$, each string of length n. The receiver inputs vector $\boldsymbol{\sigma} \in \mathbb{F}_2^m$, and as a result obtains the ith left secret $\mathbf{l}_i$ if $\sigma_i = 0$, or the ith right secret $\mathbf{r}_i$ if $\sigma_i = 1$. We will also make use of a *correlation robust function*. We name the output of the CRF as the *hash* of the input. Some times we will write H instead of CRF. The definition can be found in [IKNP03].

3 The IKNP Approach

In 2003, in their breakthrough, Ishai, Kilian, Nissim and Petrank [IKNP03] opened the door for practical OT extensions. They provided two protocols for this task. Throughout this paper we will sometimes refer to the passive version as the IKNP extension. We consider the standard OT functionality [CLOS02] in its multi session version, the only difference is that the adversary is allowed to abort the execution. This is necessary because of how we deal with a cheating sender (see Fig. 3).

3.1 IKNP in a Nutshell

For any $m = \text{poly}(n)$, the ideal functionality $\mathcal{OT}_n^m$ is realized making a single call to $\mathcal{OT}_m^n$, where the security parameter of the reduction depends on n. This in turn implies a reduction to $\mathcal{OT}_n^n$ using a pseudorandom generator. It works as follows: Let $\boldsymbol{\sigma} \in \mathbb{F}_2^m$ be the input of Bob to $\mathcal{OT}_n^m$, he chooses a $m \times 2n$ binary matrix $[\mathbf{L}, \mathbf{R}]$ for which it holds $\mathbf{l}^j \oplus \mathbf{r}^j = \boldsymbol{\sigma}$, $j \in [n]$, but is otherwise random, and inputs it to an inner $\mathcal{OT}_m^n$ primitive. Alice inputs a random vector $\mathbf{a} \in \mathbb{F}_2^n$. As a result of the call Alice obtains (row) vectors $\{\mathbf{q}_i\}_{i\in[m]}$, for which hold $\mathbf{q}_i = \mathbf{l}_i \oplus \sigma_i \cdot \mathbf{a}$. Now, if Alice wants to obliviously transfer one of her two ith secrets $(\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)})$, she XORs them with $\mathbf{p}_i^{(0)} = \mathbf{q}_i$ and $\mathbf{p}_i^{(1)} = \mathbf{q}_i \oplus \mathbf{a}$ respectively, and sends masks $\mathbf{y}_i^{(0)}$, $\mathbf{y}_i^{(1)}$ to Bob, who can obtain $\mathbf{x}_i^{(b_i)}$ from $\mathbf{y}_i^{(b_i)}$ and $\mathbf{l}_i$. This can be used to implement one transfer out of the m that Bob wishes to receive, but can not cope with more: the OTP used for the ith transfer, with pads $(\mathbf{p}_i^{(0)}, \mathbf{p}_i^{(1)})$, prohibits to use $(\mathbf{p}_j^{(0)}, \mathbf{p}_j^{(1)})$ in the jth transfer, because they are correlated (the *same* $\mathbf{a}$ is implicit in both pairs[2]). To move from a situation with correlated pads to a situation with uncorrelated ones, IKNP uses a CRF; i.e. Alice masks $\mathbf{x}_i^{(c)}$ with the hash of $\mathbf{p}_i^{(c)}$. The construction is perfectly secure

[2] Bob would learn e.g. the distance of two non-transmitted secrets. It is trivial to check that if two correlated pairs are used by Alice, then $\mathbf{x}_i^{(1+b_i)} \oplus \mathbf{x}_j^{(1+b_j)} = \mathbf{y}_i^{(1+b_i)} \oplus \mathbf{y}_j^{(1+b_j)} \oplus \mathbf{l}_i \oplus \mathbf{l}_j$.

against a malicious sender Alice^*, and statistically secure against a *semi-honest* receiver Bob^*.

Intuitively, each input bit, σ_i, of Bob is protected by using n independent additive sharings as inputs to the inner $\mathcal{OT}^n_m$. As for Alice's privacy, the crucial point being that as long as $\mathbf{a}$ is not known to Bob, then $\mathbf{x}^{(1+b_i)}$ remains hidden from him; in that situation, one of the pads in each pair is independent of Bob's view. Unfortunately, the above crucially relies on Bob following the protocol specifications. In fact, it is shown in [IKNP03] how Bob^* can break privacy if he chooses carefully what he gives to the inner $\mathcal{OT}^n_m$.

3.2 Modularizing the Extension

We define an ideal functionality that acts as a wrapper of the inner call to the $\mathcal{OT}$ primitive.[3] It behaves as follows: (1) On an honest input $\mathbf{B} = [\mathbf{L}, \mathbf{R}]$ from Bob (i.e. $\mathbf{B}$ defines n sharings of some vector $\boldsymbol{\sigma}$), the functionality gives to Alice a pair $(\mathbf{a}, \mathbf{Q})$ that she will use to implement the extended transfers. The secret $\mathbf{a}$ is randomly distributed in Bob's view. (2) An ideal adversary $\mathcal{S}$ can guess d bits of $\mathbf{a}$, in this case the functionality takes the guesses with probability 2^{-d}. The secret $\mathbf{a}$ has $n-d$ bits randomly distributed in Bob's view.

The functionality is denoted with $c\mathcal{PAD}_{m,n}$ to emphasize that it gives m correlated pairs of pads, under the same $\mathbf{a}$ to Alice (of length n). See Fig. 1 for a formal description. We emphasize that $c\mathcal{PAD}$ without the malicious behaviour was implicit in [IKNP03], and with the malicious behaviour in [Nie07]. We have just made the probability of aborting more explicit. The novelty of our approaches lies in how is realized.

For completeness, we have included the IKNP extension protocol, see Fig. 2 for details. The only difference is that the pads $(\mathbf{p}_i^{(0)}, \mathbf{p}_i^{(1)})_{i \in [m]}$ that Alice uses to generate uncorrelated ones via the CRF are assumed to be given by $c\mathcal{PAD}_{m,n}$.

3.3 The Reduction

The proof is on the same lines of the reduction of [IKNP03]. For the case the receiver is actively corrupted, with $c\mathcal{PAD}_{m.n}$ at play, Bob^* is forced to take a guess *before* the actual extended transfers are executed. He is not caught with probability 2^{-d}, in which case $n-d$ bits of $\mathbf{a}$ are completely unknown to him. This correspondence between adversarial advantage and uncertainty (observed in [Nie07]) is the key to argue security in the active case. What we observe is that within the set F that indexes the $n-d$ unknown bits, either $\mathbf{a}$ or the flipped vector $\mathbf{a} \oplus \mathbf{1}$ has at least $(n-d)/2$ bits set to one. Consequently, the same number of bits of one of the pads that Alice uses remains unknown to Bob^*. Using bounding techniques borrowed from [Nie07] it is not difficult to simulate ρ with security *essentially half* the security of the IKNP extension.

[3] The purpose of the otherwise seemingly artificial functionality is to give a neat security analysis, both inwardly and outwardly.

Functionality $c\mathcal{PAD}_{m,n}$

$c\mathcal{PAD}$ runs with a pad's receiver Alice, a pad's creator Bob, and an adversary $\mathcal{S}$. It is parametrized with the numbers of transfers m, and the length of the bit strings n.

- Upon receiving input $(\mathsf{receiver}, sid)$ from Alice and $(\mathsf{creator}, sid, [\mathbf{L}, \mathbf{R}])$ from Bob, where $[\mathbf{L}, \mathbf{R}] \in \mathcal{M}_{m \times 2n}$ defines n sharings of the same vector $\boldsymbol{\sigma}$, sample at random $\mathbf{a} \in \mathbb{F}_2^n$. Then record the tuple $(\mathbf{a}, [\mathbf{L}, \mathbf{R}])$, send (sid) to $\mathcal{S}$ and halt.
- Upon receiving message $(\mathsf{deliver}, sid)$ from $\mathcal{S}$, compute matrix $\mathbf{Q} \in \mathcal{M}_{m \times n}$ as

$$\mathbf{q}_i = \mathbf{l}_i \oplus (l_i^1 \oplus r_i^1) \cdot \mathbf{a}.$$

  Output $(\mathsf{delivered}, sid, \mathbf{a}, \mathbf{Q})$ to Alice, $(\mathsf{delivered}, sid)$ to Bob and $\mathcal{S}$, and halt.
- Upon receiving $(\mathsf{corruptAlice}, sid, \tilde{\mathbf{a}})$ from $\mathcal{S}$, where $\tilde{\mathbf{a}} \in \mathbb{F}_2^n$, give $\tilde{\mathbf{Q}} = [\mathbf{l}^j \oplus \tilde{a}^j(\mathbf{l}^j \oplus \mathbf{r}^j]_{j \in [n]}$ to $\mathcal{S}$. If additionally $\mathcal{S}$ sends $(\mathsf{corruptAlice}, sid, \perp)$, output (abort, sid) to Alice and Bob and halt.
- Upon receiving message $(\mathsf{corruptBob}, sid, [\tilde{\mathbf{L}}, \tilde{\mathbf{R}}], \tilde{\mathbf{a}}, G)$ from $\mathcal{S}$, where $G \subseteq [n]$ is of size d, and $\tilde{\mathbf{a}} \in \mathbb{F}_2^d$, do:
  - with probability $p = 1 - 2^{-d}$, output $(\mathsf{corruptBob}, sid)$ to Alice and $\mathcal{S}$ and halt. Else,
  - replace $\mathbf{a}_{|G}$ with $\tilde{\mathbf{a}}$, and compute matrix $\mathbf{Q}$ subject to

$$\mathbf{q}_i = \tilde{\mathbf{l}}_i \oplus (\tilde{\mathbf{l}}_i \oplus \tilde{\mathbf{r}}_i) * \mathbf{a}.$$

  Output $(\mathsf{delivered}, sid, \mathbf{a}, \mathbf{Q})$ to Alice, $(\mathsf{delivered}, sid)$ to Bob and $\mathcal{S}$, and halt.

