A New Protocol for Conditional Disclosure of Secrets And Its Applications

Sven Laur[1] and Helger Lipmaa[2]

[1] Helsinki University of Technology, Finland
[2] University College London, UK

Abstract. Many protocols that are based on homomorphic encryption are private only if a client submits inputs from a limited range $\mathcal{S}$. Conditional disclosure of secrets (CDS) helps to overcome this restriction. In a CDS protocol for a set $\mathcal{S}$, the client obtains server's secret if and only if the client's inputs belong to $\mathcal{S}$ and thus the server can guard itself against malformed queries. We extend the existing CDS protocols to work over additively homomorphic cryptosystems for every set from $\mathbf{NP/poly}$. The new construction is modular and easy to apply. As an example, we derive a new oblivious transfer protocol with log-squared communication and a millionaire's protocol with logarithmic communication. We also implement private, universally verifiable and robust multi-candidate electronic voting so that all voters only transmit an encryption of their vote. The only hardness assumption in all these protocols is that the underlying public-key cryptosystem is IND-CPA secure and the plaintext order does not have small factors.

Keywords. Conditional disclosure of secrets, crypto-computing, homomorphic encryption, oblivious transfer, two-party computation.

1 Introduction

Homomorphic encryption is a powerful tool that provides efficient private implementations for many basic operations such as scalar product, oblivious transfer and oblivious polynomial evaluation. However, basic versions of these protocols without zero-knowledge proofs of correctness are secure only in a semihonest model, where all parties submit inputs from a limited range, and are not protected against malicious behaviour. Consequently, a malicious adversary can completely or partially learn the secret inputs. Conditional disclosure of secrets [GIKM00,AIR01], also known as input verification gadget [BGN05], is a protection mechanism against such attacks. Unfortunately, current solutions [AIR01,BGN05] are secure only if the plaintext space has a prime order, whereas most additively homomorphic encryption schemes have a composite plaintext order. We provide the first conditional disclosure of secrets protocol that works in conjunction with *all* currently known additively homomorphic encryption schemes. Hence, we can efficiently and more securely solve many practical problems.

Formally, we consider only two-party protocols between a client and a server, though our results can be extended to the multiparty setting. At the end of such a protocol the client should learn the desired value whereas the server should learn nothing. Our main goal is to achieve *relaxed-security*; that is, the protocol must be secure against malicious clients and semihonest servers. Such a model is widely used in current cryptographic literature [NP99,AIR01] and is well-justified in practical applications: as the number of possible service providers is relatively small compared to the clients, it is possible to force semihonest behaviour with auditing. Moreover, service providers must preserve their reputation and thus they are less likely to act maliciously.

For clarity and brevity, we state our main results in the public key model, where the client is guaranteed to know a valid secret key and the server knows the corresponding public key. The choice of the model is not too restrictive: with a proper initialisation phase all our protocols can be implemented in the standard model, see Sect. 7. On the other hand, such a model enables to prove security of parallel compositions. Composability together with our new basic construction leads to a simpler and more modular way to construct complex protocols. Shortly put, relaxed-security follows directly from the protocol design and there is no need to handcraft the proof. More precisely, we show how to decompose a protocol into elementary tasks that can be efficiently implemented with any additively homomorphic IND-CPA secure cryptosystem, provided that the plaintext order does not have unknown small factors.

In Sect. 3, we establish basic security notions and derive a necessary machinery to analyse parallel compositions. The core results of our papers are presented in Sect. 4. We note that most existing additively homomorphic protocols are based on the possibility of computing the next three basic primitives on ciphertexts: addition of ciphertexts, multiplication with a constant, and *disclose-if-equal* (DIE). In a disclose-if-equal protocol, the server obliviously releases secret β only if the client sends a valid encryption of x, where the coefficient x can be freely chosen by the server. The current cryptographic literature is full of many useful and efficient two-message protocols that are based on these three primitives. Unfortunately, the standard DIE protocol defined say in [AIR01], and then used in many subsequent papers, is secure only if the plaintext space has a prime order and thus can only be used in conjunction with the lifted ElGamal cryptosystem where one has to compute discrete logarithms to decrypt. We provide a new DIE protocol that works in conjunction with *all* currently known additively homomorphic encryption schemes. As a result, we can naturally simplify or extend many protocols that utilise the DIE functionality, e.g. [AIR01,Ste98,Lip05,BK04,FNP04,LLM05].

The rest of the paper provides many useful applications of these generic building blocks. In Sect. 5, we present a two-message protocol for conditional disclosure of secrets (CDS), where the client learns a secret β only if his message $\mathfrak{q}$ is a valid encryption of $\boldsymbol{x} \in \mathcal{S}$, where $\mathcal{S}$ is a publicly known set. Hence, the server can use β as a one-time pad to protect the protocol output, i.e., the client learns nothing unless $\mathsf{Dec}_{\mathsf{sk}}(\mathfrak{q}) \in \mathcal{S}$. The latter forms a basis of the CDS transformation that can guard any two-message protocol, where the first message is a vector of ciphertexts, against malicious clients. A slightly extended CDS construction provides an efficient solution to the millionaire problem and conditional oblivious transfer. Another extension of CDS provides a way to implement electronic voting and auctions without non-interactive zero-knowledge proofs in the multi-party setting using threshold-decryption. Finally, we compare our results with conventional cryptographic methods to provide some interesting insights and show the theoretical significance of our results, see Sect. 7.

History. The new DIE protocol, together with the CDS protocol and the CDS transformation date from August 2004 and has been available on eprint since 2005.

2 Cryptographic Preliminaries

Distributions. For a a finite set X, let $\mathcal{U}(X)$ denote the uniform distribution over X and $x \leftarrow X$ denote a uniform draw from X. Two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ over a discrete support X are statistically ε-close, $\mathcal{D}_1 \overset{\varepsilon}{\sim} \mathcal{D}_2$, if their statistical difference $\max_{S \subseteq X} |\Pr[\mathcal{D}_1 \in S] - \Pr[\mathcal{D}_2 \in S]| \leq \varepsilon$. A shorthand $\mathcal{D}_1 \equiv \mathcal{D}_2$ denotes $\mathcal{D}_1 \overset{0}{\sim} \mathcal{D}_2$.

Homomorphic encryption. A public-key cryptosystem π is defined by three algorithms. A key generation algorithm Gen returns a secret and public key pair $(\mathsf{sk}, \mathsf{pk})$. Corresponding $\mathsf{Enc}_{\mathsf{pk}}(\cdot)$ and $\mathsf{Dec}_{\mathsf{sk}}(\cdot)$ algorithms are used to encrypt and decrypt messages. Let $\mathcal{M}$ and $\mathcal{C}$ denote the corresponding message and ciphertext spaces. Then we require $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Enc}_{\mathsf{pk}}(x)) = x$ for every $x \in \mathcal{M}$ and assume that there exists efficient membership test for the ciphertext space $\mathcal{C}$. Privacy of encrypted messages is guaranteed by IND-CPA security. For any *stateful* probabilistic algorithm A, its IND-CPA advantage quantifies the ability to distinguish ciphertexts:

$$\mathsf{Adv}_{\pi}^{\text{IND-CPA}}(A) = 2 \cdot \left| \Pr\left[\begin{matrix} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, (x_0, x_1) \leftarrow A(\mathsf{pk}), i \leftarrow \{0,1\} \\ c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(x_i) : A(x_0, x_1, c) = i \end{matrix} \right] - \frac{1}{2} \right| \quad ,$$

where the probability is taken over coin tosses of all relevant algorithms. A cryptosystem π is (ε, τ)-*IND-CPA-secure* if $\mathsf{Adv}_{\pi}^{\text{IND-CPA}}(A) \leq \varepsilon$ for any τ-time adversary A.

A cryptosystem π is *additively homomorphic*, if $\mathcal{M} = \mathbb{Z}_N$ for some N, and for any $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}$ and valid messages $x_1, x_2 \in \mathcal{M}$ the distribution of products $\mathsf{Enc}_{\mathsf{pk}}(x_1) \cdot \mathsf{Enc}_{\mathsf{pk}}(x_2)$ coincides with the distribution of ciphertexts $\mathsf{Enc}_{\mathsf{pk}}(x_1 + x_2)$. To be precise, the equivalence

$$\mathsf{Enc}_{\mathsf{pk}}(x_1) \cdot \mathsf{Enc}_{\mathsf{pk}}(x_2) \equiv \mathsf{Enc}_{\mathsf{pk}}(x_1 + x_2)$$

must hold for any fixed ciphertext $\mathsf{Enc}_{\mathsf{pk}}(x_1)$. That is, given $\mathsf{Enc}_{\mathsf{pk}}(x_1) \cdot \mathsf{Enc}_{\mathsf{pk}}(x_2)$, even an unbounded adversary learns nothing beyond $x_1 + x_2$. A cryptosystem π is *multiplicatively homomorphic*, if $\mathsf{Enc}_{\mathsf{pk}}(x_1) \cdot \mathsf{Enc}_{\mathsf{pk}}(x_2) \equiv \mathsf{Enc}_{\mathsf{pk}}(x_1 \cdot x_2)$ for any $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}$ and $x_1, x_2 \in \mathcal{M}$, where $\mathcal{M}$ is a multiplicative group where computing the discrete logarithm is hard. In many practical applications, multiplicatively homomorphic cryptosystems Enc are converted to additively homomorphic cryptosystems $\overline{\mathsf{Enc}}$ by using the lifted encryption rule $\overline{\mathsf{Enc}}_{\mathsf{pk}}(x) := \mathsf{Enc}_{\mathsf{pk}}(g^x)$. Such lifted cryptosystems have reduced utility, as the new decryption rule requires computation of discrete logarithms and one can successfully decrypt only a small fraction of ciphertexts.

