Trapdoor Hash Functions and Their Applications

Nico Döttling[1], Sanjam Garg[2], Yuval Ishai[3], Giulio Malavolta[4⋆], Tamer Mour[⋆⋆5], and Rafail Ostrovsky[6]

[1] CISPA Helmholtz Center for Information Security, Saarbrücken, Germany, doettling@cispa.saarland
[2] UC Berkeley, Berkeley CA, USA, sanjamg@berkeley.edu
[3] Technion, Haifa, Israel, yuvali@cs.technion.ac.il
[4] Carengie Mellon University, Pittsburgh PA, USA, giulio.malavolta@hotmail.it
[5] Weizmann Institute of Science, Rehovot, Israel, tamer@weizmann.ac.il
[6] UCLA, Los Angeles CA, USA, rafail@cs.ucla.edu

Abstract. We introduce a new primitive, called *trapdoor hash functions* (TDH), which are hash functions $\mathsf{H} : \{0,1\}^n \to \{0,1\}^\lambda$ with additional trapdoor function-like properties. Specifically, given an index $i \in [n]$, TDHs allow for sampling an encoding key ek (that hides i) along with a corresponding trapdoor. Furthermore, given $\mathsf{H}(x)$, a hint value $\mathsf{E}(\mathsf{ek}, x)$, and the trapdoor corresponding to ek, the i^{th} bit of x can be efficiently recovered. In this setting, one of our main questions is: How small can the hint value $\mathsf{E}(\mathsf{ek}, x)$ be? We obtain constructions where the hint is only *one bit* long based on DDH, QR, DCR, or LWE.

This primitive opens a floodgate of applications for low-communication secure computation. We mainly focus on two-message protocols between a receiver and a sender, with private inputs x and y, resp., where the receiver should learn $f(x, y)$. We wish to optimize the *(download) rate* of such protocols, namely the asymptotic ratio between the size of the output and the sender's message. Using TDHs, we obtain:

1. The first protocols for (two-message) *rate-1 string OT* based on DDH, QR, or LWE. This has several useful consequences, such as:
   (a) The first constructions of PIR with communication cost poly-logarithmic in the database size based on DDH or QR. These protocols are in fact rate-1 when considering block PIR.
   (b) The first constructions of a *semi-compact* homomorphic encryption scheme for branching programs, where the encrypted output grows only with the program *length*, based on DDH or QR.
   (c) The first constructions of lossy trapdoor functions with input to output ratio approaching 1 based on DDH, QR or LWE.
   (d) The first *constant-rate* LWE-based construction of a 2-message "statistically sender-private" OT protocol in the plain model.
2. The first *rate-1* protocols (under any assumption) for n parallel OTs and matrix-vector products from DDH, QR or LWE.

We further consider the setting where f evaluates a RAM program y with running time $T \ll |x|$ on x. We obtain the first protocols with communication sublinear in the size of x, namely $T \cdot \sqrt{|x|}$ or $T \cdot \sqrt[3]{|x|}$, based on DDH or, resp., pairings (and correlated-input secure hash functions).

⋆ Part of the work done while at Friedrich-Alexander-Universität Erlangen-Nürnberg.
⋆⋆ Part of the work done while at Technion.

1 Introduction

Seminal results from the 1980s [55, 31] showed that it is possible for a group of mutually distrustful parties to compute a joint function on their private inputs without revealing anything more than the output of the computation. These foundational results were seen as providing the first theoretical proofs of concept. However, significant theoretical and practical advances over the years provide support for the idea that perhaps secure computation can be as practical and ubiquitous as public-key cryptography.

In the quest to make secure computation efficient, realizing *communication efficient* secure computation protocols has emerged as a central theme of research. Moreover, secure computation protocols with large communication cost can often be prohibitive in practice. Consequently, substantial effort has been put towards realizing communication efficient protocols. Nonetheless, our understanding of communication efficient secure computation protocols remains significantly limited. Specifically, known protocols for circuits with communication independent of the circuit size are only known using fully homomorphic encryption (FHE) [27] and can only be based on variants of LWE. In the two-party case, the communication complexity of such protocols is comparable to the length of the *shorter input* plus the length of the output. For simpler functions that can be represented by log-depth circuits or polynomial-size branching programs, similar protocols were recently constructed from other assumptions such as DDH [9] or a circular-secure flavor of DCR [23]. Here the communication is comparable to the total length of *both inputs* plus the length of the output.

The above state of affairs leaves several types of gaps between secure and insecure communication complexity.[1] First, even when applying the best known FHE schemes, there is a constant-factor gap for functions whose output length is comparable to (or longer than) the length of the shorter input.[2] Second, the communication gap can be even bigger when considering restricted interaction. For instance, when one input is much shorter than the other, FHE cannot be used to get communication-efficient 2-message protocols where the party holding the *long* input sends the first message. Finally, and most importantly for this work, under standard assumptions other than LWE, the gaps between secure and insecure communication are much bigger, especially when considering functions with unbalanced input sizes.

To illustrate the current gaps, consider the fundamental problem of private information retrieval (PIR) [44, 17] over m-bit records, where a client wants to privately learn the i^{th} record of a server's database that consists of n records of length m each. Here, a protocol that achieves near-optimal communication from the server to the client (i.e., roughly m bits) is only known under DCR [18, 45].

[1] It seems plausible that these gaps can be closed using indistinguishability obfuscation [25]. However, the focus of this work is on constructions that can be based on more traditional assumptions.

[2] A simple "hybrid FHE" technique [28] can generically convert any FHE scheme into one whose encrypted (long) input is roughly as long as the input. However, no such generic technique is known for compressing the encrypted output.

For the case of retrieving m different 1-bit records, the situation is even worse. In the best known protocol, the gap between the server's message length and the output length is a big multiplicative constant [33]. Finally, even for the case $m = 1$, obtaining polylog(n) communication under (subexponential variants of) standard assumptions such as DDH or QR is open.

1.1 Our Setting and Questions of Interest

Setting: Two-Message Secure Computation. We consider two-party protocols in which a receiver and a sender have private inputs x and y, respectively. We consider protocols for evaluating a function $f(x, y)$ using only two messages. First, based on its input x, the receiver sends the first message msg_1 to the sender who, based on its input y, responds with the second message msg_2. Finally, the receiver uses its secret state and msg_2 to compute $f(x, y)$. *Sender's privacy* requires that the receiver learns nothing more about y than $f(x, y)$ and the length of y. Similarly, *receiver's privacy* requires that the sender learns nothing about x other than its length. By default, we only consider security against *semi-honest* parties.[3]

Case I: Large Receiver Output. We are primarily interested in the case where the output of f is long, and define the *download rate* of such a 2-message protocol (or *rate* for short) as the asymptotic ratio between $|f(x, y)|$ and $|\mathsf{msg}_2|$. We will also consider the *overall rate*, defined as the asymptotic ratio between the insecure communication complexity of f and that of the protocol. A fundamental functionality in this regime is oblivious transfer (OT). We start with the special case of *string OT*, implemented via a two-message protocol. Recall that in the string OT functionality the inputs of the sender and receiver are two strings $\mathsf{s}_0, \mathsf{s}_1 \in \{0, 1\}^n$ and a bit $i \in \{0, 1\}$, respectively. For this functionality, the receiver's output should be s_i. Here the download and overall rate are the asymptotic ratios $\frac{n}{|\mathsf{msg}_2|}$ and $\frac{n}{|\mathsf{msg}_1|+|\mathsf{msg}_2|}$, respectively, when the security parameter λ tends to infinity and n is a sufficiently big polynomial in λ (see Definition 3.1 for a precise formulation). We also consider *batch OT*; this functionality allows n parallel instances of bit-OT (string OT with 1-bit strings).

Even for the special case of OT, the state-of-the-art with optimal overall rate (or optimal download rate) is quite unsatisfactory.[4] Any 2-message string-OT protocol can be compiled into a similar protocol with rate 1/2 using *hybrid encryption* as follows: Given a string-OT protocol for messages of size λ, the sender uses the OT protocol to transmit one out of two symmetric keys to the receiver,

[3] Our protocols can be efficiently extended to provide security against malicious parties (under the same assumptions) using sublinear arguments [46]. This increases the number of rounds, but does not affect the asymptotic communication.

[4] The work of Cho et al. [16] on *laconic OT* gives a batch OT protocol where $|\mathsf{msg}_1|$ is independent of $|x|$. This generalizes to arbitrary functions f; however, even in the simple case of batch OT the download rate is sub-constant. The same applies to the more recent work of Quach et al. [52] on laconic function evaluation.

and uses these keys with a rate-1 symmetric encryption scheme to encrypt the actual messages. The two ciphertexts are sent along with the OT sender message. The receiver recovers one of the two keys and decrypts the corresponding ciphertext. However, going beyond rate 1/2 seems to hit barriers! Interestingly enough, for information-theoretic OT in the correlated randomness model, rate 1/2 (as e.g. in Beaver's standard reduction [6]) is optimal [54, 38]. In the computational setting, constructions based on additively homomorphic encryption or homomorphic secret sharing hit a similar barrier [53, 10, 42]. Currently, the only construction of OT known to achieve rate better than 1/2 is based on the Damgård-Jurik cryptosystem [18], which relies on the DCR assumption. Even here, optimal rate in only achieved by undesirably letting the size of the group used in the scheme grow with the size of the inputs.[5] Moreover, in the more general case of *batch OT*, rate 1 could not even be achieved based on DCR. This brings us to our first motivating question:

Can we realize OT with rate $> \frac{1}{2}$ *from assumptions other than DCR?*
Can we realize such batch OT from any assumption?

Why care about OT with rate $> \frac{1}{2}$*?* As mentioned earlier, there is a large body of work on minimizing the communication complexity of secure computation. The special case of OT is not only natural and useful as a standalone application, but it also serves as a stepping stone for other applications. Indeed, high-rate 2-message OT implies: (i) high-rate PIR with polylogarithmic communication complexity in the number of records [44, 40]; (ii) a semi-compact homomorphic encryption scheme that supports evaluation of bounded-length branching programs (in particular, finite automata, decision trees and OBDDs) over encrypted data [40], (iii) a high-rate lossy-trapdoor function [51], and (iv) statistically sender-private (SSP) two-message OT with constant rate [4, 11]. We will elaborate on these applications below. To sum up, while high-rate OT is a powerful primitive with important consequences, very little is known about how to construct it.

Case II: Large Receiver Input. Up to this point, we were mainly concerned with functions $f(x, y)$ that have a long output, where our goal was to make the communication from the sender to the receiver very close to $|f(x, y)|$. Even multi-round protocols of this type were not known prior to our work. A second setting we consider applies to two-round protocols in the case where $|x| \gg |y|$ and the output is short. In this case, an insecure protocol for f can simply have the sender communicate y to the receiver. Since secure computation with only one message is impossible (except in trivial cases), our goal is to obtain a *two-message* secure protocol with similar efficiency features, namely where the total communication complexity is comparable to $|y|$ rather than $|x|$. As a motivating example, consider the case where the receiver has a large n-bit

[5] In this work, we consider this question in the more stringent two-message setting. However, we note that no other protocols with rate $> \frac{1}{2}$ are known even when additional rounds of communication are allowed.

database x, the sender's input y describes a small RAM machine M whose running time is $T \ll n$, and the receiver's output is $M(x)$. For instance, M can select a single entry $y \in [n]$ of x, outputting $M(x) = x_y$. Note that a natural FHE-based protocol where the receiver sends an encryption of x and receives an encryption of $M(x)$ does not meet our efficiency goal of using less than n bits of communication. On the other hand, allowing for a higher round complexity, our goal can be met using any PIR protocol [46].

From here on, we restrict the attention to 2-message protocols with $o(n)$ communication. The recent *laconic function evaluation* primitive [52] provides such a protocol with overall communication of $|y| + \mathsf{poly}(\lambda, T)$, where λ is a security parameter. However, results in this setting are only known under LWE (with subexponential modulus-to-noise ratio). This brings us to our second main question:

Are there 2-message protocols computing $M(x)$ with $o(n)$ bits of communication from any assumptions other than LWE?

1.2 Our Results

In this work, we introduce a new primitive that we call trapdoor hash functions (TDHs).[6] TDHs are hash functions $\mathsf{H} : \{0,1\}^n \to \{0,1\}^\lambda$ with additional trapdoor-function-like properties. Specifically, given an index $i \in [n]$, TDHs allow for sampling an encoding key ek (that hides i) along with a corresponding trapdoor. Furthermore, given $\mathsf{H}(x)$, a hint value $\mathsf{E}(\mathsf{ek}, x)$, and the trapdoor corresponding to ek, the i^{th} bit of x can be efficiently recovered. In this setting, one of our main questions is: how small can the hint value $\mathsf{E}(\mathsf{ek}, x)$ be? We define the rate of TDH as the inverse of the size of the hint.