Fig. 1. Modeling creation of correlated pads

Protocol ρ

The protocol is parametrized with the number of extended transfers m, and the length of the transmitted vectors n.

Primitive: A $c\mathcal{PAD}_{m,n}$ functionality.

Inputs: Alice inputs $(\mathsf{sender}, (\mathbf{x}_{i,0}, \mathbf{x}_{i,1})_{i \in [m]}, sid)$, where $\mathbf{x}_{i,c} \in \mathbb{F}_2^n$, and Bob inputs $(\mathsf{receiver}, \boldsymbol{\sigma}, sid)$ with $\boldsymbol{\sigma} \in \mathbb{F}_2^m$.

Protocol:

1. Bob samples n independent sharings of $\boldsymbol{\sigma}$. Denote this sharings as $[\mathbf{L}, \mathbf{R}]$ (i.e. $\mathbf{l}^j \oplus \mathbf{r}^j = \boldsymbol{\sigma}$).
2. The parties call $c\mathcal{PAD}$. Bob inputs $(\mathsf{creator}, sid, [\mathbf{L}, \mathbf{R}])$, and Alice inputs $(\mathsf{receiver}, sid)$, as a result Alice gets $(\mathbf{a}, \mathbf{Q})$ where $\mathbf{Q}$ is a $m \times n$ binary matrix, and $\mathbf{a} \in \mathbb{F}_2^n$.
3. Let $\mathbf{p}_i^{(0)} = \mathbf{q}_i$ and $\mathbf{p}_i^{(1)} = \mathbf{q}_i \oplus \mathbf{a}$. Alice computes $\mathbf{y}_i^{(c)} = \mathbf{x}_i^{(c)} \oplus H_i^{(c)}(\mathbf{p}_i^{(c)})$ for $c = 0, 1$, and sends pairs $(\mathbf{y}_i^{(0)}, \mathbf{y}_i^{(1)})_{i \in [m]}$ to Bob.

Outputs: Bob computes $\mathbf{h}_i = H_i^{(\sigma_i)}(\mathbf{l}_i)$ and outputs $\mathbf{x}_i^{'} = \mathbf{y}_i^{(\sigma_i)} \oplus \mathbf{h}_i$. Alice outputs nothing.

Fig. 2. IKNP extension

Claim (Restatement of [IKNP03, Lemma1] for Active Adversaries). In the $c\mathcal{PAD}_{m,n}$-hybrid model, in the presence of static active adversaries, with access to at most $2^{o(n)}$ queries of a CRF, the output of protocol ρ, and the output of the ideal process involving $\mathcal{OT}_n^m$, are $2^{-n/2+o(n)+2}$-close.

For completeness it follows a proof sketch that combines the proofs of [IKNP03, Nie07]. Later, in Sect. 4.4 we will elaborate on an alternative idea for the simulation.

We focus on the case Bob* is corrupted, simulating a malicious Alice* is easy, and we refer the reader to [IKNP03] for details. To simulate a real execution of ρ,

Functionality $\mathcal{OT}_n^m$

The functionality is parametrized by the number of transfers m, and the length of bit strings n. It runs between a sender Alice, a receiver Bob and an adversary $\mathcal{S}$.

1. Upon receiving $(\mathsf{sender}, sid, (\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)})_{i\in[m]})$ from Alice, where $\mathbf{x}_i^{(c)} \in \mathbb{F}_2^n$, record tuple $(\mathbf{x}_i^{(0)}, \mathbf{x}_i^{(1)})_{i\in[m]}$. (The length n and number of transfers t is fixed and known to all parties)
2. Upon receiving $(\mathsf{receiver}, sid, \boldsymbol{\sigma})$ from Bob, where $\boldsymbol{\sigma} \in \mathbb{F}_2^m$, send (sid) to $\mathcal{S}$, record $\boldsymbol{\sigma}$ and halt.
3. Upon receiving $(\mathsf{deliver}, sid)$ from $\mathcal{S}$, send $(\mathsf{delivered}, sid, (\mathbf{x}_i^{(\sigma_i)})_{i\in[m]})$ to Bob and $(\mathsf{delivered}, sid)$ to Alice and halt.
4. Upon receiving (abort, sid) from $\mathcal{S}$, and only if $(\mathsf{deliver}, sid)$ was not previously received, send (fail, sid) to Alice and Bob and halt.

Fig. 3. The functionality of [CLOS02] augmented with aborts

- $\mathcal{S}$ internally runs steps 1 and 2 of ρ. If $\mathcal{A}$ sends $(\mathsf{deliver}, sid)$ to $c\mathcal{PAD}$, then $\mathcal{S}$ sets $\mathbf{r} \stackrel{\mathsf{def}}{=} \boldsymbol{\sigma}^*$, where $\boldsymbol{\sigma}^*$ is what $\mathcal{A}$ specified as input to $c\mathcal{PAD}$.
  Otherwise, $\mathcal{S}$ internally gets message $(\mathsf{corruptBob}, sid, [\tilde{\mathbf{L}}, \tilde{\mathbf{R}}], \tilde{\mathbf{a}}, G)$ from $\mathcal{A}$, then $c\mathcal{PAD}$ either rejects, in which case $\mathcal{S}$ externally sends (abort, sid) to $\mathcal{OT}_n^m$, outputs what ρ_{B} outputs and halts.
  If $c\mathcal{PAD}$, does not abort, let $F = [n]\backslash G$, then (for each $i \in [m]$) split it in two disjoint subsets, F_1, F_0 such that the bits of $\tilde{\mathbf{l}}_i \oplus \tilde{\mathbf{r}}_i$ indexed with F_{c_i} are equal to bit c_i. Say F_{r_i} is the largest set. $\mathcal{S}$ sets $\mathbf{r} \stackrel{\mathsf{def}}{=} (r_1, \ldots, r_m)$.
- Next, $\mathcal{S}$ externally calls $\mathcal{OT}_n^m$ on input $\mathbf{r}$ getting output $(\mathbf{z}_i)_{i\in[m]}$. It then fills the input tape of ρ_{A} with $\mathbf{x}_i^{(r_i)} = \mathbf{z}_s$ and $\mathbf{x}_i^{(r_i+1)} = \mathbf{0}^n$, executes step 3 of ρ, outputs what ρ_{B} outputs and halts.

Fig. 4. The ideal adversary for actively corrupted receivers

an ideal adversary $\mathcal{S}$ starts setting an internal copy of the real adversary $\mathcal{A}$, and runs the protocol between $\mathcal{A}$ and dummy parties ρ_{A} and ρ_{B}. The communication with the environment $\mathcal{E}$ is delegated to $\mathcal{A}$. Recall t hat $\mathcal{S}$ is also interacting with the (augmented with aborts) ideal functionality $\mathcal{OT}_n^m$ (see Fig. 3). A description of $\mathcal{S}$ for a malicious Bob^* is in Fig. 4.

