Cianciullo L, Ghodosi H. Unconditionally secure oblivious polynomial evaluation: A survey and new results. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 37(2): 443–458 Mar. 2022. DOI 10.1007/s11390-022-0878-6

Unconditionally Secure Oblivious Polynomial Evaluation: A Survey and New Results

Louis Cianciullo and Hossein Ghodosi

College of Science and Engineering, James Cook University, Townsville, QLD 4811, Australia

E-mail: louis.cianciullo@my.jcu.edu.au; hossein.ghodosi@jcu.edu.au

Received September 23, 2020; accepted January 20, 2022.

Abstract Oblivious polynomial evaluation (OPE) is a two-party protocol that allows a receiver, $\mathcal{R}$ to learn an evaluation $f(\alpha)$, of a sender, $\mathcal{S}$'s polynomial $(f(x))$, whilst keeping both α and $f(x)$ private. This protocol has attracted a lot of attention recently, as it has wide ranging applications in the field of cryptography. In this article we review some of these applications and, additionally, take an in-depth look at the special case of information theoretic OPE. Specifically, we provide a current and critical review of the existing information theoretic OPE protocols in the literature. We divide these protocols into two distinct cases (three-party and distributed OPE) allowing for the easy distinction and classification of future information theoretic OPE protocols. In addition to this work, we also develop several modifications and extensions to existing schemes, resulting in increased security, flexibility and efficiency. Lastly, we also identify a security flaw in a previously published OPE scheme.

Keywords oblivious polynomial evaluation, unconditionally secure, information theoretic

1 Introduction

Oblivious polynomial evaluation (OPE) was first discovered in 1999 by Naor and Pinkas [1]. Similar to oblivious transfer [2], OPE involves two participants, a sender $\mathcal{S}$ and a receiver $\mathcal{R}$. An OPE protocol allows $\mathcal{R}$ to privately learn an evaluation of a polynomial held by $\mathcal{S}$, in such a way that neither the evaluation point nor the polynomial is revealed publicly.

Definition 1 [3,4]. *An OPE protocol is composed of two parties, $\mathcal{S}$ who has a polynomial $f(x)$ over a finite field $\mathbb{F}$ and $\mathcal{R}$ who has an input value $\alpha \in \mathbb{F}$. Correctness is achieved if, at the end of the protocol, $\mathcal{R}$ learns $f(\alpha)$. Security is guaranteed if the following two conditions are met after the protocol has been executed.*

1) *$\mathcal{S}$ cannot reduce the sender's uncertainty of α.*

2) *$\mathcal{R}$ does not learn any information related to $f(x)$, other than $f(\alpha)$.*

An extensive amount of research has been conducted on this protocol since it was first discovered [5–11]. This valuable tool has been used in protocols such as multi-party computation (MPC) [3,12], secure mean computation [13], oblivious neural learning [4,14], oblivious keyword search [15], and privacy preserving data mining [16], to name a few. In fact, OPE is an integral part of many protocols utilised in modern cryptography. In general we can divide these protocols into two categories.

1) *Multi-Party Computation* (*MPC*) *Protocols.* MPC allows a set of n participants to securely compute any given function over their privately-held information. More formally, a set of n participants, $P_1, \cdots, P_n$ with respective private inputs, $x_1, \cdots, x_n$, can compute a given function $f(x_1, \cdots, x_n)$ without revealing any information related to their inputs.

2) *Privacy Preserving Protocols.* We choose to use this term to refer to protocols that solve a specific function or problem, with the same level of privacy utilised in MPC. We can actually view these sorts of protocols, and OPE itself, as a subset of MPC protocols.

1.1 OPE and Multi-Party Computation

Many of the recently proposed MPC protocols within the literature ([12, 17–19]) utilise OPE as an offline protocol for the generation of correlated random

Survey

data. To clarify this statement, in such protocols an MPC is split into two phases.

1) *Offline Phase.* In this phase participants compute some effectively random and shared data.

2) *Online Phase.* This phase is where participants are able to compute a function across their private data. MPC can be computed efficiently using the correlated random data computed in the offline phase.

Using OPE in such a fashion allows for a fast online phase, resulting in an efficient MPC protocol.

Cianciullo and Ghodosi [3] utilised OPE in a slightly different fashion; they showed that a certain type of multiplication could be computed efficiently and securely by utilising a modified OPE protocol in the online phase. Their work is based on the OPE protocol of Tonicelli *et al.* [11].

1.2 OPE and Privacy Preserving Protocols

As stated previously, a privacy preserving protocol can be seen as a subset of MPC. These types of protocols compute a specific function with great efficiency, and many such privacy preserving protocols utilise OPE as an essential building block.

Lindell *et al.* [20] utilised OPE as an integral part of their secure data mining protocol, which allows participants to securely run standard data mining algorithms across their privately-held information. Similar to this, Chang and Lu [4, 14] utilised OPE for the purpose of oblivious neural learning, i.e., training a neural network across private data. In [21] Hazay showed how a set of participants could securely compute the intersection of their privately-held sets. Ogata and Kurosawa [15] utilised OPE to develop an oblivious keyword protocol, wherein a participant can search among a secure database whilst keeping the information he/she was searching for private. Lastly, a secure voting scheme was developed from OPE in [22] by Otsuka and Imai.

1.3 Outline and Contributions

It is evident that OPE is a valuable protocol that has many applications and uses. However, to the best of the authors' knowledge there has not yet been any surveys or reviews published on this deeply interesting topic. We seek to rectify this by presenting a thorough review of a specific class of OPE protocols. Namely, we focus on the specific case of information theoretic (or unconditionally secure) OPE, wherein it is assumed that each of the participants (and any given adversary) has unlimited computational resources (see Section 2 for more information).

In particular, we review the current results present within the literature and then modify some of these protocols to gain improvements in efficiency, flexibility and security. To summarise, our results are threefold.

1) We provide a thorough description and critical review of all currently-known information theoretic OPE protocols. Furthermore, we show that each of the information theoretic OPE protocols within the literature can be classified under two sub-fields, three-party OPE and distributed OPE.

This result, or classification, directly corresponds to the already well known and researched area of information theoretic oblivious transfer [23–25].

2) We do not merely describe and review each protocol. We also develop modified versions of specific protocols, extending their capabilities, efficiency and security.

Additionally, we further prove the link between OT and OPE by demonstrating that a previously published distributed oblivious transfer (DOT) protocol can easily be adapted to a distributed OPE protocol.

3) Lastly, we show that a previously published "unconditionally secure OPE" scheme does not, in fact, achieve unconditional security.

The rest of this paper is organised as follows. Section 2 provides some background on information theoretic OPE. Section 3 and Section 4 investigate the two distinct categories of information theoretic OPE, reviewing current results and also describing our own research in this field. Section 5 examines the OPE scheme shown in [26] and demonstrates that it is not secure. Finally, Section 6 concludes the article.

2 Background

An information theoretic OPE is a two-party protocol that is secure against an adversary, or participants, who have unlimited computational power and resources. We take the same approach given by Rivest [25], such that if the sender or receiver needs to be computationally bounded in order to achieve security, then OPE is computationally secure. Otherwise security is information theoretic (or unconditional). Information theoretic OPE is formerly defined below.

Definition 2. *Given Definition* 1*, we define the output of an OPE protocol* (*i.e., the value* $\mathcal{R}$ *computes as his/her desired evaluation*) *as* f_α*. Let* $\mathcal{A}$ *define the set of all possible evaluation points such that* $\alpha \in A$

and let E be the set of all possible evaluations such that $f_\alpha \in E$. Lastly, let $\mathcal{V}_\mathcal{R}$ denote $\mathcal{R}$'s view (i.e., all information known and held by $\mathcal{R}$) upon the completion of OPE and let $\mathcal{V}_\mathcal{S}$ denote $\mathcal{S}$'s view.

Assuming that all participants are honest, an OPE protocol obtains information theoretic security if the following conditions hold.

