Simple Schemes in the Bounded Storage Model

Jiaxin Guan and Mark Zhandary

Princeton University, Princeton NJ 08544, USA

Abstract. The bounded storage model promises unconditional security proofs against computationally unbounded adversaries, so long as the adversary's space is bounded. In this work, we develop simple new constructions of two-party key agreement, bit commitment, and oblivious transfer in this model. In addition to simplicity, our constructions have several advantages over prior work, including an improved number of rounds and enhanced correctness. Our schemes are based on Raz's lower bound for learning parities.

1 Introduction

For the vast majority of cryptographic applications, security relies on the assumed hardness of certain computational problems, such as factoring large integers or inverting certain hash functions. Unfortunately, with the current state of complexity theory the hardness of these problems can only be conjectured. This means that the security of such schemes is always *conditional* on such conjectures being true.

Maurer proposes the Bounded Storage Model [Mau92] as an alternate model for constraining the adversary; here, instead of constraining the adversary's time, the adversary's memory is bounded. Amazingly, it is actually possible to give *unconditional* proofs of security for schemes in this model. The core idea is that the honest parties exchange so much information that the adversary cannot possibly store it all. Then, schemes are cleverly devised to exploit the adversary's lack of knowledge about the scheme.

Moreover, the space bounds are only necessary when the protocol is run, and even if the adversary later gains more space the protocol remains secure. This means schemes only need to be designed with current storage capacities in mind. This is fundamentally different than the usual approach of time-bounding adversaries, where an adversary can later break the protocol if its computational abilities increase. Hence, traditional schemes must be designed with future computational abilities in mind. This is especially important in light of recent developments in quantum computing, as Grover's algorithm [Gro96] and Shor's algorithm [Sho94] can speed up attacks on many current cryptographic protocols. Hence, much of the communication taking place today will be revealed once quantum computers become reality.

This Work. In this work, we devise very simple round-optimal protocols for bit-commitment and oblivious transfer (namely, 1 round and 2 rounds, respectively)

in the Bounded Storage Model, improving 5 rounds needed in prior works. We additionally develop a new key agreement protocol with several advantages over prior works. Our results rely on Raz's recent space lower bound for learning parities [Raz17], and in particular the simple encryption scheme based on this lower bound. Our key observation is that Raz's encryption scheme has several useful properties — including additive homomorphism and leakage resilience — that can be useful for building higher-level protocols. Our core technical contribution is a new "encrypt zero" protocol for Raz's encryption scheme, which may be of independent interest.

Our schemes are based on entirely different techniques than most of the prior literature — most of which is based on the birthday paradox — and we believe our work will therefore be a useful starting point for future work in the bounded storage model.

1.1 Prior Work in the Bounded Storage Model

Prior work in the Bounded Storage Model [Mau92, CM97, CCM98, Lu02, AR99, Din01, DHRS04] typically uses something akin to the birthday paradox to achieve security against space-bounded adversaries.

In slightly more detail, the key agreement scheme of Maurer [Mau92] works as follows. One party sends a stream of roughly n^2 random bits to the other party [1]. Each party records a random secret subset of n bits of the stream. By the birthday paradox, the two parties will have recorded one bit position in common with constant probability. They therefore share the bit positions they recorded with each other, and set their secret key to be the bit of the stream at the shared position.

An eavesdropper first sees n^2 random bits. If the eavesdropper's storage is somewhat lower than n^2, he cannot possibly remember the entire sequence of random bits. In particular, it can be shown that the adversary has little information about the bit shared by the two honest parties. This remains true even after the parties share their bit positions. Notice that the honest parties require space n, and security holds even for adversaries with space Cn^2 for some constant C. Therefore, by tuning n so that n storage is feasible, but Cn^2 is not, one obtains the desired security.

Much of the literature on the Bounded Storage Model relies on this sort of birthday attack property. Unfortunately, this leads to several difficulties:

- The two honest parties only achieve success with constant probability. In order to achieve success with high probability, the protocol either needs to be repeated many times (thus requiring more than n^2 communication) or requires the honest users to store more than n positions (thus requiring more than n space, and making the gap between the honest users and adversaries less than quadratic).

[1] In most works in the Bounded Storage Model, the random bit stream is assumed to come from a trusted third party. In this work we will insist on there being no trusted third party, and instead the bit stream comes from the parties themselves.

– Remembering n random positions out of n^2 requires $O(n \log n)$ space just to record the indices. To compress the space requirements of the honest parties, the positions are actually chosen by a pairwise independent function, complicating the scheme slightly.
– The adversary has a $1/n^2$ chance of guessing the bit position shared by the two users. As such, the adversary has a non-negligible advantage in guessing the bit. To get statistical security, a randomness extraction step is applied, adding slightly to the complexity of the protocol.
– More importantly, there is very little structure to exploit with the birthday approach. For more advanced applications such as oblivious transfer or bit commitment, the protocols end up being somewhat complicated and require several rounds.

1.2 Space Lower Bounds For Learning Parities

In this work, we exploit recent space lower bounds due to Raz [Raz17]. Raz considers a setting where one party holds a secret key $\mathbf{k} \in \{0,1\}^n$, and streams random tuples $(\mathbf{r}_i, \mathbf{r}_i \cdot \mathbf{k})$, where $\mathbf{r}_i$ is random in $\{0,1\}^n$ and the inner product is taken mod 2. Raz asks: given these random tuples, and only limited storage (namely Cn^2 for some constant C), how hard is it to recover $\mathbf{k}$? Clearly, if $C \approx 1$, then one can store n tuples, and then recover $\mathbf{k}$ using linear algebra. But if $C \ll 1$, then the adversary has no hope of storing enough tuples to perform linear algebra.

Raz proves that, for some constant C (roughly $1/20$), then either the adversary needs an exponential (in n) number of samples, or the adversary's probability of correctly guessing $\mathbf{k}$ is exponentially small.

Raz observes that his lower bound easily leads to a secret key encryption scheme in the bounded storage model. The key will be an n-bit string $\mathbf{k}$. To encrypt a message bit b, choose a random $\mathbf{r}$, and produce the ciphertext $(\mathbf{r}, \mathbf{r} \cdot \mathbf{k} \oplus b)$. Raz's lower bound shows that after seeing fewer than exponentially many encrypted messages, an adversary with Cn^2 space has an exponentially small probability of guessing $\mathbf{k}$. This means $\mathbf{k}$ always has some min-entropy conditioned on the adversaries' view. Then using the fact that the inner product is a good extractor, we have that for any new ciphertext $\mathbf{r} \cdot \mathbf{k}$ is statistically close to random, and hence masks the message b.

1.3 This Work

In this work, we use Raz's scheme in order to develop simple new constructions in the Bounded Storage Model that have several advantages over prior work.

Our main observation is that Raz's encryption scheme has several attractive properties. First, it is leakage resilient: since inner products are strong extractors, the scheme remains secure even if the adversary has partial knowledge of the key, as long as the conditional min-entropy of the key is large.

Next, we note that Raz's scheme is additively homomorphic: given encryptions $(\mathbf{r_0}, \mathbf{r_0} \cdot \mathbf{k} \oplus m_0)$ and $(\mathbf{r_1}, \mathbf{r_1} \cdot \mathbf{k} \oplus m_1)$ of m_0, m_1, we can compute an encryption

of $m_0 \oplus m_1$ by simply taking the componentwise XOR of the two ciphertexts, yielding $(\mathbf{r_0} \oplus \mathbf{r_1}, (\mathbf{r_0} \oplus \mathbf{r_1}) \cdot \mathbf{k} \oplus (m_0 \oplus m_1))$. This additive homomorphism will prove very useful. We can also toggle the bit being encrypted by toggling the last bit of a ciphertext.

For example, Rothblum [Rot11] shows that any additively homomorphic secret key encryption scheme can be converted into a public key (additively homomorphic) encryption scheme. The rough idea is that the public key consists of many encryptions of zero. Then, to devise an encryption of a bit m, simply add a random subset sum of the public key ciphertexts to get a "fresh" encryption of zero, and then toggle the encrypted bit as necessary to achieve an encryption of m.

Key Agreement. In the case of Raz's scheme, the public key will end up containing $O(n)$ ciphertexts, meaning the public key is too large for the honest users to even write down. However, we can re-interpret this protocol as a *key-agreement* protocol. Here, the public key is streamed from user A to user B, who applies the additive homomorphism to construct the fresh encryption on the fly. Now one party knows the secret key, and the other has a fresh ciphertext with a known plaintext. So the second party just sends the ciphertext back to the first party, who decrypts. The shared key is the plaintext value.

Bit Commitment. Next, we observe that the public key encryption scheme obtained above is *committing*: for any public key there is a unique secret key. Therefore, we can use the scheme to get a bit commitment scheme as follows: to commit to a bit b, the Committer simply chooses a random secret key, streams the public key to the receiver, and then sends an encryption of b. To open the commitment, the Committer simply sends the secret decryption key. The Verifier, on the other hand, constructs several fresh encryptions of 0 by reading the Committer's stream, as user B did in our key agreement protocol. Upon receiving a supposed secret key, the Verifier checks that all the encryptions do in fact decrypt to 0. If so, then it decrypts the commitment to get the committed value.

Oblivious Transfer. We can also turn this commitment scheme into an oblivious transfer protocol: the Receiver, on input b, commits to the bit b. Then the Sender, on input x_0, x_1, using the homomorphic properties of the encryption scheme, turns the encryption of b in the commitment into encryptions of $(1-b)x_0$ and bx_1. To maintain privacy of x_{1-b}, the Sender will *re-randomize* the encryptions, again using the homomorphic properties. To re-randomize, the Sender will construct some fresh encryptions of zero, again just as user B did in our key agreement protocol. The Receiver can then decrypt these ciphertexts, which yield 0 and x_b.

Malicious Security. The commitment scheme and the oblivious transfer protocol are secure as long as the public key is generated correctly. This occurs, for example, if the randomness for the encryptions of 0 is generated and streamed by a trusted third party. This is the setting considered in much of the prior work in the bounded storage model.

On the other hand, if we do not wish to rely on a trusted third party to generate the encryption randomness, a malicious Committer can choose a public key with bad randomness, which will allow him to break the commitment, as explained below. This also would let the Receiver break the security of the oblivious transfer protocol. We therefore additionally show how to modify the constructions above to obtain security for malicious parties without relying on a trusted third party. The result is round-optimal protocols for bit-commitment and oblivious transfer without a trusted third party.

1.4 Additional Technical Details

The Encrypt Zero Protocol. Notice that all of our schemes have a common feature: one user has a secret key, and the other user obtains encryptions of 0. Importantly for security, these encryptions of 0 should be independent of the view of the first user.

In order to unify our schemes, we abstract the common features required with an *Encrypt Zero* protocol for Raz's encryption scheme. The goal of the protocol is to give one party, the Keeper, a random key $\mathbf{s}$, and another party, the Recorder, λ random encryptions $\{c_1, \ldots, c_\lambda\}$ of 0. Here, λ is a parameter that will be chosen based on application. Recorder security dictates that the Keeper learns nothing about the λ encryptions stored by the Recorder (aside from the fact that they encrypt 0). Keeper security requires that the min-entropy of the key $\mathbf{s}$ conditioned on the Recorder's view is $\Omega(n)$. We additionally require that the Keeper's space is $O(n)$ (which is optimal since the Keeper must store a secret key of $O(n)$ bits), and the Recorder's space is $O(\lambda n)$ (which is also optimal, since the Recorder must store λ encryptions of $O(n)$ bits each).