Let Dist be the event that $\mathcal{E}$ distinguishes between the ideal process and the real process, we examine the simulation conditioned on three disjoint events: SH is the event defined as "$\mathcal{A}$ *sends* $(\mathsf{deliver}, sid)$ to $c\mathcal{PAD}$", Active is the event "$\mathcal{A}$ *sends* $(\mathsf{corruptBob}, sid)$ and $c\mathcal{PAD}$ does not abort", and Abort is the event "$\mathcal{A}$ *sends* $(\mathsf{corruptBob}, sid)$ and $c\mathcal{PAD}$ aborts". It is clear that conditioned on Abort the simulation is perfect (with unbounded environments), because no transfers are actually done. Now, say that $|G| = d$, then $c\mathcal{PAD}_{m,n}$ does not abort with probability 2^{-d}, so we write

$$Pr[\mathsf{Dist}] \leq Pr[\mathsf{Dist}|\mathsf{SH}] + Pr[\mathsf{Dist}|\mathsf{Active}] \cdot 2^{-d} \tag{1}$$

Conditioning on Active. In this case, the only difference between the ideal and the real process is that $\mathcal{S}$ fills with garbage the secret $\mathbf{x}_i^{(r_i+1)}$ of ρ_{A}, thus, the transcripts are indistinguishable *provided* $\mathcal{E}$ (or $\mathcal{A}$) does not submit $Q = \mathbf{p}_i^{(r_i+1)}$ to the CRF (in that case, $\mathcal{E}$ sees the zero vector in the ideal process, and the actual input of Alice in the real process). It is enough to see that this happens with negligible probability: First, pad $\mathbf{p}_i^{(r_i+1)}$ restricted at positions indexed with F_{r_i} can be expressed as

$$\begin{aligned}\mathbf{p}_{i|F_{r_i}}^{(r_i+1)} &= \mathbf{q}_{i|F_{r_i}} \oplus (r_i \oplus 1) \cdot \mathbf{a}_{|F_{r_i}} = (\tilde{\mathbf{l}}_i \oplus (\tilde{\mathbf{l}}_i \oplus \tilde{\mathbf{r}}_i) * \mathbf{a})_{|F_{r_i}}) \oplus (r_i \oplus 1) \cdot \mathbf{a}_{|F_{r_i}} \\ &= \tilde{\mathbf{l}}_{i|F_{r_i}} \oplus r_i \cdot \mathbf{a}_{|F_{r_i}} \oplus (r_i \oplus 1) \cdot \mathbf{a}_{|F_{r_i}} \\ &= \tilde{\mathbf{l}}_{i|F_{r_i}} \oplus \mathbf{a}_{|F_{r_i}}.\end{aligned}$$

Second, the size of F_{r_i} is at least $(n-d)/2$, because $F = F_0 \vee F_1$ and F_{r_i} is maximal. Third, $c\mathcal{PAD}_{m,n}$ generates $\mathbf{a}_{|F_{r_i}}$ using his own random bits. It follows that $\mathbf{p}_i^{(r_i+1)}$ has $(n-d)/2$ bits randomly distributed in $\mathcal{E}$'s view.

He may still guess such bits searching through the query space and using the CRF to compare. We next bound the probability of this happening. If $\mathcal{E}$ (or $\mathcal{A}$) guess correctly such bits, they would have handed to the CRF query $Q = \mathbf{p}_i^{(r_i+1)}$. As $(n-d)/2$ bits are unknown, the CRF returns random answers on $\mathcal{E}$'s view, the probability of hitting all the bits in $\mathbf{p}_i^{(r_i+1)}$ is bounded by $p_i \leq h_{r_i+1}2^{(d-n)/2}$ where h_{r_i+1} is the number of queries made to $H_i^{(r_i+1)}$. By the union bound, given h denoting the total number of queries, $\mathcal{E}$ and $\mathcal{A}$ jointly hit query $Q = \mathbf{p}_i^{(r_i+1)}$ for some $i \in [m]$, with probability

$$Pr[\mathsf{Dist}|\mathsf{Active}] \leq 2(\sum_{i\in[m]} h_{r_i+1}2^{(d-n)/2}) \leq h2^{d/2+1-n/2}. \tag{2}$$

Conditioning on SH. This case corresponds to semi-honest adversaries. We refer the reader to the proof of [IKNP03] for details. The only difference is that now *also* $\mathcal{A}$ can submit arbitrary queries to the CRF, hitting the offending one with the same probability than the environment would, thus

$$Pr[\mathsf{Dist}|\mathsf{SH}] \leq h2^{-n+1}. \tag{3}$$

Plugging inequalities 2 and 3 into 1, we obtain that the simulation fails with probability

$$Pr[\mathsf{Dist}] \leq h2^{-n+1} + h2^{d/2+1-n/2} \cdot 2^{-d} \leq h2^{-n/2+2}.$$

The Claim follows setting $h = 2^{o(n)}$. □

4 Generating Correlated Pads

The result of Sect. 3.3 (and previous works) shows that the IKNP extension can be upgraded to active security assuming that any adversarial strategy, on the receiver's side, amounts to guessing some of the bits of the sender's secret $\mathbf{a}$ *before* the extended transfers are executed. In this section we realize the $c\mathcal{PAD}$ functionality in a way where the only computational cost involved, beyond the underlying OT primitive on which it builds, is XORing bit strings.

Protocol $\pi_{m,n,t}$

The protocol is parametrized with the length of the input m, the number of OT seeds n, and a parameter $t \in [\frac{1}{n}, 1)$.

Primitive: An $\mathcal{OT}^{n}_{m(r+1)}$ functionality with $r = \lceil \frac{1-t}{2} n \rceil$.

Inputs: Bob inputs $[\mathbf{L}_0, \mathbf{R}_0] \in \mathcal{M}_{m\times 2n}$ defining n sharings of some vector $\boldsymbol{\sigma}_0 \in \mathbb{F}_2^m$ (i.e. $\mathbf{l}_0^j \oplus \mathbf{r}_0^j = \boldsymbol{\sigma}_0$ for $j \in [n]$). Alice inputs nothing.

Commit Phase:

1. Alice samples $\mathbf{a} \in \mathbb{F}_2^n$ at random, and Bob randomly samples r matrices $[\mathbf{L}_i, \mathbf{R}_i]$ in $\mathcal{M}_{m\times 2n}$ (i.e. $i \in [r]$). Each defining n sharings of (say) vectors $\boldsymbol{\sigma}_1, \ldots, \boldsymbol{\sigma}_r$.
2. The parties call the $\mathcal{OT}^{n}_{r(m+1)}$ functionality. Alice inputs $\mathbf{a}$, and Bob offers matrix $[[\mathbf{L}_0, \ldots, \mathbf{L}_r], [\mathbf{R}_0, \ldots, \mathbf{R}_r]]$ as his matrix of n (left,right) secrets of length $r(m+1)$ (towering up the $\mathbf{L}_i$'s together, idem with the $\mathbf{R}_i$'s). As a result Alice obtains output matrix $[\mathbf{Q}_0, \ldots, \mathbf{Q}_r] \in \mathcal{M}_{r(m+1)\times n}$.

Prove Phase: Alice challenges Bob to make sure he used a good enough input matrix.

3. Alice sends to Bob a random challenge vector $\mathbf{e} \in \mathbb{F}_2^r$.
4. For $i \in [r]$, Bob computes $\tilde{\boldsymbol{\sigma}}_i = \boldsymbol{\sigma}_i \oplus e_i \cdot \boldsymbol{\sigma}_0$, $\tilde{\mathbf{L}}_i = \mathbf{L}_i \oplus e_i \cdot \mathbf{L}_0$, and $\tilde{\mathbf{R}}_i = \mathbf{R}_i \oplus e_i \cdot \mathbf{R}_0$. It sends the r proofs $(\tilde{\boldsymbol{\sigma}}_i, [\tilde{\mathbf{L}}_i, \tilde{\mathbf{R}}_i])_{i\in[r]}$ to Alice.
5. For each $i \in [r]$, and $j \in [n]$ Alice prepares witnesses $\tilde{W}_i = (\mathbf{a}, \tilde{\mathbf{Q}}_i = \mathbf{Q}_i \oplus e_i \cdot \mathbf{Q}_0)$, and checks whether $\tilde{\mathbf{q}}_i^j \stackrel{?}{=} \tilde{\mathbf{l}}_i^j + a^j \cdot (\tilde{\mathbf{l}}_i^j \oplus \tilde{\mathbf{r}}_i^j)$, and $\tilde{\boldsymbol{\sigma}}_i \stackrel{?}{=} \tilde{\mathbf{l}}_i^j \oplus \tilde{\mathbf{r}}_i^j$ If not, she outputs corruptBob and halts.

Outputs: If Alice did not abort, she outputs $(\mathbf{a}, \mathbf{Q}_0)$ and Bob outputs nothing.

Fig. 5. Realizing $c\mathcal{PAD}_{m,n}$

4.1 Warming Up: Committing Bob to His Input

The inner $\mathcal{OT}^n_m$ of the IKNP extension can be seen, in a way, as a commitment for Bob's input $\boldsymbol{\sigma}$ to the outer $\mathcal{OT}^m_n$. The idea resembles the commitment scheme of [Cré89] generalized to m-bit strings. We split the protocol in two phases: A "commit" phase and a "prove" phase. To commit to $\boldsymbol{\sigma}$, Bob chooses n independent sharings $\mathbf{B} = [\mathbf{L}, \mathbf{R}]$ (i.e. $\mathbf{l}^j \oplus \mathbf{r}^j = \boldsymbol{\sigma}$ for $j \in [n]$) and offers them to an $\mathcal{OT}^n_m$ primitive. For the jth sharing, Alice obliviously retrieves one of the shares using her secret bit a^j. She obtains a "witness" matrix $\mathbf{Q} = [\mathbf{q}^j]_{j\in[n]}$. To prove his input Bob reveals $(\boldsymbol{\sigma}, \tilde{\mathbf{B}})$, and Alice checks she got the right share in the first place, (i.e. she checks $\mathbf{q}^j \stackrel{?}{=} \tilde{\mathbf{l}}^j \oplus a^j \cdot (\tilde{\mathbf{l}}^j \oplus \tilde{\mathbf{r}}^j)$), and that $\tilde{\mathbf{B}}$ is consistent with $\boldsymbol{\sigma}$ (i.e. $\tilde{\mathbf{l}}^j \oplus \tilde{\mathbf{r}}^j \stackrel{?}{=} \boldsymbol{\sigma}$).