We obtain constructions of rate-1 TDHs from standard assumptions, namely DDH, QR, DCR, and LWE. The surprising twist in these constructions is the close integration of techniques developed in two very recent and seemingly unrelated lines of investigation: (i) A new type of hash function for constructing identity-based encryption by Döttling and Garg [21] and its extension to constructions of trapdoor functions by Garg, Gay and Hajiabadi [26, 24] and (ii) techniques for homomorphic secret sharing by Boyle, Gilboa and Ishai [9].

Main Result: Rate-1 Two-Message String OT. Our TDHs yield the first construction of string OT with rate-1 from the {DDH, QR, LWE} assumption. Additionally, we get a new construction under DCR, for which, unlike the Damgård-Jurik construction [18], the size of the group used is independent of the size of the inputs. We stress that while our constructions use only two messages; previously, even multi-round constructions with rate $> \frac{1}{2}$ were not known under these assumptions.[7] This allows us to obtain the following new results:

[6] The notion of trapdoor hash functions is inspired by the closely related notion of hash encryption [21, 22, 13, 26] and somewhere statistically binding hash functions [36, 43, 49].

[7] Our protocols achieve asymptotic download rate 1, which is clearly optimal. However, the (additive) difference between the sender's message length and the output length

1. *Private Information Retrieval*: We obtain the first constructions of private information retrieval (PIR) from {DDH, QR, LWE} with download rate 1 (for retrieving long records). The total communication complexity grows only logarithmically with the number of records.[8] Previously, such PIR protocols were only known under DCR [45]. This also resolves the longstanding open question of building PIR with polylogarithmic communication (for 1-bit records) from {DDH, QR} [44]. Such protocols were only known under DCR, LWE, and the Phi-hiding assumptions [14, 45, 15, 50]. For example, the best known construction from DDH required $O(2^{\sqrt{\log n}} \cdot \lambda)$ bits of communication, for database size n and security parameter λ [44, 53].
2. *Semi-Compact Homomorphic Encryption for Branching Programs:* We obtain the first encryption schemes based on {DDH, QR} that allow evaluating a branching program on an encrypted input, where the encrypted output grows only with the *length* of the branching program and not with its size. Previously, such schemes were only known under {DCR, LWE} [40].
3. *Lossy Trapdoor Functions:* We obtain the first construction of lossy trapdoor functions [51] with rate 1 from the {DDH, QR, LWE} assumption. Here, rate is defined as the ratio of the input length and output length for the trapdoor function. Very recently, Garg et al. [24] obtained construction from DDH with a small constant rate. However, besides that, no constructions with constant rate were known under these assumptions.
4. *Malicious Statistically Sender Private OT:* We obtain the first LWE-based 2-message OT protocol in the plain model that offers statistical sender privacy against a malicious receiver and has a *constant* rate. This improves over the $1/\log(\lambda)$ rate of a recent LWE-based protocol of Brakerski and Döttling [11]. Similar protocols were previously known under {DDH, DCR} [48, 1, 34].

Rate-1 Protocols for Functionalities Generalizing OT. We generalize the techniques for rate-1 OT to yield secure 2-message protocols with *download* rate 1 for other useful functionalities. In these cases, we obtain the first protocols under *any* assumption. We obtain such protocols for the following functionalities.

1. *Batch OT:* Batch OT allows n instances of bit-OT to be performed in parallel. We obtain 2-message batch OT protocol with download rate 1 (but sub-constant upload rate) from QR and LWE. Allowing for inverse polynomial error probability, we obtain a similar protocol under DDH. Protocols with smaller constant download rates (and constant overall rate) were known under a variety of assumptions; see [39, 10, 8] and references therein.
2. *Batch OLE:* An oblivious linear function evaluation (OLE) scheme allows the sender to evaluate an affine function $f(x) = ax + b$ over the receiver's

grows with the security parameter λ. In the full version we show that that this gap is necessary even in the more liberal setting of secure computation with preprocessing.

[8] More specifically, as our group-based constructions are black-box in the underlying group, we can count the communication complexity in terms of the number of group elements, which in our case is $\log n \cdot \mathsf{poly}(\lambda)$, where n is the size of the database and λ is the security parameter.

private input x. We obtain the first batch OLE (over either a field of a small characteristic or smooth modulus) with download rate 1 based on QR or LWE. We also get a DDH-based construction if we allow inverse-polynomial error. For the case of fields, smaller constant download rate (and constant overall rate) could be realized under LWE [20, 42] or code-based assumptions [47, 41, 2].

3. *Matrix-Vector Products:* We generalize the above to oblivious matrix-vector product evaluation (OMV), where the sender has a matrix M, the receiver holds a vector v, and the output is Mv. A two-message OMV protocol can be thought as a relaxed form of additively homomorphic encryption. Our techniques can be generalized to construct OMVs over $\mathbb{F}_2$ with optimal download rate, based on QR or LWE. We can also generalize the LWE-based construction to fields modulo small primes or smooth integers. Compared to previous LWE-based constructions (e.g., [42]), we get better (optimal) download rate but worse overall rate.

As mentioned for rate-1 OT, all the aforementioned results were known only under the DCR assumption (and in the case of functionalities generalizing OT, were not known under any standard assumptions), where optimal rate was achieved by letting the size of the group grow with the size of the inputs. Our work improves in this setting. Specifically, assuming only DCR, our work implies all of the above results in groups of size independent of the message length.

As in the context of rate-1 OT, while we consider only two-message protocols, we stress that, prior to our work, none of the above-mentioned results were known even when additional rounds of communication are allowed.

Beyond OT: Two-Message SFE with Sublinear Communication. Armed with our new techniques, we attempt to broaden the class of functionalities for which two-message secure-function evaluation (SFE) can be achieved with sublinear communication. Specifically, we start with the following example setting: Alice would like to share her DNA sequence online so that various medical researchers can use it to provide her with valuable insights about her health. However, Alice wants to keep her "large" genetic information confidential and each researcher wants to keep the specific parts of the genetic code it looks at private. In a bit more detail, Alice wants to publish a hash $h(\mathsf{x})$ of her input x (of length n) online, such that any contractor Bob, with a private machine M with "small running time" (denoted by T) can send Alice a "short" message, enabling her to learn M^{x}, where M has random access to x. In summary, we are interested in a setting that allows Alice to evaluate Bob's private *small machine* on her private *large input* with sublinear communication.

Positive results for the above setting with sublinear communication are only known from lattice assumptions — namely, using laconic function evaluation [52]. In contrast, for the case of DDH-based constructions, such protocols need communication complexity proportional to n. We note that constructions based on

laconic OT [16] do not keep the locations accessed by M private and thus, do not suffice for this application.[9]

We obtain the first protocol for non-interactive secure computation on large inputs from DDH with communication proportional to $T \cdot \sqrt{n}$, where T is the running time of the machine and n is the size of the database. Furthermore, using pairings (and appropriate correlated-input secure hash functions [37, 30, 7, 32, 3]) we obtain a protocol with communication cost proportional to $T \cdot \sqrt[3]{n}$.

Further, in a scenario where Bob's machine M is repeatedly executed over different large inputs (possibly owned by different Alices), we achieve protocols with communication proportional to T, and *independent* of n, per execution, assuming a non-interactive "offline phase" where Bob publishes an "encoding" of M of length proportional to n or $\sqrt{n}$ (from DDH or pairings, resp.), which can be amortized over all executions.

Our results are obtained by constructing a variant of laconic OT [16], that keeps the locations accessed by M private. We call this primitive *private laconic OT*. The key technical challenge here is to realize this primitive with communication cost sublinear in the size of Alice's large input. By using private laconic OT, rather than laconic OT, in the constructions from in [16], we obtain SFE for RAM programs with sublinear communication which, as opposed to the protocol from [16], also hides the access pattern made by the machine to the input database and therefore achieves a full notion of security.

1.3 Concurrent Work

In a concurrent work, Gentry and Halevi [29] constructed an efficient rate-1 FHE schemes from LWE, which in particular also yield rate-1 OT constructions. When instantiated from LWE with polynomial modulus-to-noise ratio, their construction achieves rate $1 - \epsilon$ for any constant ϵ. In comparison, our OT constructions achieve rate $1 - 1/\lambda$ in this regime and can also be based on DDH or QR.

1.4 Paper Organization

In the following sections, we give a high level overview of the technical contributions of our work. We first introduce trapdoor hash functions, and present the ideas behind our constructions from the different assumptions. We then proceed to discuss the applications of trapdoor hash. More technical details and full formal analysis are provided in the full version.

2 Trapdoor Hash Functions

We start by providing a notational framework for the new primitive, then give an overview of our constructions.

[9] For this application, we insist on the two-message setting. Allowing $O(T)$ rounds of interaction, similar protocols can be based on any single-server PIR scheme [46].

2.1 Defining Trapdoor Hash

A *trapdoor hash scheme (TDH)* defines a family of samplable publicly-parameterized hash functions $\mathsf{H}_{\mathsf{hk}} : \{0,1\}^n \to \{0,1\}^\eta$, accompanied with the following three algorithms:

- *Key generation:* given the public hash key, Bob generates a pair of an *encoding key* and a *trapdoor* $(\mathsf{ek}, \mathsf{td}) \leftarrow \mathsf{G}(\mathsf{hk}, i)$, corresponding to a private index $i \in [n]$.
- *Encoding:* using the encoding key ek, Alice, with a private input $\mathsf{x} \in \{0,1\}^n$, can compute a *hint* $\mathsf{e} \leftarrow \mathsf{E}(\mathsf{ek}, \mathsf{x})$, which essentially encodes the bit $\mathsf{x}[i]$.
- *Decoding:* Bob, who has the secret trapdoor td, can now decode any encoding e generated for some input x as above, to recover x_i, given only the hash $\mathsf{H}_{\mathsf{hk}}(\mathsf{x})$. In fact, Bob would be able to generate two encodings $(\mathsf{e}_0, \mathsf{e}_1) \leftarrow \mathsf{D}(\mathsf{td}, \mathsf{h})$, where it is guaranteed that $\mathsf{e} = \mathsf{e}_{\mathsf{x}[i]}$.

We actually consider a more general notion of TDH where Bob with a private *predicate* $f : \{0,1\}^n \to \{0,1\}$, chosen from a predefined class of predicates $\mathcal{F}$, generates a key ek, using which Alice encodes the bit $f(x)$, and a corresponding trapdoor, using which Bob decodes. Such a scheme is called trapdoor hash for $\mathcal{F}$, and the above special case is referred to as trapdoor hash for *index predicates*.

Definition 2.1 (Trapdoor Hash Scheme). *Let $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ be a class of predicates, where each $\mathcal{F}_n$ is a set of predicates defined over over $\{0,1\}^n$, and let $\omega := \omega(\lambda) \in \mathbb{N}$ for any $\lambda \in \mathbb{N}$. A* rate-$\frac{1}{\omega}$ trapdoor hash scheme (TDH) for $\mathcal{F}$ *is a tuple of five PPT algorithms $\mathsf{TDH} = (\mathsf{S}, \mathsf{G}, \mathsf{H}, \mathsf{E}, \mathsf{D})$ with the following properties.*

- ***Syntax:***
  - $\mathsf{hk} \leftarrow \mathsf{S}(1^\lambda, 1^n)$. *The sampling algorithm takes as input a security parameter λ and an input length n, and outputs a hash key hk.*
  - $(\mathsf{ek}, \mathsf{td}) \leftarrow \mathsf{G}(\mathsf{hk}, f)$. *The generating algorithm takes as input a hash key hk and a predicate $f \in \mathcal{F}_n$, and outputs a pair of an encoding key ek and a trapdoor td.*
  - $\mathsf{h} \leftarrow \mathsf{H}(\mathsf{hk}, \mathsf{x}; \rho)$. *The hashing algorithm takes as input a hash key hk, a string $\mathsf{x} \in \{0,1\}^n$ and randomness $\rho \in \{0,1\}^*$, and deterministically outputs a hash value $\mathsf{h} \in \{0,1\}^\eta$.*
  - $\mathsf{e} \leftarrow \mathsf{E}(\mathsf{ek}, \mathsf{x}; \rho)$. *The encoding algorithm takes as input an encoding key ek, string $\mathsf{x} \in \{0,1\}^n$ and randomness $\rho \in \{0,1\}^*$, and deterministically outputs an encoding $\mathsf{e} \in \{0,1\}^\omega$.*
  - $(\mathsf{e}_0, \mathsf{e}_1) \leftarrow \mathsf{D}(\mathsf{td}, \mathsf{h})$. *The decoding algorithm takes as input a trapdoor td, a hash value $\mathsf{h} \in \{0,1\}^\eta$, and outputs a pair of a 0-encoding and a 1-encoding $(\mathsf{e}_0, \mathsf{e}_1) \in \{0,1\}^\omega \times \{0,1\}^\omega$.*
- ***Correctness:*** TDH *is* $(1-\epsilon)$-correct *(or has ϵ error probability), for $\epsilon := \epsilon(\lambda) < 1$, if the following holds for any $\lambda, n \in \mathbb{N}$, any $\mathsf{x} \in \{0,1\}^n$ and any predicate $f \in \mathcal{F}_n$.*

$$\Pr[\mathsf{e} = \mathsf{e}_{f(x)}] \geq 1 - \mathsf{negl}(\lambda) \qquad \Pr[\mathsf{e} \neq \mathsf{e}_{1-f(x)}] \geq 1 - \epsilon - \mathsf{negl}(\lambda)$$

where $\mathsf{hk} := \mathsf{S}(1^\lambda, 1^n)$, $(\mathsf{ek}, \mathsf{td}) := \mathsf{G}(\mathsf{hk}, f)$, $\mathsf{h} := \mathsf{H}(\mathsf{hk}, \mathsf{x}; \rho)$ and $\mathsf{e} := \mathsf{E}(\mathsf{ek}, \mathsf{x}; \rho)$ for $\rho \xleftarrow{\$} \{0,1\}^*$, and $(\mathsf{e}_0, \mathsf{e}_1) := \mathsf{D}(\mathsf{td}, \mathsf{h})$.