Our basic protocol for Raz's scheme works as follows:

- The Keeper chooses a random key $\mathbf{k} \in \{0,1\}^n$. Let $m = O(n)$ be a parameter. The Recorder chooses a secret matrix $\boldsymbol{\Sigma} \in \{0,1\}^{\lambda \times m}$.
- The Keeper streams m encryptions $(\mathbf{r}_i, a_i = \mathbf{r}_i \cdot \mathbf{k} + 0)$ to the Recorder, for random $\mathbf{r}_i \in \{0,1\}^n$ and $i = 1, 2, \ldots, m$. From now on, we use the convention that "+" and "$\cdot$" are carried out mod 2.
- The Recorder maintains matrix $\boldsymbol{\Psi} \in \{0,1\}^{\lambda \times n}$ and column vector $\boldsymbol{\kappa} \in \{0,1\}^\lambda$. Each row of $(\boldsymbol{\Psi}|\boldsymbol{\kappa})$ will be a random subset-sum of the encryptions sent by the Keeper, with each subset-sum chosen according to $\boldsymbol{\Sigma}$. The matrices will be computed on the fly. So when $(\mathbf{r}_i, a_i)$ comes in, the Recorder will map $\boldsymbol{\Psi} \to \boldsymbol{\Psi} + \boldsymbol{\sigma}_i \cdot \mathbf{r}_i$, $\boldsymbol{\kappa} \to \boldsymbol{\kappa} + \boldsymbol{\sigma}_i a_i$. Here, $\boldsymbol{\sigma}_i$ is the i-th column of $\boldsymbol{\Sigma}$, and $\mathbf{r}_i$ is interpreted as a row vector.
- At the end of the protocol, the Keeper outputs its key $\mathbf{s} = \mathbf{k}$, and the Recorder outputs $(\boldsymbol{\Psi}|\boldsymbol{\kappa})$, whose rows are the ciphertexts $c_1, \ldots, c_\lambda$.

Let $\mathbf{R}$ be the matrix whose rows are the $\mathbf{r}_i$'s, and let $\mathbf{a}$ be the column vector of the a_i's. Then we have that $\mathbf{a} = \mathbf{R} \cdot \mathbf{k}$, $\boldsymbol{\Psi} = \boldsymbol{\Sigma} \cdot \mathbf{R}$, and $\boldsymbol{\kappa} = \boldsymbol{\Sigma} \cdot \mathbf{a} = \boldsymbol{\Psi} \cdot \mathbf{k}$. Hence, the rows of $(\boldsymbol{\Psi}|\boldsymbol{\kappa})$ are encryptions of zero, as desired.

For Keeper security, Raz's theorem directly shows that $\mathbf{k}$ has min-entropy relative to the Recorder's view. For Recorder security, notice that $\boldsymbol{\Sigma}$ is independent of the the Keeper's view. Therefore, if the Keeper follows the protocol and m is slightly larger than n so that $\mathbf{R}$ is full rank with high probability, then $\boldsymbol{\Psi}$ is a random matrix independent of the adversary's view. Therefore the ciphertexts c_i are actually *random* encryptions of 0. Thus we get security for honest-but-curious Keepers.

Key Agreement. This protocol gives a simple key-agreement scheme. Basically, one party acts as the Keeper, and one as the Recorder. We set $\lambda = 1$. The result of the Encrypt Zero protocol is that the Recorder contains a uniformly random encryption of 0. The Recorder simply flips the bit encrypted with probability $1/2$ to get a random encryption of a random bit b, and sends the resulting ciphertext to the Keeper. The Keeper decrypts, and the shared secret key is just the resulting plaintext b.

Security of the protocol follows from the fact that after the Encrypt Zero protocol, the Keeper's key has min-entropy relative to any eavesdropper (since the eavesdropper learns no more than the Recorder). Moreover, the Keeper acts honestly, so the final ciphertext is always a fresh encryption. Finally, the encryption scheme is leakage resilient so it hides the bit b even though the adversary may have some knowledge of the key.

Notice that this scheme has *perfect* correctness, in that the two parties always arrive at a secret key. This is in contrast to the existing schemes based on the birthday paradox, where security is only statistical, and moreover this holds only if the adversary's space bounds are asymptotically smaller than n^2. In contrast, we get perfect correctness and statistical security for adversarial space bounds that are $O(n^2)$. The honest users only require $O(n)$ space.

Bit Commitment. We now describe a simple bit-commitment protocol using the above Encrypt Zero protocol. Recall that in a bit-commitment scheme, there are two phases: a commit phase where the Committer commits to a bit b, and a reveal or de-commit phase where the Committer reveals b and proves that b was the value committed to. After the commit phase, we want that the bit b is hidden. On the other hand, we want the commit phase to be binding, in that the Committer cannot later change the committed bit to something else.

The Committer and the Verifier will run the Encrypt Zero protocol, with Committer playing the role of Keeper and Verifier the role of Recorder. The protocol works as follows:

- Run the Encrypt Zero protocol, giving the Committer a random key $\mathbf{s}$ and the Verifier λ random encryptions c_i of 0.
- The Committer then sends an encryption of b relative to the key $\mathbf{s}$.
- To open the commitment, the Committer sends $\mathbf{s}$. The Verifier checks that $\mathbf{s}$ correctly decrypts all the c_i to 0. If so, it decrypts the final ciphertext to get b.

The security of the Encrypt Zero protocol and the leakage resilience of the encryption scheme show that this scheme is hiding. For binding, we note that an honest Committer will have no idea what encryptions c_i the Verifier has. As such, if the Committer later tries to change its committed bit by sending a malicious key $\mathbf{s}'$, $\mathbf{s}'$ will cause each ciphertext c_i to decrypt to 1 with probability $1/2$. Therefore, the Committer will get caught with probability $1 - 2^{-\lambda}$.

Already, this gives a very simple protocol for bit commitment that is *non-interactive*; in contrast, the prior work of Ding et al. [DHRS04] required five rounds. One limitation is that we require the Committer to behave honestly during the commit phase. For example, if the Committer chooses $\mathbf{R}$ to be low rank, then the encryptions obtained by the Verifier will not be independent of the Committer's view, and hence the Committer may be able to cheat during the de-commit phase.

To get around this, we tweak the Encrypt Zero protocol slightly to get security even against malicious Keepers. Our Enhanced Encrypt Zero protocol is as follows:

- The Keeper chooses a random key $\mathbf{k} \in \{0,1\}^n$ and an independent random secret $\mathbf{s} \in \{0,1\}^m$. We will let $m = 2n$. The Recorder chooses a secret matrix $\boldsymbol{\Sigma} \in \{0,1\}^{\lambda \times m}$.
- The Keeper streams random encryptions of the bits of s_i. We will write this in matrix form as $(\mathbf{R}, \mathbf{a} = \mathbf{R} \cdot \mathbf{k} + \mathbf{s})$.
- The Recorder computes $\boldsymbol{\Psi} = \boldsymbol{\Sigma} \cdot \mathbf{R}$ and $\boldsymbol{\kappa} = \boldsymbol{\Sigma} \cdot \mathbf{a}$.
- The Keeper then sends its key $\mathbf{k}$ *in the clear*.
- The Keeper outputs its secret $\mathbf{s}$ as the key, and the Recorder outputs $(\boldsymbol{\Sigma}, \boldsymbol{\kappa} - \boldsymbol{\Psi} \cdot \mathbf{k})$.

Notice that $\boldsymbol{\kappa} - \boldsymbol{\Psi} \cdot \mathbf{k} = \boldsymbol{\Sigma} \cdot \mathbf{s}$, a list of λ encryptions of 0 relative to the key $\mathbf{s}$, as desired. Moreover, these encryptions are random encryptions, even if $\mathbf{R}$ is chosen adversarially by the Keeper, since the Keeper has no knowledge or control over $\boldsymbol{\Sigma}$.

To prove the min-entropy of $\mathbf{s}$ relative to a malicious Recorder, we note that the real-or-random CPA security of the encryption scheme shows that just prior to receiving $\mathbf{k}$, the Recorder has essentially no information about $\mathbf{s}$. Then, since $\mathbf{k}$ is n bits, revealing it can only reveal n bits of $\mathbf{s}$. But $\mathbf{s}$ is a uniformly random $m = 2n$ bit string, meaning it has roughly n bits of min-entropy remaining, as desired. Thus we get both our security properties, even for malicious parties.

Our Enhanced Encrypt Zero protocol roughly doubles the communication, but otherwise maintains all the attractive properties of the original scheme: it is non-interactive and has perfect correctness.

Putting it all together, our bit commitment protocol is the following:

- To commit to a bit b, the Committer streams $\mathbf{R}$, $\mathbf{a} = \mathbf{R} \cdot \mathbf{k} + \mathbf{s}$ followed by $\mathbf{k}, \boldsymbol{\gamma}, c = \boldsymbol{\gamma} \cdot \mathbf{s} + b$ for random $\mathbf{R}, \mathbf{k}, \mathbf{s}, \boldsymbol{\gamma}$.
- The Verifier records $\boldsymbol{\Sigma}, \boldsymbol{\Psi} = \boldsymbol{\Sigma} \cdot \mathbf{R}, \boldsymbol{\kappa} = \boldsymbol{\Sigma} \cdot \mathbf{a}$ for a random choice of $\boldsymbol{\Sigma}$, and then once $\mathbf{k}$ comes in it computes $\boldsymbol{\phi} = \boldsymbol{\kappa} - \boldsymbol{\Psi} \cdot \mathbf{k} = \boldsymbol{\Sigma} \cdot \mathbf{s}$.
- To reveal the bit b, the Committer just sends $\mathbf{x} = \mathbf{s}$.
- The Verifier checks that $\boldsymbol{\phi} = \boldsymbol{\Sigma} \cdot \mathbf{x}$. If so, it computes $b' = c - \boldsymbol{\gamma} \cdot \mathbf{x}$.

Oblivious Transfer. We now turn to constructing an oblivious transfer (OT) protocol. In an OT protocol, one party, the Sender, has two input bits x_0, x_1. Another party, the Receiver, has a bit b. The Receiver would like to learn x_b without revealing b, and the Sender would like to ensure that the Receiver learns nothing about x_{1-b}.

In our protocol, the Receiver will play the role of Committer in our commitment scheme, committing to its input b. The Sender will play the role of Recorder in the Encrypt Zero protocol, setting $\lambda = 2$. The hiding property of the commitment scheme ensures that the space-bounded Sender learns nothing about the Receiver's bit b.

At the end of the Receiver's message, the Sender has an encryption $(\boldsymbol{\gamma}, c^* = \boldsymbol{\gamma} \cdot \mathbf{s} + b)$ of b with secret key $\mathbf{s}$. Additionally, it also has two encryptions of 0, namely $(\boldsymbol{\sigma}_0, c_0 = \boldsymbol{\sigma}_0 \cdot \mathbf{s})$ and $(\boldsymbol{\sigma}_1, c_1 = \boldsymbol{\sigma}_1 \cdot \mathbf{s})$ for random vectors $\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1$. Importantly, $\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1$ are independent of the Receiver's view, as they were chosen by the Sender.

The Sender will now exploit the additive homomorphism of the encryption scheme once more. In particular, it will compute encryptions of $(1-b)x_0$ and bx_1, which it will then send back to the Receiver. To compute an encryption of bx_1, it simply multiplies the ciphertext $(\boldsymbol{\gamma}, c^*)$ by x_1. Similarly, to compute an encryption of $(1-b)x_0$, it toggles c^* (to get an encryption of $1-b$) and then multiplies the entire ciphertext by x_0.