- *Correctness.*

$$Pr[f_\alpha = f(\alpha)] = 1.$$

- *Security for $\mathcal{R}$. Let $\beta \in A$ be a possible value for α chosen by $\mathcal{S}$.*

$$Pr[\beta = \alpha | \mathcal{V}_\mathcal{S}] = Pr[\beta = \alpha] = \frac{1}{A}.$$

- *Security for $\mathcal{S}$. Given that $\mathcal{R}$ has obtained the evaluation point $f_\alpha = f(\alpha)$, we define $b \in A$ where $b \neq \alpha$ and let $f_b \in E$ be a possible output value chosen by $\mathcal{R}$.*

$$Pr[f(b) = f_b | \mathcal{V}_\mathcal{R}] = Pr[f(b) = f_b] = \frac{1}{E}.$$

The above definition is a more expanded version of Definition 1 that formalises the security requirements for OPE. Put simply, Definition 2 states that, if all participants are honest, then the value computed by $\mathcal{R}$ at the end of the protocol will be equal to $f(\alpha)$. In terms of security, the above definition states that upon the completion of OPE, $\mathcal{S}$ cannot reduce their uncertainty of $\mathcal{R}$'s evaluation point (α) and $\mathcal{R}$ cannot reduce their uncertainty of $\mathcal{S}$'s polynomial, i.e., for any given input all possible outputs are equally likely.

Computationally secure protocols need not rely on such stringent measures of security. Instead, in a computationally secure protocol, security is assured if $\mathcal{S}$ or $\mathcal{R}$ can reduce their uncertainty of (respectively) α and $f(x)$ only by expending some (defined) limit of computational power and/or time. For example, an OPE scheme could be considered secure if $\mathcal{R}$ could reduce his/her uncertainty of $f(x)$ in exponential time only.

We note that although information theoretic protocols have a far higher level of security, this comes with a trade-off in communication complexity and the number of participants. Specifically, most purely information theoretic protocols within the literature tend to have a high communication complexity. The upside to this, however is that information theoretic protocols are often computationally efficient. Additionally, it has long been understood that it is not possible to have information theoretic security with only two participants [27, 28].

This very statement seems to preclude any possibility of an information theoretic OPE protocol. However, numerous researchers have cleverly avoided this conundrum by introducing a third party (or a set of third parties) who takes part in, but gains no information from the OPE protocol. We specifically refer to what we dub as both three-party OPE and distributed OPE which are (informally) described below.

1) *Distributed OPE* (*DOPE*). A DOPE protocol consists of $n + 2$ participants, the sender and the receiver, along with n servers. In this type of protocol $\mathcal{R}$ and $\mathcal{S}$ compute OPE by communicating with n servers. At the start of the protocol $\mathcal{S}$ distributes some information among the servers. Later, $\mathcal{R}$ contacts a subset of these servers, who provide him/her with enough information to compute his/her evaluation.

As before, both the evaluation point and the polynomial should remain private, and furthermore a coalition of servers should not be able to compute anything related to either $\mathcal{R}$'s or $\mathcal{S}$'s private information. An additional requirement is that a coalition composed of a subset of servers and $\mathcal{R}$ cannot compute anything related to $\mathcal{S}$'s polynomial.

The benefit of this type of protocol is that after $\mathcal{S}$ distributes the sender's data, the sender needs not (and is not expected) to take any further part in the protocol. This provides a high level of availability for $\mathcal{R}$ in that he/she is able to compute his/her OPE at any given time, without waiting on $\mathcal{S}$. This is further improved when we consider that $\mathcal{R}$ needs to only contact a subset of servers; thus if some servers are not available, $\mathcal{R}$ may still compute their evaluation.

2) *Three-Party OPE* (*TOPE*). A TOPE protocol involves a single third party who takes part in the protocol alongside $\mathcal{S}$ and $\mathcal{R}$. This third party is mutually trusted and provides information to both participants which allow them to efficiently and securely compute an OPE.

As with the servers in DOPE, the third party should not be able to compute any information related to $\mathcal{S}$'s polynomial or $\mathcal{R}$'s evaluation point. However, unlike DOPE, it is expected that the third party will not actively try to cheat by sharing (private) information with either $\mathcal{S}$ or $\mathcal{R}$. We note that TOPE protocols are far more efficient than DOPE protocols and often a lot simpler and easier to understand.

In this work, we examine information theoretic OPE schemes that are secure in the presence of a semi-honest (a.k.a. honest but curious) adversity. This assumes that all of the participants will follow the protocol exactly, but will try to learn as much extra information as possible. In the case of DOPE this means that coalitions of servers will attempt to compute some information, whilst in TOPE we assume that no coalitions are formed, but both $\mathcal{R}$ and $\mathcal{S}$ will individually attempt to compute information related to both α and $f(x)$ (whichever they themselves do not directly know).

Within the literature, there currently exist very few information theoretic OPE protocols. However, the usefulness of these protocols (as seen in Section 1) is not in doubt. TOPE has been used in both MPC and privacy preserving protocols, whilst DOPE draws strong parallels to distributed oblivious transfer (DOT) which is a well established and thoroughly researched field with many rich applications and protocols.

Thus, it is our hope to further illuminate information theoretic OPE by reviewing the current protocols and providing our own research in this field. In Sections 3–5 we formally define DOPE and TOPE and investigate the protocols present within the literature.

3 Distributed OPE

To the best of the authors' knowledge there exist only two DOPE protocols within the literature, the "distributed oblivious function evaluation" (DOFE) of Li *et al.*[9] and the DOPE protocol of Cianciullo and Ghodosi[5]. Although both DOPE protocols in the literature differ in their security requirements and definitions, we can provide a blanket model that suffices to broadly explain the requirements of both DOPE protocols. We will, of course, specify the specific requirements of each DOPE protocol before examining them. Our broad model and security requirements of a DOPE protocol are given below.

3.1 Model

A DOPE protocol is an OPE protocol (as per Definition 1) with a set of $n \geqslant 2$ additional participants, $s_1, \cdots, s_n$ called servers. We assume the presence of private synchronous communication between all existing participants (standard in MPC and other such protocols). The sender, $\mathcal{S}$'s polynomial ($f(x)$) is of a degree $k \geqslant 1$ over the field $\mathbb{F}$, whilst $\mathcal{R}$ has the value $\alpha \in \mathbb{F}$, where $|\mathbb{F}| = q$ such that $q > \max(n, k)$ is a prime number. A DOPE protocol can be split into two phases[5].

1) *Initialisation.* $\mathcal{S}$ privately distributes information to the set of servers. After sending this information $\mathcal{S}$ takes no further part in the protocol.

2) *Evaluation.* $\mathcal{R}$ contacts a subset of the n servers who then respond to $\mathcal{R}$ by sending him/her some information. $\mathcal{R}$ uses this information to compute the required evaluation, $f(\alpha)$.

We can divide the security requirements of a DOPE protocol into a set of four conditions. These conditions were given in [5, 29], which, in turn were adapted from the conditions of Blundo *et al.*[30].

1) *Correctness.* $\mathcal{R}$ is able to compute the requested evaluation after receiving information from t or more servers.

2) *Receiver's Privacy.* A coalition of $t-1$ servers cannot compute any information related to α.

3) *Sender's Privacy.* After the initialisation phase (but before the evaluation phase) a coalition composed of $t-1$ servers and $\mathcal{R}$ cannot compute any information related to $f(x)$.

4) *Sender's Privacy After Protocol Execution.* After the communication between $\mathcal{R}$ and the servers has occurred and $\mathcal{R}$ has computed $f(\alpha)$, a coalition composed of $t-1$ servers and $\mathcal{R}$ cannot compute any information related to $f(x)$, other than what the evaluation of $\mathcal{R}$'s chosen value (i.e., $f(\alpha)$) has already revealed.