Witnessing. The above protocol is of no use in our context, as for Bob to show he behaved correctly, he would have to reveal his input $\boldsymbol{\sigma}$ to the outer $\mathcal{OT}^m_n$. Nevertheless, we retain the concept of Alice obtaining a "witness" of what Bob* gave to the inner $\mathcal{OT}$. Such object is a pair $W = (\mathbf{a}, \mathbf{Q})$ obtained as the output of an $\mathcal{OT}^n_m$ primitive. Two witnesses W, W' are *consistent* if $\mathbf{a} = \mathbf{a}'$. Similarly, a "proof for witness W" is a pair $(\boldsymbol{\sigma}, \tilde{\mathbf{B}})$ such that $\tilde{\mathbf{B}}$ defines n sharings of $\boldsymbol{\sigma}$. We say the proof is *valid* if it is consistent with W, in the sense Alice would accept in the above protocol, when she is presented the proof and uses W to check it.

We emphasize that with this terminology, the output of $c\mathcal{PAD}_{m,n}$ is precisely a witness (see Fig. 1).

4.2 The Protocol

Suppose Alice has obtained a witness W_0 and she wants to use it to implement the extended transfers (as in protocol ρ). She is not sure if in order to give her the witness Bob used a good matrix $\mathbf{B}_0 = [\mathbf{L}_0, \mathbf{R}_0]$ or a bad one (loosely speaking a good matrix defines almost n sharings of a fixed vector $\boldsymbol{\sigma}$, whereas a bad matrix has many (left,right) pairs adding up to distinct vectors.). Now, say that Alice has not one but two witnesses W_0, W_1. If they are consistent it is not difficult to see that she also knows a witness for $\mathbf{B}_+ = \mathbf{B}_0 \oplus \mathbf{B}_1$. So what Alice can do is to ask Bob to "decommit" to $\mathbf{B}_+$ as explained in Sect. 4.1. Intuitively Bob* is able to "decommit" if $\mathbf{B}_+$ is not a bad matrix. It is also intuitive that $\mathbf{B}_+$ is not a bad matrix provided $\mathbf{B}_0$ and $\mathbf{B}_1$ are both good, or both bad. To rule out the latter possibility, Alice flips a coin and asks Bob to either "decommit" to $\mathbf{B}_1$ or to $\mathbf{B}_+$ accordingly. The process is repeated r times to achieve real soundness. Observe that a malicious Alice* can not tell anything from $\boldsymbol{\sigma}$, as an honest Bob *always* sends either $\boldsymbol{\sigma}_1$ or masked $\boldsymbol{\sigma}_0 \oplus \boldsymbol{\sigma}_1$ when he is "decommitting".

Generating r consistent witnesses with W_0 can be done very efficiently[4] using an $\mathcal{OT}^n_{r(m+1)}$ primitive. The details of the protocol are in Fig. 5.

Correctness. If the parties follow the protocol it is not difficult to see that $\pi_{m,n,t}$ outputs exactly the same as $c\mathcal{PAD}_{m,n}$. By the homomorphic property, Alice does not reject on honest inputs. Output correctness is due to the fundamental relation exploited in the IKNP extension.

4.3 Security Analysis

The rest of the section is dedicated to prove that the output of $\pi_{m,n,t}$ and the output of $c\mathcal{PAD}_{m,n}$ are statistically close. The road-map is as follows: we first explain why we can work with partitions of $[n]$, then we state some useful results, and lastly we use them to show indistinguishability.

Taxonomy of Receiver's Input. Here we are after a classification of Bob's matrix $\mathbf{B} = [\mathbf{L}, \mathbf{R}] \in \mathcal{M}_{m\times 2n}$. As an illustration consider an honest situation where Bob gives matrix $\mathbf{B}$ such that it defines n additive sharings of some vector $\boldsymbol{\sigma}$ of his choosing. This means that $\mathbf{l}^j \oplus \mathbf{r}^j = \boldsymbol{\sigma}$ for all indices in $[n]$. Clearly, the relation $j_1 \mathcal{R} j_2$ iff $\mathbf{l}^{j_1} \oplus \mathbf{r}^{j_2} = \boldsymbol{\sigma}$ is the trivial equivalence relation in $[n]$ where all indices are related to each other. In other words, the matrix $[\mathbf{L}, \mathbf{R}]$ defines the trivial partition of $[n]$, i.e. $\mathcal{P} = \{[n]\}$.

Underlying Partition. For any binary matrix $\boldsymbol{\Delta}$ in $\mathcal{M}_{m\times n}$, its underlying relation is the subset $\mathcal{R}_{\boldsymbol{\Delta}} \in [n] \times [n]$ defined as

$$\mathcal{R}_{\boldsymbol{\Delta}} = \{(i,j) \in [n] \times [n] \mid \boldsymbol{\delta}^i = \boldsymbol{\delta}^j\}.$$

As usual, we write $i\mathcal{R}_{\boldsymbol{\Delta}}j$ to mean $(i,j) \in \mathcal{R}_{\boldsymbol{\Delta}}$. It is not difficult to see that $\mathcal{R}_{\boldsymbol{\Delta}}$ is an equivalence relation[5], in particular each $\boldsymbol{\Delta}$ defines a unique partition $\mathcal{P}_{\boldsymbol{\Delta}}$

[4] The cost to pay is increasing the length of the input bit strings to the $\mathcal{OT}$, using a PRG one would only need to obliviously transfer the PRG seed.

[5] The reader can check the relation is reflexive, symmetric and transitive.

of $[n]$. Also, for any partition of $[n]$, we say is ℓ-thick if the size of its maximal parts are ℓ. Now it becomes clear that any (possibly malicious) receiver's input $\mathbf{B} = [\mathbf{L}, \mathbf{R}]$ implicitly defines a partition of $[n]$, given by matrix $\boldsymbol{\Delta} = [\mathbf{l}^1 \oplus \mathbf{r}^1, \ldots, \mathbf{l}^n \oplus \mathbf{r}^n]$. The input is ℓ-thick if its partition is ℓ-thick.

Parametrizing the Thickness. One can take a parametric definition, saying that $\mathcal{P}$ is ℓ-thick if $\ell = \frac{M}{n}$, where M is the size of a maximal part[6]. In the security analysis this notion will prove to be useful. For example, honest inputs have (high) thickness level $\ell = 1$. We will always adopt the parametric perspective.

Witnessing and Thickness. Let $W = (\mathbf{a}, \mathbf{Q})$ be a witness that Alice has. If Bob* used an ℓ-thick $\mathbf{B}$ to give W to Alice, then W is said to be ℓ-thick.

Rejecting Thin Inputs. Now we formalize the intuition that Bob* is caught with high probability if he inputs a matrix with their columns adding up to many distinct vectors.

The first lemma deals with rejections on a particular "proof" handed by Bob. The second lemma upper bounds the thickness of a witness derived from the XOR operation. The proof of the rejection lemma exploits the correctness of the $\mathcal{OT}$ primitive. Both proofs make heavy use of the underlying partition defined in Sect. 4.3. The reader might want to skip them in the first lecture, and go directly to Proposition 1.

Lemma 1. *Let $W = (\mathbf{a}, \mathbf{Q})$ be a witness that is known to* Alice. *Then,* Bob *knows a valid proof for W only if he knows at least $n(1-\ell)$ bits of $\mathbf{a}$, where ℓ is the thickness of W. In particular, if* Alice *is honest this happens with probability $p \leq 2^{-n(1-\ell)}$.*

Proof. Let $\mathbf{B} = [\mathbf{L}, \mathbf{R}]$ be the input of Bob to the $\mathcal{OT}$ from which Alice obtained witness $(\mathbf{a}, \mathbf{Q})$, and let $(\boldsymbol{\sigma}, \tilde{\mathbf{B}})$ be the proof held by Bob. Also, let $\boldsymbol{\Delta} = [\mathbf{l}^1 \oplus \mathbf{r}^1, \ldots, \mathbf{l}^n \oplus \mathbf{r}^n]$, and say that $\boldsymbol{\Delta}$ defines partition $\mathcal{P} = \{P_1, \ldots, P_p\}$ of $[n]$.