– ***Function Privacy:*** TDH *is* function-private *if for any polynomial-length* $\{1^{n_\lambda}\}_{\lambda\in\mathbb{N}}$ *and any* $\{f_n\}_{n\in\mathbb{N}}$ *and* $\{f'_n\}_{n\in\mathbb{N}}$ *such that* $f_n, f'_n \in \mathcal{F}_n$ *for all* $n \in \mathbb{N}$*, it holds that*

$$\{(\mathsf{hk}_\lambda, \mathsf{ek}_\lambda)\}_{\lambda\in\mathbb{N}} \stackrel{c}{\equiv} \{(\mathsf{hk}_\lambda, \mathsf{ek}'_\lambda)\}_{\lambda\in\mathbb{N}}$$

where $\mathsf{hk}_\lambda \stackrel{\$}{\leftarrow} \mathsf{S}(1^\lambda, 1^{n_\lambda})$*,* $(\mathsf{ek}_\lambda, \mathsf{td}_\lambda) \stackrel{\$}{\leftarrow} \mathsf{G}(\mathsf{hk}_\lambda, f_{n_\lambda})$ *and* $(\mathsf{ek}'_\lambda, \mathsf{td}'_\lambda) \stackrel{\$}{\leftarrow} \mathsf{G}(\mathsf{hk}_\lambda, f'_{n_\lambda})$*.*

– ***Input Privacy:*** TDH *is* input-private *if for any polynomial-length* $\{\mathsf{x}_\lambda\}_{\lambda\in\mathbb{N}}$ *and* $\{\mathsf{x}'_\lambda\}_{\lambda\in\mathbb{N}}$ *such that* $n_\lambda := |\mathsf{x}_\lambda| = |\mathsf{x}'_\lambda|$*, it holds that*

$$\{(\mathsf{hk}_\lambda, \mathsf{h}_\lambda)\}_{\lambda\in\mathbb{N}} \stackrel{c}{\equiv} \{(\mathsf{hk}_\lambda, \mathsf{h}'_\lambda)\}_{\lambda\in\mathbb{N}}$$

where $\mathsf{hk}_\lambda \stackrel{\$}{\leftarrow} \mathsf{S}(1^\lambda, 1^{n_\lambda})$*,* $\mathsf{h}_\lambda = \mathsf{H}(\mathsf{hk}_\lambda, \mathsf{x}_\lambda; \rho)$ *and* $\mathsf{h}' = \mathsf{H}(\mathsf{hk}_\lambda, \mathsf{x}'_\lambda; \rho')$ *for* $\rho, \rho' \stackrel{\$}{\leftarrow} \{0,1\}^*$*. We also define statistical input privacy in the natural sense.*

– ***Compactness:*** *we require that the image length of the hash function,* η*, is independent of* n*, and is bounded by some polynomial in* λ*.*

For this outline, we think of trapdoor hashing as a protocol where Alice and Bob play the roles of a sender with input x and, respectively, a receiver who wants to learn $\mathsf{x}[i]$ (or, generally, $f(\mathsf{x})$). For now, we will mostly focus on receiver privacy, i.e. function privacy, as sender's privacy is much easier to achieve. Our main goal is to construct trapdoor hash with optimal rate of 1, that is a scheme where the hint e consists of a single bit.

2.2 Trapdoor Hash from DDH

We start with our DDH-based construction of trapdoor hash for index predicates. Recall that, roughly speaking, the *Decisional Diffie-Hellman (DDH)* assumption says that an element g^{ab} of a group $\mathbb{G}$ with prime order p, where $g \in \mathbb{G}$ is a generator and $a, b \in \mathbb{Z}_p$ are uniform, is indistinguishable from a uniform group element, given g^a and g^b. We formally state our first result below.

Theorem 2.2. *There exists a rate-1 trapdoor hash scheme for index predicates with error probability* $1/\lambda$*, statistical input privacy, and function privacy based on the DDH assumption.*

The Basic Hash Function. The starting point of is the following group-based hash function mapping $\{0,1\}^n$ to a group $\mathbb{G}$:

$$\mathsf{H}(\mathbf{A}, \mathsf{x}) = \prod_{j=1}^{n} g_{j,\mathsf{x}[j]}$$

where $\mathsf{x} \in \{0,1\}^n$ is the input and $\mathbf{A} = (g_{j,b})_{j\in[n],b\in\{0,1\}} \stackrel{\$}{\leftarrow} \mathbb{G}^{2\times n}$ is chosen uniformly at random and serves as the hash key hk. By choosing n larger than the representation size of a group element in $\mathbb{G}$, this function becomes compressing. Collision resistance of this function can be routinely established from the discrete logarithm assumption in $\mathbb{G}$.

This surprisingly powerful function plays a central role in recent constructions of identity based encryption [21], trapdoor functions [26], deterministic encryption and lossy trapdoor functions [24].

Adding Trapdoors. We show how this function can be made invertible, using techniques of [24]. Clearly, the hash value $\mathsf{h} \leftarrow \mathsf{H}(\mathsf{hk}, \mathsf{x})$ is too short to information-theoretically specify x. Thus we will add additional *hints*, which we also call *encodings*, to allow recovery of x. We will first discuss how the receiver can recover a single bit $\mathsf{x}[i]$ of x.

Let $i \in [n]$ be an index of the receiver's choice. The receiver will generate a matrix $\mathbf{B} \in \mathbb{G}^{2\times n}$, that serves as an encoding key ek, such that the following holds for all $\mathsf{x} \in \{0,1\}^n$: If $\mathsf{H}(\mathbf{A}, \mathsf{x}) = \mathsf{h}$, then $\mathsf{H}(\mathbf{B}, \mathsf{x}) = \mathsf{h}^s \cdot g^{\mathsf{x}[i]}$ for some $s \in \mathbb{Z}_p$. We can construct such a matrix $\mathbf{B} = (u_{j,b})_{j\in[n],b\in\{0,1\}}$ by choosing $s \xleftarrow{\$} \mathbb{Z}_p$ uniformly at random, and setting

$$u_{j,b} = g^s_{j,b}$$

for all $j \neq i$ and

$$u_{i,b} = g^s_{i,b} \cdot g^b. \tag{2.1}$$

Since s is uniform, we immediately get, via the DDH assumption, that all $g^s_{j,b}$ are pseudorandom, and consequently, the matrix $\mathbf{B}$ is pseudorandom as well. Thus, the matrix $\mathbf{B}$ computationally hides the index i.

Given the values $\mathsf{h} = \mathsf{H}(\mathbf{A}, \mathsf{x})$ and the hint $\mathsf{e} = \mathsf{H}(\mathbf{B}, \mathsf{x})$, as well as a trapdoor consisting of s, the receiver can recover x as follows. As by the above property it holds that $\mathsf{e} = \mathsf{h}^s \cdot g^{\mathsf{x}[i]}$, we can recover $\mathsf{x}[i]$ by testing $\mathsf{e} \stackrel{?}{=} \mathsf{h}^s \cdot g^b$ for both $b \in \{0,1\}$ and setting $\mathsf{x}[i] \leftarrow b$ for the b which satisfies this test. While we can construct a trapdoor hash function in this way, its rate will be far from 1: To encode a single bit $\mathsf{x}[i]$ of x, we need to spend one full group element e. Assuming that a group element has size λ, this will give us a construction of rate $1/\lambda$.

Towards Rate 1. Clearly, sending a group element e to encode a single bit $\mathsf{x}[i]$ is wasteful. However, we make the following observation: The term e can only assume two different values, namely h^s and $\mathsf{h}^s \cdot g$, depending on whether the bit $\mathsf{x}[i]$ is 0 or 1. So what we need is a way for the sender to signal to the receiver that either $\mathsf{e} = \mathsf{h}^s$ or $\mathsf{e} = \mathsf{h}^s \cdot g$, without actually sending e. Yet, since the sender does not know i, he generally does not know whether he is encoding 0 or 1, that is, he does not know whether e is of the form h^s or $\mathsf{h}^s \cdot g$.

However, assume the sender could somehow determine the *distance to a nearby reference point* of e which is insensitive to small perturbations. This would for instance be the case if the group $\mathbb{G}$ had a subgroup $\mathbb{G}'$, such that we can efficiently test membership in $\mathbb{G}'$ and the quotient $\mathbb{G}/\mathbb{G}'$ is of polynomial size. Since $|\mathbb{G}/\mathbb{G}'|$ is only of polynomial size, we can efficiently compute the *distance* to $\mathbb{G}'$ for every $\mathsf{e} \in \mathbb{G}$. That is, the function $\mathsf{Dist}(\mathsf{e})$ which exhaustively searches for the smallest $z \in \mathbb{Z}$ such that $\mathsf{e} \cdot g^z \in \mathbb{G}'$ is efficiently computable. Assuming further for simplicity that $|\mathbb{G}/\mathbb{G}'|$ is even, it holds for every $\mathsf{e} \in \mathbb{G}$ that

$$\mathsf{Dist}(\mathsf{e} \cdot g) \mod 2 = (\mathsf{Dist}(\mathsf{e}) + 1) \mod 2.$$

This means that h^s and $\mathsf{h}^s \cdot g$ never map to the same bit under the function $\mathsf{Dist}(\cdot) \mod 2$. Via this observation, the sender can signal to the receiver whether e is

h^s or $\mathsf{h}^s \cdot g$ as follows. Instead of sending e itself to the receiver, he just sends the bit $\hat{\mathsf{e}} = \mathsf{Dist}(\mathsf{e}) \mod 2 \in \{0,1\}$ to the receiver.

Modifying the recovery procedure of above, the receiver can recover $\mathsf{x}[i]$ by testing $\hat{\mathsf{e}} \stackrel{?}{=} \mathsf{Dist}(\mathsf{h}^s g^b) \mod 2$ for $b \in \{0,1\}$ and setting $\mathsf{x}[i] \leftarrow b$ for the b which satisfies this test. This procedure recovers the correct bit $\mathsf{x}[i]$ with $\hat{\mathsf{e}} = \mathsf{Dist}(\mathsf{h}^s \cdot g^{\mathsf{x}[i]})$, as the value e computed by the sender must have been either h^s or $\mathsf{h}^s \cdot g$, and by the above $\mathsf{Dist}(\mathsf{h}^s) \mod 2 \neq \mathsf{Dist}(\mathsf{h}^s \cdot g) \mod 2$.

Achieving Rate 1. Alas, since $\mathbb{G}$ is typically a cyclic group of prime order, it has no non-trivial subgroups. But upon closer inspection, the signalling technique above does not really rely on any additional group structure. All we need is that $\mathsf{Dist}(\mathsf{e} \cdot g) = \mathsf{Dist}(\mathsf{e}) + 1$.

Fortunately, a technique to determine the distance to a reference point was proposed by Boyle, Gilboa and Ishai [9] in the context of homomorphic secret sharing. In a nutshell, instead of computing the distance to a subgroup, we compute the distance to a *moderately dense pseudorandom subset* of $\mathbb{G}$. Such a pseudorandom subset can be succinctly represented via the key of a pseudorandom function by setting S_K to be the set of all points $h \in \mathbb{G}$ for which $\mathsf{PRF}_K(h)$ starts with $k = O(\log(\lambda))$ zeros. By tuning the parameter k appropriately, one can achieve an average separation of the points in S_K by an arbitrary polynomial amount. We can now define $\mathsf{Dist}(\mathsf{e})$ to be the smallest $z \in \mathbb{Z}$ such that $\mathsf{e} \cdot g^z \in S_K$, i.e. $\mathsf{PRF}_K(\mathsf{e} \cdot g^z)$ starts with k zeros. Note that this function can be computed efficiently for the above choice of k.