Many well-known homomorphic cryptosystems are IND-CPA secure under reasonable complexity assumptions, e.g. [Elg85,Pai99,DJ01]. Existing additively homomorphic cryptosystems have a composite plaintext order with large factors. For example, the plaintext order of the Paillier cryptosystem [Pai99] is an RSA modulus and thus its smallest prime factor is approximately $\sqrt{N}$. The Goldwasser-Micali cryptosystem [GM82] is the only known exception, as it is additively homomorphic over $\mathbb{Z}_2$. Such plaintext space is too small for many applications. All known cryptosystems with a large prime plaintext order are multiplicative, e.g., the ElGamal cryptosystem [Elg85].

3 Basic Properties of Two-Message Protocols

Throughout the paper, we consider two-message protocols where a client sends a query $\mathfrak{q}$ to a server that replies with $\mathfrak{a}$, and then the client computes a desired output from $\mathfrak{a}$. The server should learn nothing about the query. The client should learn $f(\boldsymbol{\alpha}, \boldsymbol{\beta})$, where $\boldsymbol{\alpha}$ denotes client's private input vector and $\boldsymbol{\beta}$ denotes server's private input vector. Mostly, we consider the relaxed-security against unbounded clients and computationally bounded servers, but sometimes we consider also the setting where both parties are computationally bounded. A protocol is *correct* if the client always recovers $f(\boldsymbol{\alpha}, \boldsymbol{\beta})$ when both parties are honest. A priori we do not assume correctness from all protocols, as sometimes it is sufficient to know that a client cannot learn anything beyond $f(\boldsymbol{\alpha}, \boldsymbol{\beta})$.

In the simplest case, the query $\mathfrak{q}$ consists of encrypted inputs $(\alpha_1, \ldots, \alpha_m)$ and the server uses properties of additively homomorphic encryption to compose an appropriate reply. We call such protocols *additively homomorphic two-message protocols.* Here, we explicitly assume that the server knows public key pk and thus can efficiently verify that the query consists of valid ciphertexts and ignore malformed queries. Notably, many interesting tasks can be solved with additively homomorphic two-message protocols. Computationally-private information retrieval [AIR01,Ste98,Lip05], solutions to millionaire's problem [BK04,Fis01], and various protocols for privacy-preserving data mining tasks [FNP04,WY04,GLLM04] form only a small set of such protocols.

Relaxed-security in the PKI model. As usual, we define security by comparing the real and ideal model. However, we explicitly assume that the client knows the secret key, the server knows the corresponding public key and only the client can deviate from the protocol specification. Formally, a trusted key generator initially runs the key generation algorithm Gen for a cryptosystem π, and then privately sends $(\mathsf{sk}, \mathsf{pk})$ to the client and pk to the server. In particular, the server knows that pk corresponds to this fixed client. This key pair is then possibly used in many different protocol runs.

Note that the PKI model is normal and even desirable in many applications, e.g. e-voting. Still, we stress that we use the PKI model only for the sake of simplicity of security proofs. In Sect. 7, we show how to replace the trusted key generator by a key transfer protocol with a marginal degradation of security.

Since the server obtains no output and is always semihonest, we can decompose the standard security definition into two orthogonal requirements: client-privacy and server-privacy. A two-message protocol is (ε, τ)*-client-private*, if for any τ-time stateful adversary A, the next inequality holds:

$$2 \cdot \left| \Pr\left[\begin{matrix} (\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, (\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_1) \leftarrow A(\mathsf{pk}), \\ i \leftarrow \{0, 1\}, \mathfrak{q} \leftarrow \mathfrak{q}_{\mathsf{pk}}(\boldsymbol{\alpha}_i) : A(\boldsymbol{\alpha}_0, \boldsymbol{\alpha}_1, \mathfrak{q}) = i \end{matrix} \right] - \frac{1}{2} \right| \leq \varepsilon \enspace ,$$

where $\mathfrak{q}_{\mathsf{pk}}(\boldsymbol{\alpha}_i)$ denotes the first message computed by the honest client. Server-privacy has a slightly more complicated definition, since we must transform any efficient adversary from the real world to an efficient adversary in the ideal model, where a trusted third party (TTP) computes $f(\boldsymbol{\alpha}, \boldsymbol{\beta})$. Hence, the definition incorporates a simulator Sim and a distinguisher B and we need to explicitly quantify their efficiency. The simulator

Sim gets $(\mathsf{sk}, \mathfrak{q})$ as an input and can send $\boldsymbol{\alpha}^*$ once to the TTP. Then Sim obtains the value of $f^* = f(\boldsymbol{\alpha}^*, \boldsymbol{\beta})$ and can proceed with the simulation. For brevity, let us define

$$p_{\mathrm{r}} = \Pr\left[(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, (\boldsymbol{\beta}, \mathfrak{q}) \leftarrow A(\mathsf{sk}), \mathfrak{a} \leftarrow \mathfrak{a}_{\mathsf{pk}}(\mathfrak{q}, \boldsymbol{\beta}) : B(\boldsymbol{\beta}, \mathfrak{q}, \mathfrak{a}) = 1\right] \ ,$$
$$p_{\mathrm{i}} = \Pr\left[(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}, (\boldsymbol{\beta}, \mathfrak{q}) \leftarrow A(\mathsf{sk}), \hat{\mathfrak{a}} \leftarrow \mathsf{Sim}_{\mathsf{sk}}(\mathfrak{q}, f^*) : B(\boldsymbol{\beta}, \mathfrak{q}, \hat{\mathfrak{a}}) = 1\right] \ ,$$

where $\mathfrak{a}(\mathfrak{q}, \boldsymbol{\beta})$ denotes the answer of the honest server with the input $\boldsymbol{\beta}$ to the query $\mathfrak{q}$. A protocol implements $(\tau, \delta, t, \varepsilon)$-*server-privately* a function f, if for any τ-time adversary A there exists a $(t + \delta)$-time simulator Sim such that $|p_{\mathrm{r}} - p_{\mathrm{i}}| \leq \varepsilon$ for any t-time distinguisher B. In the information-theoretical setting, algorithms A, Sim and B are unbounded. A protocol is ε-*server-private* if for any adversary A there exists a simulator Sim such that their output distributions are statistically ε-close. We say that a protocol is $(\varepsilon_1, \tau; \varepsilon_2)$-*relaxed-secure* if it is (ε_1, τ)-client-private and ε_2-server-private. Relaxed-security is widely used standard security assumption, see [NP99,AIR01].

Extractability and simulatability. Usually, the client-privacy follows directly from security assumptions. For example, additively homomorphic protocols are client-private by the construction, provided that the cryptosystem is IND-CPA secure. Proofs of server-privacy can be significantly simplified by considering the following notions of extractability and simulatability. As client can be malicious, the simulator Sim must somehow deduce the intended input $\boldsymbol{\alpha}^*$. In the PKI model, the simulator can use sk to determine the input $\boldsymbol{\alpha}^*$ directly from $\mathfrak{q}$. A two-message protocol is *extractable* if there exists an efficient algorithm $\mathsf{Ext}_{\mathsf{sk}}(\cdot)$ such that $\mathsf{Ext}_{\mathsf{sk}}(\mathfrak{q}_{\mathsf{pk}}(\boldsymbol{\alpha})) = \boldsymbol{\alpha}$ for all valid inputs and $\mathsf{Ext}_{\mathsf{sk}}(\mathfrak{q}) = \perp$ for all invalid queries $\mathfrak{q}$ that do not correspond to any input.

In many protocols, the server's reply can be perfectly or almost perfectly simulated knowing only the corresponding client's output f^* and a secret key sk. We formalise this as simulatability. Consider a protocol transcript $(\mathfrak{q}, \mathfrak{a})$ between the honest client and server. Let $f^* = f(\boldsymbol{\alpha}, \boldsymbol{\beta})$ be the corresponding client's output. Then the server's reply is ε_2-*simulatable* if there exists an efficient algorithm $\mathsf{Sim}^*_{\mathsf{sk}}$ such that the output distributions $(\mathfrak{q}, \mathfrak{a})$ and $(\mathfrak{q}, \hat{\mathfrak{a}})$ are statistically ε_2-close even for a fixed $\mathfrak{q}$, where $\hat{\mathfrak{a}} \leftarrow \mathsf{Sim}^*_{\mathsf{sk}}(\mathfrak{q}, f^*)$. The notion of (t, ε_2)-simulatability is defined analogously. Extractability together with simulatability implies server-privacy:

Theorem 1. *If a two-message protocol is extractable, ε_2-simulatable and the server ignores malformed queries, then the protocol is also ε_2-server-private in the PKI model.*

Proof. We construct a universal simulator Sim as follows. If the query $\mathfrak{q}$ is malformed then the simulator ignores it. Otherwise, Sim extracts the intended input $\boldsymbol{\alpha}^* \leftarrow \mathsf{Ext}_{\mathsf{sk}}(\mathfrak{q})$ and sends $\boldsymbol{\alpha}^*$ to the TTP. Given the reply $f^* = f(\boldsymbol{\alpha}^*, \boldsymbol{\beta})$ from the TTP, the simulator uses $\mathsf{Sim}^*_{\mathsf{sk}}(\mathfrak{q}, f^*)$ to simulate the reply $\hat{\mathfrak{a}}$. Since malformed queries are discarded in both worlds, the distributions $(\boldsymbol{\beta}, \mathfrak{q}, \mathfrak{a})$ and $(\boldsymbol{\beta}, \mathfrak{q}, \hat{\mathfrak{a}})$ are statistically ε_2-close. $\square$

Forked composition. We can use Thm. 1 to prove that a parallel composition of extractable and simulatable protocols preserves server-privacy. It makes sense to consider protocols that share the query phase as we can always merge different queries into a single query. Let two-message protocols $\Pi_1, \dots, \Pi_s$ share the first message $\mathfrak{q}$. Then the *forked composition* $\mathsf{Forked}[\Pi_1, \dots, \Pi_s]$ is defined as follows:

1. The client computes the query $\mathfrak{q}$ and sends it to the server.
2. The server uses $\mathfrak{q}$ to compute replies $\mathfrak{a}_1, \ldots, \mathfrak{a}_s$ according to $\Pi_1, \ldots, \Pi_s$.
3. The server sends $\mathfrak{a}_1, \ldots, \mathfrak{a}_s$ to the client.
4. The client computes the private output $(f_1, \ldots, f_s)$ according to $\Pi_1, \ldots, \Pi_s$.