Now clearly these two ciphertexts reveal both x_0 and x_1, so the Sender cannot send them directly to the Receiver. Instead, it will *re-randomize* them by adding the two encryptions of 0. Now it obtains *fresh* encryptions of $(1-b)x_0$ and bx_1:

$$\begin{aligned} \boldsymbol{\sigma}_0 + x_0\boldsymbol{\gamma},\ c_0 + x_0(1-c^*) &= (\boldsymbol{\sigma}_0 + x_0\boldsymbol{\gamma}) \cdot \mathbf{s} + \big((1-b)x_0\big) \\ \boldsymbol{\sigma}_1 + x_1\boldsymbol{\gamma},\ c_1 + x_1 c^* &= (\boldsymbol{\sigma}_1 + x_1\boldsymbol{\gamma}) \cdot \mathbf{s} + \big(bx_1\big) \end{aligned}$$

It sends these ciphertexts to the Receiver, who then decrypts. All the Receiver learns then is $(1-b)x_0$ and bx_1. One of these plaintexts will be x_b as desired, and the other will be 0. Thus, the Receiver learns nothing about x_{1-b}.

Our protocol is round-optimal, since it involves only a single message in each direction. This improves on the best prior work of Ding et al. [DHRS04] requiring 5 rounds. Additionally, our protocol is much simpler than the prior work.

1.5 Discussion

Just as homomorphic encryption has been an extremely useful tool in traditional cryptography, our work demonstrates that the homomorphic properties of Raz's encryption scheme are also fruitful for the Bounded Storage model. We believe our work will be a useful starting point for much future work in this area.

1.6 Other Related Work

A recent work by Ball et al. [BDKM18] shows another application of Raz's encryption scheme, where they use it to construct unconditional non-malleable codes against streaming, space-bounded tempering.

2 Preliminaries

Here, we recall some basic cryptographic notions, translated into the setting of the bounded storage model. In the following definitions, n will be a security parameter.

A symmetric encryption scheme is a pair of algorithms $\Pi = (\mathtt{Enc}, \mathtt{Dec})$ with an associated key space $\mathcal{K}_n$, message space $\mathcal{M}$, and ciphertext space $\mathcal{C}_n$. Notice that the key space and ciphertext space depend on n; the message space will not depend on n. We require that:

- $\mathtt{Enc} : \mathcal{K}_n \times \mathcal{M} \to \mathcal{C}_n$ is a probabilistic polynomial time (PPT) algorithm
- $\mathtt{Dec} : \mathcal{K}_n \times \mathcal{C}_n \to \mathcal{M}$ is a deterministic polynomial time algorithm.
- Correctness: for any $k \in \mathcal{K}_n$ and any message $m \in \mathcal{M}$,

$$\Pr[\mathtt{Dec}(k, \mathtt{Enc}(k, m)) = m] = 1.$$

Additionally, we will require a security notion. In this work, we will focus on the following notion.

Definition 1 (Real-or-Random-Ciphertext (RoRC) Security). *Let $\mathcal{A}$ be an adversary. $\mathcal{A}$ plays the following game $\mathsf{RoRC}_{\mathcal{A},\Pi,b}(n, q)$:*

- *The challenger's input is a bit $b \in \{0, 1\}$.*
- *The challenger chooses a random key $k \in \mathcal{K}_n$*
- *$\mathcal{A}$ makes q adaptive queries on messages $m_1, \ldots, m_q \in \mathcal{M}$.*
- *In response to each query, the challenger does the following:*
  - *If $b = 0$, the challenger responds with $c_i \leftarrow \mathtt{Enc}(k, m_i)$.*
  - *If $b = 1$, the challenger responds with a random ciphertext $c_i \in \mathcal{C}_n$.*
- *Finally, $\mathcal{A}$ outputs a guess b' for b.*

We say that Π is $(S(n), Q(n), \epsilon)$-secure if for all adversaries that use at most $S(n)$ memory bits and $Q(n)$ queries (i.e. $q \leq Q(n)$),

$$|\Pr[\mathsf{RoRC}_{\mathcal{A},\Pi,0}(n, q) = 1] - \Pr[\mathsf{RoRC}_{\mathcal{A},\Pi,1}(n, q) = 1]| \leq \epsilon.$$

In this work, a lot of the proofs are based on the Leftover Hash Lemma for Conditional Min-Entropy due to Impagliazzo, Levin, and Luby [ILL89].

For random distributions X and Y, let $H_\infty(X|Y)$ denote the min-entropy of X conditioned on Y. Let $X \approx_\epsilon Y$ denote that the two distributions are ϵ-close, i.e. the statistical distance between these two distributions $\Delta(X, Y) \leq \epsilon$. Furthermore, let U_m denote a uniformly distributed random variable of m bits for some positive integer m.

Lemma 1 (Leftover Hash Lemma for Conditional Min-Entropy [ILL89]). *Let X, E be a joint distribution. If $H_\infty(X|E) \geq k$, and $m = k - 2\log(1/\epsilon)$, then*

$$(H(X), H, E) \approx_{\epsilon/2} (U_m, U_d, E),$$

where m is the output length of a universal hash function H, and d is the length of the description of H.

3 Raz's Encryption Scheme

Our constructions of the commitment scheme and the oblivious transfer scheme are largely based on the bit encryption scheme from parity learning proposed by Raz [Raz17]. Raz sketches how his lower bound for learning implies the security of his encryption scheme. Below we reproduce the construction of the encryption scheme, and formalize the security proof.

Construction 1 (Bit Encryption Scheme from Parity Learning). For a given security parameter n, the encryption scheme consists of a message space $\mathcal{M} = \{0,1\}$, a ciphertext space $\mathcal{C}_n = \{0,1\}^n \times \{0,1\}$, a key space $\mathcal{K}_n = \{0,1\}^n$, and a pair of algorithms $\Pi = (\mathtt{Enc}, \mathtt{Dec})$ as specified below:

- $\mathtt{Enc}(\mathbf{k}, m \in \mathcal{M})$: Samples a random row vector $\mathbf{r} \leftarrow \{0,1\}^n$, computes $a = \mathbf{r} \cdot \mathbf{k} + m$, and outputs the ciphertext $c = (\mathbf{r}, a)$ as a pair.
- $\mathtt{Dec}(\mathbf{k}, c = (\mathbf{r}, a) \in \mathcal{C}_n)$: Computes and outputs $m' = \mathbf{r} \cdot \mathbf{k} + a$.

To prove Real-or-Random-Ciphertext security of the above scheme, we rely on a result from Raz [Raz17], reproduced below.

Lemma 2 ([Raz17]). *For any $C < \frac{1}{20}$, there exists $\alpha > 0$, such that: for uniform $\mathbf{k} \in \{0,1\}^n$, $m \leq 2^{\alpha n}$, and algorithm $\mathcal{A}$ that takes a stream of $(\mathbf{x}_1, y_1)$, $(\mathbf{x}_2, y_2)$, $\ldots$, $(\mathbf{x}_m, y_m)$, where $\mathbf{x}_i$ is a uniform distribution over $\{0,1\}^n$ and $y_i = \mathbf{x}_i \cdot \mathbf{k}$ for every i, under the condition that $\mathcal{A}$ uses at most Cn^2 memory bits and outputs $\tilde{\mathbf{k}} \in \{0,1\}^n$, then $Pr[\tilde{\mathbf{k}} = \mathbf{k}] \leq O(2^{-\alpha n})$.*

We also rely on the Goldreich-Levin Algorithm, reproduced below.

Lemma 3 (Goldreich-Levin Algorithm [GL89]). *Assume that there exists a function $f : \{0,1\}^n \to \{0,1\}$ s.t. for some unknown $\mathbf{x} \in \{0,1\}^n$, we have*

$$\Pr_{\mathbf{r} \in \{0,1\}^n}[f(\mathbf{r}) = \langle \mathbf{x}, \mathbf{r} \rangle] \geq \frac{1}{2} + \epsilon$$

for $\epsilon > 0$.

Then there exists an algorithm $\mathcal{GL}$ that runs in time $O(n^2 \epsilon^{-4} \log n)$, makes $O(n\epsilon^{-4} \log n)$ orcale queries into f, and outputs $\mathbf{x}$ with probability $\Omega(\epsilon^2)$.

Instead of directly proving RoRC security of the encryption scheme, we prove Modified Real-or-Random-Ciphertext (RoRC') security, which differs from RoRC security in that for all but the last query, the challenger always responds with the valid encryption of the messsage; for the last query, the challenger responds either with a valid encryption or a random ciphertext, each with probability 1/2. A detailed definition is given below.

Definition 2 (Modified Real-or-Random-Ciphertext (RoRC') Security). *Let $\mathcal{A}$ be an adversary. $\mathcal{A}$ plays the following game $\mathsf{RoRC}'_{\mathcal{A},\Pi,b}(n,q)$:*

- *The challenger's input is a bit $b \in \{0,1\}$.*

– *The challenger chooses a random key* $k \in \mathcal{K}_n$.
– $\mathcal{A}$ *makes* q *adaptive queries on messages* $m_1, \ldots, m_q \in \mathcal{M}$.
– *In response to query* m_i *with* $1 \leq i \leq q-1$, *the challenger responds with* $c_i \leftarrow \mathtt{Enc}(k, m_i)$.
– *In response to query* m_q, *the challenger does the following:*
  • *If* $b = 0$, *the challenger responds with* $c_q \leftarrow \mathtt{Enc}(k, m_q)$.
  • *If* $b = 1$, *the challenger responds with a random ciphertext* $c_q \in \mathcal{C}_n$.
– *Finally,* $\mathcal{A}$ *outputs a guess* b' *for* b.

We say that Π *is* $(S(n), Q(n), \epsilon)$*-secure if for all adversaries that use at most* $S(n)$ *memory bits and* $Q(n)$ *queries (i.e.* $q \leq Q(n)$*),*

$$|\Pr[\mathsf{RoRC'}_{\mathcal{A},\Pi,0}(n,q) = 1] - \Pr[\mathsf{RoRC'}_{\mathcal{A},\Pi,1}(n,q) = 1]| \leq \epsilon.$$

We now show that RoRC' security implies RoRC security.

Lemma 4. *An encryption scheme that is* $(S(n), Q(n), \epsilon)$*-secure under the RoRC' setting is* $(S(n), Q(n), Q(n)\epsilon)$*-secure under the RoRC setting.*

Proof. We prove this using a hybrid argument. For any $q \leq Q(n)$, consider the hybrid security games H_0, H_1, …, H_q, where H_j describes the following hybrid game:

– The challenger chooses a random key $k \in \mathcal{K}_n$.
– $\mathcal{A}$ makes q adaptive queries on messages $m_1, \ldots, m_q \in \mathcal{M}$.
– In response to query m_i with $1 \leq i \leq j$, the challenger responds with $c_i \leftarrow \mathtt{Enc}(k, m_i)$.
– In response to query m_i with $j+1 \leq i \leq q$, the challenger responds with a random ciphertext $c_i \in \mathcal{C}_n$.

Particularly, notice that H_0 corresponds to a game where the challenger always responds with random ciphertexts, and that H_q corresponds to a game where the challenger always responds with valid encryptions of the messages. In that way, the $\mathsf{RoRC}_{\mathcal{A},\Pi,b}(n,q)$ game is equivalent to distinguishing H_q from H_0.