However, as the vigilant reader might have observed already, when using this distance function, the above signalling procedure does not have perfect correctness anymore. If h^s and $\mathsf{h}^s \cdot g$ decode to different points in S_K, it might be that $\mathsf{Dist}(\mathsf{h}^s) \mod 2 = \mathsf{Dist}(\mathsf{h}^s \cdot g) \mod 2$, in which case the receiver cannot infer whether $\mathsf{x}[i] = 0$ or $\mathsf{x}[i] = 1$ and must declare an erasure.

Fortunately, by choosing k large enough, we can make the probability of such an erasure happening an arbitrarily small polynomial fraction $1/p(\lambda)$, while still ensuring that the decoding procedure runs in polynomial time[10] As it turns out, in many applications, we can deal with this small erasure probability by resorting to standard coding techniques.

Sender Privacy. So far we have not addressed issues concerning the privacy of the sender's inputs. However, in our DDH-based construction this is easy to achieve by providing an additional random input to the hash function H. That is, we define H as

$$\mathsf{H}(\mathbf{A}, \mathsf{x}; r) = g^r \cdot \prod_{j=1}^{n} g_{j,\mathsf{x}[j]},$$

[10] We can ensure that both sender and receiver run in strict polynomial time by introducing a suitable polynomial upper bound for the number of trials in the exhaustive search step of $\mathsf{Dist}(\cdot)$. For small erasure probabilities, a near-quadratic improvement in the running time can be obtained via the recent optimal "distributed discrete log" algorithm of Dinur, Keller, and Klein [19].

for a uniformly random $r \xleftarrow{\$} \mathbb{Z}_p$. The hash value $\mathsf{h} = \mathsf{H}(\mathbf{A}, \mathsf{x}; r)$ is now uniformly random (over the choice of r) and therefore does not leak information about x. Furthermore, given the trapdoor s and a single bit $\mathsf{x}[j]$ of the input x we can perfectly simulate e by computing $\mathsf{e} \leftarrow \mathsf{h}^s \cdot g^{\mathsf{x}[j]}$. From e we can compute $\hat{\mathsf{e}}$ as before. Thus, the modified construction has perfect sender privacy.

2.3 Trapdoor Hash from QR and LWE

We now briefly discuss instantiations of these techinques based on the Quadratic Residuosity (QR) and Learning With Errors (LWE) assumptions to achieve trapdoor hash for the even more general class of linear predicates. As it turns out, in both these cases we will have structures with exact subgroups. However, in both cases there will also be new challenges which will have to be addressed with slightly different ideas.

Theorem 2.3. *There exists a rate-1 trapdoor hash scheme for linear predicates with negligible error probability, statistical input privacy, and function privacy based on the {QR,LWE} assumption.*

Construction from QR. We will start with the QR-based construction. Instead of relying on the QR assumption directly, we will use the fact that we can construct a group $\mathbb{G}$ in which the subgroup indistinguishability problem is hard under QR [12]. More specifically, the group $\mathbb{G}$ we use has a subgroup $\mathbb{G}'$ such that $|\mathbb{G}/\mathbb{G}'| = 2$. We can represent every $h \in \mathbb{G}$ as $h = (-1)^b \cdot a$, where $b \in \{0,1\}$ and $a \in \mathbb{G}'$. For the hash function H, we can use exactly the same construction as above, that is $\mathsf{H}(\mathsf{hk}, \mathsf{x}) = \prod_{j=1}^{n} g_{j,\mathsf{x}[j]}$. The only difference is that we choose the elements in the key $\mathsf{hk} = \mathbf{A}$ from the subgroup $\mathbb{G}'$ instead of $\mathbb{G}$, that is, $\mathbf{A} = (g_{j,b})_{j\in[n],b\in\{0,1\}} \xleftarrow{\$} {\mathbb{G}'}^{2\times m}$. Similar as in the DDH-based construction, for an index $i \in [n]$, the matrix $\mathbf{B}$ generated by G now has the form $\mathbf{B} = (u_{j,b})_{j\in[n],b\in\{0,1\}}$ where

$$u_{j,b} = g_{j,b}^s$$

for all $j \neq i$ and

$$u_{i,b} = g_{i,b}^s \cdot (-1)^b.$$

Here, s is uniformly random in an appropriate domain. The crucial difference is that in $u_{i,b}$, we have replaced the generator g in the DDH-based construction by -1. It follows directly via the subgroup indistinguishabilty assumption that $g_{i,b}^s \cdot (-1)^b$ is indistinguishable from $g_{i,b}^s$. Thus, as before, the index i is hidden.

By a similar analysis as before, it holds that if $\mathsf{h} = \mathsf{H}(\mathbf{A}, \mathsf{x})$, then $\mathsf{e} = \mathsf{H}(\mathbf{B}, \mathsf{x}) = \mathsf{h}^s \cdot (-1)^{\mathsf{x}[i]}$. However, there is a crucial difference now: As $\mathsf{e} = \mathsf{h}^s \cdot (-1)^{\mathsf{x}[i]}$, the sender can also compute $\mathsf{e} \cdot (-1) = \mathsf{h}^s \cdot (-1)^{1-\mathsf{x}[i]}$. That is, one of these two elements is h^s and the other one is $\mathsf{h}^s \cdot (-1)$. Recall that the receiver can also compute these two elements using the hash value h and the trapdoor s. Thus, the only task left for the sender is to signal to the receiver which one of the two elements the element e he got is. This can be easily done by communicating a single bit: The sender compares e and $\mathsf{e} \cdot (-1)$ under some

total order $\succ$, say, by representing both elements as bit strings, and computing the lexicographic order. Now, he sends the bit $\hat{\mathsf{e}} = 0$ if $\mathsf{e} \succ \mathsf{e} \cdot (-1)$ and $\hat{\mathsf{e}} = 1$ otherwise. The receiver can recover $\mathsf{x}[i]$ as follows: If $\mathsf{h}^s \succ \mathsf{h}^s \cdot (-1)$ and $\hat{e} = 0$ he sets $\mathsf{x}[i] = 0$, otherwise $\mathsf{x}[i] = 1$.

The main difference of this instantiation compared to our DDH-based construction is that there is no decoding error. We can even leverage this fact to achieve a stronger functionality. So far, we have only discussed how the receiver can recover individual bits $\mathsf{x}[i]$ of the sender's input, namely realize trapdoor hash for *index predicates*. We will now show how this can be upgraded in a way such that the receiver can learn an inner product $\langle \mathsf{y}, \mathsf{x} \rangle \mod 2$, and therefore obtain trapdoor hash for the more general class of *linear predicates*. The vector y is chosen by the receiver and is used to generate the matrix $\mathbf{B}$. Concretely, for a vector $\mathsf{y} \in \{0,1\}^n$ the receiver sets

$$u_{j,b} = g_{j,b}^s \cdot (-1)^{b \cdot \mathsf{y}[j]}$$

for all $j \in [n]$ and $b \in \{0,1\}$. As before, we can use the subgroup indistinguishability assumption to establish that the matrix $\mathbf{B}$ hides the vector y.

A simple calculation shows that $\mathsf{H}(\mathbf{B}, \mathsf{x}) = \mathsf{h}^s \cdot (-1)^{\langle \mathsf{y}, \mathsf{x} \rangle}$. The encoding and decoding procedures are exactly the same as before, with the difference that now the receiver learns the inner product $\langle \mathsf{y}, \mathsf{x} \rangle \mod 2$. While this modification to our construction is nearly straightforward, it has several important applications.

Construction from LWE. We will finally turn to our construction from LWE. On a conceptual level, the construction is very similar to the QR-based construction. We will directly explain the construction for linear predicates, i.e. inner products over $\mathbb{F}_2$. In this instantiation, let $q = 2p$ be an even modulus. The hashing key $\mathsf{hk} = \mathbf{A}$ is a $2 \times n$ matrix of uniformly random column vectors $\mathbf{a}_{j,b} \in \mathbb{Z}_q^k$, that is, each component of this matrix is a vector itself. The hash of an input $\mathsf{x} \in \{0,1\}^n$ is now computed as the sum of the corresponding $\mathbf{a}_{j,b}$, that is

$$\mathsf{H}(\mathbf{A}, \mathsf{x}) = \sum_{j=1}^{n} \mathbf{a}_{j,\mathsf{x}[j]}.$$

The encoding key contains a matrix $\mathbf{B} = (u_{j,b})_{j \in [n], b \in \{0,1\}}$, which consists of elements $u_{j,b} \in \mathbb{Z}_q^k$ which are computed by

$$u_{j,b} = \mathbf{s}^\top \mathbf{a}_{j,b} + e_{j,b} + \mathsf{y}[j] \cdot b \cdot (q/2),$$

where $\mathbf{s}$ is chosen uniformly from $\mathbb{Z}_q^k$ and the $e_{j,b}$ are sampled from a short LWE-error distribution such as a discrete gaussian. By the LWE assumption, we immediately get that the values $\mathbf{s}^\top \mathbf{a}_{j,b} + e_{j,b}$ are pseudorandom, and consequently the matrix $\mathbf{B}$ hides the vector y. Assume further that PRF is a pseudorandom function from $\mathbb{Z}_q^k$ to $\mathbb{Z}_q$. For this instantiation, the receiver will also include a uniformly random PRF-key $K \xleftarrow{\$} \{0,1\}^\lambda$ into the encoding key.

As before, the sender computes $\mathsf{h} = \mathsf{H}(\mathbf{A}, \mathsf{x})$ and $\mathsf{e} = \mathsf{H}(\mathbf{B}, \mathsf{x})$. Notice that it holds that

$$\mathsf{e} = \sum_{j=1}^{n} u_{j,\mathsf{x}[j]} = \mathbf{s}^\top \sum_{j=1}^{n} \mathbf{a}_{j,\mathsf{x}[j]} + \sum_{j=1}^{n} e_{j,\mathsf{x}[j]} + \langle \mathsf{y}, \mathsf{x} \rangle \cdot (q/2) = \mathbf{s}^\top \mathsf{h} + e' + \langle \mathsf{y}, \mathsf{x} \rangle \cdot (q/2),$$

where $e' = \sum_{j=1}^{n} e_{j,\mathsf{x}[j]}$ is a small error.

The challenge in this instantiation is that e is noisy, so the comparison-based technique from the QR-based construction will not work here. Nevertheless, a standard tool to robustly deal with this kind of error in the world of LWE is the rounding technique, introduced by Banerjee, Peikert and Rosen [5]. Define the rounding function $\lfloor \cdot \rceil_2$ by $\lfloor z \rceil_2 = \lfloor z \cdot 2/q \rceil \mod 2$. The sender now computes $\hat{\mathsf{e}}$ by

$$\hat{\mathsf{e}} = \lfloor \mathsf{H}(\mathbf{B}, \mathsf{x}) + \mathsf{PRF}_K(\mathsf{h}) \rceil_2$$

and sends h along with the bit $\hat{\mathsf{e}}$ to the receiver. The receiver now computes and outputs $(\hat{\mathsf{e}} - \lfloor \mathbf{s}^\top \mathsf{h} + \mathsf{PRF}_K(\mathsf{h}) \rceil_2) \mod 2$.

To establish correctness, we will use the fact that, for a sufficiently large q, the rounding function is insensitive to small perturbations. That is, for a uniformly random $z \xleftarrow{\$} \mathbb{Z}_q$, and a sufficiently *small* noise e, it holds that $\lfloor z + e \rceil_2 = \lfloor z \rceil_2$, except with small probability over the choice of z. Now, since the term $\mathsf{PRF}_K(\mathsf{h})$ is pseudorandom in $\mathbb{Z}_q$, it holds that

$$\begin{aligned} \hat{\mathsf{e}} = \lfloor \mathsf{H}(\mathbf{B}, \mathsf{x}) + \mathsf{PRF}_K(\mathsf{h}) \rceil_2 &= \lfloor \mathbf{s}^\top \mathsf{h} + e' + \langle \mathsf{y}, \mathsf{x} \rangle \cdot (q/2) + \mathsf{PRF}_K(\mathsf{h}) \rceil_2 \\ &= \lfloor \mathsf{s}^\top \mathsf{h} + \mathsf{PRF}_K(\mathsf{h}) \rceil_2 + \langle \mathsf{y}, \mathsf{x} \rangle, \end{aligned}$$

except with small probability over the choice over K. This is the reason why we include the key K in the receiver's message, that is, to enable the sender to randomize $\mathsf{H}(\mathbf{B}, \mathsf{x})$ without increasing the size of the sender message. Correctness of the scheme follows.