For the most part, research in this area (and DOT) has focused on producing protocols that achieve security against the highest possible group or coalition, something that is, oftentimes, quite hard to achieve. As we will see, the security for one participant ($\mathcal{R}$ or $\mathcal{S}$) often comes at a price, i.e., lower security for the other participant.

3.2 Shamir's Secret Sharing Scheme

Before we begin evaluating the DOPE protocols, we must first describe a key building block of many distributed protocols, known as secret sharing. A secret sharing scheme is a cryptographic primitive that consists of n participants and a dealer $\mathcal{D}$. The specific case we are concerned with is known as a (t, n) threshold secret sharing scheme which is defined as follows.

Definition 3. *A (t, n) threshold secret sharing scheme allows a dealer $\mathcal{D}$ with secret value S to distribute a set of shares among n participants, in such a way that any t or more of these participants can use their shares to compute S. Security is maintained if a set of $t-1$ or less participants cannot compute any information related to S.*

Below we present the well known secret sharing scheme given by Shamir in his seminal work[31], which is based on polynomials over a finite field.

Let all computations be performed in the field $\mathbb{F}_q$, where $q > n$ is a prime number. We label the n participants as $P_1, \cdots, P_n$ and their respective shares of S as $V_1, \cdots, V_n$; then Shamir's scheme is as follows.

Sharing Phase. 1) The dealer, $\mathcal{D}$ constructs a random polynomial, $g(x)$, of a degree at most $t-1$, such that $g(0) = S$. 2) A given participant, P_i, is privately assigned the share $V_i = g(i)$.

Reconstruction Phase. 1) A set of t or more participants pool their shares and perform Lagrange interpolation to compute $g(x)$. 2) The participants take $g(0)$ as the secret.

3.3 DOFE Protocol

Li *et al.*[9] described a set of three DOPE protocols, each with varying levels of security and flexibility. For our purposes, we examine their second scheme; however in Subsection 3.5 we show that a sub-protocol for a DOT scheme shown in [32] can be slightly altered to produce the first DOPE scheme described by Li *et al.*[9].

In the DOPE protocol of Li *et al.*[9] the sender's security is guaranteed against a coalition composed of $l-1$ servers and $\mathcal{R}$. Whilst the receiver's privacy is guaranteed against a subset of $b-1$ servers and $\mathcal{S}$, such that $b + l < t \leqslant n$ where the security of both $\mathcal{R}$ and $\mathcal{S}$ is guaranteed against a coalition composed of $t-1$ or less servers. The full protocol is given in Fig.1. This is a flexible scheme that can be easily altered to suit a given environment. However, it is evident that increasing the security or privacy threshold for $\mathcal{S}$ would result in a decrease of security for $\mathcal{R}$ and vice versa. Li *et al.*[9] showed how to avoid this, achieving a threshold of $b = l = t$, unfortunately though this increase in security comes with a cost to both privacy and complexity.

Aside from the overall complexity of the scheme increasing, the security modifications also allow $\mathcal{R}$ to learn extra information about $f(x)$. Although this information is not enough for $\mathcal{R}$ to compute anything in an isolated setting, it does mean that this scheme may not be suitable for implementation as part of a larger protocol.

To overcome the privacy and complexity concerns we noted above, Cianciullo and Ghodosi[5] proposed a DOPE protocol secure (for both $\mathcal{S}$ and $\mathcal{R}$) against $t-1$ servers where $t \leqslant n$. However, we note that this protocol is not without faults of its own, which we shall examine in Subsection 3.4. Before doing so we briefly list the communication overhead of the DOFE protocol, to serve as a comparison to the scheme of [5] investigated in Subsection 3.4.

Overhead. In the initialisation phase the sender has a communication overhead of $O(kn)$, sending $O(k)$ values to each of the n servers. In the evaluation phase the receiver must send $k+1$ values to a set of t servers,

Input: $\mathcal{S}$ has the polynomial $f(x) = a_0 + a_1 x + \cdots + a_k x^k$ and $\mathcal{R}$ the value α.
Output: $\mathcal{R}$ receives $f(\alpha)$ and $\mathcal{S}$ gets nothing.

Initialisation

1. $\mathcal{S}$ selects and broadcasts $k+1$ random, distinct values: $x_0, x_1, \cdots, x_k$.
2. Using these values $\mathcal{S}$ privately computes $y_0, y_1, \cdots, y_k$ such that $y_i = f(x_i)$ where $i = 0, \cdots, k$.
3. Next, $\mathcal{S}$ computes $k+1$ random polynomials, $f_0(x), f_1(x), \cdots, f_k(x)$ where $f_0(0) = y_0, f_1(0) = y_1 - y_0, \cdots, f_k(0) = y_k - y_0$ such that the degree of $f_0(x)$ is at most $t-1$ and the other k polynomials have a degree at most $l-1$.
4. $\mathcal{S}$ sends to each server, s_j, the share $A_j = (f_0(j), \cdots, f_k(j))$.

Evaluation

1. $\mathcal{R}$ computes the random values $d_0, \cdots, d_k$ such that they satisfy $\boldsymbol{\alpha} = d_0 x_0 + d_1 x_1 + \cdots + d_k x_k$. Here, for any value x, the value x_i denotes $(1, x, x^2, \cdots, x^k)$.
2. $\mathcal{R}$ then uses these values to compute a set of random polynomials, $Q_1(x), \cdots, Q_k(x)$, of a degree at most $b-1$, where $Q_i(0) = d_i$ for $i = 1, \cdots, k$.
3. $\mathcal{R}$ selects a subset of t servers denoted as ω. He/she then sends to each $s_j \in \omega$ the values $B_j = (1, Q_1(j), \cdots, Q_k(j))$.
4. Each $s_j \in \omega$ computes and sends to $\mathcal{R}$ the value $R(j) = \langle A_j, B_j \rangle$, i.e., the inner (dot) product of the two vectors.
5. $\mathcal{R}$ computes $f(\alpha)$ by interpolating over the set of $R(j)$ values he/she received to compute the polynomial $R(x)$. He/she takes $R(0)$ as his/her evaluation.

Fig.1. DOFE protocol[9].

and $\mathcal{R}$ then receives $O(1)$ messages from these t servers, achieving a communication complexity of $O(kt)$ where $t \leqslant n$.

3.4 Cianciullo and Ghodosi's DOPE Protocol

DOPE presented in this subsection is given by Cianciullo and Ghodosi in [5]. This protocol is both simple and efficient, achieving the highest level of security possible, i.e., $\mathcal{S}$'s privacy is guaranteed against a coalition composed of $t-1$ servers and $\mathcal{R}$ and $\mathcal{R}$'s privacy is guaranteed against a coalition of $t-1$ servers, where $t \leqslant n$. The protocol does not need secure communication channels that allow $\mathcal{R}$ to privately communicate to the servers. It only needs a one-way private channel that allows each server to privately communicate to $\mathcal{R}$. So $\mathcal{R}$ communicates using a public channel (broadcast) to all the servers. Whilst the servers can each privately communicate information to $\mathcal{R}$.

This allows for a more robust scheme in which $\mathcal{R}$ simply broadcasts (publicly) some information to which a set of servers then respond. This means that $\mathcal{R}$ does not necessarily need to pick the specific servers they communicate with, but the servers can either all respond or just a minimum subset of required servers can respond. The full protocol is given in Fig.2.

Whilst the security benefits of this scheme are obvious, it is not without flaws. For instance, this protocol requires $\mathcal{S}$ to communicate directly with $\mathcal{R}$ during the initialisation stage. The author's shown how this can be rectified, however, this modification also results in more communication complexity.

Furthermore, and perhaps more alarmingly, the protocol requires (like many previously published DOT protocols) that $\mathcal{S}$ does not communicate with any of the servers after the initialisation phase. It is easy to see that if this were to occur it would be a trivial matter for $\mathcal{S}$ to compute the exact value of α.