To put this formally, let D be an arbitrary distinguisher, and $h \leftarrow H_j$ denote a randomly sampled instance of the game H_j, we have

$$\begin{aligned} &|\Pr[\mathsf{RoRC}_{\mathcal{A},\Pi,0}(n,q) = 1] - \Pr[\mathsf{RoRC}_{\mathcal{A},\Pi,1}(n,q) = 1]| \\ =&\left| \Pr_{h \leftarrow H_q}[D(h) = 1] - \Pr_{h \leftarrow H_0}[D(h) = 1] \right|. \end{aligned}$$

By the hybrid argument, there exists j, s.t. $0 \leq j < q$ and

$$\left| \Pr_{h \leftarrow H_q}[D(h) = 1] - \Pr_{h \leftarrow H_0}[D(h) = 1] \right| \leq q \left| \Pr_{h \leftarrow H_{j+1}}[D(h) = 1] - \Pr_{h \leftarrow H_j}[D(h) = 1] \right|.$$

To distinguish between H_{j+1} and H_j, consider the following security game $\mathsf{Dist}_{\mathcal{A},\Pi,b}(n,q,j)$:

– The challenger's input is a bit $b \in \{0,1\}$.
– The challenger chooses a random key $k \in \mathcal{K}_n$.
– $\mathcal{A}$ makes q adaptive queries on messages $m_1, \ldots, m_q \in \mathcal{M}$.
– In response to query m_i with $1 \leq i \leq j$, the challenger responds with $c_i \leftarrow \mathtt{Enc}(k, m_i)$.
– In response to query m_{j+1}, the challenger does the following:
  • If $b = 0$, the challenger responds with $c_{j+1} \leftarrow \mathtt{Enc}(k, m_{j+1})$.
  • If $b = 1$, the challenger responds with a random ciphertext $c_{j+1} \in \mathcal{C}_n$.
– In response to query m_i with $j + 1 < i \leq q$, the challenger responds with a random ciphertext $c_i \in \mathcal{C}_n$.
– Finally, $\mathcal{A}$ outputs a guess b' for b.

This directly gives us

$$\begin{aligned}&\left|\Pr_{h \leftarrow H_{j+1}}[D(h) = 1] - \Pr_{h \leftarrow H_j}[D(h) = 1]\right| \\ =&\left|\Pr[\mathsf{Dist}_{\mathcal{A},\Pi,0}(n,q,j) = 1] - \Pr[\mathsf{Dist}_{\mathcal{A},\Pi,1}(n,q,j) = 1]\right|.\end{aligned}$$

Next, we show that we can use an adversary $\mathcal{A}$ for the $\mathsf{Dist}_{\mathcal{A},\Pi,b}(n,q,j)$ game to construct an adversary $\mathcal{A}'$ for the $\mathsf{RoRC}'_{\mathcal{A}',\Pi,b}(n,j+1)$ game. Notice that the only difference between $\mathsf{RoRC}'_{\mathcal{A}',\Pi,b}(n,j+1)$ and $\mathsf{Dist}_{\mathcal{A},\Pi,b}(n,q,j)$ is that $\mathsf{Dist}_{\mathcal{A},\Pi,b}(n,q,j)$ has $(q-j-1)$ extra queries at the end. An adversary $\mathcal{A}'$ for $\mathsf{RoRC}'_{\mathcal{A}',\Pi,b}(n,j+1)$ can simulate $\mathsf{Dist}_{\mathcal{A},\Pi,b}(n,q,j)$ for adversary $\mathcal{A}$ by forwarding each of $\mathcal{A}$'s first $(j+1)$ queries to the challenger in $\mathsf{RoRC}'_{\mathcal{A}',\Pi,b}(n,j+1)$, and similarly forward the responses from the challenger back to $\mathcal{A}$. For the additional $(q-j-1)$ queries in the end, $\mathcal{A}'$ can simply respond by drawing random ciphertexts from $\mathcal{C}_n$. $\mathcal{A}'$ will output whatever is output by $\mathcal{A}$.

Notice that adversary $\mathcal{A}'$ does *not* require any additional memory space besides the space used by adversary $\mathcal{A}$. All that $\mathcal{A}'$ needs to do is to forward $\mathcal{A}$'s queries and the challenger's responses, and to sample random ciphertexts from $\mathcal{C}_n$. These operations do not require $\mathcal{A}'$ to store any persistent states.

Therefore, we have

$$\begin{aligned}&\left|\Pr[\mathsf{Dist}_{\mathcal{A},\Pi,0}(n,q,j) = 1] - \Pr[\mathsf{Dist}_{\mathcal{A},\Pi,1}(n,q,j) = 1]\right| \\ \leq&\left|\Pr[\mathsf{RoRC}'_{\mathcal{A},\Pi,0}(n,j+1) = 1] - \Pr[\mathsf{RoRC}'_{\mathcal{A},\Pi,1}(n,j+1) = 1]\right|.\end{aligned}$$

Bringing all these parts together, assuming that the encryption scheme Π is $(S(n), Q(n), \epsilon)$-secure yields

$$\begin{aligned}
&|\Pr[\mathsf{RoRC}_{\mathcal{A},\Pi,0}(n,q)=1] - \Pr[\mathsf{RoRC}_{\mathcal{A},\Pi,1}(n,q)=1]| \\
=&\left|\Pr_{h\leftarrow H_q}[D(h)=1] - \Pr_{h\leftarrow H_0}[D(h)=1]\right| \\
\leq& q\left|\Pr_{h\leftarrow H_{j+1}}[D(h)=1] - \Pr_{h\leftarrow H_j}[D(h)=1]\right| \\
=& q\,|\Pr[\mathsf{Dist}_{\mathcal{A},\Pi,0}(n,q,j)=1] - \Pr[\mathsf{Dist}_{\mathcal{A},\Pi,1}(n,q,j)=1]| \\
\leq& q\,|\Pr[\mathsf{RoRC'}_{\mathcal{A},\Pi,0}(n,j+1)=1] - \Pr[\mathsf{RoRC'}_{\mathcal{A},\Pi,1}(n,j+1)=1]| \\
\leq& q\epsilon \leq Q(n)\epsilon.
\end{aligned}$$

Therefore, Π is $(S(n)), Q(n), Q(n)\epsilon)$-secure under the RoRC setting. □

Theorem 1. *For any $C < \frac{1}{20}$, there exists $\alpha > 0$, s.t. the bit encryption scheme from parity learning is $(Cn^2, 2^{\alpha n}, O(2^{-\alpha n/2}))$-secure under the RoRC' setting.*

Proof. We prove this result by reducing a parity learning game to an RoRC' game.

To start off, we consider a weaker variant of the parity learning game described in Lemma 2, denoted as $\mathsf{PL}_{\mathcal{A},b}(n,q)$:

- The challenger's input is a bit $b \in \{0,1\}$.
- The challenger chooses a random $\mathbf{k} \in \{0,1\}^n$.
- The challenger streams $(\mathbf{x}_1, y_1)$, $(\mathbf{x}_2, y_2)$, …, $(\mathbf{x}_{q-1}, y_{q-1})$, where $\mathbf{x}_i$ is uniformly distributed over $\{0,1\}^n$ and $y_i = \mathbf{x}_i \cdot \mathbf{k}$ for all i.
- The challenger sends $(\mathbf{x}_q, y_q)$, where $\mathbf{x}_q$ is uniformly distributed over $\{0,1\}^n$ and:
  - If $b = 0$, $y_q = \mathbf{x}_q \cdot \mathbf{k}$.
  - If $b = 1$, y_q is a random bit.
- Finally, $\mathcal{A}$ outputs a guess b' for b.

We now show how we can use an adversary $\mathcal{A}$ for $\mathsf{RoRC'}_{\mathcal{A},\Pi,b}(n,q)$ to build an adversary $\mathcal{A}'$ for $\mathsf{PL}_{\mathcal{A}',b}(n,q)$. The adversary $\mathcal{A}'$ works as follows:

- Simulate for $\mathcal{A}$ an $\mathsf{RoRC'}_{\mathcal{A},\Pi,b}(n,q)$ game.
- For every query m_i submitted by $\mathcal{A}$, respond with $(\mathbf{x}_i, y_i + m_i)$ where $\mathbf{x}_i$ and y_i come from the i-th pair of the $\mathsf{PL}_{\mathcal{A}',b}(n,q)$ game.
- If the adversary $\mathcal{A}$ outputs 0, output 0. Otherwise, output 1.

This should be easily verifiable. First, notice that $\mathcal{A}'$ faithfully simulates $\mathsf{RoRC'}_{\mathcal{A},\Pi,b}(n,q)$. For $1 \leq i \leq q-1$, $\mathcal{A}$ receives $(\mathbf{x}_i, y_i + m_i) = (\mathbf{x}_i, \mathbf{x}_i \cdot \mathbf{k} + m_i)$, which is a valid encryption of m_i. Also, for the last query m_q, $\mathcal{A}$ receives either $(\mathbf{x}_q, y_q + m_q) = (\mathbf{x}_q, \mathbf{x}_q \cdot \mathbf{k} + m_q)$, i.e. a valid encryption, or $(\mathbf{x}_q, y_q + m_q)$ for a random bit y_q, i.e. a random ciphertext. Secondly, if $\mathcal{A}$ outputs 0, that implies $(\mathbf{x}_q, y_q + m_q) = \mathtt{Enc}(\mathbf{k}, m_q) = (\mathbf{x}_q, \mathbf{x}_q \cdot \mathbf{k} + m_q)$, and hence $y_q = \mathbf{x}_q \cdot \mathbf{k}$ and $\mathcal{A}'$

should output 0. Lastly, if $\mathcal{A}$ outputs 1, we have $y_q + m_q$ being a random bit. Since m_q is fixed, we have y_q a random bit and hence $\mathcal{A}'$ should output 1.

This yields

$$\begin{aligned}&|\Pr[\mathsf{RoRC'}_{\mathcal{A},\Pi,0}(n,q)=1] - \Pr[\mathsf{RoRC'}_{\mathcal{A},\Pi,1}(n,q)=1]|\\ \leq &|\Pr[\mathsf{PL}_{\mathcal{A},0}(n,q)=1] - \Pr[\mathsf{PL}_{\mathcal{A},1}(n,q)=1]|\,.\end{aligned}$$

Let $\beta = |\Pr[\mathsf{PL}_{\mathcal{A},0}(n,q)=1] - \Pr[\mathsf{PL}_{\mathcal{A},1}(n,q)=1]|$. Then we have an algorithm that distinguishes between $(\mathbf{x}_q, y_q = \mathbf{x}_q \cdot \mathbf{k})$ and $(\mathbf{x}_q, y_q \leftarrow \{0,1\})$ with probability $(1+\beta)/2$, i.e. it outputs 0 if y_q is a valid inner product and 1 if it is random. This can be easily converted into an algorithm that given $\mathbf{x}_q$, outputs $\mathbf{x}_q \cdot \mathbf{k}$ with probability $(1+\beta)/2$ (simply XOR the output of the previous algorithm with y_q). Let f be the function computed by this algorithm. Then for given $\mathbf{x}_q \in \{0,1\}^n$ and unknown $\mathbf{k} \in \{0,1\}^n$, $f(\mathbf{x}_q) = \langle \mathbf{k}, \mathbf{x}_q \rangle$ with probability $(1+\beta)/2$. By applying Lemma 3, there is an algorithm that runs in time $O(n^2\beta^{-4}\log n)$ and outputs $\mathbf{k}$ with probability at least $\Omega(\beta^2)$.