The magnitude of the correctness error depends on the modulus-to-noise ratio. If we choose a superpolynomial modulus-to-noise ratio, the correctness error becomes negligible. For a polynomial modulus-to-noise ration the correctness error will be inverse polynomial and we have to compensate with coding techniques.

3 Rate-1 Oblivious Transfer and More

We now present the first family of applications of trapdoor hash. We show how to use rate-1 trapdoor hash to securely realize basic sender-receiver functionalities through two-message protocols with optimal sender-receiver communication, i.e. optimal *download rate*.

Formally speaking, a two-message protocol for functionality $f : X \times Y \to Z$ is defined through a triple of PPT algorithms $\Pi = (\Pi_1, \Pi_2, \Pi_3)$ where, at first, the receiver computes a message $\mathsf{msg}_1 \leftarrow \Pi_1(1^\lambda, x)$ for security parameter λ and input $x \in X$ and sends it to the sender. The sender with input $y \in Y$ responds

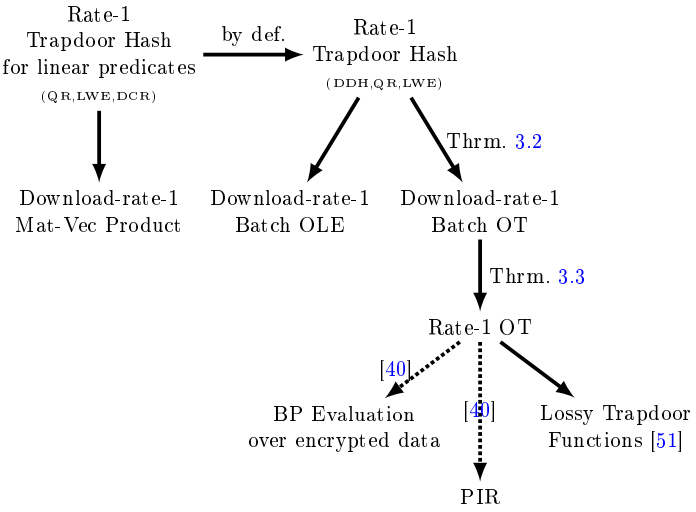

Fig. 1: Overview of the results in this work, Part I: optimal-rate protocols for OT-like sender-receiver functionalities and their applications. Dotted lines correspond to corollaries from prior work.

by a message $\mathsf{msg}_2 \leftarrow \Pi_2(\mathsf{msg}_1, y)$. Lastly, given the second message msg_2 and possibly a local state st computes the output $f(x,y) = \Pi_3(\mathsf{msg}_2, \mathsf{st})$. We require standard notions of receiver privacy and sender privacy (against a semi-honest receiver). The download rate of a two-message is defined as follows.

Definition 3.1 (Download Rate of a Two-Message Protocol). *Let $0 \leq \omega \leq 1$. We say that a two-message protocol Π for functionality $f : X \times Y \to Z$ has download rate ω if there exists a polynomial $B(\lambda)$ such for all polynomial-length input sequences $\{(x_\lambda, y_\lambda)\}_{\lambda \in \mathbb{N}}$ in the domain of f such that $|f(x_\lambda, y_\lambda)| \geq B(\lambda)$ for all λ, we have*

$$\liminf_{\lambda \to \infty} \frac{|f(x_\lambda, y_\lambda)|}{m_\lambda} = \omega$$

where m_λ is the maximal length of the sender-receiver message when Π runs on inputs (x_λ, y_λ) and security parameter λ.

The first fundamental functionality we investigate is *oblivious transfer* (OT), where a receiver with private input bit $i \in \{0,1\}$ communicates with a sender with secrets $\mathsf{s}_0, \mathsf{s}_1$ in order to obtain secret s_i. Rate-1 OT has several important applications, for which we are able to achieve the first constructions under various assumptions, using our trapdoor hash constructions. We also discuss a couple of related primitives: *oblivious linear function evaluation* (OLE), where the sender has a linear function $f(x) = ax + b$ and the goal is to evaluate f on the receiver's private input x, and the more general *matrix-vector product* where the sender

has a matrix M, the receiver has a vector v, and the goal is to compute the product $Mv^\top$.

3.1 Rate-1 Oblivious Transfer from Trapdoor Hash

Equipped with our newly developed tool, we show how to construct 2-message OT protocols with rate 1 given any trapdoor hash with the same rate. We consider two flavours of OT where download-rate-1 can be achieved. The first is *batch OT*, where a batch of OT instances with single-bit secrets are invoked in parallel, and the second is *string OT*, which consists of a single OT instance with secrets that are assumed to be sufficiently long. In the latter case, we get optimal *overall* rate (where also receiver-sender communication is taken into account).

Batch OT. Recall that a trapdoor hash scheme for index predicates allows one to recover the i^{th} bit of a string x given the hash value $\mathsf{H}(\mathsf{hk}, \mathsf{x})$ and a single additional bit e (which we denote $\hat{\mathsf{e}}$ above). With this tool at hand, we can realize the 1-out-of-2 bit OT functionality by letting the receiver specify the hash key hk and the encoding key ek corresponding to the choice bit $i \in \{0,1\}$. The sender then sets its input $\mathsf{x} := \mathsf{s}_0 \| \mathsf{s}_1$ to be the concatenation of the two secret bits and computes $\mathsf{h} = \mathsf{H}(\mathsf{hk}, \mathsf{x})$ together with the encoding e. Given such an information, the receiver can recover the chosen secret bit by running the decoding algorithm. The obvious shortcoming of this approach is that it is wasteful in terms of download rate, in the sense that the hash of the string must be included to recover a single bit.

The key observation here is that the hash key hk can be reused across several executions. Therefore the size of the hash h can be amortized across multiple independent bit OT protocols. That is, if the bit OTs are executed in a batch, we can boost the download rate of the construction to approach 1: Given n independent instances of bit OT, the receiver samples a hash key hk as before, this time for inputs of length $2n$ rather than 2, and samples a set of encoding keys $(\mathsf{ek}_1, \ldots, \mathsf{ek}_n)$, where the j^{th} key allows the receiver to learn the input bit at position $(2j + i_j)$, where $i_j \in \{0,1\}$ is the choice bit of the j^{th} OT instance. It is important that all of the encoding keys are generated with respect to the same hk, since it will allow us to re-use the corresponding hash. The sender defines $\mathsf{x} := \mathsf{s}_{1,0} \| \mathsf{s}_{1,1} \| \ldots, \| \mathsf{s}_{n,0} \| \mathsf{s}_{n,1}$, where $\mathsf{s}_{j,0}, \mathsf{s}_{j,1}$ are the secrets for the j^{th} instance, and computes the hash $\mathsf{h} = \mathsf{H}(\mathsf{hk}, \mathsf{x})$ as before, in addition to the additional hints, i.e. TDH encodings, $(\mathsf{e}_1, \ldots, \mathsf{e}_n)$. The recovery procedure is then run in parallel for each bit OT instance. Note that the sender's message consists of a hash (i.e., a single group element) and n bits. That is, the impact of h in the communication vanishes as n grows, and thus, the download rate of the scheme approaches 1. The above outline gives the following theorem.

Theorem 3.2. *Assume there exists a rate-1 trapdoor hash scheme* TDH*, with error probability $\epsilon(\lambda)$. Then, there exists a 2-message batch OT protocol with download rate 1 and independent error probability of $\epsilon(\lambda)$ in every (single-bit) OT instance. Further, if* TDH *is statistically input-private, then the obtained batch OT protocol is statistically sender-private.*

String OT. We showed how to obliterate the impact of the hash value h in the second OT message by executing multiple bit OT instances in a batch. The same can be accomplished for a single OT instance, when executed on sufficiently long secret strings (rather than single bits)[11]. The protocol can be derived generically from the batch OT by adapting the encodings of the inputs: The receiver executes the batch OT protocol of above by replicating the same choice bit i over each of the n instances, whereas the sender parses the two strings $(\mathsf{s}_0, \mathsf{s}_1) \in \{0,1\}^n$ as n pairs of bits and encodes the string x as before. Since the choice bit of the receiver is the same in all positions, the decoding algorithm will recover the string s_i in its entirety.

In the above discussion we omitted a few important aspects of our transformation that need to be addressed in order to obtain a fully-fledged rate-1 OT. More specifically, (i) some instances of trapdoor hash have a correctness error, in the sense that the secret might not be recoverable with a certain probability ϵ. Furthermore, (ii) the upload rate of the construction is inverse polynomial in λ. To resolve the former point we preprocess the sender's inputs with a sufficiently strong error-correcting code. One has to be careful that the encoding function does not affect the download rate of the protocol. Fortunately, our error probability ϵ lies in a regime of parameters that allow us to efficiently instantiate the encoding function. For the latter issue, we show that any string OT with download rate 1 can be generically bootstrapped to a string OT with overall rate 1. Our method is based upon the simple observation that the first message of an OT is always reusable and therefore can be amortized by executing the same OT over blocks of a sufficiently long string. Thus, overall, our main result in this context is as follows.

Theorem 3.3. *Assume there exists a 2-message batch OT protocol with download rate 1 and independent error probability of $\epsilon(\lambda) = O(1/\lambda)$ in every (single-bit) OT instance. Then, there exists a 2-message string OT protocol with overall rate 1 and negligible error.*

The same techniques can be generalized to 1-out-of-k OT, for any $k \in \mathbb{N}$.

3.2 Applications of Rate-1 OT

We now discuss few interesting applications of rate-1 OT.

Private Information Retrieval. Given a 1-out-of-2 string OT with rate 1, a (block) single-server PIR protocol [44], with optimal download rate and polylogarithmic overall communication, follows as a simple corollary of the main theorem of Ishai and Paskin [40]. We hereby recall the transformation for completeness.

Recall that in (block) PIR, a client queries a server, that holds a database consisting of N blocks, each of length β bits, in order to privately retrieve a block of his choice. Observe that a 1-out-of-2 string OT can be seen as a hash

[11] In fact, string OT can be thought of as a special case of batch OT, where all the choice bits i_j are equal.

function that compresses the size of its input by a factor of roughly two. The idea is to use such a hash function and let the server compute a Merkle tree over the database $\mathsf{x} \in \{0,1\}^{N \cdot \beta}$. Every node in the tree consists of a block and, for simplicity, we assume that $N = 2^d$ for some $d \in \mathbb{N}$, which is the depth of the tree. Thus, the lowest level in the tree consists of the N database blocks: $\mathsf{x}_0, \ldots, \mathsf{x}_{N-1}$, and every other level $\ell = 1, \ldots, d$ in the tree consists of $N/2^\ell$-many blocks: $\mathsf{h}_{\ell,0}, \ldots, \mathsf{h}_{\ell,N/2^\ell-1}$, that are hashes of the nodes in level ℓ. Notice that every index $i \in \{0, \ldots, N-1\}$ corresponds to a path in the tree, which we denote by $(i_1, \ldots, i_d)$, which represents the path from database block x_i to the root of the tree.

The protocol proceeds as follows: First, the client generates the receiver message $\mathsf{msg}_1^{(\ell)}$ of an OT for strings of appropriate length, for each layer $\ell = 1, \ldots, d$ in the tree, where the choice bit is set to be the index i_ℓ. Then the client sends $(\mathsf{msg}_1^{(1)}, \ldots, \mathsf{msg}_1^{(\ell)})$ to the server, who computes all of the hash values in the Merkle tree, i.e. OT sender messages, and sends the root $\mathsf{msg}_2^{(d)}$ to the client. The client can recover the entry of interest by recursively applying the decoding algorithm of the OT, starting from the top level d.

Evaluating Branching Programs over Encrypted Data. Another result in the work of Ishai and Paskin [40], which can be seen as a generalization of the above, is a compiler that takes any 2-message rate-1 OT[12] into a *semi-compact* homomorphic encryption scheme for branching programs (a superclass of NC^1), where the size of the evaluated ciphertexts depends only on the length of the branching program but not on its size. This immediately yields a sublinear secure function evaluation protocol where the client's work is independent of the size of the branching program (which is in fact hidden to its eyes).

Lossy Trapdoor Functions. As a yet another application, we show a simple construction of lossy trapdoor functions [51, 35] with optimal rate from any 2-message rate-1 OT, and therefore obtain schemes based on DDH, QR, or LWE. Prior to our work, rate optimal schemes were known to exist only under the DCR assumption.