Specifically, all it would take is one server revealing to $\mathcal{S}$ the value of ϵ_i. This would allow $\mathcal{S}$ to compute $\alpha = \sqrt[i]{\epsilon_i + r_i}$, the result of which is a complete loss of privacy for $\mathcal{R}$.

Unfortunately this predicament is an inherent problem present in all such schemes that achieve the maximum level of security. As noted by Cheong *et al.*[32], achieving this level of security in a DOT protocol also results in the same issue. To overcome this problem within the field of DOT, Cheong *et al.*[32] developed a robust and flexible DOT scheme that achieved what they described as the maximum security a DOT scheme could possibly achieved. One in which the security issue is highlighted above does not exist. As before we

Input: $\mathcal{S}$ has the polynomial $f(x) = a_0 + a_1 x + \cdots + a_k x^k$ and $\mathcal{R}$ the value α.
Output: $\mathcal{R}$ receives $f(\alpha)$ and $\mathcal{S}$ gets nothing.

Initialisation

1. $\mathcal{S}$ creates a set of random values $r_1, \cdots, r_k$ and computes k values of the form $\gamma_i = r_i \times a_i$ for $i = 1, \cdots, k$.
2. For each coefficient, a_h $(h = 0, \cdots, k)$, $\mathcal{S}$ computes a random polynomial, $A_h(x)$ of a degree at most $t-1$ such that $A_h(0) = a_h$. He/she does the same for each γ_i value, computing k polynomials of the form $\Gamma_i(x)$ with free term $\Gamma_i(0) = \gamma_i$.
3. Using Shamir's secret sharing scheme $\mathcal{S}$ distributes these values among the servers, giving server s_j $(j = 1, \cdots, n)$ the following information:
   - k shares of the form $\gamma_{i_j} = \Gamma_i(j)$,
   - $k+1$ shares of the form $a_{h_j} = A_h(j)$.
4. $\mathcal{S}$ privately sends to $\mathcal{R}$ the values $r_1, \cdots, r_k$ and then takes no further part in the protocol.

Evaluation

1. $\mathcal{R}$ broadcasts to all servers a set of k values of the form $\epsilon_i = \alpha^i - r_i$.
2. A set of t or more servers, denoted as $\mathcal{W}$ respond to $\mathcal{R}$'s broadcast values. Each server, $s_j \in \mathcal{W}$, computes and sends to $\mathcal{R}$ the share:
$$z_j = a_{0_j} + \sum_{i=1}^{k} (a_{i_j} \times \epsilon_i + \gamma_{i_j}).$$
3. As per Shamir's secret sharing scheme, $\mathcal{R}$ performs Lagrange interpolation across each z_j value to compute the polynomial $Z(x)$ with free term $Z(0) = f(\alpha)$.

Fig.2. DOPE protocol [5].

examine the communication complexity of this protocol.

Overhead. As with the DOFE scheme, the sender has a communication overhead of $O(kn)$, sending $O(||)$ values to each of the n servers. In the evaluation phase the receiver broadcasts a set of k values and then is answered by a set of t servers which each send a single value. If we assume that the broadcast message is simply $\mathcal{R}$ sending each server a message, then the communication complexity of this protocol is given as $O(kn+t)$. Interestingly, the communication complexity of this protocol is slightly more than that of the DOFE protocol; however since $t \leqslant n$ we note that the overall complexity is roughly the same. Furthermore the efficiency of this protocol also depends greatly on the method used to achieve the broadcast channel utilised in the first step of the evaluation.

In Subsection 3.5 we show that an interesting subprotocol used in the DOT scheme by Cheong *et al.*[32] can be modified to provide a secure DOPE protocol, with the flexible security thresholds of the DOFE protocol. In fact, we show that this sub-protocol can be adapted into the first DOPE protocol described in [9] by Li *et al.* with minimal effort.

3.5 Flexible DOPE from DOT

The security parameters of the DOT protocol devised by Cheong *et al.*[32] operate in much the same fashion as the DOFE protocol, in that security is guaranteed against two different thresholds. Put simply, $\mathcal{R}$'s privacy is guaranteed against a group consisting of γ_1 servers and $\mathcal{S}$, whilst $\mathcal{S}$'s privacy is guaranteed against a set of γ_2 servers and $\mathcal{R}$, where $\gamma_1 + \gamma_2 < t \leqslant n$. Furthermore, a group of $t-1$ or less servers cannot compute anything related to either $\mathcal{S}$'s or $\mathcal{R}$'s private information. In this subsection we show how to adapt a sub-protocol of their DOT scheme, in order to produce a DOPE protocol with exactly the same highly desirable security guarantees.

The core building block of this DOT scheme (the aforementioned sub-protocol, given in Fig. 3) can be viewed as a special case of a DOPE protocol, in which the degree of $\mathcal{S}$'s polynomial is 1. Such a scheme is commonly called an oblivious linear evaluation (OLE); thus, in our case, the sub-protocol is essentially a distributed OLE (DOLE).

To create a DOPE from this DOLE protocol we simply extend the protocol, substituting the bivariate polynomial, $Q(x, y)$ with a multivariate polynomial $Q(x, y_1, \cdots, y_k)$. The full DOPE protocol is given in Fig.4.

Interestingly enough, the resulting protocol is exactly equivalent to the first of the three DOPE protocols given in [9] by Li *et al.* (as such, we point the reader to [9], for a security proof of this protocol). In fact, we note that both of these schemes (the DOT protocol of [32] and the DOPE protocol of [9]) actually utilise techniques given in the seminal work of Naor and Pinkas[33] which first introduced DOT. This result clearly shows the relationship between DOPE and DOT and establishes DOPE as an interesting field in its own right.

Input: $\mathcal{S}$ has the polynomial $f(x) = a_0 + a_1 x$ and $\mathcal{R}$ the value α.
Output: $\mathcal{R}$ receives $f(\alpha)$ and $\mathcal{S}$ gets nothing.

Initialisation

1. $\mathcal{S}$ computes two random polynomials, $B_0(x)$ and $B_1(x)$, such that $B_0(x)$ is of a degree at most $t-1$ with $B_0(0) = a_0$ and $B_1(x)$ is of a degree at most γ_2 with $B_1(0) = a_1$. He/she combines these two polynomials to compute the bivariate polynomial $Q(x, y) = B_0(x) + B_1(x)y$.
2. Each server, s_j (for $j = 1, \cdots, n$) receives from $\mathcal{S}$ the values $(B_0(j), B_1(j))$.

Evaluation

1. $\mathcal{R}$ computes the random polynomial $S(x)$, of a degree at most γ_1, where $S(0) = \alpha$.
2. Each server, s_j receives from $\mathcal{R}$ the value $S(j)$.
3. Let $Q(x, S(x)) = R(x)$, then a set of t or more servers (denoted by s_j) compute and send to $\mathcal{R}$ the value $R(j) = B_0(j) + B_1(j)S(j)$. s_j denotes one of the servers from the set of t or more servers.
4. $\mathcal{R}$ interpolates over these values to compute $R(x)$. Taking the value $R(0) = B_0(0) + B_1(0)\alpha = f(\alpha)$ as the receiver's desired evaluation.

Fig.3. DOLE protocol given in [32].

Input: $\mathcal{S}$ has the polynomial $f(x) = a_0 + a_1x + \cdots + a_kx^k$ and $\mathcal{R}$ the value α.
Output: $\mathcal{R}$ receives $f(\alpha)$ and $\mathcal{S}$ gets nothing.

Initialisation

1. $\mathcal{S}$ computes $k+1$ random polynomials, $B_0(x), \cdots, B_k(x)$, such that $B_0(x)$ is of a degree at most $t-1$ with $B_0(0) = a_0$ and $B_1(x), \cdots, B_k(x)$ are of a degree at most γ_2 with $B_i(0) = a_i$, for $i = 1, \cdots, k$. As before, he/she combines these polynomials to compute the multivariate polynomial $Q(x, y_1, \cdots, y_k) = B_0(x) + B_1(x)y_1 + \cdots + B_k(x)y_k$.
2. Each server, s_j (for $j = 1, \cdots, n$) receives from $\mathcal{S}$ the values $(B_0(j), B_1(j), \cdots, B_n(j))$.