Recall from Lemma 2 that for any $C < 1/20$, there is a positive α such that any potentially *computationally unbounded* algorithm that uses up to Cn^2 memory bits and has access to at most $2^{\alpha n}$ $(\mathbf{x}_i, y_i)$ pairs can output $\mathbf{k}$ with probability at most $O(2^{-\alpha n})$. Therefore, for adversaries that are space-bounded by Cn^2 bits and submit at most $2^{\alpha n}$ queries, $\Omega(\beta^2) \leq O(2^{-\alpha n})$. And hence $\beta = O(2^{-\alpha n/2})$

Therefore, for any $C < 1/20$, there is a positive α such that for all adversaries that use at most Cn^2 memory bits and at most $2^{\alpha n}$ queries ($q \leq 2^{\alpha n}$), we have

$$|\Pr[\mathsf{RoRC'}_{\mathcal{A},\Pi,0}(n,q)=1] - \Pr[\mathsf{RoRC'}_{\mathcal{A},\Pi,1}(n,q)=1]| \leq \beta = O(2^{-\alpha n/2}),$$

i.e. the scheme is $(Cn^2, 2^{\alpha n}, O(2^{-\alpha n/2}))$-secure under the RoRC' setting as desired. □

Corollary 1 (RoRC Security of the Bit Encryption Scheme from Parity Learning). *For any $C < \frac{1}{20}$, there exists $\alpha > 0$, s.t. the bit encryption scheme from parity learning is $(Cn^2, 2^{\alpha n/4}, O(2^{-\alpha n/2}))$-secure under the RoRC' setting (here we further bound the number of queries to $\alpha n/4$ instead of αn). By Lemma 4, this scheme is also $(Cn^2, 2^{\alpha n/4}, 2^{\alpha n/4} \cdot O(2^{-\alpha n/2}) = O(2^{-\alpha n/4}))$-secure under the RoRC setting. Put another way, for any $C < \frac{1}{20}$, there exists $\alpha'(= \alpha/4) > 0$, s.t. the bit encryption scheme from parity learning is $(Cn^2, 2^{\alpha' n}, O(2^{-\alpha' n}))$-secure under the RoRC setting.*

4 Encrypt Zero Protocols

In this section, we introduce two constructions of the Encrypt Zero Protocol. They both have the same goal: to give one party, the *Keeper*, a random key $\mathbf{s}$, and the other party, known as the *Recorder*, several encryptions of 0 under the key $\mathbf{s}$. They differ in that the simple construction is only secure against honest-but-curious Keepers, while the enhanced construction is secure even against malicious Keepers.

Before we jump into the constructions, we first define an Encrypt Zero Protocol and its security properties.

An Encrypt Zero Protocol Π involves two parties, a Keeper $\mathcal{K}$ and a Recorder $\mathcal{R}$. The protocol takes three parameters $n, m = O(n)$ and λ, and produces $(\mathbf{s}, \{c_1, c_2, \ldots, c_\lambda\}, \mathsf{trans})$, where $\mathbf{s}$ is a random key output by $\mathcal{K}$, $\{c_1, c_2, \ldots, c_\lambda\}$ is a set of ciphertexts output by $\mathcal{R}$, and trans is the transcript of their communication.

The correctness of an Encrypt Zero Protocol requires that the set of ciphertexts output by $\mathcal{R}$ are encryptions of zero under the key $\mathbf{s}$ output by $\mathcal{K}$. Put formally, we require that $\mathtt{Dec}(\mathbf{s}, c_i) = 0$ for all i.

Now, we define two desired security properties for the Encrypt Zero Protocol, namely Keeper security and Recorder security.

The security of the Keeper ensures that the Keeper's key $\mathbf{s}$ has enough min-entropy conditioned on the Recorder's view $\mathsf{view}_{\mathcal{R}}$.

Definition 3 (Keeper Security). *Let the view of the Recorder be $\mathsf{view}_{\mathcal{R}}$, we say that a protocol Π is $(S(n), h)$-secure for the Keeper if for all Recorders $\mathcal{R}$ that use up to $S(n)$ memory bits,*

$$H_\infty(\mathbf{s}|\mathsf{view}_{\mathcal{R}}) \geq h.$$

The security of the Recorder ensures that the Keeper learns nothing about $c_1, c_2, \ldots, c_\lambda$ (except that they are encryptions of zero).

For an honest-but-curious Keeper $\mathcal{K}$, this means that given all the Keeper's randomness and the transcript produced by the protocol, it is hard to distinguish the output ciphertexts $(c_1, c_2, \ldots, c_\lambda)$ from some random ciphertexts that encrypt zero.

Definition 4 (Recorder Security with Honest-but-Curious Keeper). *Let $C = \{c_1, c_2, \ldots, c_\lambda\}$ be the ciphertexts output by $\mathcal{R}$ at the end of the protocol, and $C' = \{c'_1, c'_2, \ldots, c'_\lambda\}$ where $c'_i \leftarrow \mathtt{Enc}(\mathbf{s}, 0)$ be fresh encryptions of zero under the key $\mathbf{s}$. Let $\mathsf{state}_{\mathcal{K}}$ consist of all the random coins used by $\mathcal{K}$ together with trans. Given the Keeper's state $\mathsf{state}_{\mathcal{K}}$, the key $\mathbf{s}$, the protocol Π is ϵ-secure for the Recorder if for any distinguisher D,*

$$\left| \Pr_{c \leftarrow C}[D_{\mathsf{state}_{\mathcal{K}}, \mathbf{s}}(c) = 1] - \Pr_{c \leftarrow C'}[D_{\mathsf{state}_{\mathcal{K}}, \mathbf{s}}(c) = 1] \right| \leq \epsilon.$$

In the case of a malicious Keeper $\mathcal{K}^*$ who can have arbitrary behavior, we let $\mathsf{state}_{\mathcal{K}^*}$ be the state of $\mathcal{K}^*$ at the end of the protocol. Notice that regardless of the possible behaviors that $\mathcal{K}^*$ could have, it is constrained to the state that it has stored at the end of the protocol. It has no additional information besides what it has stored in $\mathsf{state}_{\mathcal{K}^*}$.

Definition 5 (Recorder Security with Malicious Keeper). *Let $C = \{c_1, c_2, \ldots, c_\lambda\}$ be the ciphertexts output by $\mathcal{R}$ at the end of the protocol, and $C' = \{c'_1, c'_2, \ldots, c'_\lambda\}$ where $c'_i \leftarrow \mathtt{Enc}(\mathbf{s}, 0)$ be fresh encryptions of zero under the key $\mathbf{s}$.*

Given the malicious Keeper's state $\mathsf{state}_{\mathcal{K}^*}$*, the key* $\mathbf{s}$*, the protocol* Π *is* ϵ*-secure for the Recorder if for any distinguisher* D*,*

$$\left| \Pr_{c \leftarrow C}[D_{\mathsf{state}_{\mathcal{K}^*}, \mathbf{s}}(c) = 1] - \Pr_{c \leftarrow C'}[D_{\mathsf{state}_{\mathcal{K}^*}, \mathbf{s}}(c) = 1] \right| \leq \epsilon.$$

4.1 Simple Encrypt Zero Protocol

Here we present the Simple Encrypt Zero Protocol, which achieves Keeper Security and Recorder security against honest-but-curious Keeper. The main idea here is simple: the Keeper will stream a sequence of ciphertexts which are encryptions of zero, and and Recorder will obtain fresh encryptions of zero by taking random subset-sums of the ciphertexts received.

Construction 2 (Simple Encrypt Zero Protocol). A Simple Encrypt Zero Protocol instance $\mathtt{EZ}(n, m, \lambda)$ for the Keeper $\mathcal{K}$ and the Recorder $\mathcal{R}$ proceeds as follows:

- $\mathcal{K}$ chooses a random key $\mathbf{k} \in \{0,1\}^n$, and $\mathcal{R}$ chooses a random secret matrix $\boldsymbol{\Sigma} \in \{0,1\}^{\lambda \times m}$.
- $\mathcal{K}$ streams encryptions $(\mathbf{r}_i, a_i = \mathbf{r}_i \cdot \mathbf{k} + 0)$ to $\mathcal{R}$, for $i = 1, 2, \ldots, m$ and random $\mathbf{r}_i \in \{0,1\}^n$.
- $\mathcal{R}$ maintains matrix $\boldsymbol{\Psi} \in \{0,1\}^{\lambda \times n}$ and column vector $\boldsymbol{\kappa} \in \{0,1\}^{\lambda}$. Each row of $(\boldsymbol{\Psi}|\boldsymbol{\kappa})$ will be a random subset-sum of the encryptions sent by $\mathcal{K}$, with each subset-sum chosen according to $\boldsymbol{\Sigma}$. $\boldsymbol{\Psi}$ and $\boldsymbol{\kappa}$ will be computed on the fly. Specifically, when encryption $(\mathbf{r}_i, a_i)$ comes in, $\mathcal{R}$ will update $\boldsymbol{\Psi}$ to be $\boldsymbol{\Psi} + \boldsymbol{\sigma}_i \cdot \mathbf{r}_i$ and $\boldsymbol{\kappa}$ to be $\boldsymbol{\kappa} + \boldsymbol{\sigma}_i a_i$. Here, $\boldsymbol{\sigma}_i$ is the i-th column of $\boldsymbol{\Sigma}$, and $\mathbf{r}_i$ is interpreted as a row vector.
- At the end of the protocol, $\mathcal{K}$ outputs its key $\mathbf{s} = \mathbf{k}$, and $\mathcal{R}$ outputs $(\boldsymbol{\Psi}|\boldsymbol{\kappa})$, whose rows are the ciphertexts $c_1, c_2, \ldots, c_\lambda$.

Remark 1. For the ease of analysis, we combine all the encryptions sent together, and denote $\mathbf{R} = \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{r}_2 \\ \cdots \\ \mathbf{r}_m \end{bmatrix} \in \{0,1\}^{m \times n}$, and $\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \cdots \\ a_m \end{bmatrix} \in \{0,1\}^m$. This gives us

$$\mathbf{a} = \mathbf{R} \cdot \mathbf{k}.$$

Correspondingly, notice that $\mathcal{R}$ is essentially recording $\boldsymbol{\Sigma}$, $\boldsymbol{\Psi} = \boldsymbol{\Sigma} \cdot \mathbf{R}$ and $\boldsymbol{\kappa} = \boldsymbol{\Sigma} \cdot \mathbf{a} = \boldsymbol{\Sigma} \cdot \mathbf{R} \cdot \mathbf{k} = \boldsymbol{\Psi} \cdot \mathbf{k}$.

It is easy to verify that the rows of $(\boldsymbol{\Psi}|\boldsymbol{\kappa})$ are encryptions of 0 under the key $\mathbf{s} = \mathbf{k}$, as they are simply sums of encryptions of 0 under $\mathbf{s}$ and by the additive homomorphism of Raz's encryption scheme they also must encrypt 0. Therefore, this construction meets the correctness requirement for an Encrypt Zero Protocol.

Next, we show that this construction achieves Keeper security and Recorder security against honest-but-curious Keepers.

Theorem 2 (Keeper Security of EZ). *The Simple Encrypt Zero Protocol is* $(Cn^2, \Omega(\alpha n))$*-secure for the Keeper, for some* $C < \frac{1}{20}$ *and* α *dependent on* C*.*

Proof. This follows directly from Lemma 2. Here $\mathsf{view}_{\mathcal{R}}$ essentially contains m pairs of $(\mathbf{r}_i, a_i)$, where $a_i = \mathbf{r}_i \cdot \mathbf{s}$ for $i = 1, 2, \ldots, m$ and random $\mathbf{r}_i \leftarrow \{0,1\}^n$. For adversaries space-bounded to Cn^2 memory bits for some $C < \frac{1}{20}$ and α dependent on C, by applying Lemma 2, we get that the probability of an adversary outputting $\mathbf{s}$ is no more than $O(2^{-\alpha n})$. Hence, the average min-entropy of $\mathbf{s}$ conditioned on $\mathsf{view}_{\mathcal{R}}$ is $\Omega(\alpha n)$. □

Theorem 3 (Recorder Security of EZ). *The Simple Encrypt Zero Protocol with parameter* $m = 2n$ *and an honest-but-curious Keeper is* $O(2^{-n})$*-secure for the Recorder.*

Proof. Since the Keeper is honest and follows the protocol, $\mathbf{R}$ is a random $m \times n$ matrix. For $m = 2n$, we have $\mathbf{R}$ being a random $2n \times n$ matrix, which is full rank with probability $1 - O(2^{-n})$. Notice that if $\mathbf{R}$ is full rank, given that $\boldsymbol{\Sigma}$ is a random matrix conditioned on the Keeper's state $\mathsf{state}_{\mathcal{K}}$ and $\mathbf{s}$, $\boldsymbol{\Psi} = \boldsymbol{\Sigma} \cdot \mathbf{R}$ is also a random matrix conditioned on $\mathsf{state}_{\mathcal{K}}$ and $\mathbf{s}$.