It is easy to prove that a client can learn nothing beyond $f_1(\boldsymbol{\alpha}, \boldsymbol{\beta}), \ldots, f_s(\boldsymbol{\alpha}, \boldsymbol{\beta})$.

Theorem 2. *Let $\Pi_1, \ldots, \Pi_s$ be extractable and respectively ε_i-simulatable implementations of functionalities f_i. Then the composition $\mathsf{Forked}[\Pi_1, \ldots, \Pi_s]$ is an extractable and $(\varepsilon_1 + \cdots + \varepsilon_s)$-simulatable implementation of the functionality $f = (f_1, \ldots, f_s)$.*

Proof. Extractability is clear. By the definition of simulatability, there exist simulators $\mathsf{Sim}^*_{\mathsf{sk},i}$ that output simulated replies $\hat{\mathfrak{a}}_i$ such that $(\mathfrak{q}, \mathfrak{a}_i)$ and $(\mathfrak{q}, \hat{\mathfrak{a}}_i)$ are statistically ε_i-close even for fixed $\mathfrak{q}$. Now, define a simulator $\mathsf{Sim}^*_{\mathsf{sk}}$ that given $\mathfrak{q}$ and $f^* = (f_1(\boldsymbol{\alpha}^*, \boldsymbol{\beta}), \ldots, f_s(\boldsymbol{\alpha}^*, \boldsymbol{\beta}))$ runs $\mathsf{Sim}^*_{\mathsf{sk},i}(\mathfrak{q}, f_i^*)$ for $i \in \{1, \ldots, s\}$ and outputs $\hat{\mathfrak{a}}_1, \ldots, \hat{\mathfrak{a}}_s$. By the construction, the distributions $(\mathfrak{q}, \mathfrak{a}_1, \ldots, \mathfrak{a}_s)$ and $(\mathfrak{q}, \hat{\mathfrak{a}}_1, \ldots, \hat{\mathfrak{a}}_s)$ are statistically $(\varepsilon_1 + \cdots + \varepsilon_s)$-close even for a fixed $\mathfrak{q}$ and the simulatability follows. □

Reducing communication further with CPIR. In many two-message protocols, the client must access only a short part of the reply $\mathfrak{a}$ to recover the output $f(\boldsymbol{\alpha}, \boldsymbol{\beta})$ whereas the rest of $\mathfrak{a}$ consists of random noise. Hence, we can significantly decrease the total communication $|\mathfrak{q}| + |\mathfrak{a}|$, if the client could fetch only useful parts of $\mathfrak{a}$. The latter can be done using *computationally private information retrieval* (CPIR). In a 1-out-of-n CPIR protocol, the server maintains a database $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_n)$ of ℓ-bit strings and the client can fetch β_i so that a computationally bounded server learns nothing. The basic properties of CPIR protocols are determined by parameters n and ℓ. It is trivial to achieve communication complexity $\Theta(n\ell)$ just by sending the whole database so one considers only CPIR protocols with sublinear communication. There is a wide range of such protocols. Recent protocols achieve communication that is low-degree polylogarithmic in the database size, see [Lip05,GR05] for further references.

Now, assume that the server's reply has a structure $\mathfrak{a} = (\mathfrak{a}_1, \ldots, \mathfrak{a}_n)$ and the client needs to recover at most t elements. Then the client can use t parallel CPIR queries to fetch desired parts $\mathfrak{a}_{i_1}, \ldots, \mathfrak{a}_{i_t}$. Note that the CPIR queries can be sent together with the protocol Π messages, provided that the CPIR instance is run independently from Π or joining queries does not decrease client-privacy. Server-privacy cannot decrease, as the replies of CPIR queries are computed from the original reply $\mathfrak{a}$.

4 Three Basic Crypto-Computing Primitives

"Crypto-computing" is often used to describe two-message protocols, where a server uses some basic operations on client's garbled inputs to compute reply that reveals only $f(\boldsymbol{\alpha}, \boldsymbol{\beta})$. The first comparative study [SYY99] showed how to crypto-compute predicates with logarithmic circuit depth using the Goldwasser-Micali cryptosystem. Later, this construction was somewhat generalised to compute the greater-than predicate [Fis01]. Here, we provide three basic crypto-computing primitives for additively homomorphic cryptosystems with large factors of the plaintext space. Note that the server can crypto-compute ciphertexts of sums and products with one public factor obliviously from ciphertexts, as

$$\mathsf{Enc}_{\mathsf{pk}}(x_1 + x_2) \equiv \mathsf{Enc}_{\mathsf{pk}}(x_1) \cdot \mathsf{Enc}_{\mathsf{pk}}(x_2) \cdot \mathsf{Enc}_{\mathsf{pk}}(0) \quad (1)$$

$$\mathsf{Enc}_{\mathsf{pk}}(x \cdot y) \equiv \mathsf{Enc}_{\mathsf{pk}}(y)^x \cdot \mathsf{Enc}_{\mathsf{pk}}(0) \quad (2)$$

hold by the definition of additively homomorphic cryptosystems. Here the multiplication by $\mathsf{Enc}_{\mathsf{pk}}(0)$ is necessary to re-randomise the replies.

But there is also a third generic operation that implicitly "tests" whether a ciphertext c is an encryption of x. The existence of this operation depends additionally on the order of the plaintext group. More precisely, a *disclose-if-equal* (DIE) protocol allows releasing of a secret β only if $\mathsf{Dec}_{\mathsf{sk}}(c) = x$ where the server can freely choose x. The idealised functionality of DIE protocol is defined as follows

$$f(\alpha, \beta) = \begin{cases} \beta, & \text{if } \alpha = x \enspace , \\ \bot, & \text{if } \alpha \neq x \enspace . \end{cases}$$

The simplest implementation of DIE protocol was given in the paper [AIR01]:

1. The client sends $c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\alpha)$ to the server.
2. If $c \in \mathcal{C}$ then the server sends a reply $\mathfrak{a} \leftarrow (c \cdot \mathsf{Enc}_{\mathsf{pk}}(-x))^r \cdot \mathsf{Enc}_{\mathsf{pk}}(\beta)$ for $r \leftarrow \mathcal{M}$.
3. The client outputs $\mathsf{Dec}_{\mathsf{sk}}(\mathfrak{a}) = (\alpha - x)r + \beta$.

If the plaintext space has a prime order, then $(\alpha - x)r$ has uniform distribution over $\mathcal{M}$ when $x \neq \alpha$. Consequently, the protocol is perfectly simulatable: if $f(\alpha, \beta) = \bot$ a simulator should output a random encryption $\mathsf{Enc}_{\mathsf{pk}}(m)$ for $m \leftarrow \mathcal{M}$ and $\mathsf{Enc}_{\mathsf{pk}}(\beta)$ otherwise. Therefore, the basic DIE protocol is also relaxed-secure.

On the other hand, the protocol is not correct, since the client obtains a random output when $\mathsf{Dec}_{\mathsf{sk}}(c) \neq x$. If x is public then the correctness is not an issue, as the client knows whether $\mathsf{Dec}_{\mathsf{sk}}(c) = x$ or not. Otherwise, the construction guarantees only that the client learns nothing about β when $\mathsf{Dec}_{\mathsf{sk}}(c) \neq x$. Moreover, if the server sets the first k-bits of β to 0, then the honest client can detect $\alpha \neq x$ with failure probability 2^{-k}, i.e., there is a trade-off between reliability and throughput.

Unfortunately, the basic DIE protocol is not secure if the message space has a composite order. As an example, consider the Paillier cryptosystem, where $N = pq$ is an RSA modulus. If a malicious client sends $c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(p + x)$ then $\mathsf{Dec}_{\mathsf{sk}}(\mathfrak{a}) = \beta + rp \mod N$ and the client can recover $\beta \mod p$ although $\mathsf{Dec}_{\mathsf{sk}}(c) \neq x$. Since the DIE protocol is a building block in many existing protocols, then such leakage might cause a domino effect that can completely reveal server's input. For example, the circuit CDS protocol in Sect. 5 is extremely vulnerable against such attacks. Therefore, we devise a new DIE protocol that works in conjunction with all currently known additively homomorphic cryptosystems. As a result, we can naturally simplify many protocols [AIR01,Ste98,Lip05,BK04,FNP04,LLM05] that use the lifted ElGamal cryptosystem or zero-knowledge correctness proofs to guarantee security of the DIE protocol.

New general construction for DIE. Server-privacy of the basic DIE protocol hinges on the fact that $\alpha\mathbb{Z}_N = \{\alpha r : r \in \mathbb{Z}_N\} = \mathbb{Z}_N$ for any $\alpha \neq 0$. If the message space

Query phase:
The client sends $\mathfrak{q} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\alpha)$ to the server.
Transfer phase:
If the ciphertext is invalid $\mathfrak{q} \notin \mathcal{C}$ then the server returns $\perp$.
Otherwise, the server returns $\mathfrak{a} \leftarrow (c \cdot \mathsf{Enc}_{\mathsf{pk}}(-x))^r \cdot \mathsf{Enc}_{\mathsf{pk}}(\mathsf{encode}(\beta))$ for $r \leftarrow \mathcal{M}$.
Post-processing:
The client computes $y = \mathsf{Dec}_{\mathsf{sk}}(\mathfrak{a})$ and returns $\mathsf{decode}(y)$.