Evaluation

1. $\mathcal{R}$ computes the random polynomials $S_1(x), \cdots, S_k(x)$, of a degree at most γ_1, where $S_j(0) = \alpha^j$.
2. Each server, s_j receives from $\mathcal{R}$ the values $(S_1(j), \cdots, S_k(j))$.
3. Let $Q(x, S_1(x), \cdots, S_k(x)) = R(x)$, and then a set of t or more servers (denoted by s_j) computes and sends to $\mathcal{R}$ the value
$$R(j) = B_0(j) + B_1(j)S_1(j) + \cdots + B_k(j)S_k(j).$$
4. $\mathcal{R}$ interpolates over these values to compute $R(x)$. Taking the value $R(0) = B_0(0) + B_1(0)\alpha = f(\alpha)$ as their desired evaluation.

Fig.4. Flexible DOPE Protocol from DOT [32]. This is Equivalent to the first DOPE protocol given in [9].

Overhead. Since the sender computes a multivariate polynomial from a set of k random polynomials and each of the servers receives a value from said polynomials, the sender has a communication overhead of $O(kn)$. The receiver sends a set of k values to each of the servers and receives a value from t or more of these servers, giving a communication complexity of $O(kn+t)$, identical to the previously examined DOPE protocol.

In Section 4 we examine the existing TOPE protocols within the literature.

4 Three-Party OPE

TOPE substitutes the n servers of DOPE for just one extra participant who takes part in the protocol. As with the servers in DOPE, this third participant should learn nothing related to either $f(x)$ or α, and furthermore it is expected that the third party does not actively collaborate with either $\mathcal{R}$ or $\mathcal{S}$.

The major benefit that a TOPE protocol has over a DOPE protocol is the need for only one extra participant. This drastically cuts down on communication complexity as there is no need to send/receive messages from a large set of servers. The downside to TOPE is, of course, that there is a central point of failure. To clarify, if the third party is compromised and/or corrupted in some way and freely shares information with either $\mathcal{S}$ or $\mathcal{R}$, then all security is lost. Another issue is that the third party must be available and present throughout the entire protocol. As such, if there is no single trusted third party available then the protocol cannot be computed. DOPE overcomes these issues by essentially spreading out the function of the third party among the set of n servers, achieving security and availability at a far higher cost to efficiency (namely, efficiency of communication).

Of the two TOPE protocols reviewed in this section, the first uses the third participant as an active and ever present party who takes full part in the protocol, whilst the second simply uses the third participant to provide some unrelated and random information at the start of the protocol. For this reason we shall not present an overall model of security and communication for TOPE, and rather, the security and communication requirements for each of these protocols are given as required. However, we can provide a broad and informal definition that covers both TOPE protocols reviewed here.

Definition 4. *A TOPE protocol is an OPE protocol with an extra participant who does not (illegally, i.e., in secret or against the protocol) collaborate with either $\mathcal{S}$ or $\mathcal{R}$. Security is maintained as per the requirements stated in Definition* 1*, along with the additional requirement that the extra participant cannot compute anything related to either $f(x)$ or α.*

4.1 Active Third-Party TOPE

The TOPE protocol given by Chang and Lu [14] requires an active third party who takes full part in, and is present for the entire TOPE protocol. We call this third party the mediator, denoted as $\mathcal{M}$. There are three rounds of communication in this protocol, in the first round $\mathcal{R}$ sends some information to the other participants, the second round has $\mathcal{S}$ sending information and lastly $\mathcal{M}$ sends information to $\mathcal{R}$ who then computes his/her evaluation. All computations are performed over a finite field $\mathbb{F}$. The full protocol is given in Fig.5.

Overhead. As would be expected in an OPE protocol utilising a polynomial of a degree k, the communication complexity is just $O(k)$, which is extremely efficient, especially in contrast to the previously examined DOPE protocols. Furthermore this protocol utilises finite fields of characteristics at most $k + 1$, whereas DOPE protocols require fields of characteristics $q > \max(k, n)$. The net result is that for a small k value the TOPE protocol would be far more efficient than a DOPE protocol where $n > k$. However, this active TOPE requires a total of six messages to be sent between the participants.

Evidently, the use of an actively involved third party allows for an extremely efficient protocol; however, $\mathcal{M}$ has an integral role that is tied in with the entire protocol. As such, there may be issues with both security and availability. To elaborate, if $\mathcal{M}$ is not available for the entire protocol then the OPE cannot be computed. Furthermore, the fact that $\mathcal{M}$ is present for the entire protocol may result in some security concerns. To rectify these problems, Hanoaoka *et al.*[34] and Tonicelli *et al.*[11] developed a TOPE protocol in which the third party does not receive communications from either of the other two participants and only needs be present for the start of the protocol.

4.2 Commodity-Based TOPE

In this subsection we look at what is known as the commodity-based TOPE given in [11,34]. Commodity-based cryptography was originally described by Beaver in [35] and involves the participants "buying" or being assigned some (essentially random) information from a neutral third party at the start of (or before, i.e., "offline") a given protocol. We dub this third party the initialiser, denoted as $\mathcal{I}$, and divide the TOPE protocol into two phases.

1) *Setup.* In this phase the initialiser individually assigns some correlated random information to both $\mathcal{S}$ and $\mathcal{R}$.

2) *Computation.* Here, $\mathcal{S}$ and $\mathcal{R}$ securely and privately compute an OPE using the correlated information assigned to them by $\mathcal{I}$.

As with the previously depicted OPE protocols, security is maintained if, after the protocol has been executed, $\mathcal{R}$ cannot compute any information related to $f(x)$ (other than $f(\alpha)$) and $\mathcal{S}$ cannot compute any information related to α. All computations are performed in the finite field $\mathbb{F}_q$ where $q > k$ is a prime number, and the protocol is given in Fig.6.

Hanaoka *et al.*[34] and Tonicelli *et al.*[11] proved that their TOPE is (for the conditions that they have defined) optimal in terms of communication complexity and overall efficiency.

Overhead. As before the communication complexity of this protocol is bounded by the size of the sender's polynomial, $O(k)$. In contrast to the active TOPE protocol, however, this commodity-based protocol only requires sending four messages to achieve OPE. Similarly the protocol operates in a field of characteristic $q > k$, thereby achieving greater communication efficiency than the active TOPE protocol.

This commodity-based scheme is elegant in its simplicity and does away with the restrictions of the active third party TOPE given in [14]. Specifically, we note that the setup phase of the protocol can be done at any time, even before either $\mathcal{S}$ or $\mathcal{R}$ has their respective privately held information ($f(x)$ and α). Tonicelli *et al.*[11]

Input: $\mathcal{S}$ has the polynomial $f(x) = a_0 + a_1 x + \cdots + a_k x^k$ and $\mathcal{R}$ the value α.
Output: $\mathcal{R}$ receives $f(\alpha)$ and $\mathcal{S}$ gets nothing.

TOPE Protocol

1. $\mathcal{R}$ sends to $\mathcal{S}$ a set of random values, $\boldsymbol{r}_1, \cdots \boldsymbol{r}_k \in \mathbb{F}^k$. He/she also sends to $\mathcal{M}$ the values, $\alpha'_1, \cdots, \alpha'_k$ where $\alpha'_i = \alpha^i + r_i$ for $i = 1, \cdots, k$.
2. $\mathcal{S}$ sends to $\mathcal{R}$ a set of random values, $s_0, \cdots, s_k \in \mathbb{F}^{k+1}$. Additionally, $\mathcal{M}$ gets a set of $k + 1$ values, $a'_0, \cdots, a'_k$ from $\mathcal{S}$, where $a'_0 = a_0 + s_0 + \sum_{i=1}^{k} a_i r_i$ and $a'_i = a_i + s_i$.
3. $\mathcal{M}$ sends to $\mathcal{R}$ the value $y = a'_0 + \sum_{i=1}^{k} a' \alpha'_i$. Finally, $\mathcal{R}$ computes $f(\alpha) = y - (s_0 + \sum_{i=1}^{k} a'_i s_i)$.