If the proof $(\boldsymbol{\sigma}, \tilde{\mathbf{B}})$ is *valid*, then for all $j \in [n]$ we can derive the equations

$$(1)\ \mathbf{q}^j = \mathbf{l}^j \oplus a^j \cdot (\mathbf{l}^j \oplus \mathbf{r}^j)\ ,\ (2)\ \boldsymbol{\delta}^j = \mathbf{l}^j \oplus \mathbf{r}^j,$$
$$(3)\ \mathbf{q}^j = \tilde{\mathbf{l}}^j \oplus a^j \cdot (\tilde{\mathbf{l}}^j \oplus \tilde{\mathbf{r}}^j)\ ,\ (4)\ \boldsymbol{\sigma} = \tilde{\mathbf{l}}^j \oplus \tilde{\mathbf{r}}^j.$$

where (1) and (2) are given by the correctness of the $\mathcal{OT}_m^n$ executed on Bob's input $\mathbf{B} = [\mathbf{L}, \mathbf{R}]$, and (3) and (4) follow from assuming $(\boldsymbol{\sigma}, \tilde{\mathbf{B}})$ is valid. Adding (1) and (3), and plugging (2) and (4) in the result, we write $\mathbf{l}^j \oplus \tilde{\mathbf{l}}^j = a^j \cdot (\boldsymbol{\delta}^j \oplus \boldsymbol{\sigma})$. Assume first there exist $j_0 \in [n]$, such that $\boldsymbol{\sigma} = \boldsymbol{\delta}^{j_0}$. Say wlog. that $j_0 \in P_1$. Now, by definition of $\mathcal{P}$, we have $\boldsymbol{\sigma} = \boldsymbol{\delta}^j$ iff $j \mathcal{R}_{\boldsymbol{\Delta}} j_0$. In other words, for $2 \leq k \leq p$ and $j \in P_k$ we have $\boldsymbol{\sigma} \neq \boldsymbol{\delta}^j$. It follows that there exists $i \in [m]$ such that $\delta_i^j \neq \sigma_i$, and therefore $a^j = l_i^j \oplus \tilde{l}_i^j$. The RHS of the last equation is known to Bob, so is a^j. This is true for all $j \in P_k$, and all $k \geq 2$, therefore Bob knows $|P_2 \vee \ldots \vee P_p| = n - |P_1| \geq n(1-\ell)$ bits of $\mathbf{a}$, where the last inequality follows

[6] Parameter ℓ lies in $[\frac{1}{n}, 1]$.

because $\mathcal{P}$ is ℓ-thick. On the other hand, if $\boldsymbol{\sigma} \neq \boldsymbol{\delta}^j$ for all $j \in [n]$, then Bob knows the entire vector $\mathbf{a}$. Adding up, Bob* knows at least $n(1-\ell)$ bits of $\mathbf{a}$.

Since $\mathbf{a}$ is secured via the $\mathcal{OT}^n_m$, Bob knows such bits by guessing them at random. We conclude that Alice accepts any $\boldsymbol{\sigma}$ with probability $p < 2^{n(1-\ell)}$, provided Alice samples $\mathbf{a}$ at random, which is indeed the case.

Lemma 2. *If $W = (\mathbf{a}, \mathbf{Q})$ is ℓ-thick and $\tilde{W} = (\mathbf{a}, \tilde{\mathbf{Q}})$ is $\tilde{\ell}$-thick , then $W_+ = (\mathbf{a}, \mathbf{Q} \oplus \tilde{\mathbf{Q}})$ is ℓ_+-thick with $\ell_+ \leq 1 - |\ell - \tilde{\ell}|$.*

Proof. Say that $\epsilon = |\ell - \tilde{\ell}|$. and let $[\mathbf{L}, \mathbf{R}]$, $[\tilde{\mathbf{L}}, \tilde{\mathbf{R}}]$ be the Bob's inputs from which Alice obtained witnesses W and $\tilde{W}$. Say that they define partitions $\mathcal{P} = \mathcal{P}_{[\mathbf{L},\mathbf{R}]}$, $\tilde{\mathcal{P}} = \mathcal{P}_{[\tilde{\mathbf{L}},\tilde{\mathbf{R}}]}$. Similarly one defines partition $\mathcal{P}_{\boldsymbol{\Delta} \oplus \tilde{\boldsymbol{\Delta}}}$ for witness $(\mathbf{a}, \mathbf{Q} \oplus \tilde{\mathbf{Q}})$.

First, suppose $\ell \leq \tilde{\ell}$, and let $\tilde{P}_{max}$ a maximal part of $\tilde{\mathcal{P}}$. Consider the refinement $\mathcal{P}^{\cap}_{max} = \tilde{P}_{max} \cap \mathcal{P}$. If j_1, j_2 lie in the same part of $\mathcal{P}^{\cap}_{max}$ then $j_1 \mathcal{R}_{\boldsymbol{\Delta} \oplus \tilde{\boldsymbol{\Delta}}} j_2$ iff $j_1 \mathcal{R}_{\boldsymbol{\Delta}} j_2$. This follows from the fact that if j_1 and j_2 are both in $\tilde{P}_{max}$, then $\tilde{\boldsymbol{\delta}}^{j_1} = \tilde{\boldsymbol{\delta}}^{j_2}$. In particular, each part of $\mathcal{P}^{\cap}_{max}$ lies in a different part of $\mathcal{P}_{\boldsymbol{\Delta} \oplus \tilde{\boldsymbol{\Delta}}}$.

Now, look at the auxiliar partition $\{[n] \backslash \tilde{P}_{max}, \mathcal{P}^{\cap}_{max}\}$. The maximum size we can hope for a part in $P_{\boldsymbol{\Delta} \oplus \tilde{\boldsymbol{\Delta}}}$ occurs when $[n] \backslash \tilde{P}_{max}$ collapses with a single maximal part of $\mathcal{P}^{\cap}_{max}$. Even in this case, the size of a maximal part of $\mathcal{P}_{\boldsymbol{\Delta} \oplus \tilde{\boldsymbol{\Delta}}}$ is upper bounded by

$$n(1 - \tilde{\ell}) + n\ell = n(1 - (\ell + \epsilon) + \ell) = n(1 - \epsilon).$$

This follows from observing that $\tilde{P}_{max}$ is of size $n\tilde{\ell}$, and $\mathcal{P}^{\cap}_{max}$ have parts upper bounded by $n\ell$. The case $\tilde{\ell} \leq \ell$ is analogous (using auxiliar partition $\{[n] \backslash P_{max}, P_{max} \cap \tilde{\mathcal{P}}\}$).

Next, we estimate the acceptance probability of $\pi_{m,n,t}$ on any possible input of Bob. Note that the first witness obtained in the commit phase is the output of $\pi_{m,n,t}$.

Proposition 1. *Let $W = (\mathbf{a}, \mathbf{Q})$ the first witness that* Alice *obtains in the commit phase of $\pi_{m,n,t}$. Then, if W has thickness $\ell \leq t$,* Alice *accepts any adversarial proof with probability $p \leq 2^{-n(1-t)/2+2}$. In that case,* Bob *knows at least $n(1-t)/2$ bits of* $\mathbf{a}$.

Proof. Recall that in the protocol $r = \lceil \frac{1-t}{2} n \rceil$, and let $E = (E_1, \ldots, E_r)$ be the random variable (uniformly distributed over $\mathbb{F}_2^r$) that Alice uses to challenge Bob*. For $i \in [r]$, let $B_i^* = (L_i^*, R_i^*)$ be the adversarial random variables that Bob* uses to sample the r matrices in the commit phase of π. Let $[\mathbf{L}_i, \mathbf{R}_i] = \mathbf{B}_i \leftarrow B_i^*$ the actual matrices. Denote with $\boldsymbol{\Delta}_i$ their correspondent underlying matrices. Each $\boldsymbol{\Delta}_i$ defines a unique partition $\mathcal{P}_i$ of $[n]$, with thickness $\ell_i \in [\frac{1}{n}, 1]$.

We want to upper bound the probability of Alice accepting in $\pi_{m,n,t}$ with $\ell \leq t$. Denote with Accept this event. Consider the r.v. $E^* = (E_1^*, \ldots, \mathsf{E}_r^*)$, given by:

$$E_i^* = \begin{cases} 0 & i \text{ if } \ell_i > t' + \ell \\ 1 & \text{ if } \ell_i \leq t' + \ell \end{cases}$$

where $t' = \frac{1-t}{2}$ is positive if $t \in [\frac{1}{n}, 1)$. We first look at the probability of Alice accepting the ith proof,

$$\begin{aligned} &P[\mathsf{Accept}_i] = \frac{1}{2}(P[\mathsf{Accept}_i \mid E_i \to 0] + P[\mathsf{Accept}_i \mid E_i \to 1]) \\ &\leq \frac{1}{2}(P[\mathsf{Accept}_i \mid E_i \to 0, E_i^* \to 0] + (P[\mathsf{Accept}_i \mid E_i \to 0, E_i^* \to 1] \\ &+ P[\mathsf{Accept}_i \mid E_i \to 1, E_i^* \to 0] + P[\mathsf{Accept}_i \mid E_i \to 1, E_i^* \to 1]) \\ &= p_{0,0} + p_{0,1} + p_{1,0} + p_{1,1}. \end{aligned}$$

Consider the cases:

$(e_i, e_i^*) = (0,1)$. If Alice uses $\tilde{W}_i = W_i$ and $\ell_i \leq t' + \ell$ (i.e. $1 - \ell_i \geq 1 - t' - \ell$), by Lemma 1 we bound $p_{0,1} \leq 2^{-n(1-\ell_i)} \leq 2^{-n(1-t'-\ell)}$.