Protocol 1: Disclose-if-equal protocol of ℓ-bit secrets for the constraint $\mathsf{Dec}_{\mathsf{sk}}(\mathfrak{q}) = x$

contains non-trivial additive subgroups (ideals) $\{0\} \neq \mathbb{G} \subsetneq \mathbb{Z}_N$ then the client can choose a ciphertext c so that the reply $\mathfrak{a}$ enables to restore the coset $\beta + \mathbb{G}$. Consequently, a malicious client can learn up to $\log_2 N - \log_2 \Phi$ bits of information, where Φ is the minimal size of the non-trivial subgroup $\mathbb{G}$. To seal the leakage, we must use a probabilistic encoding for β such that the total entropy of $\mathbb{G}$ together with the encoding $\mathsf{encode}(\beta)$ is roughly $\log_2 N$. Let us define an encoding for ℓ-bit strings

$$\mathsf{encode}(\beta) = \beta + 2^\ell \cdot t \mod N \quad \text{for} \quad t \leftarrow \mathbb{Z}_T \ ,$$
$$\mathsf{decode}(y) = (y \mod N) \mod 2^\ell \ ,$$

where $T = \lfloor 2^{-\ell} \cdot N \rfloor$ and $\ell < \lfloor \log_2 N \rfloor$. As there are no modular wrappings, the decoding is always correct. More importantly, Prot. 1 is now secure for small enough ℓ.

Theorem 3. *Let π be an additively homomorphic cryptosystem such that the smallest factor of the plaintext order is larger than $\gamma > 2$. Then Protocol 1 for transferring ℓ-bit strings is extractable and $(2^{\ell-1}/\gamma)$-simulatable.*

Proof. Extractability is clear and thus we consider only simulatability. If $\alpha \neq x$, then by construction $y = \mathsf{encode}(\beta) + g$ where g is chosen uniformly from a non-zero subgroup $\mathbb{G} \subseteq \mathbb{Z}_N$. If $\mathbb{G} = \mathbb{Z}_N$ then y is uniformly distributed over $\mathbb{Z}_N$. Otherwise $\mathbb{G}$ can be represented as $p\mathbb{Z}_N$, where p is a non-trivial factor of N, and $y \mod p \equiv \beta + 2^\ell \cdot t \mod p$, where $t \leftarrow \mathbb{Z}_T$ and $T = \lfloor 2^{-\ell} \cdot N \rfloor$. Since 2 and p are relatively prime, $\{2^\ell \cdot t : t \in \mathbb{Z}_p\} = \mathbb{Z}_p$ and the term $2^\ell \cdot t \mod p$ covers all elements of $\mathbb{Z}_p$ almost uniformly. More precisely, the elements of $\mathbb{Z}_p$ can be divided into two sets:

$$\mathcal{T}_0 = \left\{c \in \mathbb{Z}_p : \Pr[\beta + 2^\ell \cdot t \mod p = c] = \tfrac{a}{T}\right\} \qquad \text{with} \qquad |\mathcal{T}_0| = p - b \ ,$$
$$\mathcal{T}_1 = \left\{c \in \mathbb{Z}_p : \Pr[\beta + 2^\ell \cdot t \mod p = c] = \tfrac{a+1}{T}\right\} \qquad \text{with} \qquad |\mathcal{T}_1| = b \ ,$$

where $a = \left\lfloor \frac{T}{p} \right\rfloor$ and $b = T - ap$. Consequently, the statistical difference between $y \mod p$ and the uniform distribution $\mathcal{U}(\mathbb{Z}_p)$ can be expressed as

$$\varepsilon = \frac{|\mathcal{T}_0|}{2} \cdot \left(\frac{1}{p} - \frac{a}{T}\right) + \frac{|\mathcal{T}_1|}{2} \cdot \left(\frac{a+1}{T} - \frac{1}{p}\right) = \frac{b(p-b)}{Tp} \leq \frac{p}{4T} \leq \frac{N}{4\gamma T} \ ,$$

as $p(p-b) \leq p^2/4$ and $p \leq N/\gamma$. Since $2^{\ell+1} \leq N$ we get $T = \lfloor 2^{-\ell} N \rfloor \geq N/2^{\ell+1}$ and thus $\varepsilon \leq 2^{\ell-1}/\gamma$. Now note that the distributions $\mathsf{encode}(\beta) + \mathcal{U}(p\mathbb{Z}_N)$ and $\mathcal{U}(\mathbb{Z}_N)$

are still ε-close, as we can express

$$\Pr\left[\mathsf{encode}(\beta) + \mathcal{U}(p\mathbb{Z}_N) = c \mod N\right] = \frac{p}{N} \cdot \Pr\left[\mathsf{encode}(\beta) = c \mod p\right] \ .$$

Hence, we can use a simulator $\mathsf{Sim}^*_{\mathsf{sk}}(\mathfrak{q}, f^*)$ that outputs $\mathsf{Enc}_{\mathsf{pk}}(\mathsf{encode}(\beta))$ if $f^* = \beta$ and $\mathsf{Enc}_{\mathsf{pk}}(m)$ for $m \leftarrow \mathbb{Z}_N$ otherwise. □

Corollary 1. *Let π be an (τ, ε_1)-IND-CPA-secure additively homomorphic cryptosystem such that the smallest factor of the plaintext order is larger than $\gamma > 2$. Then Protocol 1 for transferring ℓ-bit strings is $(\tau, \varepsilon_1; \varepsilon_2)$-relaxed-secure for $\varepsilon_2 = 2^{\ell-1}/\gamma$.*

The Maximal Throughput of DIE Protocol. First, note that if we want to achieve ε-server-privacy then we must choose $\ell = \lfloor \log_2(2\varepsilon\gamma) \rfloor$, where γ is the lower bound to non-trivial factors of N. Usually, it is sufficient to take $\varepsilon = 2^{-80}$ and thus N cannot have smaller factors than 2^{80} if the server wants to release Boolean secrets. For the Paillier cryptosystem the smallest factor of N is approximately $\sqrt{N}$, and consequently, one can transfer $\ell = \lfloor \log_2(2\sqrt{N}\varepsilon) \rfloor \approx 0.5 \log_2 N + \log_2 \varepsilon$ bits. For standard 1024-bit RSA modulus and $\varepsilon = 2^{-80}$, one can take $\ell = 433$.

As our DIE protocol is extractable and simulatable, a forked composition of t protocols enables transfer of a $t\ell$-bit secret, where the achieved server-privacy is $\frac{|\beta|}{\ell\gamma} \cdot 2^{\ell-1}$. Smaller values of ℓ increase the maximal length of β but also decrease the ratio between the desired communication $|\beta|$ and the total communication $|\mathfrak{q}| + |\mathfrak{a}|$ and make the protocol less efficient. In other words, a bad encoding $\mathsf{encode}(\beta)$ with a small capacity can significantly decrease efficiency. As our target distribution is $\mathcal{U}(\mathbb{Z}_N)$ then it is straightforward to derive entropy bounds for the capacity: $H(\mathbb{Z}_N) \approx H(\mathsf{encode}(\beta) + p\mathbb{Z}_N) \leq H(\mathsf{encode}(\beta)) + H(p\mathbb{Z}_N) \leq \log_2 |\mathsf{encode}(\beta)| + H(p\mathbb{Z}_N)$, where $|\mathsf{encode}(\beta)|$ denotes the size of the support. As the encoding must be uniquely decodable, the capacity of a single reply $\ell \leq \log_2 \frac{N}{|\mathsf{encode}(\beta)|} \lesssim \min_p H(p\mathbb{Z}_n) = \log_2 \Phi$, where Φ is the smallest prime factor of N. Thus, the encoding is optimal up to a constant additive term $\log_2 \varepsilon$. The result can be generalised for any target distribution using a more detailed analysis.

5 Generic Construction for Conditional Disclosure of Secrets

Many protocols are secure only if client submits inputs $\boldsymbol{\alpha}$ from a limited range $\mathcal{S}$. Cleverly chosen $\boldsymbol{\alpha} \notin \mathcal{S}$ can either partially or completely reveal the server's input $\boldsymbol{\beta}$. Therefore, the server must somehow verify that $\boldsymbol{\alpha} \in \mathcal{S}$. Classically, this is done by a zero-knowledge proof that $\mathsf{Dec}_{\mathsf{sk}}(\boldsymbol{c}) \in \mathcal{S}$. However, this either increases the number of messages or requires a security model with a common reference string or random oracles. A conditional disclosure of secrets (CDS) reaches the same goal without extra messages and exotic assumptions. In a CDS protocol, the client should learn a secret β only if $\mathsf{Dec}_{\mathsf{sk}}(\mathfrak{q}) \in \mathcal{S}$, where the query vector $\mathfrak{q}$ consists of ciphertexts $\mathsf{Enc}_{\mathsf{pk}}(\alpha_1), \ldots, \mathsf{Enc}_{\mathsf{pk}}(\alpha_m)$ and the set $\mathcal{S}$ is public. Since the server can use β as a one-time pad to encrypt the original reply $\mathfrak{a}$, the client learns nothing about the outputs of the original protocol if $\boldsymbol{\alpha} \notin \mathcal{S}$ and the modified protocol becomes server-private.

A CDS protocol can be straightforwardly constructed as a forked composition of individual DIE protocols for $\{\mathsf{Dec}_{\mathsf{sk}}(c) = x\}_{x \in \mathcal{S}}$ that share the same secret β but such composition is inefficient. Therefore, we show how to use Benaloh-Leichter secret sharing scheme [BL88] together with slightly extended DIE protocols to achieve a more computation and communication efficient CDS protocol (*circuit CDS*).