In this way, conditioned on $\mathsf{state}_{\mathcal{K}}$ and $\mathbf{s}$, $(\boldsymbol{\Psi}|\boldsymbol{\kappa})$ contains *random* encryptions of 0. Therefore, by definition, these encryptions $\{c_1, \ldots, c_\lambda\}$ cannot be distinguished from $\{c'_1, \ldots, c'_\lambda\}$ where c'_i is a random encryption of 0. Hence, the probability of distinguishing C from C' is bounded by the probability that $\mathbf{R}$ is *not* full rank, which is $O(2^{-n})$. Thus we have

$$\left| \Pr_{c \leftarrow C}[D_{\mathsf{trans},\mathbf{s}}(c) = 1] - \Pr_{c \leftarrow C'}[D_{\mathsf{trans},\mathbf{s}}(c) = 1] \right| \leq 2O(2^{-n}) = O(2^{-n})$$

as desired. □

Kindly notice that this simple construction of an Encrypt Zero protocol is only secure for the Recorder if the Keeper is honest. For malicious Keepers, they could, for example, generate the matrix $\mathbf{R}$ with bad randomness so that it is very likely to be low rank.

One way to tackle this is to have the random matrix $\mathbf{R}$ generated and streamed by a trusted third party, which is a common practice in much of the prior work in the bounded storage model. However, if we do not wish to rely on a trusted third party (notice that the model without a trusted third party is stronger than one with a trusted third party), we show in the following subsection how we can tweak our simple construction to have Recorder security even against malicious Keepers.

4.2 Enhanced Encrypt Zero Protocol

In the Enhanced Encrypt Zero Protocol construction, we tweak the simple construction slightly to account for malicious Keepers.

Construction 3 (Enhanced Encrypt Zero Protocol). An Enhanced Encrypt Zero Protocol instance $\mathtt{EZ}^+(n, m, \lambda)$ with the Keeper $\mathcal{K}$ and the Recorder $\mathcal{R}$ proceeds as follows:

- $\mathcal{K}$ chooses a random key $\mathbf{k} \in \{0,1\}^n$ and an independent random secret $\mathbf{s} \in \{0,1\}^m$. $\mathcal{R}$ chooses a random secret matrix $\boldsymbol{\Sigma} \in \{0,1\}^{\lambda \times m}$.
- $\mathcal{K}$ streams random encryptions of the bits in $\mathbf{s}$. Namely, in matrix form, $\mathcal{K}$ sends $(\mathbf{R}, \mathbf{a} = \mathbf{R} \cdot \mathbf{k} + \mathbf{s})$ for random $\mathbf{R} \in \{0,1\}^{m \times n}$.
- $\mathcal{R}$ maintains matrix $\boldsymbol{\Psi} = \boldsymbol{\Sigma} \cdot \mathbf{R}$ and column vector $\boldsymbol{\kappa} = \boldsymbol{\Sigma} \cdot \mathbf{a}$.
- $\mathcal{K}$ sends its key $\mathbf{k}$ *in the clear*, and $\mathcal{R}$ uses that to compute $\boldsymbol{\phi} = \boldsymbol{\kappa} - \boldsymbol{\Psi} \cdot \mathbf{k}$.
- $\mathcal{K}$ outputs $\mathbf{s}$ as its key, and $\mathcal{R}$ outputs $(\boldsymbol{\Sigma}|\boldsymbol{\phi})$, whose rows are the ciphertexts $c_1, c_2, \ldots, c_\lambda$.

Notice that $\boldsymbol{\phi} = \boldsymbol{\kappa} - \boldsymbol{\Psi} \cdot \mathbf{k} = \boldsymbol{\Sigma} \cdot \mathbf{s}$, and hence the rows of $(\boldsymbol{\Sigma}|\boldsymbol{\phi})$ are indeed encryptions of 0 using key $\mathbf{s}$, as desired in the correctness property.

Theorem 4 (Keeper Security of $\mathtt{EZ}^+$). *The Simple Encrypt Zero Protocol is $(Cn^2, \Omega(n))$-secure for the Keeper, for some $C < \frac{1}{20}$ and α dependent on C.*

Proof. First, notice that before the Keeper sends over $\mathbf{k}$, the two distributions $(\mathbf{s}, \mathbf{R}, \mathbf{R} \cdot \mathbf{k} + \mathbf{s})$ and $(\mathbf{s}, \mathbf{R}, \mathbf{R} \cdot \mathbf{k} + \mathbf{s}')$ for random $\mathbf{s}' \in \{0,1\}^m$ are statistically indistinguishable, due to the RoRC security of Raz's encryption scheme.

Now, notice that in the second distribution, the probability the Recorder can guess $\mathbf{s}$ is 2^{-m}. In this case, if it later receives $\mathbf{k}$, the probability it guesses $\mathbf{s}$ is still at most 2^{n-m}, which is 2^{-n}.

Now, we use the following simple fact: suppose two distributions X, Y are ϵ-close. Then there is a procedure P which first samples $x \leftarrow X$, and then based which x it samples, it may replace x with a different sample x'. P satisfies the property that (1) its output distribution is identical to Y, and (2) the probability it re-samples is ϵ.

We use this simple fact by assigning X to $(\mathbf{s}, \mathbf{R}, \mathbf{R} \cdot \mathbf{k} + \mathbf{s}')$ for random $\mathbf{s}' \in \{0,1\}^m$ and Y to $(\mathbf{s}, \mathbf{R}, \mathbf{R} \cdot \mathbf{k} + \mathbf{s})$.

Now consider the probability of guessing $\mathbf{s}$. In the case X, we know it is 2^{-n}. So if we consider Y sampled from P, we know that the probability of guessing $\mathbf{s}$ in the non-replacing case is 2^{-n}. But the replacing case only happens with probability ϵ, meaning overall the probability of outputting $\mathbf{s}$ is at most $\epsilon + 2^{-n}$. □

Theorem 5 (Recorder Security of $\mathtt{EZ}^+$). *The Enhanced Encrypt Zero Protocol with parameter $m = 2n$ and any possibly malicious Keeper $\mathcal{K}^*$ is perfectly secure for the Recorder.*

Proof. Notice that regardless of the Keeper's state $\mathsf{state}_{\mathcal{K}^*}$ (even if one of a malicious Keeper), $\boldsymbol{\Sigma}$ is always random conditioned on $\mathsf{state}_{\mathcal{K}^*}$ and $\mathbf{s}$, since it is solely sampled by the Recorder. Therefore, $(\boldsymbol{\Sigma}|\boldsymbol{\phi})$ is already random encryptions of 0 conditioned on $\mathsf{state}_{\mathcal{K}^*}$ and $\mathbf{s}$. Hence, to distinguish it from other random encryptions of 0, one can do no better than a random guess. Thus, the advantage that any distinguisher D could have in distinguishing C and C' is 0 as desired. □

5 Two-Party Key-Agreement Protocol

Consider a pair of interactive PPT algorithms $\Pi = (\mathsf{A}, \mathsf{B})$. Each of A, B take n as input. We will let $(a, b, \mathsf{trans}) \leftarrow \Pi(n)$ denote the result of running the protocol on input n. Here, a is the output of A, b the output of B, and trans is the transcript of their communication.

A two-party key-agreement protocol is a protocol $\Pi = (\mathsf{A}, \mathsf{B})$ with the correctness property that $\Pr[a = b] = 1$. In this case, we will define $\hat{k} = a = b$ and write $(\hat{k}, \mathsf{trans}) \leftarrow \Pi(n)$. Additionally, we will require eavesdropping security:

Definition 6 (Eavesdropping Security of Two-Party Key-Agreement Protocol). *We say that Π is $(S(n), \epsilon)$-secure if for all adversaries $\mathcal{A}$ that use at most $S(n)$ memory bits,*

$$\begin{aligned} &|\Pr[\mathcal{A}(\hat{k}, \mathsf{trans}) = 1 : (\hat{k}, \mathsf{trans}) \leftarrow \Pi(n)] \\ &- \Pr[\mathcal{A}(k', \mathsf{trans}) = 1 : k' \leftarrow \mathcal{K}_n, (k, \mathsf{trans}) \leftarrow \Pi(n)]| \leq \epsilon. \end{aligned}$$

In this section we demonstrate how we can use the Simple Encrypt Zero Protocol to implement a two-party key-agreement protocol. For simplicity, we consider a key space of one single bit.

Construction 4 (Two-Party Key-Agreement Protocol). For two parties $\mathcal{P}$ and $\mathcal{Q}$ trying to derive a shared key $\hat{k} \in \{0, 1\}$, they will first run a Simple Encrypt Zero Protocol $\mathtt{EZ}(n, m, \lambda = 1)$ with $\mathcal{P}$ as the Keeper and $\mathcal{Q}$ as the Recorder. At the end of the $\mathtt{EZ}$ protocol, $\mathcal{P}$ gets a key $\mathbf{s}$, and $\mathcal{Q}$ gets an encryption of 0 using $\mathbf{s}$, namely $(\boldsymbol{\Psi}|\boldsymbol{\kappa})$ (notice that $\boldsymbol{\kappa}$ is of dimension $\lambda \times 1$, and hence is a single bit here). To derive a shared key, $\mathcal{Q}$ sends $\boldsymbol{\Psi}$ to $\mathcal{P}$. The shared key is thus $\boldsymbol{\kappa}$, which is known to $\mathcal{Q}$, and is computable by $\mathcal{P}$ as $\boldsymbol{\kappa} = \boldsymbol{\Psi} \cdot \mathbf{s}$.

Remark 2. For key spaces $\{0, 1\}^d$, we can simply tune the protocol to use $\lambda = d$, and that will yield a shared key $\hat{\mathbf{k}} \in \{0, 1\}^d$.

Theorem 6. *The two-party key-agreement protocol presented above is $(Cn^2, O(2^{-\alpha n/2}))$-secure against eavesdropping adversaries.*

Proof. First, by the Keeper security of the $\mathtt{EZ}$ protocol, for adversaries with up to Cn^2 memory bits for some $C < \frac{1}{20}$, $H_\infty(\mathbf{s}|\mathsf{view}_\mathcal{R}) \geq \Omega(\alpha n)$. Subsequently, $H_\infty(\boldsymbol{\Psi}, \mathbf{s}|\mathsf{view}_\mathcal{R}) \geq \Omega(\alpha n)$. Let $H : \{0, 1\}^n \times \{0, 1\}^n \to \{0, 1\}$ compute the inner product. Using the fact that the inner product is a universal hash function and applying Lemma 1, we have

$$(H(\boldsymbol{\Psi}, \mathbf{s}), H, \mathsf{view}_\mathcal{R}) \approx_{\epsilon/2} (U_1, U_d, \mathsf{view}_\mathcal{R}),$$

where $1 + 2\log(1/\epsilon) = \Omega(\alpha n)$. Solving for ϵ yields that $\epsilon = O(2^{-\alpha n/2})$, i.e. an adversary has advantage at most $O(2^{-\alpha n/2})$ in distinguishing $H(\boldsymbol{\Psi}, \mathbf{s})$ and U_1. Recall that in the eavesdropping security game for Two-Party Key-Agreement Protocols, the adversary need to distinguish between actual derived keys $\hat{k} =$

$\boldsymbol{\Psi} \cdot \mathbf{s}$ from random k' sampled directly from the key space $\{0,1\}$. Observe that $H(\boldsymbol{\Psi}, \mathbf{s}) = \boldsymbol{\Psi} \cdot \mathbf{s} = \hat{k}$, and k' is drawn from U_1. Therefore, we have

$$\begin{aligned} &|\Pr[\mathcal{A}(\hat{k}, \mathsf{trans}) = 1 : (\hat{k}, \mathsf{trans}) \leftarrow \Pi(n)] \\ &- \Pr[\mathcal{A}(k', \mathsf{trans}) = 1 : k' \leftarrow \mathcal{K}_n, (k, \mathsf{trans}) \leftarrow \Pi(n)]| \leq \epsilon = O(2^{-\alpha n/2}) \end{aligned}$$

as desired. □

6 Bit Commitment Scheme

Let n and λ be security parameters. A bit commitment scheme Π consists of a tuple of algorithm $(\texttt{Commit}, \texttt{Reveal}, \texttt{Verify})$ for a committer $\mathcal{C}$ and a verifier $\mathcal{V}$.