$(e_i, e_i^*) = (1,0)$. If Alice uses $\tilde{W}_i = W_+ = W_i + W$ and $\ell_i \geq t' + \ell$ (i.e. $\ell_i - \ell \geq t'$, with $t' \geq 0$ is equivalent to $|\ell_i - \ell| \geq t'$), by Lemma 2 we have $\ell_+ \leq 1 - |\ell_i - \ell| \leq 1 - t'$, and Lemma 1 bounds $p_{1,0} \leq 2^{-n(1-\ell_+)} \leq 2^{-n(1-(1-t'))} = 2^{-nt'}$.

Now, observe that by hypothesis $\ell \leq t$, and therefore $\frac{1-t}{2} = t' = min\{1 - t' - \ell, t'\}$. From the above we deduce, (1) if Bob* does not guess Alice's coin E_i with his own coins E_i^* then he has to guess at least $n(1-t)/2$ bits of $\mathbf{a}$, (2) in that case we bound $p_{b,b+1} \leq 2^{-nt'} = 2^{-n(1-t)/2}$.

We have to give up in bounding $p_{0,0}$ and $p_{1,1}$ as Bob* can always choose ℓ_i appropriately to pass the test with high probability (e.g. $\ell_i = 1$, $\ell_i = \ell$ respectively). As observed, in these cases Bob* is guessing Alice's coin e_i with his own coin e_i^*. It is now easy to finish the proof as follows:

Let Guess be the event $\{\mathbf{e} \leftarrow E\} \cap \{\mathbf{e} \leftarrow E^*\}$, is clear that if $\neg$Guess, then exist i_0 s.t. $e_{i_0} \leftarrow E_{i_0}$ and $e_{i_0} \oplus 1 \leftarrow E_{i_0}^*$, we can write

$$\begin{aligned} &P[\mathsf{Accept}] = P[\cap_{i=1}^r \mathsf{Accept}_i] \\ &\leq P[(\cap_i^r \mathsf{Accept}_i) \cap \mathsf{Guess}] + P[(\cap_i^r \mathsf{Accept}_i) \mid \neg\mathsf{Guess}] \\ &\leq P[\mathsf{Guess}] + P[\mathsf{Accept}_{i_0} \mid E_{i_0} \to e_i, E_{i_0}^* \to e_i + 1] \\ &\leq 2^{-r} + 2^{-n\frac{1-t}{2}+1} \end{aligned}$$

Therefore Alice accepts and Bob* knows $n(1-t)/2$ bits of $\mathbf{a}$ with probability at most $2^{-n(1-t)/2+2}$.

Remark on Proposition 1. The above result ensures two things: First, if Bob inputs a matrix whose columns do not add to the same constant value *he is forced to take a guess on some bits of* $\mathbf{a}$. As we saw in Sect. 3.3 this is enough to implement the extended transfers securely. Second, setting the thick parameter t appropriately we can rule out a wide range of adversarial inputs with overwhelming probability in n. For example, the adversarial input $\mathbf{I}_{IKNP} = [\mathbf{L}, \mathbf{R}]$ of the attack in the IKNP extension has all its columns adding up to distinct elements, i.e. its underlying partition is the *thinnest* possible partition of $[n]$, $\mathcal{P}_{IKNP} = \{\{1\}, \ldots, \{n\}\}$. Since $t \geq \frac{1}{n}$, this input is rejected with overwhelming probability.

Simulating a malicious Alice* $\mathcal{S}$ externally sends $(\mathsf{Alice}, \mathsf{corrupt})$ to $c\mathcal{PAD}_{m,n}$. Next, it runs an internal execution of π. In step 1 $\mathcal{S}$ does nothing (acting as π_{B}). In step 2, $\mathcal{S}$ internally gets adversarial $\tilde{\mathbf{a}}$ as input to the inner $\mathcal{OT}$. $\mathcal{S}$ externally sends $(\mathsf{corruptAlice}, sid, \tilde{\mathbf{a}})$ to $c\mathcal{PAD}_{m,n}$, obtaining matrix $\tilde{\mathbf{Q}}_0$. It samples at random r vectors $\boldsymbol{\sigma}_i \in \mathbb{F}_2^m$, and for each it sets n sharings $[\mathbf{L}_i, \mathbf{R}_i]$ (i.e. $\mathbf{l}_i^j \oplus \mathbf{r}_i^j = \boldsymbol{\sigma}_i$ for $j \in [n]$). Let $\mathbf{Q}_i$ a $m \times n$ matrix such that $\mathbf{q}_i^j = \mathbf{l}_i^j \oplus \tilde{a}^j \cdot \boldsymbol{\sigma}_i$, $\mathcal{S}$ internally gives $[\tilde{\mathbf{Q}}_0, \mathbf{Q}_1, \ldots, \mathbf{Q}_r]$ to $\mathcal{A}$ in step 2.
Let $\tilde{\mathbf{e}} \in \mathbb{F}_2^r$ the adversarial challenge that $\mathcal{S}$ internally gets from $\mathcal{A}$ in step 3. If $e_i = 0$, $\mathcal{S}$ prepares proof $(\boldsymbol{\sigma}_i, [\mathbf{L}_i, \mathbf{R}_i])$. If $e_i = 1$, $\mathcal{S}$ prepares proof $(\boldsymbol{\sigma}_{i,+}, [\mathbf{L}_{i,+}, \mathbf{R}_{i,+}])$, where $\boldsymbol{\sigma}_{i,+}$ is sampled at random, and $[\mathbf{L}_{i,+}, \mathbf{R}_{i,+}]$ defines n sharings of $\boldsymbol{\sigma}_{i,+}$. Then $\mathcal{S}$ internally sends the r proofs to $\mathcal{A}$ in step 4. If π_{A} aborts in step 5, $\mathcal{S}$ externally sends to $c\mathcal{PAD}_{m,n}$ message $(\mathsf{corruptAlice}, sid, \perp)$. Lastly, $\mathcal{S}$ outputs whatever π_{A} outputs and halts.

Simulating a malicious Bob* $\mathcal{S}$ externally sends $(\mathsf{Bob}, \mathsf{corrupt})$ to $c\mathcal{PAD}_{m,n}$ and as a response obtains input $\mathbf{B}_{\mathcal{E}} = [\mathbf{L}_{\mathcal{E}}, \mathbf{R}_{\mathcal{E}}]$. It then sets π_{B}'s input to $\mathbf{B}_{\mathcal{E}}$, and runs an internal execution of π up to step 5 (π_{B} is controlled by $\mathcal{A}$). In step 2, $\mathcal{A}$ specifies an $m(r+1) \times 2n$ matrix $[[\mathbf{L}_0, \ldots \mathbf{L}_r], [\mathbf{R}_0, \ldots \mathbf{R}_r]]$ as input to $\mathcal{OT}^n_{m(r+1)}$, and in step 4 $\mathcal{A}$ specifies r proofs $(\tilde{\boldsymbol{\sigma}}_i, [\tilde{\mathbf{L}}_i, \tilde{\mathbf{R}}_i])_{i \in [r]}$.
Next, $\mathcal{S}$ runs step 5 of its internal copy of $\pi_{m,n,t}$, it sets flag Rabort to true iff it resulted in abort, but it does not tell $\mathcal{A}$ whether or not she passed. If Rabort is true $\mathcal{S}$ externally sends $(\mathsf{corruptBob}, sid, \perp)$ to $c\mathcal{PAD}_{m,n}$, outputs what π_{B} outputs and halts. Otherwise, it computes the $r+1$ associated $m \times n$ matrices $\boldsymbol{\Delta}_i$ of the (adversarial) input [a] given to $\mathcal{OT}^n_{m(r+1)}$. For each $i \in [r]$, $\mathcal{S}$ finds the indices $j \in [n]$ such that $\tilde{\boldsymbol{\sigma}}_i \neq \boldsymbol{\delta}_i^j \oplus e_i \cdot \boldsymbol{\delta}_0^j$ (if any). Denote this subset of $[n]$ as G. Now, for those $j \in G$, $\mathcal{S}$ finds the first $k \in [m]$ such that $\tilde{\sigma}_{i,k} \neq \delta^j_{i,k}$, then it sets $\tilde{a}^j = l^j_{i,k} \oplus \tilde{l}^j_{i,k}$. Lastly, if G is empty, $\mathcal{S}$ externally sends $(\mathsf{deliver}, sid)$ to $c\mathcal{PAD}_{m,n}$. Otherwise it sends $(\mathsf{corruptBob}, sid, [\mathbf{L}_0, \mathbf{R}_0], \tilde{\mathbf{a}}, G)$ to $c\mathcal{PAD}_{m,n}$. $\mathcal{S}$ tells to abort to $\mathcal{A}$ iff $c\mathcal{PAD}_{m,n}$ says so, outputs what π_{B} outputs and halts.

Simulating an honest execution $\mathcal{S}$ gets (sid) from $c\mathcal{PAD}_{m,n}$, runs an internal execution of π and halts.

[a] Recall how they are defined, i.e. $\boldsymbol{\Delta}_i$ has columns $\boldsymbol{\delta}_i^j = \mathbf{l}_i^j \oplus \mathbf{r}_i^j$ for $i \in [r] \cup \{0\}$, $j \in [n]$.