Conjunctive affine zero tests. First, we present an optimisation for specific sets. Recall that our DIE protocol is secure since $\mathsf{encode}(\beta) + \mathcal{U}(\mathbb{G}) \overset{\varepsilon}{\sim} \mathcal{U}(\mathbb{Z}_N)$ if $\mathbb{G} \neq \{0\}$. Similarly, we can construct CDS protocols for *conjunctive affine zero tests* $\Psi_0(\boldsymbol{\alpha}) = \bigwedge_{j=1}^{v}[\sum_{i=1}^{m} s_{ij}\alpha_i \overset{?}{=} x_j]$, where $\{x_i\}$ and $\{s_{ij}\}$ are public constants:

1. The client sends $\mathfrak{q} = (c_1, \ldots, c_m)$ where $c_i = \mathsf{Enc}_{\mathsf{pk}}(\alpha_i)$.
2. The server halts if some $c_1, \ldots, c_n$ is not a valid ciphertext, otherwise it replies $\mathfrak{a} = \prod_{j=1}^{v} \left(\prod_{i=1}^{m} c_i^{s_{ij}} \cdot \mathsf{Enc}_{\mathsf{pk}}(-x_j)\right)^{r_j} \cdot \mathsf{Enc}_{\mathsf{pk}}(\mathsf{encode}(\beta))$ for $r_1, \ldots, r_v \leftarrow \mathbb{Z}_N$.
3. The client restores $y = \mathsf{Dec}_{\mathsf{sk}}(\mathfrak{a})$ and outputs $\mathsf{decode}(y)$.

As $y = \sum_{j=1}^{v} (\sum_{i=1}^{m} \alpha_i s_{ij} - x_i)\, r_j + \mathsf{encode}(\beta) = \mathsf{encode}(\beta) + \mathbb{G}_1 + \cdots + \mathbb{G}_v$, then $y = \mathsf{encode}(\beta) + \mathcal{U}(\mathbb{G})$ for a non-zero sub-group $\mathbb{G}$ if some zero-tests do not hold. The latter follows from the fact that $r_1, \ldots, r_v$ are independently chosen. Hence, the claims of Thm. 3 hold also for the CDS protocol given above. Of course, when the plaintext order is prime then there is no need to use probabilistic encoding and we can use the construction given in [AIR01]. Notably, such simplified construction has been used in [BGN05] together with a cryptosystem that has a composite plaintext order. Paradoxically, the latter construction is still computationally secure, as the client must compute arbitrary discrete logarithms to recover a coset $\beta + \mathbb{G}$.

Circuit CDS protocol. For any set $\mathcal{S}$, we can write the predicate $\Psi_{\mathcal{S}}(\boldsymbol{\alpha}) := [\boldsymbol{\alpha} \in \mathcal{S}]$ as a monotonous combination of affine zero tests, i.e., the formula consists of Boolean operations $\wedge$ and $\vee$ together with atomic terms $\Psi_0(\boldsymbol{\alpha}) = \bigwedge_{j=1}^{v}[\sum_{i=1}^{m} s_{ij}\alpha_i \overset{?}{=} x_j]$. For efficiency reasons, we might express the input $\boldsymbol{\alpha}$ as a bit-vector. The server can later use properties of additively homomorphic encryption to restore the original ciphertexts.

First, the server uses the Benaloh-Leichter secret sharing scheme to assign sub-secrets β_i to each leaf test $\Psi_0(\boldsymbol{\alpha})$ so that the client can reconstruct the secret $\beta \in \{0,1\}^{\ell}$ if $\Psi(\boldsymbol{\alpha})$ holds and the secrets of true leaves are revealed. Fig. 1 illustrates how secret β is propagated through the circuit of $\Psi(\alpha) = [\alpha > x]$ without optimisation. Namely, the master secret β is assigned to the topmost gate of the circuit. For every $\vee$-gate, the output secret is just pushed downwards. For every $\wedge$-gate ψ with u children and a secret β_ψ assigned to it, sub-secrets $\beta_1, \ldots, \beta_{u-1} \leftarrow \{0,1\}^{\ell}$ and $\beta_u \leftarrow \beta_\psi - \beta_1 - \cdots - \beta_{u-1} \mod 2^{\ell}$ are assigned to the children. One can also use threshold operations: $\textsc{Thr}_v(x_1, \ldots, x_s) = 0$ if and only if at least v values x_j are equal to 1. For a $\textsc{Thr}_v$ gate, generate a random $(v-1)$-degree polynomial f_ψ with $f_\psi(0) = \beta_\psi$ and assign the secret $f_\psi(i)$ to its ith child. Finally, the server uses a forked composition of CDS protocols for leaf tests Ψ_0 to release sub-secrets associated to each leaf. The client recomputes the secret from leaf values by inversely following the secret generation.

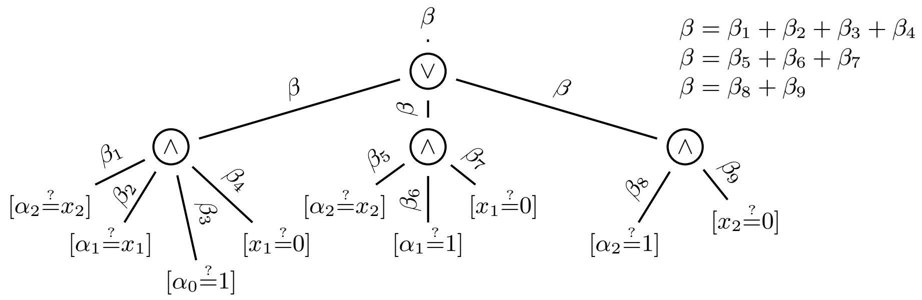

Fig. 1. An unoptimised circuit for $\Psi(\alpha) = [\alpha > x]$ where secrets are pushed down to DIE leafs. The circuit can be further optimised by replacing $\wedge$-gates with conjunctive affine equality tests

Theorem 4. *If the leaf CDS protocol is extractable and ε_2-simulatable, then the circuit CDS protocol for $\Psi_{\mathcal{S}}$ is extractable and $\mathcal{L}(\Psi_{\mathcal{S}}) \cdot \varepsilon_2$-simulatable, where $\mathcal{L}(\Psi_{\mathcal{S}})$ is the number of leaves. If the cryptosystem is (τ, ε_1)-IND-CPA secure and $\mathfrak{q}$ consists of m ciphertexts, then the protocol is $(\tau - \mathcal{O}(1), m\varepsilon_1; \mathcal{L}(\Psi_{\mathcal{S}}) \cdot \varepsilon_2)$-relaxed-secure.*

Proof. Given the main secret β it is straightforward to reconstruct the leaf-level secrets. Otherwise, if $\Psi_{\mathcal{S}}(\boldsymbol{\alpha}) = 0$ then the sub-secrets β_i that are assigned to true atoms $\Psi_0(\boldsymbol{\alpha}) = 1$ are independent and are uniformly distributed. Hence, a world with $\mathcal{L}(\Psi_{\mathcal{S}})$ ideally implemented leaf CDS protocols can be perfectly simulated in the world where β is released only if $\boldsymbol{\alpha} \in \mathcal{S}$. Now, the simulatability follows directly from Thm. 2. The second claim follows from Thm. 1 and the basic properties of IND-CPA encryption. □

If the CDS protocol is based on the new DIE protocol, then we can estimate how many bits are needed to transfer ℓ-bit secrets. For the 1024-bit Paillier cryptosystem and 2^{-80}-sever-privacy, a single ciphertext can fit 393 bits provided that the corresponding circuit has less than 2^{40} leaves; the message expansion is roughly $|\mathfrak{a}| / \ell \approx 5.2 \cdot \mathcal{L}(\Psi)$.

As negations can be expressed by conjunctive affine zero tests, then they can appear only in the leaf level, i.e., the formula $\Psi(\boldsymbol{\alpha})$ must be in a negation normal form (NNF). Many practically interesting sets have compact NNF-s, but for some circuits Ψ such normal form is exponentially larger. We can circumvent the problem by using auxiliary inputs $\boldsymbol{w}$. Consider the circuit representation of Ψ that consists of unary $\neg$-gates and binary $\wedge$- and $\vee$-gates. Denote all output wires of logical gates by auxiliary labels w_i. Now, we can represent assignments $w_u \leftarrow w_s \wedge w_t$ and $w_u \leftarrow w_s \vee w_t$ with the formulae

$$[w_u \stackrel{?}{=} 1] \wedge [w_s \stackrel{?}{=} 1] \wedge [w_t \stackrel{?}{=} 1] \vee [w_u \stackrel{?}{=} 0] \wedge [w_s \stackrel{?}{=} 0] \vee [w_u \stackrel{?}{=} 0] \wedge [w_s \stackrel{?}{=} 0] \enspace .$$
$$[w_u \stackrel{?}{=} 0] \wedge [w_s \stackrel{?}{=} 0] \wedge [w_t \stackrel{?}{=} 0] \vee [w_u \stackrel{?}{=} 1] \wedge [w_s \stackrel{?}{=} 1] \vee [w_u \stackrel{?}{=} 1] \wedge [w_s \stackrel{?}{=} 1] \enspace ,$$

and $w_u \leftarrow \neg w_s$ as $[w_u \stackrel{?}{=} 0] \wedge [w_s \stackrel{?}{=} 1] \vee [w_u \stackrel{?}{=} 1] \wedge [w_s \stackrel{?}{=} 0]$. Therefore, we can in principle construct a new formula $\overline{\Psi}(\boldsymbol{\alpha}, \boldsymbol{w})$ in NNF such that $\Psi(\boldsymbol{\alpha}) = 1 \iff \exists \boldsymbol{w} : \overline{\Psi}(\boldsymbol{\alpha}, \boldsymbol{w}) = 1$ and the size of $\overline{\Psi}(\boldsymbol{\alpha}, \boldsymbol{w})$ is proportional to the gate count of $\Psi(\boldsymbol{\alpha})$. Consequently, we can always construct efficient circuit CDS protocols for efficiently recognisable sets.

Query phase:
The client sends $\mathfrak{q} = (c_1, \dots, c_m)$ to the server, where $c_i \leftarrow \mathsf{Enc}_{\mathsf{pk}}(\alpha_i)$ for $i \in \{1, \dots, m\}$.