3.3 Rate-Optimal Protocols for Other OT-like Functionalities

It turns out that using trapdoor hash for index predicates, we can already capture a wide variety of predicate classes through a simple transformation. More specifically, if a given predicate class $\mathcal{F}$ is "small", i.e. contains $\mathsf{poly}(n)$ predicates for input size n, then we can obtain TDH for $\mathcal{F}$ on input x by applying TDH for index predicates on input x', where the i^{th} bit in x' is the evaluation of the i^{th} predicate in $\mathcal{F}$ on x.

We use this observation to extend the range of functionalities for which we can construct rate-optimal protocols. For instance, an interesting special case of

[12] In fact they require an OT protocol with a strong notion of sender privacy, which is satisfied by all of our constructions.

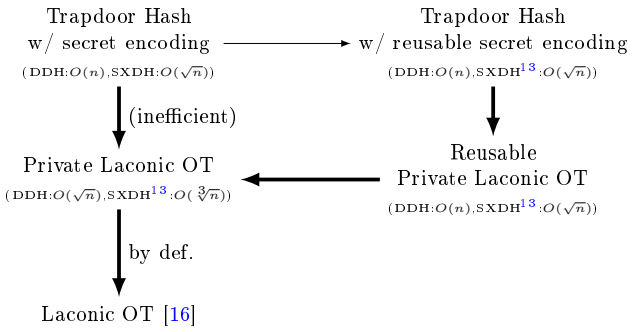

Fig. 2: Overview of the results in this work, Part II: secure function evaluation with sublinear communication. Thin lines correspond to non-generic transformations.

small predicate classes are functions $f(x) = ax + b$ over $\mathbb{F}_2$, which essentially allow realizing batch oblivious linear function evaluation (OLE) [47] by replacing the TDH for index predicates, in the batch OT construction described above, with TDH for such predicates. Further, one can extend the idea to OLE over other constant size rings (e.g. fields $\mathbb{F}_p$ for constant prime p), by evaluating each output bit separately.

An even more general functionality, that allows evaluating matrix-vector products over $\mathbb{F}_2$ (with the vector and matrix respectively being the receiver's and sender's input), can be realized using the same technique by relying on TDH for linear predicates, which can be instantiated, as mentioned earlier, under the LWE and QR assumptions. The LWE-based TDH scheme can be further extended to allow trapdoor-evaluation of linear functions over small fields, thus yielding *oblivious matrix vector multiplication* (OMV) over such fields. It is worth mentioning that OMV can be also seen as a variant of rate-1 additively homomorphic encryption, where inner products (and in particular matrix multiplication) can be evaluated over encrypted vectors.

Lastly, we note that using OLE and OMV schemes over small fields, we can realize similar functionalities over larger algebraic structures through standard algebraic manipulations. More specifically, we can get OLE and OMV over smooth rings, via the Chinese Remainder Theorem, and over extension fields of small characteristic using basic extension field algebra.

4 Private Laconic Oblivious Transfer

In this section we outline another application of trapdoor hash: *private laconic oblivious transfer*. As discussed in the introduction, private laconic OT has strong applications in secure computation. In particular, following the outline presented

[13] We also assume correlated-input secure hash over bilinear groups.

in [16] to utilize laconic OT for non-interactive secure RAM computation with unprotected memory access, we can use private laconic OT to obtain secure RAM computation where the access pattern to the memory is also hidden, and therefore achieve a stronger notion of security.

Recall that in laconic OT ($\ell\mathsf{OT}$) [16], a receiver with an input database $\mathsf{x} \in \{0,1\}^n$ communicates with a sender, with two secrets $\mathsf{s}_0, \mathsf{s}_1 \in \{0,1\}$ and an index $i \in [n]$ as input, in order to learn $\mathsf{s}_{\mathsf{x}[i]}$,while keeping both x and $\mathsf{s}_{1-\mathsf{x}[i]}$ private. In *private* laconic OT ($p\ell\mathsf{OT}$), we also require that the index i remains hidden from the receiver.

Our end goal is to realize the $p\ell\mathsf{OT}$ functionality through a two-message protocol where the overall communication is sublinear in n in order to obtain sublinear SFE protocols (due to [16]).

As a start, however, we aim for *receiver-compact* $p\ell\mathsf{OT}$ where the upload communication (i.e., the communication from the receiver to the sender) is independent of the receiver's database size n, and set no restrictions on the communication from the sender to the receiver. We then describe such a receiver-compact $p\ell\mathsf{OT}$ construction with linear sender-receiver communication through our DDH-based trapdoor hash, and then show to get sublinear communication (namely $\sqrt{n}$) using pairings.

Lastly, we show that if we are willing to compromise receiver-compactness, then we can balance our protocols using what we call *reusable private laconic OT* and obtain more efficient SFE protocols with sublinear communication under both DDH and pairings. We start with the basic definition of private laconic OT.

Definition 4.1 (Private Laconic OT). *A* private laconic OT *scheme is a tuple of four PPT algorithms* $p\ell\mathsf{OT} = (\mathsf{Gen}, \mathsf{Hash}, \mathsf{Send}, \mathsf{Receive})$ *with the following properties.*

- ***Syntax:***
  - $\mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda, 1^n)$. *The generating algorithm takes as input the security parameter* 1^λ*, and the size of the database* n*, and outputs public parameters* $\mathsf{pp} \in \{0,1\}^*$.
  - $\mathsf{h} \leftarrow \mathsf{Hash}(\mathsf{pp}, \mathsf{x}; \rho)$. *The hashing algorithm takes as input the public parameters* pp*, a database* $\mathsf{x} \in \{0,1\}^n$*, and randomness* $\rho \in \{0,1\}^*$*, and deterministically outputs a hash value* $\mathsf{h} \in \{0,1\}^\eta$.
  - $\mathsf{ct} \leftarrow \mathsf{Send}(\mathsf{pp}, \mathsf{h}, i, (\mathsf{s}_0, \mathsf{s}_1))$. *The sending algorithm takes as input the public parameters* pp*, a hash value* h*, an index* $i \in [n]$*, and a pair of secrets* $(\mathsf{s}_0, \mathsf{s}_1) \in \{0,1\} \times \{0,1\}$*, and outputs a ciphertext* $\mathsf{ct} \in \{0,1\}^*$.
  - $\mathsf{s} \leftarrow \mathsf{Receive}(\mathsf{pp}, \mathsf{ct}, \mathsf{x}; \rho)$. *The receiving algorithm takes as input the public parameters* pp*, a ciphertext* ct*, a database* $\mathsf{x} \in \{0,1\}^n$*, and randomness* $\rho \in \{0,1\}^*$*, and deterministically outputs a secret* $\mathsf{s} \in \{0,1\}$.
- ***Correctness:*** $p\ell\mathsf{OT}$ *is* correct *if there exists a negligible function* $\epsilon(\lambda)$ *such that the following holds for all* $\lambda, n \in \mathbb{N}$*, any database* $\mathsf{x} \in \{0,1\}^n$*, any index*

$i \in [n]$, *and any pair of secrets* $\mathsf{s}_0, \mathsf{s}_1 \in \{0,1\}$.

$$\Pr\left[\mathsf{s} = \mathsf{s}_{\mathsf{x}[i]} \;\middle|\; \begin{array}{l} \mathsf{pp} \leftarrow \mathsf{Gen}(1^\lambda, n) \\ \rho \xleftarrow{\$} \{0,1\}^* \\ \mathsf{h} \leftarrow \mathsf{Hash}(\mathsf{pp}, \mathsf{x}; \rho) \\ \mathsf{ct} \leftarrow \mathsf{Send}(\mathsf{pp}, \mathsf{h}, i, (\mathsf{s}_0, \mathsf{s}_1)) \\ \mathsf{s} \leftarrow \mathsf{Receive}(\mathsf{pp}, \mathsf{ct}, \mathsf{x}; \rho) \end{array}\right] \geq 1 - \epsilon(\lambda).$$

- ***Receiver Privacy:*** $p\ell\mathsf{OT}$ *is statistically, resp., computationally,* receiver-private *if for any polynomial-length* $\{\mathsf{x}_\lambda, \mathsf{x}'_\lambda\}_{\lambda\in\mathbb{N}}$ *where* $n_\lambda := |\mathsf{x}_\lambda| = |\mathsf{x}'_\lambda|$ *for all* $\lambda \in \mathbb{N}$*, the following two distribution ensembles*

$$\{(\mathsf{pp}_\lambda, \mathsf{h}_\lambda)\}_{\lambda\in\mathbb{N}} \qquad\qquad \{(\mathsf{pp}_\lambda, \mathsf{h}'_\lambda)\}_{\lambda\in\mathbb{N}}$$

  where $\mathsf{pp}_\lambda \xleftarrow{\$} \mathsf{Gen}(1^\lambda, 1^{n_\lambda})$ *and* $\mathsf{h}_\lambda := \mathsf{Hash}(\mathsf{pp}_\lambda, \mathsf{x}_\lambda; \rho)$, $\mathsf{h}'_\lambda := \mathsf{Hash}(\mathsf{pp}_\lambda, \mathsf{x}'_\lambda; \rho')$ *for* $\rho, \rho' \xleftarrow{\$} \{0,1\}^*$*, are statistically, resp. computationally, indistinguishable.*
- ***Sender Privacy (against a semi-honest receiver):*** $p\ell\mathsf{OT}$ *is (computationally)* sender-private *if there exists a PPT algorithm* Sim *such that for any* $\mathsf{s}_0, \mathsf{s}_1 \in \{0,1\}$*, any polynomial-length* $\{\mathsf{x}_\lambda\}_{\lambda\in\mathbb{N}}$ *and any* $\{i_\lambda\}_{\lambda\in\mathbb{N}}$*, where* $n_\lambda := |\mathsf{x}_\lambda|$ *and* $i_\lambda \in [n_\lambda]$ *for all* $\lambda \in \mathbb{N}$*, the distribution ensembles* $\{\mathsf{Real}_\lambda\}_{\lambda\in\mathbb{N}}$ *and* $\{\mathsf{Ideal}_\lambda\}_{\lambda\in\mathbb{N}}$*, where*

$$\mathsf{Real}_\lambda = (\mathsf{pp}_\lambda, \mathsf{x}_\lambda, (\mathsf{ct}_\lambda, \rho)) \qquad \mathsf{Ideal}_\lambda = (\mathsf{pp}_\lambda, \mathsf{x}_\lambda, \mathsf{Sim}(1^\lambda, \mathsf{pp}_\lambda, \mathsf{x}_\lambda, \mathsf{s}_{\mathsf{x}_\lambda[i]}))$$

  such that $\rho \xleftarrow{\$} \{0,1\}^*$, $\mathsf{pp}_\lambda \xleftarrow{\$} \mathsf{Gen}(1^\lambda, 1^{n_\lambda})$ *and* $\mathsf{ct}_\lambda \xleftarrow{\$} \mathsf{Send}(\mathsf{pp}_\lambda, \mathsf{h}_\lambda, i_\lambda, (\mathsf{s}_0, \mathsf{s}_1))$ *for* $\mathsf{h}_\lambda = \mathsf{Hash}(\mathsf{pp}_\lambda, \mathsf{x}_\lambda; \rho)$*, are computationally indistinguishable.*
- ***Receiver Compactness:*** $p\ell\mathsf{OT}$ *is* receiver-compact *if the output length of* Hash*,* η*, is independent in* n*, and is bounded by some polynomial in the security parameter* λ*.*

4.1 Basic Construction from Trapdoor Hash

Let us first try to realize the relaxed notion of $\ell\mathsf{OT}$ using trapdoor hash in a straight-forward way. In order to that, the roles of Alice and Bob from Section 2, as a sender and a receiver, must be swapped.

Given a TDH for index predicates, the construction proceeds as follows. The public parameters of the $\ell\mathsf{OT}$ scheme simply consist of a hash key hk of the TDH. The receiver (which is now played by Alice) computes a hash value of his database $\mathsf{h} = \mathsf{H}(\mathsf{hk}, \mathsf{x})$, which he sends to the sender (now Bob). Observe that the size of h is independent of the size of x, and thus satisfying our requirement regarding upload communication.

The sender generates a pair $(\mathsf{ek}, \mathsf{td})$ of an encoding key and a trapdoor corresponding to the hash key hk and his input index i. Using td and h, he computes two symmetric encryption keys $(\mathsf{e}_0, \mathsf{e}_1) = \mathsf{D}(\mathsf{td}, \mathsf{h})$, using which he encrypts his secret inputs s_0 and s_1, respectively, to obtain two ciphertexts. He now sends the

key ek as well as the two ciphertexts to the receiver, who will be able to decrypt one of them by recovering $\mathsf{e}_{\mathsf{x}[i]}$ using the encoding algorithm E of the TDH.