Fig. 6. The ideal adversary for $c\mathcal{PAD}$

Putting the Pieces in the UC Framework. We have not yet captured the notion of having blocks of $\mathbf{a}$ randomly distributed in Bob's view, it is resolved with a simulation argument. More concretely, we show a reduction to $\mathcal{OT}^n_{m(r+1)}$ with *perfect* security against Alice*, and *statistical* security against Bob*.

Theorem 1. *In the $\mathcal{OT}^n_{m(r+1)}$-hybrid, in the presence of* static active adversaries*, the output of protocol $\pi_{m,n,t}$ and the output of the ideal process involving $c\mathcal{PAD}_{m,n}$ are $2^{-n(1-t)/2+2}$ close.*

Proof. Let $\mathcal{E}$ denote the environment, and $\mathcal{S}$ be the ideal world adversary. $\mathcal{S}$ starts invoking an internal copy of $\mathcal{A}$ and setting dummy parties π_{A} and π_{B}. It then runs an internal execution of π between $\mathcal{A}$, π_{A}, π_{B}, where every incoming communication from $\mathcal{E}$ is forwarded to $\mathcal{A}$ as if it were coming from $\mathcal{A}$'s environment, and any outgoing communication from $\mathcal{A}$ is forwarded to $\mathcal{E}$. The description of $\mathcal{S}$ is in Fig. 6.

We now argue indistinguishability. Let Dist be the event of having $\mathcal{E}$ distinguishing between the ideal and real process. We bound the probability of Dist occurring conditioned on corrupting at most one of the parties.

Perfect security for Bob *($\mathsf{EXEC}_{\pi,\mathcal{E},\mathcal{A}} \equiv \mathsf{EXEC}_{\phi,\mathcal{E},\mathcal{S}}$).* If Alice is malicious, then what $\mathcal{E}$ gets to see from Bob's ideal transcript is $(\mathbf{B}, [\tilde{\mathbf{Q}}_0, \mathbf{Q}_1 \ldots, \mathbf{Q}_r]), (\tilde{\boldsymbol{\sigma}}_i,$

$[\tilde{\mathbf{L}}_i, \tilde{\mathbf{R}}_i])_{i\in[r]})$, where $\mathbf{B} = [\mathbf{L}, \mathbf{R}]$ is the input, i.e. n sharings of, say, $\boldsymbol{\sigma}_0$. Matrix $\tilde{\mathbf{Q}}_0$ is consistent with $\mathbf{B}$ and with adversarial choice $\tilde{\mathbf{a}}$ (see Fig. 1), hence by definition of $\mathcal{S}$ and the robustness of $\mathcal{OT}^n_{m(r+1)}$, matrix $[\tilde{\mathbf{Q}}_0, \mathbf{Q}_1, \ldots, \mathbf{Q}_r]$ is exactly distributed as in the real process. Furthermore, if $\tilde{e}_i = 1$, then $\tilde{\boldsymbol{\sigma}}_i = \boldsymbol{\sigma}_{i,+}$ is randomly sampled, whereas in the real process $\tilde{\boldsymbol{\sigma}}_i = \boldsymbol{\sigma}_0 \oplus \boldsymbol{\sigma}_i$, with $\boldsymbol{\sigma}_i$ being in the *private* part of Bob's transcript. Therefore the proofs of the ideal and real process are identically distributed. We conclude that real and ideal transcripts are identically distributed, and therefore $Pr[\mathsf{Dist}|\mathsf{corruptAlice}] = 0$.

Statistical security for Alice ($\mathsf{EXEC}_{\pi,\mathcal{E},\mathcal{A}} \overset{s}{\approx} \mathsf{EXEC}_{\phi,\mathcal{E},\mathcal{S}}$). For the case Bob is corrupted, we first note that up to step 5, both processes are *identically* distributed because $\mathcal{S}$ runs an internal copy of $\pi_{m,n,t}$ using input $[\mathbf{L}_{\mathcal{E}}, \mathbf{R}_{\mathcal{E}}]$ specified by $\mathcal{E}$. Next, say $[\mathbf{L}_0, \mathbf{R}_0]$ is ℓ-thick. Then, if $\ell \leq t$, by Proposition 1, the size of G is at least $n(1-t)/2$ with overwhelming probability (in n), thus $c\mathcal{PAD}_{m,n}$ does not abort with probability $p \leq 2^{-n(1-t)/2}$. By Proposition 1 again the ideal and the real processes abort on thin inputs except with probability $p \leq 2^{-n(1-t)/2+2}$ (i.e. we do not care if $\mathcal{E}$ distinguishes in this case). On the other hand, if $\ell > t$ and the internal copy of $\pi_{m,n,t}$ did not abort (if aborts, so does $c\mathcal{PAD}_{m,n}$ by definition of $\mathcal{S}$), then we claim that the output of both processes are identically distributed. This follows from (1) the output matrix $[\mathbf{L}_0, \mathbf{R}_0]$ is extracted by $\mathcal{S}$, and looking closely at the proof of Lemma 1, we deduce (2) if $j \in G$, then for some $i \in [r]$, Bob* "decommits" to $\tilde{\boldsymbol{\sigma}}_i \neq \boldsymbol{\delta}_i^j \oplus e_i \cdot \boldsymbol{\delta}_0^j$, the real bit a^j is exactly as the one extracted by $\mathcal{S}$; (3) if $j \notin G$, then j is such that for each $i \in [r]$, Bob* is decommitting to $\tilde{\boldsymbol{\sigma}}_i = \boldsymbol{\delta}_i^j \oplus e_i \cdot \boldsymbol{\delta}_0^j$. In this case, the system of equations given in the proof of Lemma 1 collapses to $\mathbf{l}^j = \tilde{\mathbf{l}}^j$; $\mathbf{r}^j = \tilde{\mathbf{r}}^j$. One sees that if $\mathcal{E}$ could tell anything from $\mathbf{a}_{|[n]\setminus G}$, he could equally tell the same *before* the prove phase, contradicting the security of the underlying $\mathcal{OT}^n_{m(r+1)}$.

We have argued $Pr[\mathsf{Dist}|\mathsf{corruptBob}] \leq 2^{-n(1-t)/2+2}$.

Completeness. For the case none of the parties are corrupted, indistinguishability follows from the security of the underlying $\mathcal{OT}^n_{m(r+1)}$.

Adding up, $\mathcal{E}$ distinguishes with probability $Pr[\mathsf{Dist}] \leq 2^{-n(1-t)/2+2}$. This concludes the proof.

4.4 Another Look at the Outer Reduction

Here we take a different perspective for the IKNP reduction that fits better with our partition point of view (as defined in Sect. 4.3). We aim to give some intuition of the underlying ideas, and the reader should by no means take the following discussion as formal arguments.

For an illustrative example let us first look at the attack of the protocol in the IKNP extension. A malicious Bob* was giving input matrix $\mathbf{B}$ with all the columns adding up to distinct elements. Consequently its underlying partition is $\mathcal{P}_{IKNP} = \{\{1\}, \ldots, \{n\}\}$. This structure on $\mathbf{B}$ is such that all but one of the bits of both pads are known to Bob*. One can see this as splitting the query space $\mathbb{F}_2^n$ as n copies of $\mathbb{F}_2$, namely $Q = \bigoplus_{i=1}^n \mathbb{F}_2$. To search for the secret vector $\mathbf{a}$, one just have to brute-force each summand separately and use the CRF to compare. After $n \cdot |\mathbb{F}_2| = 2n$ calls the query space is exhausted, i.e. even computationally bounded environments would distinguish between the ideal and the real process.

We want to assign to each possible matrix input $\mathbf{B} = [\mathbf{L}, \mathbf{R}]$ a unique structure of the query space that the environment is forced to use towards distinguishing. In other words, we want to establish a correspondence between the partition implicitly defined in $\mathbf{B}$, and the possible ways to split the query space $Q = \mathbb{F}_2^n$.

Let $\mathcal{P}$ be any partition of $[n]$, express it as $\mathcal{P} = \{P_{1,1}, \ldots, P_{q_1,1}, \ldots, P_{1,n}, \ldots, P_{q_n,n}\}$ where for $i \in [n]$, $j \in [q_i]$, part $P_{j,i}$ is either empty or is of size i (i.e. there are q_i parts of size i in $\mathcal{P}$). The *type* of $\mathcal{P}$ is the vector $\mathbf{q} = (q_1, \ldots, q_n) \in \{[n] \cup \{0\}\}^n$. The $\mathbf{q}$-type query space, is the vectorial space $Q_{\mathbf{q}} = \bigoplus_{i=1}^n Q_{\mathbf{q},i}$, where $Q_{\mathbf{q},i}$ is the ith *block of* $Q_{\mathbf{q}}$, and stands for q_i copies of an $\mathbb{F}_2$-vectorial space of dimension i.

Thus, the type of $\mathcal{P}_{IKNP}$ corresponds to vector $\mathbf{q} = n \cdot \mathbf{e}_1$, and the query space the environment was using to brute-force Alice's secret $\mathbf{a}$ is precisely $Q_{n \cdot \mathbf{e}_1}$. On the other hand, honest inputs always define the trivial partition $\mathcal{P}_H = \{[n]\}$ with type $\mathbf{q} = \mathbf{e}_n$, the reduction against a semi-honest receiver in [IKNP03], based security arguing that the environment would have to brute-force $\mathbb{F}_2^n$, which is the query space $Q_{\mathbf{e}_n}$.