Transfer phase:
The server computes the reply $\mathfrak{a} = (d_1, \dots, d_n)$ according to the original protocol Π
The server applies one-time pad $e_i \leftarrow d_i \cdot \mathsf{Enc}_{\mathsf{pk}}(t_i)$ for $t_i \leftarrow \mathbb{Z}_N$ and $i \in \{1, \dots, m\}$.
The server computes the CDS reply $\mathfrak{a}_{\mathrm{cds}}$ for $\mathsf{Dec}_{\mathsf{sk}}(\mathfrak{q}) \in \mathcal{S}$ with secret a $\beta = t_1 \| \dots \| t_m$.
The server replies $(e_1, \dots, e_n)$ and $\mathfrak{a}_{\mathrm{cds}}$.

Post-processing:
The client recovers the secrets t_i from $\mathfrak{a}_{\mathrm{cds}}$ and computes $\hat{d}_i \leftarrow e_i \cdot \mathsf{Enc}_{\mathsf{pk}}(-t_i)$.
Next, the client proceeds with the original protocol Π.

Protocol 2: CDS transformation for additively homomorphic two-message protocols

CDS Transformation. It is straightforward to use a CDS protocol to transform any additively homomorphic two-message protocol that is secure in the semihonest model to a modified two-message protocol that is relaxed-secure. Let the query $\mathfrak{q}$ consist of m ciphertexts $(c_1, \dots, c_m)$. Protocol Π is secure in the semihonest model, when there exist a set of valid inputs $\mathcal{S}$ such that the client learns only $f(\boldsymbol{\alpha}, \boldsymbol{\beta})$, provided that $\mathsf{Dec}_{\mathsf{sk}}(\mathfrak{q}) = (\alpha_1, \dots, \alpha_m) \in \mathcal{S}$. Let us use a sufficiently long secret β as a one-time pad to decrypt the original reply $\mathfrak{a}$ and release the secret only if $\mathsf{Dec}_{\mathsf{sk}}(\mathfrak{q}) \in \mathcal{S}$. Then the corresponding protocol is clearly relaxed-secure. In many cases, the reply $\mathfrak{a}$ consists of re-randomised ciphertexts and we can reduce the length of the secret β, see Prot. 2.

Theorem 5. *If the two-message additively homomorphic protocol Π is correct, (τ, ε_1)-client-private and ε_2-simulatable for $\boldsymbol{\alpha} \in \mathcal{S}$ and the CDS protocol for the set $\mathcal{S}$ is ε_3-simulatable, then Protocol 2 is correct and $(\tau, \varepsilon_1; \max\{\varepsilon_2, \varepsilon_3\})$-relaxed-secure.*

Proof. Due to the re-randomisation, the recovered replies $\hat{d}_i$ have the same distribution as d_i, thus correctness is evident. Client-privacy is evident as both protocols share the query $\mathfrak{q}$. For server-privacy, note that if $\boldsymbol{\alpha} \in \mathcal{S}$, we can first use the original simulator to simulate d_i and then apply the CDS transformation to the simulation output. The corresponding simulation is ε_2-close to the real run, since the original reply is not more than ε_2 away from the simulated one. Otherwise, $(e_1, \dots, e_n)$ are random ciphertexts and thus perfectly simulatable. Now if we add a simulated CDS reply $\hat{\mathfrak{a}}_{\mathrm{cds}}$, then the aggregated reply $\hat{\mathfrak{a}}_{\mathrm{cds}}, e_1, \dots, e_n$ is ε_3-close to the real protocol transcript, as the CDS is ε_3-simulatable. The claim follows, as $\mathsf{Dec}_{\mathsf{sk}}(\mathfrak{q})$ is either in $\mathcal{S}$ or not. □

Optimisations. If all replied ciphertexts of the original protocol are in the fixed range, i.e., $\mathsf{Dec}_{\mathsf{sk}}(d_i) \in \{0,1\}^{\ell}$ then full recovery of t_i is not necessary. It is sufficient to send $t_i \mod 2^{\ell}$ together with a extra bit needed to indicate a possible wrapping $t_i \geq N - 2^{\ell}$ and the message expansion rate can be less than $\mathcal{L}(\Psi)$. Secondly, note that the communication overhead of the CDS transformation is linear in $|\mathfrak{a}|$. Therefore, the transformation is quite inefficient when $|\mathfrak{a}|$ is long. To get better performance, the server can use symmetric encryption to garble the original reply and a CDS protocol to release the corresponding key. The output is still computationally simulatable and thus we achieve computational server-privacy. A block cipher in counter mode is the best encryption method, as then the client can efficiently decrypt only necessary parts of $\mathfrak{a}$.

6 Practical Applications of Crypto-Computing Techniques

In this section, we show how to use additively homomorphic two-message protocols to solve several important cryptographic tasks. Here, the query $\mathfrak{q}$ is a vector of ciphertexts, and the reply is computed by combining the identities (1) and (2) with Prot. 1. Note that the outputs of crypto-computed sums and products are perfectly simulatable provided that the end result is re-randomised. Consequently, client-privacy follows form (τ, ε_1)-IND-CPA security and server-privacy follows from the basic properties of forked composition, see Thm. 1 and 2. Shortly put, the resulting protocols are $(\tau - \mathcal{O}(1), m\varepsilon_1; n\varepsilon_2)$-relaxed-secure, where m is the number of ciphertexts and n is the number of DIE instances, provided that the basic DIE protocol is ε_2-simulatable.

Sometimes we must also prove that knowledge of $f(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is equivalent to the knowledge of $f_1(\boldsymbol{\alpha}, \boldsymbol{\beta}), \ldots, f_s(\boldsymbol{\alpha}, \boldsymbol{\beta})$, i.e., design a protocol for f based on generic operations. As for 1024-bit Paillier and 2^{-80}-server-privacy, we can transfer 393 bits in the individual DIE reply whenever the number of DIE instances is less than 2^{40}, the resulting protocols are really efficient.

Oblivious Transfer. Recall that a 1-*out-of-*n *oblivious transfer* (OT) protocol implements an ideal functionality $f(\alpha; \beta_1, \ldots, \beta_n) = \beta_\alpha$ if $\alpha \in \{1, \ldots, n\}$ and $\perp$ otherwise. Already in [AIR01], the authors showed that such a protocol can be expressed as a forked composition of n individual DIE protocols:

- release β_1 if $\alpha = 1$,

  $\vdots$
- release β_n if $\alpha = n$.

Therefore, we get a relaxed-secure implementation of oblivious transfer by using Prot. 1 to implement all instances of DIE protocols. Moreover, a client can use any CPIR protocol to obliviously choose the αth reply of the DIE. Hence, we have just described a generic transformation from any CPIR to a relaxed-secure oblivious transfer.

An alternative approach was taken by Chang [Cha04] who proved that the basic DIE protocol from [AIR01] leaks at most $\beta_{\alpha_1} \mod p_1$ and $\beta_{\alpha_2} \mod p_2$ whenever the plaintext order is a product of two primes p_1 and p_2. In the corresponding 1-out-of-n OT protocol an honest client has to encrypt values that depend on the secret key and thus the client-privacy does not follow directly from IND-CPA security.

Millionaire's protocol with logarithmic communication. The millionaire's problem is: given client's private input α and server's private input x, decide whether $\alpha > x$. Although numerous solutions have been proposed for this problem, none of the proposals is completely satisfactory. For example, the two-message protocol of Blake and Kolesnikov [BK04] is server-secure only in the semihonest model since encrypted inputs must be in correct range, it can leak information otherwise. To solve that type of problems, consider a circuit CDS protocol for a public set $\mathcal{S}_x = \{\alpha \in \{0,1\}^m : \alpha > x\}$. Writing α bit by bit $(\alpha_{m-1}, \dots, \alpha_0)$, we obtain

$$\begin{aligned}\Psi_{\mathcal{S}_x}(\alpha) =& ([\alpha_{m-1} \stackrel{?}{=} 1] \wedge [x_{m-1} \stackrel{?}{=} 0]) \vee \\ & ([\alpha_{m-1} \stackrel{?}{=} x_{m-1}] \wedge [\alpha_{m-2} \stackrel{?}{=} 1] \wedge [x_{m-2} \stackrel{?}{=} 0]) \vee \\ & ([\alpha_{m-1} \stackrel{?}{=} x_{m-1}] \wedge [\alpha_{m-2} \stackrel{?}{=} x_{m-2}] \wedge [\alpha_{m-3} \stackrel{?}{=} 1] \wedge [x_{m-3} \stackrel{?}{=} 0]) \vee \dots \vee \\ & ([\alpha_{m-1} \stackrel{?}{=} x_{m-1}] \wedge [\alpha_{m-2} \stackrel{?}{=} x_{m-2}] \wedge \dots \wedge [\alpha_1 \stackrel{?}{=} x_1] \wedge [\alpha_0 \stackrel{?}{=} 1] \wedge [x_0 \stackrel{?}{=} 0]) \enspace .\end{aligned}$$

Here, every row corresponds to one conjunctive affine equality test. Fig. 1 depicts the corresponding unoptimised circuit. Now consider the modified protocol where β_0 is a publicly fixed ℓ-bit secret and the server randomly reorders the leaf CDS replies $\mathfrak{a}_1, \dots, \mathfrak{a}_m$. Finally, the client outputs 1 if one of the recovered CDS outputs is β_0. As the formula $\Psi_{\mathcal{S}_x}(\alpha)$ is a disjunction of affine zero tests, then in the ideal world, the client learns a randomly shuffled set $\{\beta_0, \perp, \dots, \perp\}$ if $\alpha > x$ and $\{\perp, \perp, \dots, \perp\}$ otherwise. Hence, the modified protocol is server-private even if x is private and we have obtained a relaxed-secure solution to the millionaire problem that fails with probability $2^{-\ell}$. The total communication of our solution is $2m$ ciphertexts, the client's computation is $\Theta(m)$ and the server's computation is $\Theta(m^2)$, and we only assume that the underlying additively homomorphic cryptosystem is IND-CPA secure. The server's workload can be reduced $\Theta(m)$ as in the Blake-Kolesnikov protocol, if we first crypto-compute a recursion $t_i = (\alpha_i - x_i)r_i + \dots + (\alpha_{m-1} - x_{m-1})r_{m-1}$ for $r_i \leftarrow \mathbb{Z}_N$ and then re-randomise it by crypto-computing $u_i = t_i s_i$ for $s_i \leftarrow \mathbb{Z}_N$.