Fig.5. TOPE with active third party[14].

Input: $\mathcal{R}$ has a value α and $\mathcal{S}$ the polynomial $f(x)$ of a degree at most k.
Output: $\mathcal{R}$ obtains $f(\alpha)$ and $\mathcal{S}$ gets nothing.

Setup $\mathcal{I}$ privately sends:

1. A random polynomial, $S(x)$, of a degree at most k to $\mathcal{S}$.
2. A random value, d and the value $g = S(d)$ to $\mathcal{R}$.

Computation

1. $\mathcal{R}$ sends the value $l = \alpha - d$ to $\mathcal{S}$.
2. $\mathcal{S}$ then computes and sends to $\mathcal{R}$ the polynomial $V(x) = f(l + x) + S(x)$.
3. $\mathcal{R}$ computes $f(\alpha) = V(d) - g$.

Fig.6. Commodity-based TOPE [11, 34].

also rigorously proved the security of their scheme under the simulation-based paradigm and their work has been used as a building block in protocols for MPC [3] and secure voting [22].

In the next subsection we take the TOPE protocol shown here and extend its capabilities further modifying the underlying scheme. Our modifications are relatively simple and do not result in any dramatic changes to the efficiency of the protocol.

4.3 Extending TOPE

We demonstrate three modifications to the commodity-based TOPE displayed in Subsection 4.2.

1) *Multivariate Polynomial Capabilities.* Without any loss of security, we extend the protocol to handle multivariate polynomials. Our modified scheme is just as efficient as the original scheme achieving a communication complexity of $O(hk)$ where h denotes the number of variables within the multivariate polynomial.

2) *Randomised Multi-Evaluation Capabilities.* We show how $\mathcal{R}$ can compute not only the receiver's desired evaluation $(f(\alpha))$, but also a random set of $k-1$ extra evaluations, simply by having $\mathcal{I}$ send extra information to $\mathcal{S}$ and $\mathcal{R}$ in the setup phase. Our modification does not add any extra communication or complexity to the computation phase, achieving an overall communication complexity of $O(k)$ with the same amount of messages sent as the original protocol.

3) *Multi-Evaluation Capabilities.* By relaxing the security constraints and slightly adapting the above modified scheme, we show how to allow $\mathcal{R}$ to compute a given set of k evaluations (that are not randomised). This protocol is also extremely efficient, achieving the same communication complexity as the original, at the cost of sending only one extra message (for a total of five messages sent).

4.3.1 *TOPE with Multivariate Polynomial*

Our first modification is to allow the commodity-based TOPE protocol to handle multivariate polynomials. In this case $\mathcal{S}$ has the multivariate polynomial $f(x_1, \cdots, x_h)$, whilst $\mathcal{R}$ holds a range of values, $\alpha_1, \cdots, \alpha_h$ and wishes to learn the evaluation of $f(\alpha_1, \cdots, \alpha_h)$. All computations are performed in the finite field $\mathbb{F}_q$, where $q > \max(h, k)$ is a prime number. The full multivariate protocol is given in Fig.7.

Evaluation. The security and correctness of this extended protocol is an obvious extension of the original protocol (see [11] for its full proof). It is easy to see that our protocol is still extremely efficient, only adding a multiplicative factor h onto the communication complexity of the original univariate protocol.

The probability of error for $\mathcal{R}$, i.e., the chance that $\mathcal{S}$ correctly guesses $\alpha_1, \cdots, \alpha_h$ (or $d_1, \cdots, d_h$), is given as $\frac{1}{q^h}$. As to do this $\mathcal{S}$ would have to correctly guess h values over the finite field $\mathbb{F}_q$. $\mathcal{S}$ has a $\frac{1}{q}$ chance to correctly guess the evaluation value, as this requires only guessing one number. However, we note that security/privacy for $\mathcal{R}$ is reliant on $\mathcal{S}$ not being able to correctly compute $\mathcal{R}$'s privately-held values, $\alpha_1, \cdots, \alpha_h$. Thus the probability of error here is $\frac{1}{q^h}$.

We note, however, that if this value is lowered to $\frac{1}{q}$, then we can achieve even a greater efficiency. To do this, we simply have $\mathcal{I}$ send one d value to $\mathcal{R}$ in the setup phase of the protocol. In the computation phase of the protocol $\mathcal{R}$ utilises the same d value as a mask for all of his/her $\alpha_1, \cdots, \alpha_h$ values. This modification results in a more communication efficient protocol, at the

Input: $\mathcal{R}$ has a set of values $\alpha_1, \cdots, \alpha_h$ and $\mathcal{S}$ the multivariate polynomial $f(x_1, \cdots, x_h)$ of a degree at most k in any variable.
Output: $\mathcal{R}$ obtains $f(\alpha_1, \cdots, \alpha_h)$ and $\mathcal{S}$ gets nothing.

Setup $\mathcal{I}$ privately sends:

1. A random multivariate polynomial, $S(x_1, \cdots, x_h)$, of a degree at most k in all variables, to $\mathcal{S}$.
2. A set of random values, $d_1, \cdots, d_h$ and the value $g = S(d_1, \cdots, d_h)$ to $\mathcal{R}$.

Computation

1. $\mathcal{R}$ sends the values $l_1, \cdots, l_h$ to $\mathcal{S}$, where $l_i = \alpha_i - d_i$ for $i = 1, \cdots, h$.
2. $\mathcal{S}$ then computes and sends to $\mathcal{R}$ the polynomial:
$$V(x_1, \cdots, x_h) = f(l_1, x_1, \cdots, l_h, x_h) + S(x_1, \cdots, x_h).$$
3. $\mathcal{R}$ computes:
$$f(\alpha_1, \cdots, \alpha_h) = V(d_1, \cdots, d_h) - g.$$

Fig.7. Commodity-based multivariate TOPE.

cost of security. The full extension to the multivariate protocol is given in Fig.8.

As discussed, the probability of error here is only $\frac{1}{q}$ as all $\mathcal{S}$ has to do is to correctly guess $d \in \mathbb{F}_q$ to easily compute all $\alpha_1, \cdots, \alpha_h$ values.

4.3.2 TOPE with Randomised Multi-Evaluation

Our next modification allows $\mathcal{R}$ to compute not only his/her evaluation $f(\alpha)$, but also a set of $k-1$ extra (random) evaluations as well.

This modification is extremely efficient and only effects the setup phase, allowing for a computation phase that is just as efficient as the original unmodified protocol. The benefit is evident when we consider that often-times in protocols such as MPC, a lot of computation is delegated to the offline or setup phase in order to make the actual computation phase (or the "online" phase) as efficient as possible. This is because the setup/offline phase can be carried out at any time, well in advance of the actual online protocol.

When the setup phase $\mathcal{I}$ sends to $\mathcal{R}$ a set of extra values, $d_1, \cdots, d_{k-1}$, this will allow $\mathcal{R}$ to compute the extra (random) evaluations. The computation phase then proceeds (from a communication point of view) exactly as before, with the only change being some small, extra, computations performed privately by $\mathcal{R}$. The full protocol is given in Fig.9.

The main benefit of our protocol is that we are able

Input: $\mathcal{R}$ has a set of values $\alpha_1, \cdots, \alpha_h$ and $\mathcal{S}$ the multivariate polynomial $f(x_1, \cdots, x_h)$ of a degree at most k in any variable.
Output: $\mathcal{R}$ obtains $f(\alpha_1, \cdots, \alpha_h)$ and $\mathcal{S}$ gets nothing.

Setup $\mathcal{I}$ privately sends:

1. A random multivariate polynomial, $S(x_1, \cdots, x_h)$, of a degree at most k in all variables, to $\mathcal{S}$.
2. The values d and $g = S(d, \cdots, d)$ to $\mathcal{R}$.