Now, the map $f : \mathbf{B} \mapsto Q_{\mathbf{q}}$, where $\mathcal{P}_{\mathbf{B}}$ is $\mathbf{q}$-type, is well defined. To see this, just observe that the relation in $\mathcal{P}^{[n]}$ defined as $\mathcal{P} \sim \mathcal{P}'$ iff "both partitions are of same type" is an equivalence relation, and look at the chain

$$\begin{aligned} \mathcal{M}_{m \times n} &\xrightarrow{g_1} \mathcal{P}^{[n]} \xrightarrow{g_2} (\mathcal{P}^{[n]} / \sim) \xrightarrow{g_3} \mathcal{V} \\ \boldsymbol{\Delta} &\mapsto \mathcal{P}_{\boldsymbol{\Delta}} \mapsto [\mathcal{P}_{\boldsymbol{\Delta}}]_{\sim} = \mathbf{q} \mapsto Q_{\mathbf{q}} \end{aligned}$$

We see that $f = g_3 \circ g_2 \circ g_1$ is well defined.

From this one can imagine how the reduction would work. $c\mathcal{PAD}$ could check the thickness of the adversarial $\mathbf{B}$, and reject if is less than a fixed parameter t. This ensures that the structure of the query space contains at least one block of size big enough, wasting the chances of the environment to search through it in reasonable time. Unfortunately, with this reduction the composition of the inner and outer protocols renders worst choices of parameters.

5 Concluding the Construction

In this section we prove the main result of the paper. For a given security parameter n recall that t is a parameter lying in interval $[\frac{1}{n}, 1)$, and $r = \lceil \frac{1-t}{2} n \rceil$. Observe that the results of Sect. 4 break down for $t = 1$. This corresponds to a superfluous $\pi_{m,n,1}$ (no checks at all). In other words, a malicious Bob* can input *any* possible bad-formed matrix $\mathbf{B}$ to the IKNP extension, in which case there is no security.

Corollary 1. *In the* $\mathcal{OT}^n_{m(r+1)}$*-hybrid, for any* $t \in [\frac{1}{n}, 1)$ *protocol* $\rho^{\pi_{m,n,t}/c\mathcal{PAD}_{m,n}}$ UC-realizes $\mathcal{OT}^m_n$ *in the presence of* static active adversaries*, provided the environment is given access to at most* $2^{o(n)}$ *queries to a* CRF*.*

Proof. The result follows applying the Composition Theorem of [Can01]. By Claim the error simulation for ρ is $e_\rho = 2^{-n/2+o(n)+2}$, and by Theorem 1 the error simulation for $\pi_{m,n,t}$ is $e_\pi = 2^{-n(1-t)/2+2}$. Using that $(1-t)/2 < 1/2$ if $t > 0$, and the transitivity of the composition operation, the error simulation for $\rho^{\pi_{m,n,t}/c\mathcal{PAD}_{m,n}}$ is $e = e_\rho + e_\pi \leq 2^{-n(1-t)/2+o(n)+3}$.

5.1 Complexity and Choice of Parameters

For the computational overhead, we emphasize that a cryptographic primitive is still needed to implement the actual extended transfers (we are using the IKNP extension). To implement $m = poly(n)$ transfers, in the test Alice and Bob have to XOR $rm(2n+1)$ bits. Thus, per extended OT each participant needs to XOR $O((1-t)n^2)$ bits. The communication complexity (number of bits transferred per OT) turns out to be equivalent. The test adds a *constant* number of rounds to the overall construction, concretely 2 extra rounds.

In terms of the seed expansion we can do it better. For a security level of s bits in the reduction, one need roughly $n \approx \frac{2}{1-t}(s + o(n) + 3)$ OT seeds. One can measure the quality of the reduction looking at the seed expansion factor $exp(t) = \frac{2}{1-t}$. It is clear that $exp(t)$ tends to 2, when $t \to \frac{1}{n}$ and $n \to \infty$. One only need to halve the security parameter of the IKNP reduction (asymptotically).

Practical choice of parameters are also very efficient. For example, to implement about $1,000,000$ transfers, with security of $s = 64$ bits, setting $t = \frac{1}{16}$, one needs roughly $n \approx 186$ OT seeds. For security level $s = 128$, one would need roughly 323 OT seeds.

5.2 Open Problems

In the reductions for ρ and π the security parameter suffers an expansion factor of 2. We ask whether one can remove this overhead whilst still maintaining security against computational unbounded receivers in the inner protocol.

In the area of secure function evaluation, recently OT has been used to boost the efficiency of two-party protocols [NNOB12] and their counterparts in the multiparty case [LOS14]. A key part on the design of such protocols was the generation of authenticated bits, which in turn borrows techniques from the IKNP extension. It would be interesting to see whether (a suitable modification of) our protocol π can be used to generate such authenticated bits. This would immediately give unconditional security (currently both constructions need a random oracle), in terms of efficiency we do not know if this replacement would bring any improvement at all.

Acknowledgments. This work has been supported in part by EPSRC via grant EP/I03126X.

References

BCR86a. Brassard, G., Crépeau, C., Robert, J.M.: All-or-nothing disclosure of secrets. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 234–238. Springer, Heidelberg (1987)

BCR86b. Brassard, G., Crépeau, C., Robert, J.-M.: Information theoretic reductions among disclosure problems. In: FOCS, pp. 168–173 (1986)

Bea96. Beaver, D.: Correlated pseudorandomness and the complexity of private computations. In: STOC, pp. 479–488 (1996)

Can01. Canetti, R.: Universally composable security: a new paradigm for cryptographic protocols. In: FOCS, pp. 136–145 (2001)

CK88. Crépeau, C., Kilian, J.: Weakening security assumptions and oblivious transfer. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 2–7. Springer, Heidelberg (1990)

CLOS02. Canetti, R., Lindell, Y., Ostrovsky, R., Sahai, A.: Universally composable two-party and multi-party secure computation. In: STOC, pp. 494–503 (2002)

Cré87. Crépeau, C.: Equivalence between two flavours of oblivious transfers. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 350–354. Springer, Heidelberg (1988)

Cré89. Crépeau, C.: Verifiable disclose for secrets and applications. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 150–154. Springer, Heidelberg (1990)

EGL85. Even, S., Goldreich, O., Lempel, A.: A randomized protocol for signing contracts. Commun. ACM **28**(6), 637–647 (1985)

GMW86. Goldreich, O., Micali, S., Wigderson, A.: Proofs that yield nothing but their validity and a methodology of cryptographic protocol design (extended abstract). In: FOCS, pp. 174–187 (1986)

GV87. Goldreich, O., Vainish, R.: How to solve any protocol problem - an efficiency improvement. In: CRYPTO, pp. 73–86 (1987)

HIKN08. Harnik, D., Ishai, Y., Kushilevitz, E., Nielsen, J.B.: OT-combiners via secure computation. In: Canetti, R. (ed.) TCC 2008. LNCS, vol. 4948, pp. 393–411. Springer, Heidelberg (2008)

IKNP03. Ishai, Y., Kilian, J., Nissim, K., Petrank, E.: Extending oblivious transfers efficiently. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 145–161. Springer, Heidelberg (2003)

IPS08. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding cryptography on oblivious transfer – efficiently. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 572–591. Springer, Heidelberg (2008)

IR89. Impagliazzo, R., Rudich, S.: Limits on the provable consequences of one-way permutations. In: STOC, pp. 44–61 (1989)

Kil88. Kilian, J.: Founding cryptography on oblivious transfer. In: STOC, pp. 20–31 (1988)

LOS14. Larraia, E., Orsini, E., Smart, N.P.: Dishonest majority multi-party computation for binary circuits. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part II. LNCS, vol. 8617, pp. 495–512. Springer, Heidelberg (2014)

LZ13. Lindell, Y., Zarosim, H.: On the feasibility of extending oblivious transfer. In: Sahai, A. (ed.) TCC 2013. LNCS, vol. 7785, pp. 519–538. Springer, Heidelberg (2013)

Nie07. Nielsen, J.B.: Extending oblivious transfers efficiently - how to get robustness almost for free. IACR Cryptology ePrint Arch. **2007**, 215 (2007)

NNOB12. Nielsen, J.B., Nordholt, P.S., Orlandi, C., Burra, S.S.: A new approach to practical active-secure two-party computation. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 681–700. Springer, Heidelberg (2012)

PVW08. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 554–571. Springer, Heidelberg (2008)

Rab81. Rabin, M.O.: How to exchange secrets with oblivious transfer. IACR Cryptology ePrint Arch. 187 (1981)

Wie83. Wiesner, S.: Conjugate coding. SIGACT News **15**, 78–88 (1983)

Yao82. Yao, A.C.-C.; Protocols for secure computations (extended abstract). In: FOCS, pp. 160–164 (1982)