To establish security of this $\ell\mathsf{OT}$ construction, we need the following to ensure that (i) the hash h hides x (*receiver privacy*), and (ii) the encoding key ek hides $\mathsf{e}_{1-\mathsf{x}[i]}$ (*sender privacy*). While receiver privacy is implied directly from the input privacy of the underlying TDH, sender privacy does not generically follow. Thus, we need to augment our definition of TDH with the requirement that, for every i, the value $\mathsf{e}_{1-\mathsf{x}[i]}$ is uniformly random given hk, ek and x, where $(\mathsf{ek}, \mathsf{td}) \xleftarrow{\$} \mathsf{G}(\mathsf{hk}, i)$ and $(\mathsf{e}_0, \mathsf{e}_1) = \mathsf{D}(\mathsf{td}, \mathsf{H}(\mathsf{hk}, \mathsf{x}))$. A TDH that satisfies this requirement is said to have *secret encoding*.

Notice that the secret encoding property is in conflict with achieving high rate in a TDH. In particular, in any rate-1 TDH, correctness requires that $\mathsf{e}_{1-\mathsf{x}[i]} = 1 - \mathsf{e}_{\mathsf{x}[i]}$ with a high probability.

Fortunately, the basic DDH-based TDH construction without the rate optimization, which is sketched above, fulfills the secret encoding property under DDH. This is not surprising, as so far, this construction is very similar to the original $\ell\mathsf{OT}$ construction of [16].

In fact, the above outlined protocol realizes the stronger notion of $p\ell\mathsf{OT}$. By relying on the function privacy of the underlying TDH, we immediately get that the sender's input index i is kept hidden from the receiver, and hence, we get the following theorem.

Theorem 4.2. *There exists a receiver-compact $p\ell\mathsf{OT}$ scheme, with statistical receiver privacy and sender privacy under the DDH assumption, that has communication complexity $O(\mathsf{poly}(\lambda)n)$.*

As hinted earlier, the above construction suffers from an undesired property: download communication is linear in the size of the receiver's database. We propose two solutions. The first relies on the SXDH assumption over bilinear groups and uses pairings in order to reduce the communication to $O(\sqrt{n})$. In the second, we introduce a *resuablitiy* notion of $p\ell\mathsf{OT}$, that can be realized under both DDH and SXDH with similar communication. We then show how to transform any reusable $p\ell\mathsf{OT}$ into a (non-reusable) $p\ell\mathsf{OT}$ scheme while reducing the overall communication complexity, to obtain efficient $p\ell\mathsf{OT}$ protocols under DDH, resp. SXDH, with download communication proportional to $\sqrt{n}$ and $\sqrt[3]{n}$, respectively.

4.2 Shrinking the Keys Using Pairings

The bottleneck in the efficiency of the DDH-based $p\ell\mathsf{OT}$ scheme from above lies in the size of the public parameters and sender's message, namely the keys hk and ek of the trapdoor hash, which both grow linearly in n.

Towards achieving sublinear communication, we start with the following observation. The high entropy of the public parameters, i.e. matrix $\mathbf{A}$ in hk, is not essential for security in the DDH-based TDH scheme. Thus, if we could produce such a matrix $\mathbf{A} = (g_{j,b})_{j\in[n],b\in\{0,1\}}$ using a shorter "seed", and then let Alice

compute a short "seed" that expands to a matrix $\mathbf{B} = (u_{j,b})_{j\in[n],b\in\{0,1\}}$ which can be used as an encoding key ek, then we are able to reduce the size of hk and ek and, therefore, the communication of the resulted $p\ell\mathsf{OT}$.

Roughly speaking, we choose the seed for $\mathbf{A}$ to be two $2 \times \sqrt{n}$ matrices, $\mathbf{A_1} \in \mathbb{G}^{2\times\sqrt{n}}$ and $\mathbf{A_2} \in \mathbb{H}^{2\times\sqrt{n}}$, for two different groups $\mathbb{G}$ and $\mathbb{H}$. We then use a bilinear map $e : \mathbb{G} \times \mathbb{H} \to \hat{\mathbb{G}}$ to pair elements in $\mathbf{A_1}$ with elements in $\mathbf{A_2}$ and get $2\times n$ elements in $\hat{\mathbb{G}}$, which we use as the hash key $\mathsf{hk} = \mathbf{A} \in \hat{\mathbb{G}}^{2\times n}$. To generate a seed to the corresponding encoding key $\mathbf{B}$, we begin by defining $\mathbf{B_1} = \mathbf{A_1}^{s_1}$ and $\mathbf{B_2} = \mathbf{A_2}^{s_2}$, which would expand to $\mathbf{B} = \mathbf{A}^{s_1+s_2}$ using the pairing. To achieve functionality, we would want to "puncture" the $(i, 1)^{th}$ entry in $\mathbf{B}$ and multiplying it by a random group element (see Equation 2.1). For this task, we use a bilinear pairing with a special property, that allows us to multiply every element in $\mathbf{B_1}$ and $\mathbf{B_2}$ by carefully sampled random elements from $\mathbb{G}$ and $\mathbb{H}$ (resp.). We do this in a way that when pairing elements from the two matrices to generate $\mathbf{B}$, these random factors cancel each other out, except at the $(i, 1)^{th}$ element, which will be randomly distributed. The above idea gives us the following result.

Theorem 4.3. *There exists a receiver-compact $p\ell\mathsf{OT}$ scheme, with statistical receiver privacy and sender privacy under the SXDH assumption, that has communication complexity $O(\mathsf{poly}(\lambda)\sqrt{n})$.*

4.3 Balanced Protocols Through Resuable Private Laconic OT

Having shown how to obtain receiver-compact private laconic OT from DDH and SXDH, we next describe in a high level how to transform such "unbalanced" schemes to $p\ell\mathsf{OT}$ schemes which, despite being non-receiver-compact, have lower overall communication, and in particular, give us sublinear non-interactive secure computation protocols also from DDH.

Theorem 4.4. *There exists a (non-receiver-compact) $p\ell\mathsf{OT}$ scheme, with statistical receiver privacy and sender privacy under the {DDH,SXDH}[14] assumption, that has overall communication complexity $\{O(\mathsf{poly}(\lambda)\sqrt{n}), O(\mathsf{poly}(\lambda)\sqrt[3]{n})\}$.*

How to Reuse the Sender's Message. Let us reexamine the $p\ell\mathsf{OT}$ scheme from TDH. The sender's message consists of an encoding key ek and encryptions of the two sender's secrets, each under a corresponding TDH encoding. We observe that the encoding key ek, which is actually the larger part of the sender's message, is actually independent of the hash value $\mathsf{h} = \mathsf{H}(\mathsf{hk}, \mathsf{x})$. It can therefore be *reused* for different $p\ell\mathsf{OT}$ invocations corresponding to different values of the receiver's database x and the sender's secrets $\mathsf{s}_0, \mathsf{s}_1$ (but that share the same index i).

This brings us to define a notion of *reusable* $p\ell\mathsf{OT}$, where we distinguish between two parts of the sender's message: (i) a *reusable* part, of size sublinear in n, that depends only on i and, therefore, can be reused for different inputs

[14] for the SXDH-based construction we further assume the existence of correlated-input secure hash for group-induced correlations over bilinear groups [32, 3].

$\mathsf{x}, \mathsf{s}_0, \mathsf{s}_1$, and (ii) a *compact* part, of size independent in n, that is generated w.r.t. a specific receiver's database x and sender's secrets s_0 and s_1.

As mentioned above, the $p\ell\mathsf{OT}$ construction from TDH already gives reusability, with ek being reusable. However, a subtle issue concerning the sender privacy has to be resolved. Take for instance the $p\ell\mathsf{OT}$ construction from the DDH-based TDH. Above, we argued that the encoding $\mathsf{e}_{1-\mathsf{x}[i]}$ is uniformly distributed given hk and ek, in what we called the *secret encoding property*. Notice, however, that this is not sufficient for reusable $p\ell\mathsf{OT}$, where many such encodings $\mathsf{e}_{1-\mathsf{x}[i]}$, namely symmetric encryption keys, are generated w.r.t. different values of x. Although each of these encryption keys is individually uniform, they are highly correlated. Encryption under correlated keys is clearly insecure. Thus, we do not get sender privacy when ek is reused.

We handle this issue by defining a related *reusable secret encoding* property for TDH. Both the DDH-based and pairings-based TDH schemes can be extended to have reusable secret encoding using suitable *correlated-input secure hash* [3], which can be fortunately realized under DDH and, resp., appropriate hardness assumptions over bilinear groups.

Reusable $p\ell\mathsf{OT}$ can be useful by itself for applications in secure computation, in particular when we allow to amortize the communication cost over many computations of the same functionality on different inputs. Further, as mentioned earlier, reusable $p\ell\mathsf{OT}$ turns out to be useful to achieve $p\ell\mathsf{OT}$ schemes which, although non-reusable, have smaller download communication.

Exploiting Reusability for More Efficient Schemes. Lastly, we show how to use reusable $p\ell\mathsf{OT}$ to achieve more efficient $p\ell\mathsf{OT}$ schemes. Our final results are a DDH-based $p\ell\mathsf{OT}$ with communication proportional to $\sqrt{n}$, and a pairing-based $p\ell\mathsf{OT}$ with communication proportional to $\sqrt[3]{n}$. Although the construction is generic, it is parameterized differently according to the underlying reusable $p\ell\mathsf{OT}$. For presentation, we take the DDH-based reusable $p\ell\mathsf{OT}$, where the public parameters and sender's message grow linearly in n as a special case.

The idea is as follows. We divide the receiver's database x to $\sqrt{n}$ smaller databases, $\mathsf{x}_1, \ldots, \mathsf{x}_{\sqrt{n}}$, each of size $\sqrt{n}$. Consequently, every index $j \in [n]$ is interpreted as $(j_1, j_2) \in [\sqrt{n}]^2$ (particularly, $i = (i_1, i_2)$) where $\mathsf{x}_j := \mathsf{x}_{j_1}[j_2]$. On the sender's side, each of the secrets $\mathsf{s}_0, \mathsf{s}_1$ is additively shared to $\mathsf{s}_{0,1}, \ldots, \mathsf{s}_{0,\sqrt{n}} \in \{0,1\}$ and, respectively, $\mathsf{s}_{1,1}, \ldots, \mathsf{s}_{1,\sqrt{n}} \in \{0,1\}$ s.t. $\sum_j \mathsf{s}_{j,b} = \mathsf{s}_b$ for $\mathsf{b} \in \{0,1\}$. In fact, the sender generates the shares such that $\mathsf{s}_{j,0} = \mathsf{s}_{j,1}$ for any $j \neq i_1$.

The idea is to use the underlying reusable $p\ell\mathsf{OT}$ to send to the receiver, for every $j \in [\sqrt{n}]$, either $\mathsf{s}_{j,0}$ or $\mathsf{s}_{j,1}$, conditioned on $\mathsf{x}_j[i_2]$. For any $j \neq i_1$, both bits are equal, and therefore, the receiver obtains s_j, regardless of the value of $\mathsf{x}_j[i_2]$. The only database bit that matters is $\mathsf{x}[i] := \mathsf{x}_{i_1}[i_2]$, which determines whether the receiver receives $\mathsf{s}_j := \mathsf{s}_{j,0}$, and therefore can compute $\sum_j \mathsf{s}_j = \mathsf{s}_0$, or $\mathsf{s}_j := \mathsf{s}_{j,1}$, which would allow him to compute $\sum_j \mathsf{s}_j = \mathsf{s}_1$.

5 Acknowledgements

We thank Craig Gentry, Shai Halevi, Srinath Setty, and Vinod Vaikuntanathan for helpful discussions and pointers.

S. Garg supported by DARPA/ARL SAFEWARE Award W911NF15C0210, AFOSR Award FA9550-15-1-0274, AFOSR YIP Award, DARPA and SPAWAR under contract N66001-15-C-4065, a Hellman Award and research grants by the Okawa Foundation, Visa Inc., and Center for Long-Term Cybersecurity (CLTC, UC Berkeley). The views expressed are those of the author and do not reflect the official policy or position of the funding agencies.

Y. Ishai supported by ERC Project NTSC (742754), ISF grant 1709/14, NSF-BSF grant 2015782, and a grant from the Ministry of Science and Technology, Israel and Department of Science and Technology, Government of India.

G. Malavolta supported by a gift from Ripple, a gift from DoS Networks, a grant from Northrop Grumman, a Cylab seed funding award, and a JP Morgan Faculty Fellowship.

T. Mour supported by BSF grant 2012378, and NSF-BSF grant 2015782.

R. Ostrovsky supported by NSF grant 1619348, BSF grant 2015782, DARPA SafeWare subcontract to Galois Inc., DARPA SPAWAR contract N66001-15-C-4065, JP Morgan Faculty Research Award, OKAWA Foundation Research Award, IBM Faculty Research Award, Xerox Faculty Research Award, B. John Garrick Foundation Award, Teradata Research Award, and Lockheed-Martin Corporation Research Award. The views expressed are those of the authors and do not reflect position of the Department of Defense or the U.S. Government.