- The `Commit` algorithm is run by the committer, and it takes as input the security parameter n and a bit b to be committed to. A transcript of the communication, a committer state, and a verifier state $(\mathsf{trans}, \mathsf{state}_{\mathcal{C}}, \mathsf{state}_{\mathcal{V}}) \leftarrow \texttt{Commit}(n, \lambda, b)$ is output by the `Commit` algorithm.
- The `Reveal` algorithm is also run by the committer, and it takes as input a committer state $\mathsf{state}_{\mathcal{C}}$ and a bit b'. It outputs a revealing, denoted as x, together with the committed bit b'.
- The `Verify` algorithm is run by the Verifier and takes input a verifier state $\mathsf{state}_{\mathcal{V}}$ and outputs of a `Reveal` algorithm, (x, b'). It outputs a bit u.

There are two desired security properties for a bit commitment scheme, namely hiding and binding. We will give out formal definitions below.

The hiding property of a bit commitment scheme essentially states that the committed bit b should be hidden from the Verifier given the Verifier's view after the `Commit` algorithm. Notice that the Verifier's view after the `Commit` algorithm consists of exactly trans and $\mathsf{state}_{\mathcal{V}}$. Put formally:

Definition 7 (Hiding Property of a Bit Commitment Scheme). *For some given security parameters n, λ and a bit b, let $(\mathsf{trans}, \mathsf{state}_{\mathcal{C}}, \mathsf{state}_{\mathcal{V}}) \leftarrow \texttt{Commit}(n, \lambda, b)$, we say that the bit commitment scheme is $(S(n), \epsilon)$-hiding if for all Verifiers $\mathcal{V}$ with up to $S(n)$ memory bits,*

$$(b, \mathsf{trans}, \mathsf{state}_{\mathcal{V}}) \approx_{\epsilon} (r, \mathsf{trans}, \mathsf{state}_{\mathcal{V}})$$

for random r uniformly sampled from $\{0,1\}$.

The binding property of a bit commitment scheme essentially requires that a committer is not able to open a commitment to both 0 and 1. Notice that this applies to all committers, who can be potentially malicious. A malicious committer $\mathcal{A}$ can run an arbitrary Commit^* procedure, which has no guarantees except that it produces some $(\mathsf{trans}, \mathsf{state}_{\mathcal{A}}, \mathsf{state}_{\mathcal{V}})$. Note that this Commit^* procedure does not necessarily commit to a bit b, so it does not take b as a parameter.

Definition 8 (Binding Property of a Bit Commitment Scheme). *Let $\mathcal{A}$ be an adversary. $\mathcal{A}$ plays the following game* $\mathsf{Binding}_{\mathcal{A},\Pi}(n,\lambda)$ *for some given security parameters n and λ:*

- *The adversary $\mathcal{A}$ runs an arbitrary commit procedure (potentially malicious)* $\mathtt{Commit}^*(n,\lambda)$ *with an honest Verifier $\mathcal{V}$ and produces* $(\mathsf{trans}, \mathsf{state}_{\mathcal{A}}, \mathsf{state}_{\mathcal{V}})$.
- *The adversary produces* $(x_0, 0)$ *and* $(x_1, 1)$.
- *The game outputs* 1 *if both* $\mathtt{Verify}(\mathsf{state}_{\mathcal{V}}, (x_0, 0))$ *and* $\mathtt{Verify}(\mathsf{state}_{\mathcal{V}}, (x_1, 1))$ *output* 1*, and* 0 *otherwise.*

We say that Π is ϵ-binding if for all adversary $\mathcal{A}$

$$\Pr[\mathsf{Binding}_{\mathcal{A},\Pi}(n,\lambda) = 1] \leq \epsilon.$$

Now we present the construction for a bit commitment scheme using the Enhanced Encrypt Zero Protocol.

Construction 5 (Bit Commitment Scheme from Parity Learning). For security parameters n, λ and committer input bit b, we construct the bit commitment scheme by specifying each of the $(\mathtt{Commit}, \mathtt{Reveal}, \mathtt{Verify})$ algorithms.

- $\mathtt{Commit}(n, b)$: Runs the Enhanced Encrypt Zero Protocol $\mathtt{EZ}^+(n, 2n, \lambda)$ with $\mathcal{C}$ as the Keeper and $\mathcal{V}$ as the Recorder. Set trans to be the transcript of the $\mathtt{EZ}^+$ protocol, $\mathsf{state}_{\mathcal{C}}$ to be the output of $\mathcal{C}$ after the $\mathtt{EZ}^+$ protocol, i.e. a secret key $\mathbf{s}$, and $\mathsf{state}_{\mathcal{V}}$ to be the output of $\mathcal{V}$ after the $\mathtt{EZ}^+$ protocol, namely $(\boldsymbol{\Sigma}|\boldsymbol{\phi})$, which contains multiple encryptions of 0 under the key $\mathbf{s}$. Additionally, samples random $\boldsymbol{\gamma} \in \{0,1\}^{2n}$, and sends $(\boldsymbol{\gamma}, c = \boldsymbol{\gamma} \cdot \mathbf{s} + b)$ to the Verifier (notice that this also gets appended to trans).
- $\mathtt{Reveal}(\mathsf{state}_{\mathcal{C}}, b')$: Outputs $(\mathbf{x}, b') = (\mathbf{s}, b')$.
- $\mathtt{Verify}(\mathsf{state}_{\mathcal{V}}, \mathbf{x}, b')$: Checks that $\boldsymbol{\phi} = \boldsymbol{\Sigma} \cdot \mathbf{x}$, and that $c = \boldsymbol{\gamma} \cdot \mathbf{x} + b'$. If any of the checks fail, output 0; otherwise, output 1.

Theorem 7. *The bit commitment construction above is $(Cn^2, O(2^{-n/2}))$-hiding for some $C < 1/20$.*

Proof. First, by the Keeper security of the $\mathtt{EZ}^+$ protocol, for adversaries with up to Cn^2 memory bits for some $C < \frac{1}{20}$, $H_\infty(\mathbf{s}|\mathsf{view}_{\mathcal{V}}) \geq \Omega(n)$. Recall that $\mathsf{view}_{\mathcal{V}}$ is exactly $(\mathsf{trans}, \mathsf{state}_{\mathcal{V}})$. Subsequently, $H_\infty(\boldsymbol{\gamma}, \mathbf{s}|\mathsf{trans}, \mathsf{state}_{\mathcal{V}}) \geq \Omega(n)$. Let $H : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ compute the inner product. Using the fact that the inner product is a universal hash function and applying Lemma 1, we have

$$(H(\boldsymbol{\gamma}, \mathbf{s}), H, \mathsf{trans}, \mathsf{state}_{\mathcal{V}}) \approx_{\epsilon/2} (U_1, U_d, \mathsf{trans}, \mathsf{state}_{\mathcal{V}}),$$

where $1 + 2\log(1/\epsilon) = \Omega(n)$. Furthermore, we have

$$(H(\boldsymbol{\gamma}, \mathbf{s}) + c, H, \mathsf{trans}, \mathsf{state}_{\mathcal{V}}) \approx_{\epsilon/2} (U_1 + c, U_d, \mathsf{trans}, \mathsf{state}_{\mathcal{V}}),$$

Solving for ϵ yields that $\epsilon = O(2^{-n/2})$, i.e. an adversary has advantage at most $O(2^{-n/2})$ in distinguishing $H(\boldsymbol{\gamma}, \mathbf{s}) + c$ and $U_1 + c$. Notice that $H(\boldsymbol{\gamma}, \mathbf{s}) + c =$

$\boldsymbol{\gamma} \cdot \mathbf{s} + c = b$, and that $U_1 + c$ is yet another uniformly random bit $r \leftarrow \{0,1\}$. Therefore, we have

$$(b, H, \mathsf{trans}, \mathsf{state}_{\mathcal{V}}) \approx_{\epsilon/2} (r, U_d, \mathsf{trans}, \mathsf{state}_{\mathcal{V}})$$

for $\epsilon = O(2^{-n/2})$ and r a uniformly random bit. Thus, by

$$(b, \mathsf{trans}, \mathsf{state}_{\mathcal{V}}) \approx_{\epsilon'} (r, \mathsf{trans}, \mathsf{state}_{\mathcal{V}})$$

for $\epsilon' = \frac{1}{2}O(2^{-n/2}) = O(2^{-n/2})$ and r a uniformly random bit, we have shown that the bit commitment scheme presented above is $(Cn^2, O(2^{-n/2}))$-hiding as desired. □

Theorem 8. *The bit commitment scheme presented above is* $(2^{-\lambda})$*-binding.*

Proof. We show that the scheme is statistically binding by arguing that the probability that an adversary can win the Binding game is no more than $\frac{1}{2^\lambda}$.

Notice that in order for the adversary to win the game, the adversary need to output $(\mathbf{x}_0, 0)$ and $(\mathbf{x}_1, 1)$ that both pass the Verify algorithm. Recall that the Verify Algortihm checks for two things:

- $c = \boldsymbol{\gamma} \cdot \mathbf{x}_0 + 0$ and $c = \boldsymbol{\gamma} \cdot \mathbf{x}_1 + 1$ where c and $\boldsymbol{\gamma}$ are part of the transcipt trans and are stored in the Verifier's state $\mathsf{state}_{\mathcal{V}}$. This leads to that $\boldsymbol{\gamma} \cdot \mathbf{x}_0 \neq \boldsymbol{\gamma} \cdot \mathbf{x}_1$ and hence $\mathbf{x}_0 \neq \mathbf{x}_1$.
- $\boldsymbol{\phi} = \boldsymbol{\Sigma} \cdot \mathbf{x}_0 = \boldsymbol{\Sigma} \cdot \mathbf{x}_1$ where $\boldsymbol{\Sigma}$ and $\boldsymbol{\phi}$ are sampled and computed by the Verfier and stored in $\mathsf{state}_{\mathcal{V}}$. Notice this leads to $\boldsymbol{\Sigma} \cdot (\mathbf{x}_0 - \mathbf{x}_1) = 0$.