Interestingly enough, one can view our solution as an efficient generalisation of the Fischlin protocol [Fis01]. The latter can be alternatively described as a CDS protocol based on additively homomorphic cryptosystem over $\mathbb{Z}_2$. Due to the small message space, the Fischlin's protocol requires a parallel run of ℓ protocols to achieve the same reliability as our protocol, i.e., our protocol is ℓ times more efficient.

Conditional OT. In a *conditional oblivious transfer* protocol for public predicate Ψ, the client has a private input α and the server has a private input (x, β_0, β_1). The client obtains β_1 if $\Psi(\alpha, x) = 1$ and β_0 otherwise. Assume that the master secret β is reconstructed identically for the circuits without witnesses Ψ and $\neg\Psi$ and the reconstruction process and the number of true leaves leaks nothing about x except $\Psi(\alpha, x)$. In particular, assume that the master secret can be reconstructed from randomly shuffled shares. Let $\mathcal{B}_{\Psi(\alpha,x)}$ and $\mathcal{B}_{\neg\Psi(\alpha,x)}$ be the shuffled CDS replies in the ideal world. Then given a shuffled set of sets $\{\mathcal{B}_{\Psi(\alpha,x)}, \mathcal{B}_{\neg\Psi(\alpha,x)}\}$, one can learn only $\beta_{\Psi(\alpha,x)}$ and nothing more, provided that the number of leaf tests is equal $|\mathcal{B}_{\Psi(\alpha,x)}| = |\mathcal{B}_{\neg\Psi(\alpha,x)}|$.

This leads to the following COT protocol. First, the server assigns β_0 to $\neg\Psi$ and β_1 to Ψ and adds trailing zeroes to leaf secrets of one circuit and trailing ones to the remaining sub-secrets. Next, the server constructs replies for each leaf CDS and sends randomly shuffled replies back. Finally, the client restores sets $\mathcal{B}_\Psi(\alpha, x)$ and $\mathcal{B}_{\neg\Psi(\alpha,x)}$ and reconstructs $\beta_{\Psi(\alpha,x)}$. The failure probability is bounded by $2^{-k} \cdot \mathcal{L}(\Psi)$ where k is the number of trailing zeroes and ones. Since $[\alpha > x]$ and $[\alpha \leq x]$ have such symmetrical circuits, we can construct a COT protocol for $[\alpha > x]$ and for many other relations.

Electronic voting and auctions without random oracles. E-voting and auction protocols based on homomorphic encryption [CGS97,DJ01,LAN02] are natural extensions of homomorphic two-message protocols, since the secret key is known by the election tallier (or a coalition of talliers) to whom the server forwards the second message. In such protocols, conditional disclosure of secrets can be used to guarantee security of the election authority against malicious voters and a semihonest server. As in [BGN05], consider an electronic voting protocol where every voter sends an encryption $c_i \leftarrow \mathsf{Enc}_{\mathsf{pk}}(v_i)$ to talliers. We assume that the protocol is secure if $v_i \in \mathcal{S}$ for some publicly known set $\mathcal{S}$; this is true in typical e-voting protocols [CGS97,DJ01].

In the existing protocols, it is usually assumed that every voter accompanies his or her vote with a non-interactive zero-knowledge proof that $v_i \in \mathcal{S}$. Instead, the talliers can jointly apply the CDS protocol, with output secret 0, to c_i (this can be done very efficiently if $\mathcal{S}$ is the set of powers of a fixed integer) and then threshold-decrypt the result. If the plaintext is equal to 0, talliers accept the vote as correct. Of course, every step of the talliers has to be accompanied by a zero-knowledge proof of correctness (to each other and to every possible outside observer), but since the number of talliers is significantly smaller than the number of voters, this is doable in practise, see [BGN05].

As the result, we get a voter-private, universally verifiable and robust e-voting scheme where the voters only have to perform one encryption, assuming only that there exists an IND-CPA secure additively homomorphic public-key cryptosystem. The same trick can be used to eliminate the need for random oracles in a similar electronic auction scheme of [LAN02] and in many other similar protocols. Compared to the protocols of [BGN05], our protocols are more efficient since they are based on genuine additive homomorphic cryptosystem whereas [BGN05] uses a lifted version of ElGamal and thus there one has to compute discrete logarithms. Moreover, their cryptosystem is secure under less established security assumptions.

Multiplicative relations and polynomial arithmetic Finally, we illustrate the power of using auxiliary witnesses. It is well known that multiplicative relation $[z \stackrel{?}{=} xy]$ does not have a compact NNF. However, we can still construct efficient circuit CDS protocol by introducing a suitable witness $\boldsymbol{w}$. Let $x, y \in \{0,1\}^m$ and $z \in \{0,1\}^{2m}$ be sent to the server by individually encrypting each bit of x, y, z and let $w_0, \dots, w_{m-1}$ be auxiliary variables such that $w_i = xy_i$. Then $xy = w_0 + 2w_1 + \cdots + 2^{m-1}w_{m-1}$ and the formula $\Psi_{[z=xy]}$ can be expressed as a conjunction of tests: (1) $x_{m-1}, \dots, x_0 \in \{0,1\}$, (2) $[y_i \stackrel{?}{=} 0] \wedge [w_i \stackrel{?}{=} 0] \vee [y_i \stackrel{?}{=} 1] \wedge [w_i \stackrel{?}{=} x]$ for $i \in \{0, \dots, m-1\}$ and x is crypto-computed as $x_0 + \cdots + 2^{m-1}x_{m-1}$, and (3) $[z \stackrel{?}{=} w_0 + \cdots + 2^{m-1}w_{m-1}]$.

Several papers, see e.g. [KS05], use additively homomorphic two-message protocols in a setting where one encrypts the coefficients of some polynomials, where the important quantity is the set of roots of this polynomial. For example, if F_1 is the set of roots of $f_1(x)$ and F_2 is the set of roots of $f_2(x)$ then $F_1 \cup F_2$ is the set of roots of $f_1(x) \cdot f_2(x)$. Consequently, we can also construct a CDS protocol for the set to prove that $g(x) = f_1(x) \cdot f_2(x)$, as the ith coefficient $g_i = f_{10}f_{2i} + \cdots + f_{1i}f_{20}$. Now, we can also verify that for some sets F_1, F_2 and G, it holds that $F_1 \cup F_2 = G$.

7 Theoretical Implications

Although we stated our results in the PKI model, where a trusted key generator generates a key pair $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{Gen}$ and privately transfers $(\mathsf{sk}, \mathsf{pk})$ to the client and pk to the server, they can be easily implemented in the standard model. Namely, we can eliminate the PKI assumption if the client executes once, separately and in an isolated manner (that is, no other messages of different protocols are sent by the client at the same time), with every server a zero-knowledge proof of knowledge that pk is valid and that he knows the corresponding secret key. This is followed by the real protocol. In the security proof, the simulator extracts the secret key by rewinding and thereafter continues to work as previously. Since we require statistical server-security—and thus can use an unbounded simulator—then it is actually sufficient to have a zero-knowledge proof that the key is correct: the simulator just computes the secret key corresponding to the (correct) public key. It is even irrelevant whether the client computes the public key with a correct distribution, since for the proof we only need the existence of the secret key. Therefore, the amortised message complexity is still two-messages in the standard model, as the verification of a public key must be carried out only once.

It is well known that secure two-party protocols require at least three messages, therefore, it is impossible to obtain full security of two-message protocols in the malicious model. In fact, one cannot achieve more than relaxed-security in two messages even in the PKI model. Consequently, the CDS-transformation presented in Sect. 5 is a universal round-optimal transformation from semihonest model to relaxed-secure model whenever the first message contains only ciphertexts. Moreover, computational and communication resources are linear in the size of the circuit that is needed to test a validity of an input. More formally, assume that for sets $\mathcal{S}_m$ of m-bit strings exists a polynomial-size formula $\Psi(\boldsymbol{\alpha}, \boldsymbol{w})$ such that $\boldsymbol{\alpha} \in \mathcal{S}_m$ iff $\exists \boldsymbol{w} : \Psi(\boldsymbol{\alpha}, \boldsymbol{w}) = 1$. Then there exists also a polynomial-size formula $\overline{\Psi}(\boldsymbol{\alpha}, \overline{\boldsymbol{w}})$ in a negation normal form such that $\boldsymbol{\alpha} \in \mathcal{S}_m$ iff $\exists \overline{\boldsymbol{w}} : \overline{\Psi}(\boldsymbol{\alpha}, \overline{\boldsymbol{w}}) = 1$. Therefore, there exist a family of polynomial-time CDS protocols for an arbitrary set $\mathcal{S}$ in $\mathbf{NP/poly}$. Such protocols can be automatically generated in polynomial time for every set $\mathcal{S}$ that can be described by any $\mathbf{NP}$ relation.

Alternative classical round-preserving methods that guard against malicious clients are based on non-interactive zero-knowledge proofs, i.e., we have to either rely on random oracles or use the *common reference string* (CRS) model. While CRS is a plausible model for protocol design, constructing efficient non-interactive zero-knowledge protocols for $\mathbf{NP}$ in the CRS model has been a long-standing open problem. Thus, our result is also appealing from the complexity-theoretical viewpoint.