References

1. William Aiello, Yuval Ishai, and Omer Reingold. Priced oblivious transfer: How to sell digital goods. In *EUROCRYPT 2001*, pages 119–135, 2001. 6
2. Benny Applebaum, Ivan Damgård, Yuval Ishai, Michael Nielsen, and Lior Zichron. Secure arithmetic computation with constant computational overhead. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 223–254, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. 7
3. Nuttapong Attrapadung, Takahiro Matsuda, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Constrained PRFs for NC^1 in traditional groups. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 543–574, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. 8, 24, 25
4. Saikrishna Badrinarayanan, Sanjam Garg, Yuval Ishai, Amit Sahai, and Akshay Wadia. Two-message witness indistinguishability and secure computation in the plain model from new assumptions. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017, Part III*, volume 10626 of *Lecture Notes in Computer Science*, pages 275–303, Hong Kong, China, December 3–7, 2017. Springer, Heidelberg, Germany. 4

5. Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 719–737, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany. 15
6. Donald Beaver. Precomputing oblivious transfer. In *CRYPTO '95*, pages 97–109, 1995. 4
7. Mihir Bellare and David Cash. Pseudorandom functions and permutations provably secure against related-key attacks. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 666–684, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Heidelberg, Germany. 8
8. Alexander R. Block, Hemanta K. Maji, and Hai H. Nguyen. Secure computation with constant communication overhead using multiplication embeddings. In *INDOCRYPT 2018*, pages 375–398, 2018. 6
9. Elette Boyle, Niv Gilboa, and Yuval Ishai. Breaking the circuit size barrier for secure computation under DDH. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016, Part I*, volume 9814 of *Lecture Notes in Computer Science*, pages 509–539, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany. 2, 5, 12
10. Elette Boyle, Niv Gilboa, and Yuval Ishai. Group-based secure computation: Optimizing rounds, communication, and computation. In *EUROCRYPT 2017, Part II*, pages 163–193, 2017. 4, 6
11. Zvika Brakerski and Nico Döttling. Two-message statistically sender-private OT from LWE. In *TCC 2018: 16th Theory of Cryptography Conference, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 370–390, Panaji, India, November 11–14, 2018. Springer, Heidelberg, Germany. 4, 6
12. Zvika Brakerski and Shafi Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability - (or: Quadratic residuosity strikes back). In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 1–20, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Heidelberg, Germany. 13
13. Zvika Brakerski, Alex Lombardi, Gil Segev, and Vinod Vaikuntanathan. Anonymous IBE, leakage resilience and circular security from new assumptions. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2018, Part I*, volume 10820 of *Lecture Notes in Computer Science*, pages 535–564, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany. 5
14. Christian Cachin, Silvio Micali, and Markus Stadler. Computationally private information retrieval with polylogarithmic communication. In Jacques Stern, editor, *Advances in Cryptology – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 402–414, Prague, Czech Republic, May 2–6, 1999. Springer, Heidelberg, Germany. 6
15. Yan-Cheng Chang. Single database private information retrieval with logarithmic communication. In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *ACISP 04: 9th Australasian Conference on Information Security and Privacy*, volume 3108 of *Lecture Notes in Computer Science*, pages 50–61, Sydney, NSW, Australia, July 13–15, 2004. Springer, Heidelberg, Germany. 6
16. Chongwon Cho, Nico Döttling, Sanjam Garg, Divya Gupta, Peihan Miao, and Antigoni Polychroniadou. Laconic oblivious transfer and its applications.

In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 33–65, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. 3, 8, 20, 21, 23

17. Benny Chor, Oded Goldreich, Eyal Kushilevitz, and Madhu Sudan. Private information retrieval. In *36th Annual Symposium on Foundations of Computer Science*, pages 41–50, Milwaukee, Wisconsin, October 23–25, 1995. IEEE Computer Society Press. 2
18. Ivan Damgård and Mats Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In Kwangjo Kim, editor, *PKC 2001: 4th International Workshop on Theory and Practice in Public Key Cryptography*, volume 1992 of *Lecture Notes in Computer Science*, pages 119–136, Cheju Island, South Korea, February 13–15, 2001. Springer, Heidelberg, Germany. 2, 4, 5
19. Itai Dinur, Nathan Keller, and Ohad Klein. An optimal distributed discrete log protocol with applications to homomorphic secret sharing. In *CRYPTO 2018, Part III*, pages 213–242, 2018. 12
20. Yevgeniy Dodis, Shai Halevi, Ron D. Rothblum, and Daniel Wichs. Spooky encryption and its applications. In *CRYPTO 2016, Part III*, pages 93–122, 2016. 7
21. Nico Döttling and Sanjam Garg. Identity-based encryption from the Diffie-Hellman assumption. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017, Part I*, volume 10401 of *Lecture Notes in Computer Science*, pages 537–569, Santa Barbara, CA, USA, August 20–24, 2017. Springer, Heidelberg, Germany. 5, 10
22. Nico Döttling, Sanjam Garg, Mohammad Hajiabadi, and Daniel Masny. New constructions of identity-based and key-dependent message secure encryption schemes. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018: 21st International Conference on Theory and Practice of Public Key Cryptography, Part I*, volume 10769 of *Lecture Notes in Computer Science*, pages 3–31, Rio de Janeiro, Brazil, March 25–29, 2018. Springer, Heidelberg, Germany. 5
23. Nelly Fazio, Rosario Gennaro, Tahereh Jafarikhah, and William E. Skeith III. Homomorphic secret sharing from paillier encryption. In *ProvSec 2017*, pages 381–399, 2017. 2
24. Sanjam Garg, Romain Gay, and Mohammad Hajiabadi. New techniques for efficient trapdoor functions and applications. Cryptology ePrint Archive, Report 2018/872, 2018. https://eprint.iacr.org/2018/872. 5, 6, 10, 11
25. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual Symposium on Foundations of Computer Science*, pages 40–49, Berkeley, CA, USA, October 26–29, 2013. IEEE Computer Society Press. 2
26. Sanjam Garg and Mohammad Hajiabadi. Trapdoor functions from the computational Diffie-Hellman assumption. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018, Part II*, volume 10992 of *Lecture Notes in Computer Science*, pages 362–391, Santa Barbara, CA, USA, August 19–23, 2018. Springer, Heidelberg, Germany. 5, 10
27. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 169–178, Bethesda, MD, USA, May 31 – June 2, 2009. ACM Press. 2

28. Craig Gentry, Jens Groth, Yuval Ishai, Chris Peikert, Amit Sahai, and Adam D. Smith. Using fully homomorphic hybrid encryption to minimize non-interative zero-knowledge proofs. *J. Cryptology*, 28(4):820–843, 2015. 2
29. Craig Gentry and Shai Halevi. Compressible fhe with applications to pir. Technical report, 2019. (personal communication). 8
30. David Goldenberg and Moses Liskov. On related-secret pseudorandomness. In Daniele Micciancio, editor, *TCC 2010: 7th Theory of Cryptography Conference*, volume 5978 of *Lecture Notes in Computer Science*, pages 255–272, Zurich, Switzerland, February 9–11, 2010. Springer, Heidelberg, Germany. 8
31. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th Annual ACM Symposium on Theory of Computing*, pages 218–229, New York City, NY, USA, May 25–27, 1987. ACM Press. 2
32. Vipul Goyal, Adam O'Neill, and Vanishree Rao. Correlated-input secure hash functions. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 182–200, Providence, RI, USA, March 28–30, 2011. Springer, Heidelberg, Germany. 8, 24
33. Jens Groth, Aggelos Kiayias, and Helger Lipmaa. Multi-query computationally-private information retrieval with constant communication rate. In *PKC 2010*, pages 107–123, 2010. 3
34. Shai Halevi and Yael Tauman Kalai. Smooth projective hashing and two-message oblivious transfer. *J. Cryptology*, 25(1):158–193, 2012. 6
35. Brett Hemenway and Rafail Ostrovsky. Extended-ddh and lossy trapdoor functions. In *Public Key Cryptography - PKC 2012 - 15th International Conference on Practice and Theory in Public Key Cryptography, Darmstadt, Germany, May 21-23, 2012. Proceedings*, pages 627–643, 2012. 19
36. Pavel Hubacek and Daniel Wichs. On the communication complexity of secure function evaluation with long output. Cryptology ePrint Archive, Report 2014/669, 2014. http://eprint.iacr.org/2014/669. 5
37. Yuval Ishai, Joe Kilian, Kobbi Nissim, and Erez Petrank. Extending oblivious transfers efficiently. In Dan Boneh, editor, *Advances in Cryptology – CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 145–161, Santa Barbara, CA, USA, August 17–21, 2003. Springer, Heidelberg, Germany. 8
38. Yuval Ishai, Eyal Kushilevitz, Sigurd Meldgaard, Claudio Orlandi, and Anat Paskin-Cherniavsky. On the power of correlated randomness in secure computation. In Amit Sahai, editor, *TCC 2013: 10th Theory of Cryptography Conference*, volume 7785 of *Lecture Notes in Computer Science*, pages 600–620, Tokyo, Japan, March 3–6, 2013. Springer, Heidelberg, Germany. 4
39. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. In *STOC 2008*, pages 433–442, 2008. 6
40. Yuval Ishai and Anat Paskin. Evaluating branching programs on encrypted data. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 575–594, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Heidelberg, Germany. 4, 6, 16, 18, 19
41. Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. Secure arithmetic computation with no honest majority. In *TCC 2009*, pages 294–314, 2009. 7
42. Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha Chandrakasan. GAZELLE: A low latency framework for secure neural network inference. In *USENIX Security Symposium*, pages 1651–1669, 2018. 4, 7

43. Venkata Koppula, Allison Bishop Lewko, and Brent Waters. Indistinguishability obfuscation for turing machines with unbounded memory. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *47th Annual ACM Symposium on Theory of Computing*, pages 419–428, Portland, OR, USA, June 14–17, 2015. ACM Press. 5
44. Eyal Kushilevitz and Rafail Ostrovsky. Replication is NOT needed: SINGLE database, computationally-private information retrieval. In *38th Annual Symposium on Foundations of Computer Science*, pages 364–373, Miami Beach, Florida, October 19–22, 1997. IEEE Computer Society Press. 2, 4, 6, 18
45. Helger Lipmaa. An oblivious transfer protocol with log-squared communication. In *ISC 2005*, pages 314–328, 2005. 2, 6
46. Moni Naor and Kobbi Nissim. Communication preserving protocols for secure function evaluation. In *33rd Annual ACM Symposium on Theory of Computing*, pages 590–599, Crete, Greece, July 6–8, 2001. ACM Press. 3, 5, 8
47. Moni Naor and Benny Pinkas. Oblivious transfer and polynomial evaluation. In *31st Annual ACM Symposium on Theory of Computing*, pages 245–254, Atlanta, GA, USA, May 1–4, 1999. ACM Press. 7, 20
48. Moni Naor and Benny Pinkas. Efficient oblivious transfer protocols. In *SODA 2001*, pages 448–457, 2001. 6
49. Tatsuaki Okamoto, Krzysztof Pietrzak, Brent Waters, and Daniel Wichs. New realizations of somewhere statistically binding hashing and positional accumulators. In Tetsu Iwata and Jung Hee Cheon, editors, *Advances in Cryptology – ASIACRYPT 2015, Part I*, volume 9452 of *Lecture Notes in Computer Science*, pages 121–145, Auckland, New Zealand, November 30 – December 3, 2015. Springer, Heidelberg, Germany. 5
50. Rafail Ostrovsky and William E. Skeith III. A survey of single-database private information retrieval: Techniques and applications. In *Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 393–411. Springer, 2007. 6
51. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 187–196, Victoria, British Columbia, Canada, May 17–20, 2008. ACM Press. 4, 6, 16, 19
52. Willy Quach, Hoeteck Wee, and Daniel Wichs. Laconic function evaluation and applications. *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 859–870, 2018. 3, 5, 7
53. Julien P. Stern. A new efficient all-or-nothing disclosure of secrets protocol. In Kazuo Ohta and Dingyi Pei, editors, *Advances in Cryptology – ASIACRYPT'98*, volume 1514 of *Lecture Notes in Computer Science*, pages 357–371, Beijing, China, October 18–22, 1998. Springer, Heidelberg, Germany. 4, 6
54. Severin Winkler and Jürg Wullschleger. On the efficiency of classical and quantum oblivious transfer reductions. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 707–723, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Heidelberg, Germany. 4
55. Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science*, pages 162–167, Toronto, Ontario, Canada, October 27–29, 1986. IEEE Computer Society Press. 2