Computation

1. $\mathcal{R}$ sends the values $l_1, \cdots, l_h$ to $\mathcal{S}$, where $l_i = \alpha_i - d$ for $i = 1, \cdots, h$.
2. $\mathcal{S}$ then computes and sends to $\mathcal{R}$ the polynomial:
$$V(x_1, \cdots, x_h) = f(l_1, x_1, \cdots, l_h, x_h) + S(x_1, \cdots, x_h).$$
3. $\mathcal{R}$ computes:
$$f(\alpha_1, \cdots, \alpha_h) = V(d, \cdots, d) - g.$$

Fig.8. A more efficient commodity-based multivariate TOPE.

Input: $\mathcal{R}$ has the value α and $\mathcal{S}$ the polynomial $f(x)$ of a degree k or more.
Output: $\mathcal{R}$ obtains $f(\alpha)$ as well as $k-1$ random evaluations of $\mathcal{S}$'s polynomial of the form: $f(\beta_1), \cdots, f(\beta_{k-1})$. As before, $\mathcal{S}$ gets nothing.

Setup $\mathcal{I}$ privately sends:

1. A random polynomial, $S(x)$, of a degree k or more to $\mathcal{S}$.
2. A set of random values, $d_0, \cdots, d_{k-1}$, and the values $g_i = S(d_i)$ to $\mathcal{R}$, for $i = 0, \cdots, k-1$.

Computation

1. $\mathcal{R}$ computes and sends to $\mathcal{S}$ the value $l = \alpha - d_0$. Privately, $\mathcal{R}$ computes $k-1$ values, of the form $\beta_i = l + d_i$, for $i = 1, \cdots, k-1$.
2. $\mathcal{S}$ then computes and sends to $\mathcal{R}$ the polynomial $V(x) = f(l+x) + S(x)$.
3. $\mathcal{R}$ computes their evaluation as: $f(\alpha) = V(d_0) - g_0$. The $k-1$ random evaluations are computed in much the same fashion: $f(\beta_i) = V(d_i) - g_i$.

Fig.9. Commodity-based randomised multi-evaluation TOPE.

to compute k evaluations for the same communication complexity (in the computation phase) as the original protocol. A naive approach to this would result in a multiplicative increase of k, something our protocol manages to avoid by simply designating all extra communication to the setup phase.

Evaluation. From $\mathcal{S}$'s point of view the actual protocol is unchanged from the original protocol. Thus it only remains to show that $\mathcal{R}$ cannot compute anything extra from the multiple evaluation points he/she has received.

Theorem 1. *The randomised multi-evaluation TOPE protocol maintains privacy for* $\mathcal{S}$.

The proof is quite simple and is a result of Shamir's secret sharing scheme[31].

Proof. At the end of the modified OPE protocol $\mathcal{R}$ will have obtained k evaluations of a k-degree polynomial: $f(x) = a_0 + a_1 x + \cdots + a_k x^k$. As such, $\mathcal{R}$ holds a system composed of $k+1$ unknowns and k independent equations. In other words, $\mathcal{R}$ has k shares of a Shamir polynomial of a degree k. As per Shamir's scheme[31], it is known that $k+1$ shares are needed to compute the polynomial. Therefore, $\mathcal{R}$ cannot compute anything extra about $\mathcal{S}$'s polynomial, other than what his/her evaluation points have already told him/her. □

The benefits of this protocol are evident when looking at the use of the original protocol. For example the original OPE is used as a multiplication protocol in MPC[3] and is also the backbone of a secure voting protocol[22]. Our modifications would allow for increases in efficiency, for both of these purposes, i.e., computing multiple multiplications simultaneously and/or evaluating multiple votes simultaneously in each respective protocol.

To be more specific, the multiplication of numbers in [3] is carried out via each participant carrying out an OPE protocol with the other participants. Using this new scheme participants in the MPC could compute multiple multiplication results with just the one OPE, rather than one result per OPE. The secure voting scheme of [22] uses the commodity-based OPE to verify the ballots of voters. With some modifications to their underlying protocol and the replacement of the original OPE scheme with our enhanced version, the verification process could be conducted more efficiently, verifying multiple votes using just the one OPE.

4.3.3 TOPE with Multi-Evaluation

In this final extension to the commodity-based TOPE protocol we show that by lessening the original security requirements of the protocol, we can modify the previously discussed randomised multi-evaluation scheme to allow $\mathcal{R}$ to actually choose all evaluation points. To do this we modify the setup phase and allow $\mathcal{R}$ to communicate with $\mathcal{I}$. This modification does not actually lessen security in any fashion, as $\mathcal{I}$ will still not be able to learn anything related to either $\mathcal{S}$'s or $\mathcal{R}$'s private information. Furthermore, this new protocol has the same communication complexity as the randomised multi-evaluation protocol and we still do not require $\mathcal{I}$ to be an active participant throughout the protocol (as in [14]), i.e., $\mathcal{I}$ is only present for the setup phase.

To summarise our modification, we take the randomised multi-evaluation protocol and add an extra level of communication in the setup phase, whereby $\mathcal{R}$ sends to $\mathcal{I}$ a specific set of $d_1, \cdots, d_k$ points to be evaluated by $S(x)$. This allows $\mathcal{R}$ to choose the evaluation points he/she requires, as opposed to them being random. The full protocol is given in Fig.10.

As with the randomised protocol, the benefits of this scheme have applications in both MPC and privacy preserving protocols. However, with this scheme there is no need to use randomised evaluations, and rather a set of predetermined evaluations can be chosen, allowing for a far more useful and versatile protocol.

Evaluation. From a security perspective this protocol is (for $\mathcal{S}$ and $\mathcal{R}$) exactly the same as the randomised protocol, in that neither $\mathcal{R}$ nor $\mathcal{S}$ can compute any information they are not explicitly assigned. To put this in another way, the protocol is identical in terms of the messages and information shared between $\mathcal{R}$ and $\mathcal{S}$. As a result, we need only to prove that the extra information shared between $\mathcal{I}$ and $\mathcal{R}$ does not lead to any breakdown of security.

When looking at the extra information given to $\mathcal{I}$ by $\mathcal{R}$ it is easy to see that $\mathcal{I}$ cannot compute anything related to any of $\mathcal{R}$'s evaluation points. Because each of the d_i values is essentially random from $\mathcal{I}$'s point of view, to correctly guess any α_i the initialiser would have to guess $l \in \mathbb{F}_q$. This gives $\mathcal{I}$ a $\frac{1}{q}$ chance of correctly computing any extra information, the same probability as in the original scheme given by Tonicelli *et al.*[11], and the same probability that $\mathcal{S}$ has of correctly guessing the evaluation points.

We note that performing k OPEs would result in an overall probability of $\frac{1}{qk}$; however this comes at a far greater cost to communication.

5 Flaws in Bo *et al.*'s OPE Scheme

In this section we show that the OPE protocol devised by Bo *et al.*[26] is not secure. The essential premise of this scheme (as per the authors' claims) is that it achieves unconditional security with only two participants, $\mathcal{R}$ and $\mathcal{S}$. However, as we mentioned in Section 2, it has long been established that a two-party unconditionally secure protocol is impossible[27,28]. To demonstrate this fact we display Bo *et al.*'s protocol[26] in Fig.11 and then discuss the flaws in their proposed OPE.

The idea behind this protocol is to utilise a series of random values as masks, in order to preserve the privacy of both $\mathcal{S}$ and $\mathcal{R}$. However, after learning the evaluation ($f(\alpha)$) the receiver, $\mathcal{R}$, is able to go back and actually compute the masks used by $\mathcal{S}$. This then allows $\mathcal{R}$ to break the protocol by computing $f(x)$. The exact method by which this is possible is given below in an example in which we set $k = 1$ (the degree of $\mathcal{S}$'s polynomial).