As stated already in Sect. 6, the DIE-based OT protocol leads to a general transformation from CPIR to information-theoretically server-private OT, as the client can use the CPIR protocol to fetch only the answer of the αth DIE protocol. In particular, there exists a generic CPIR construction for any IND-CPA secure additively homomorphic cryptosystem [Ste98] with sublinear-but-superpolylogarithmic communication. Therefore, there exists also an OT protocol with comparable communication under the sole assumption that IND-CPA secure additively homomorphic cryptosystems exists. Under the assumption that IND-CPA secure length-flexible additively homomorphic cryptosystem exist, one can construct a CPIR protocol [Lip05] with communication $\Theta(\mathbf{k} \cdot \log^2 n + \ell \cdot \log n)$ where $\mathbf{k}$ is the security parameter. Consequently, we can construct an OT with communication $\Theta(\mathbf{k}\cdot\log^2 n+\ell\cdot\log n)$, if an IND-CPA secure length-flexible additively homomorphic cryptosystem exists. Finally due to the results of Gentry and Ramzan [GR05], there also exists an OT protocol with optimal communication $\Theta(\log n+\ell+\mathbf{k})$, if we assume that Φ-Hiding is hard and that an IND-CPA secure additively homomorphic cryptosystem exists.

Another two-message OT protocol was proposed by Kalai [Kal05]. Her protocol is secure in the standard model, whereas our protocol requires a zero-knowledge proof that the public key is valid. On the other hand, the query of Kalai's protocol does not consist of ciphertexts and thus cannot be used for the CDS protocol. Moreover, Thm. 3 holds even with incorrectly formed pk provided that the corresponding encryption rule is additively homomorphic and it is still possible to detect invalid ciphertexts. Therefore, we can omit the zero-knowledge proofs for pk provided that we can verify that the plaintext order does not have too small factors. For small enough γ and public plaintext order this can be done efficiently by using Lenstra's Elliptic Curve Method, see App. A for further details. Hence, it is possible to achieve two messages as non-amortised round-complexity in the standard model under stronger computational assumptions.

Finally, note that small detectable factors of N can be effectively eliminated. Namely, a server can eliminate a known factor p by multiplying a ciphertext $\mathsf{Enc}_{\mathsf{pk}}(x)$ with $\mathsf{Enc}_{\mathsf{pk}}(pr)$ for $r\leftarrow\mathbb{Z}_N$. Then the client can learn only a coset $x+p\mathbb{Z}_N$, i.e., we have established a new cryptosystem over a new message space $\mathbb{Z}_N/p\mathbb{Z}_n\simeq\mathbb{Z}_{N/p}$.

Acknowledgements. We would like to thank Phil Carmody, Yuval Ishai and Vladimir Kolesnikov for useful comments. The work was partially supported by the Finnish Academy of Sciences and by the Estonian Science Foundation, grant 6848.

References

[AIR01] William Aiello, Yuval Ishai, and Omer Reingold. Priced Oblivious Transfer: How to Sell Digital Goods. In *Advances in Cryptology — EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, 2001. Springer-Verlag.

[BGN05] Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. Evaluating 2-DNF Formulas on Ciphertexts. In *The Second Theory of Cryptography Conference, TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, 2005. Springer Verlag.

[BK04] Ian F. Blake and Vladimir Kolesnikov. Strong Conditional Oblivious Transfer and Computing on Intervals. In *Advances on Cryptology — ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*, 2004. Springer-Verlag.

[BL88] Josh Benaloh and Jerry Leichter. Generalized Secret Sharing and Monotone Functions. In *Advances in Cryptology—CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, 1988. Springer-Verlag, 1990.

[CGS97] Ronald Cramer, Rosario Gennaro, and Berry Schoenmakers. A Secure and Optimally Efficient Multi-Authority Election Scheme. In *Advances in Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, 1997. Springer-Verlag.

[Cha04] Yan-Cheng Chang. Single Database Private Information Retrieval with Logarithmic Communication. In *The 9th Australasian Conference on Information Security and Privacy (ACISP 2004)*, volume 3108 of *Lecture Notes in Computer Science*, 2004. Springer-Verlag.

[DJ01] Ivan Damgård and Mads Jurik. A Generalisation, a Simplification and Some Applications of Paillier's Probabilistic Public-Key System. In *Public Key Cryptography 2001*, volume 1992 of *Lecture Notes in Computer Science*, 2001. Springer-Verlag.

[Elg85] Taher Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4), 1985.

[Fis01] Marc Fischlin. A Cost-Effective Pay-Per-Multiplication Comparison Method for Millionaires. In *Topics in Cryptology - CT-RSA 2001, The Cryptographer's Track at RSA Conference 2001*, volume 2020 of *Lecture Notes in Computer Science*, 2001. Springer-Verlag.

[FNP04] Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient Private Matching and Set Intersection. In *Advances in Cryptology — EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, 2004. Springer-Verlag.

[GIKM00] Yael Gertner, Yuval Ishai, Eyal Kushilevitz, and Tal Malkin. Protecting Data Privacy in Private Information Retrieval Schemes. *Journal of Computer and System Sciences*, 60(3), June 2000.

[GLLM04] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. On Private Scalar Product Computation for Privacy-Preserving Data Mining. In *Information Security and Cryptology - ICISC 2004*, volume 3506 of *Lecture Notes in Computer Science*, 2004. Springer-Verlag.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic Encryption and How to Play Mental Poker Keeping Secret All Partial Information. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, 1982. ACM.

[GR05] Craig Gentry and Zulfikar Ramzan. Single-Database Private Information Retrieval with Constant Communication Rate. In *The 32nd International Colloquium on Automata, Languages and Programming, ICALP 2005*, volume 3580 of *Lecture Notes in Computer Science*, 2005. Springer-Verlag.

[Kal05] Yael Tauman Kalai. Smooth Projective Hashing and Two-Message Oblivious Transfer. In *Advances in Cryptology — EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, 2005. Springer-Verlag.

[KS05] Lea Kissner and Dawn Song. Privacy-Preserving Set Operations. In *Advances in Cryptology — CRYPTO 2005, 25th Annual International Cryptology Conference*, volume 3621 of *Lecture Notes in Computer Science*, 2005. Springer-Verlag.

[LAN02] Helger Lipmaa, N. Asokan, and Valtteri Niemi. Secure Vickrey Auctions without Threshold Trust. In *Financial Cryptography — Sixth International Conference*, volume 2357 of *Lecture Notes in Computer Science*, 2002. Springer-Verlag.

[Len87] Hendrik W. Lenstra, Jr. Factoring integers with Elliptic Curves. *Annals of Mathematics*, 126(2), 1987.

[Lip05] Helger Lipmaa. An Oblivious Transfer Protocol with Log-Squared Communication. In *The 8th Information Security Conference (ISC'05)*, volume 3650 of *Lecture Notes in Computer Science*, 2005. Springer-Verlag.

[LLM05] Sven Laur, Helger Lipmaa, and Tanel Mielikäinen. Private Itemset Support Counting. In *Information and Communications Security, 7th International Conference, ICICS 2005*, volume 3783 of *Lecture Notes in Computer Science*, 2005. Springer-Verlag.

[NP99] Moni Naor and Benny Pinkas. Oblivious Transfer and Polynomial Evaluation. In *Proceedings of the Thirty-First Annual ACM Symposium on the Theory of Computing*, 1999. ACM Press.

[Pai99] Pascal Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *Advances in Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, 1999. Springer-Verlag.

[Ste98] Julien P. Stern. A New and Efficient All or Nothing Disclosure of Secrets Protocol. In *Advances on Cryptology — ASIACRYPT '98*, volume 1514 of *Lecture Notes in Computer Science*, 1998. Springer-Verlag.

[SYY99] Tomas Sander, Adam Young, and Moti Yung. Non-Interactive CryptoComputing For NC^1. In *40th Annual Symposium on Foundations of Computer Science*, 1999. IEEE Computer Society.

[WY04] Rebecca N. Wright and Zhiqiang Yang. Privacy-Preserving Bayesian Network Structure Computation on Distributed Heterogeneous Data. In *Proceedings of The Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004. ACM.

[ZD06] Paul Zimmermann and Bruce Dodson. 20 Years of ECM. In *The Algorithmic Number Theory Symposium*, volume 4076 of *Lecture Notes in Computer Science*, 2006. Springer-Verlag.

[Zim06b] Paul Zimmermann. Optimal Parameters for ECM. Available at `http://www.loria.fr/~zimmerma/records/ecm/params.html`, as of May, 2006.

A Non-interactive Partial Public Key Validation

Next, we propose another technique to transform the proposed protocols to be secure in the the standard model. It does not need extra messages but needs an extra amount of computations by an honest server. Namely, Thm. 3 holds even with incorrectly formed pk provided that the corresponding encryption rule is additively homomorphic and it is still possible to detect invalid ciphertexts. In particular, the Paillier cryptosystem is homomorphic even if a public modulus N is incorrectly formed. Thus, the verification of pk can just consist of computing a lower bound γ on factors of N. For small enough γ this can be done efficiently by using Lenstra's Elliptic Curve Method [Len87] which works in time $\exp((\sqrt{2}+o(1))\sqrt{\ln p \cdot \ln \ln p})$ where p is the smallest factor of N [ZD06]. If we want the server's computation to be polynomial in $\log N$ then we have to take a sufficiently small ℓ. To provide some concrete numbers note that ECM allows "efficient" detection of 88-bit factors. Assume that the desired server-privacy level is 2^{-40}. Such a choice of ε_2 is most probably sufficient in practise. Then, in the case of the DIE protocol, one has $\ell = 47$, which is sufficient for several applications. In Spring 2006, we verified this approach by using the suggested optimal parameters from [Zim06b], on an AMD Athlon 64 3000+ processor by using the GMP-ECM software. As an example, if $N = pq$, where p is an 88-bit prime and q is an $(1024 - 88)$-bit prime then one has to run the ECM algorithm on an expected 206 curves with bounds $B1 = 50\,000$ and $B2 = 5\,000\,000$. Testing on one curve with these parameters takes approximately 2.5 seconds, and thus testing that the smallest factor is greater than 2^{89} takes 9 minutes on average. On the other hand, if q is an 66-bit prime then it takes an expected 77 curves with bounds $B1 = 11\,000$ and $B2 = 1\,100\,000$. On the same platform, testing one curve with these parameters takes approximately 0.66 seconds and checking the bound 2^{67} takes 51 seconds on average. Given the advances in the ECM, we would expect the quoted timings to decrease dramatically over the next few years.