Now let $\mathbf{x}' = \mathbf{x}_0 - \mathbf{x}_1$. From $\mathbf{x}_0 \neq \mathbf{x}_1$, we know that $\mathbf{x}' \neq \mathbf{0}$. Therefore, we need to find a non-trivial root for the equation $\boldsymbol{\Sigma} \cdot \mathbf{x}' = \mathbf{0}$. Recall that by the Recorder's perfect security of the $\mathtt{EZ}^+$ protocol, the matrix $\boldsymbol{\Sigma}$ stored in $\mathsf{state}_{\mathcal{V}}$ is random conditioned on the Committer's view. For each row of $\boldsymbol{\Sigma}$, denoted as $\boldsymbol{\Sigma}_i$ for the i-th row, the probability that $\boldsymbol{\Sigma}_i \cdot \mathbf{x}' = 0$ is no more than a random guess, i.e. $\frac{1}{2}$. Since to pass the `Verify` algorithm requires $\boldsymbol{\Sigma} \cdot \mathbf{x}' = \mathbf{0}$, i.e. $\boldsymbol{\Sigma}_i \cdot \mathbf{x}' = 0$ for all $i = 1, 2, \ldots, \lambda$, and recall that the rows of $\boldsymbol{\Sigma}$ are independent, the probability that the adversary can find such a $\mathbf{x}'$ is no more than $(\frac{1}{2})^\lambda = \frac{1}{2^\lambda}$. □

7 Oblivious Transfer Protocol

In an oblivious transfer (OT) protocol, one party, the Sender $\mathcal{S}$, has two input bits x_0, x_1, and the other party, known as the Receiver $\mathcal{R}'$ (not to be confused with the Recorder $\mathcal{R}$ in the Encrypt Zero Protocols), has an input bit b. After some communication between the two parties, $\mathcal{R}'$ outputs x_b. The OT protocol requires two security properties, namely Sender security and Receiver security. Sender security dictates that $\mathcal{R}'$ should have no information about x_{1-b}, and Receiver security requires that $\mathcal{S}$ has no information about b.

Before we proceed to our construction of an OT protocol, we first formally define these two security properties.

The security of the Sender ensures that an adversarial Receiver can learn about at most one of x_0 and x_1. In other words, there always exists a b' s.t. the Receiver has no information about $x_{b'}$. Put formally:

Definition 9 (Sender Security). *An OT protocol is said to be ϵ-secure for the Sender if there exists some b' s.t. for any arbitrary distinguisher D and Receiver's view $\mathsf{view}_{\mathcal{R}'}$,*

$$\left|\Pr[D_{\mathsf{view}_{\mathcal{R}'}}(x_{b'}) = 1] - \Pr[D_{\mathsf{view}_{\mathcal{R}'}}(r) = 1]\right| \leq \epsilon$$

for a uniformly random bit r.

The security of the Receiver requires that the sender $\mathcal{S}$ has no information about b. In other words, given the view of the Sender, one should not be able to distinguish between b and a random bit r. Put formally:

Definition 10 (Receiver Security). *Let $\mathsf{view}_{\mathcal{S}}$ denote the view of the sender, the OT protocol Π is said to be $(S(n), \epsilon)$-secure for the Receiver if for all possible Senders that use up to $S(n)$ memory bits,*

$$(b, \mathsf{view}_{\mathcal{S}}) \approx_{\epsilon} (r, \mathsf{view}_{\mathcal{S}}),$$

where r is a uniformly random bit.

Now we give out our construction of the OT protocol.

The key idea is that the Receiver will send a commitment of its bit b to the Sender. And the Sender therefore uses the additive homomorphism of Raz's encryption scheme to compute the encryptions of $(1-b)x_0$ and bx_1. The Sender further re-randomizes these two ciphertexts by adding fresh encryptions of zero before sending them to the Receiver. The Receiver decrypts these two ciphertexts and obtains 0 and x_b as desired.

Construction 6 (Oblivious Transfer Protocol from Parity Learning). For given security parameter n, a Sender $\mathcal{S}$ and a receiver $\mathcal{R}'$:

- Run an Enhanced Encrypt Zero Protocol $\mathtt{EZ}^{+}(n, 2n, \lambda = 2)$ with $\mathcal{R}$' as the Keeper and $\mathcal{S}$ as the Recorder. At the end of the protocol, $\mathcal{R}'$ has as output a secret key $\mathbf{s}$, and $\mathcal{S}$ has output $(\boldsymbol{\Sigma}|\boldsymbol{\phi})$, which consists of two encryptions of 0 under the key $\mathbf{s}$. Additionally, $\mathcal{R}'$ samples random $\boldsymbol{\gamma} \in \{0,1\}^{2n}$, and sends $(\boldsymbol{\gamma}, c = \boldsymbol{\gamma} \cdot \mathbf{s} + b)$ to the Sender. Kindly notice that in this step the Receiver $\mathcal{R}'$ is actually just executing $\mathtt{Commit}(n, b)$.
- For Sender $\mathcal{S}$, let $\boldsymbol{\sigma}_0$, $\boldsymbol{\sigma}_1$ be the first and second row of $\boldsymbol{\Sigma}$, and ϕ_0, ϕ_1 be the two elements in $\boldsymbol{\phi}$. Notice that $\phi_0 = \boldsymbol{\sigma}_0 \cdot \mathbf{s}$ and $\phi_1 = \boldsymbol{\sigma}_1 \cdot \mathbf{s}$. The Sender then sends to the Receiver two ciphertexts:

$$\begin{aligned} \boldsymbol{\sigma}_0 + x_0\boldsymbol{\gamma},\ \phi_0 + x_0(1-c) &= (\boldsymbol{\sigma}_0 + x_0\boldsymbol{\gamma}) \cdot \mathbf{s} + \big((1-b)x_0\big) \\ \boldsymbol{\sigma}_1 + x_1\boldsymbol{\gamma},\ \phi_1 + x_1 c &= (\boldsymbol{\sigma}_1 + x_1\boldsymbol{\gamma}) \cdot \mathbf{s} + \big(bx_1\big). \end{aligned}$$

– $\mathcal{R}'$ decrypts both ciphertexts that it has received using the key $\mathbf{s}$, and learns $(1-b)x_0$ and bx_1. Notice that one of these two values will be x_b as desired and gets output by $\mathcal{R}'$.

We then proceed to prove desired security properties for the above construction of the OT protocol.

Theorem 9. *The OT protocol described above is perfectly secure for the Sender.*

Proof. We show that right after the first part of the protocol where $\mathcal{R}'$ executes $\texttt{Commit}(n, b)$, there is a fixed $b' = c + \boldsymbol{\gamma} \cdot \mathbf{s} + 1$ such that the Receiver will have no information about $x_{b'}$. Notice that this does not break Receiver security, since although b' is fixed, $\mathcal{S}$ has no way to compute b' as $\mathbf{s}$ is only known to the Receiver $\mathcal{R}'$.

If $b' = c + \boldsymbol{\gamma} \cdot \mathbf{s} + 1 = 0$, we show that the Receiver has no information about x_0, i.e. x_0 is random given the Receiver's view. Notice that we have $1 - c = \boldsymbol{\gamma} \cdot \mathbf{s}$. And hence the two ciphertext that the Receiver receives are

$$\begin{aligned} \boldsymbol{\sigma}_0 + x_0\boldsymbol{\gamma},\ \phi_0 + x_0(1-c) &= (\boldsymbol{\sigma}_0 + x_0\boldsymbol{\gamma}) \cdot \mathbf{s} \\ \boldsymbol{\sigma}_1 + x_1\boldsymbol{\gamma},\ \phi_1 + x_1 c &= (\boldsymbol{\sigma}_1 + x_1\boldsymbol{\gamma}) \cdot \mathbf{s} + x_1. \end{aligned}$$

The only source that the Receiver might be able to gather information about x_0 is from the first ciphertext. However, since $\boldsymbol{\sigma}_0$ is uniformly random given the Receiver's view, $\boldsymbol{\sigma}_0 + x_0\boldsymbol{\gamma}$ is also uniformly random given the Receiver's view, i.e., it does not give any additional information to the Receiver. The Receiver also gets no information from $(\boldsymbol{\sigma}_0 + x_0\boldsymbol{\gamma}) \cdot \mathbf{s}$, as this value can be easily simulated by the Receiver since it knows both $\boldsymbol{\sigma}_0 + x_0\boldsymbol{\gamma}$ and $\mathbf{s}$. Therefore, x_0 is random given the Receiver's view.

If $b' = c + \boldsymbol{\gamma} \cdot \mathbf{s} + 1 = 1$, by a similar argument, we have that x_1 is random given the Receiver's view. Bringing these parts together, we have shown that for $b' = c + \boldsymbol{\gamma} \cdot \mathbf{s} + 1$, $x_{b'}$ *is* random conditioned on the Receiver's view, i.e.

$$\left|\Pr[D_{\mathsf{view}_{\mathcal{R}'}}(x_{b'}) = 1] - \Pr[D_{\mathsf{view}_{\mathcal{R}'}}(r) = 1]\right| = 0.$$

Thus, the OT protocol above is perfectly secure for the Sender as desired. □

Theorem 10. *The OT protocol described above is $(Cn^2, O(2^{-n/2}))$-secure for the Receiver, for some $C < \frac{1}{20}$.*

Proof. The proof for this is extremely straightforward. As observed above, the receiver $\mathcal{R}'$ is exactly executing $\texttt{Commit}(n, b)$, i.e. it is committing the bit b to the Sender, who is playing the role of the Verifier in the commitment scheme. Hence, by the $(Cn^2, O(2^{-n/2}))$-hiding property of the commitment scheme, we have that for all possible Sender $\mathcal{S}$ that uses at most Cn^2 memory bits,

$$(b, \mathsf{trans}, \mathsf{state}_{\mathcal{S}}) \approx_\epsilon (r, \mathsf{trans}, \mathsf{state}_{\mathcal{S}})$$

for $\epsilon = O(2^{-n/2})$ and a uniformly random bit r. Notice that $\mathsf{view}_{\mathcal{S}}$ is actually just $(\mathsf{trans}, \mathsf{state}_{\mathcal{S}})$. Therefore, the above equation can be rewritten as

$$(b, \mathsf{view}_{\mathcal{S}}) \approx_\epsilon (r, \mathsf{view}_{\mathcal{S}}).$$

This is the exact definition for (Cn^2, ϵ)-Receiver-security. Therefore, the OT protocol above is $(Cn^2, O(2^{-n/2}))$-secure for the Receiver as desired. □

References

[AR99] Yonatan Aumann and Michael O. Rabin. Information theoretically secure communication in the limited storage space model. In Michael J. Wiener, editor, *CRYPTO'99*, volume 1666 of *LNCS*, pages 65–79. Springer, Heidelberg, August 1999.

[BDKM18] Marshall Ball, Dana Dachman-Soled, Mukul Kulkarni, and Tal Malkin. Non-malleable codes from average-case hardness: AC^0, decision trees, and streaming space-bounded tampering. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 618–650. Springer, Heidelberg, April / May 2018.

[CCM98] Christian Cachin, Claude Crépeau, and Julien Marcil. Oblivious transfer with a memory-bounded receiver. In *39th FOCS*, pages 493–502. IEEE Computer Society Press, November 1998.

[CM97] Christian Cachin and Ueli M. Maurer. Unconditional security against memory-bounded adversaries. In Burton S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 292–306. Springer, Heidelberg, August 1997.

[DHRS04] Yan Zong Ding, Danny Harnik, Alon Rosen, and Ronen Shaltiel. Constant-round oblivious transfer in the bounded storage model. In Moni Naor, editor, *TCC 2004*, volume 2951 of *LNCS*, pages 446–472. Springer, Heidelberg, February 2004.

[Din01] Yan Zong Ding. Oblivious transfer in the bounded storage model. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 155–170. Springer, Heidelberg, August 2001.

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *21st ACM STOC*, pages 25–32. ACM Press, May 1989.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *28th ACM STOC*, pages 212–219. ACM Press, May 1996.

[ILL89] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstracts). In *21st ACM STOC*, pages 12–24. ACM Press, May 1989.

[Lu02] Chi-Jen Lu. Hyper-encryption against space-bounded adversaries from on-line strong extractors. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 257–271. Springer, Heidelberg, August 2002.

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

[Raz17] Ran Raz. A time-space lower bound for a large class of learning problems. In *58th FOCS*, pages 732–742. IEEE Computer Society Press, 2017.

[Rot11] Ron Rothblum. Homomorphic encryption: From private-key to public-key. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 219–234. Springer, Heidelberg, March 2011.

[Sho94] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134. IEEE Computer Society Press, November 1994.