As stated, $k = 1$, so $l = 1$ as well. Assume that the protocol has been completed and $\mathcal{R}$ has computed $f(\alpha)$. We now draw the reader's attention to step 3 of the protocol, in which $\mathcal{R}$ is assigned the following pieces of information:

1) $D_0 = Ha_0$,

2) $D_1 = Ha_1\Delta_{1_1}$ (as $a_2 = 0$),

where H, a_0 and a_1 are unknown. Now, at the end of the protocol $\mathcal{R}$ also has the equation $f(\alpha) = a_0 + \alpha a_1$, as before a_0 and a_1 are unknown. Combining these pieces of information gives the following system of equations:

$$D_0 = Ha_0,$$

Input: $\mathcal{R}$ has the values $\alpha_1, \cdots, \alpha_k$ and $\mathcal{S}$ the polynomial $f(x)$ of a degree k or more.
Output: $\mathcal{R}$ obtains $f(\alpha_1), \cdots, f(\alpha_k)$. As before, $\mathcal{S}$ gets nothing.

Setup

1. $\mathcal{I}$ sends a random polynomial, $S(x)$, of a degree k or more to $\mathcal{S}$.
2. $\mathcal{R}$ computes and sends to $\mathcal{I}$ the values $d_1, \cdots, d_k$, where $d_i = l - \alpha_i$ for $i = 1, \cdots, k$ and l is a random, private value chosen by $\mathcal{R}$.
3. $\mathcal{I}$ sends to $\mathcal{R}$ the values $g_1, \cdots, g_k$ where $g_i = S(d_i)$.

Computation

1. $\mathcal{R}$ sends to $\mathcal{S}$ the value l.
2. $\mathcal{S}$ then computes and sends to $\mathcal{R}$ the polynomial $V(x) = f(l - x) + S(x)$.
3. $\mathcal{R}$ computes his/her k evaluations as: $f(\alpha_i) = V(d_i) - g_i$ for $i = 1, \cdots, k$.

Fig.10. Commodity-based TOPE with multi-evaluation capabilities.

Input: $\mathcal{R}$ has the values α and $\mathcal{S}$ the polynomial $f(x)$ of a degree k or more.
Output: $\mathcal{R}$ obtains $f(\alpha)$ and $\mathcal{S}$ gets nothing.
Preliminaries: Let $l = \left\lfloor \frac{k}{2} \right\rfloor + 1$ if k is odd and $l = \frac{k}{2}$ if k is even. All values are drawn from the field $\mathbb{Z}_q \setminus \{0\}$ where q is a large prime.

OPE Protocol

1. $\mathcal{R}$ privately selects the random values β_1, β_2, T_1, T_2 and $r_1, \cdots, r_l$. He/she uses these values to compute l values of the form $r'_1, \cdots, r'_l$, such that $r'_j = T_2^{-1} r_j$ for $j = 1, \cdots, l$.
2. $\mathcal{R}$ then sends to $\mathcal{S}$ the values $\Delta_j = (\Delta_{j_1}, \Delta_{j_2})$ where:
$$\Delta_{j_1} = T_1 r_j \alpha^{2j-1} + \beta_1 r'_j,$$
$$\Delta_{j_2} = T_1 r_j \alpha^{2j} + \beta_2 r'_j.$$
3. $\mathcal{S}$ privately computes the random value H and sends to $\mathcal{R}$ the values $D_0, \cdots, D_l$ where $D_0 = Ha_0$ and $D_j = Ha_{2j-1}\Delta_{j_1} + Ha_{2j}\Delta_{j_2}$, where $j = 1, \cdots, l$ and $a_{2l} = 0$ if k is odd.
4. $\mathcal{R}$ computes $M = T_1 T_2 D_0 + \sum_{j=1}^{l} D_j (r'_j)^{-1}$ and then sends to $\mathcal{S}$ the value $M_1 = M\beta_1^{-1}$.
5. $\mathcal{S}$ sends to $\mathcal{R}$ the value $S_1 = M_1 - H\sum_{j=1}^{l} a_{2j-1}$.
6. Using this value, $\mathcal{R}$ sends to $\mathcal{S}$ the value $M_2 = S_1 \beta_1 \beta_2^{-1}$.
7. Following this, $\mathcal{S}$ sends to $\mathcal{R}$ the value $S_2 = \left(M_2 - H\sum_{j=1}^{l} a_{2j}\right) H^{-1}$.
8. Finally, $\mathcal{R}$ computes $f(\alpha) = S_2 \beta_2 T_1^{-1} T_2^{-1}$.

Fig.11. Flawed OPE protocol [26].

$$D_1 = Ha_1\Delta_{1_1},$$
$$f(\alpha) = a_0 + a_1\alpha.$$

To solve this system, we multiply the third equation by H and substitute the first two equations into this new third equation. Doing so gives, $f(\alpha)H = D_0 + \alpha D_1$. Solving this gives the value of $H = (D_0 + \alpha D_1)(f(\alpha))^{-1}$. Now that H is known we can compute a_0 and a_1 with ease:

$$a_0 = H^{-1} D_0,$$
$$a_1 = H^{-1}\Delta_{1_1}^{-1} D_1.$$

As a result of this $\mathcal{R}$ is able to compute the entirety of $\mathcal{S}$'s polynomial $f(x)$, thereby resulting in a flawed scheme that does not ensure either security or privacy. We note that our attack will also work with all possible cases; however $k = 1$ was used in order to easily demonstrate the flaws in this protocol.

6 Conclusions

In this article we critically reviewed the exiting information theoretic OPE protocols. Additionally we made several key contributions to the field of information theoretic OPE.

1) We adapted a DOT protocol into a flexible DOPE protocol that is equivalent to an existing DOPE protocol presented in [9], displaying the strong link between the relatively new field of DOPE and the existing and well researched field of DOT.

2) We created three extensions of a well known TOPE protocol, resulting in the following improvements:

- multivariate capabilities;
- randomised multi-evaluation capabilities;
- multi-evaluation capabilities with relaxed security.

3) Demonstration of the flaws inherent in the OPE given by Bo *et al.* [26].

OPE is a relatively new field which is continuing to grow, with new results and applications appearing frequently. As such, there is still many more opportunities for research within this field, particularly in terms of the myriad of applications to which this protocol can be put towards.

Acknowledgements We thank the reviewers for their helpful comments and improvements.

References

[1] Naor M, Pinkas B. Oblivious transfer and polynomial evaluation. In *Proc. the 31st Annual ACM Symposium on Theory of Computing*, May 1999, pp.245-254. DOI: 10.1145/301250.301312.

[2] Even S, Goldreich O, Lempel A. A randomized protocol for signing contracts. In *Proc. CRYPTO'82*, Aug. 1982, pp.205-210. DOI: 10.1007/978-1-4757-0602-4_19.

[3] Cianciullo L, Ghodosi H. Efficient information theoretic multi-party computation from oblivious linear evaluation. In *Proc. the 12th IFIP WG 11.2 International Conference on Information Security Theory and Practice*, Dec. 2019, pp.78-90. DOI: 10.1007/978-3-030-20074-9_7.

[4] Chang Y C, Lu C J. Oblivious polynomial evaluation and oblivious neural learning. In *Proc. the 7th International Conference on the Theory and Application of Cryptology and Information Security Gold Coast*, Dec. 2001, pp.369-384. DOI: 10.1007/3-540-45682-1_22.

[5] Cianciullo L, Ghodosi H. Unconditionally secure distributed oblivious polynomial evaluation. In *Proc. the 21st International Conference on Information Security and Cryptology*, Nov. 2018, pp.132-142. DOI: 10.1007/978-3-030-12146-4_9.

[6] Ghosh S, Nielsen J B, Nilges T. Maliciously secure oblivious linear function evaluation with constant overhead. In *Proc. the 23rd International Conference on the Theory and Applications of Cryptology and Information Security*, Dec. 2017, pp.629-659. DOI: 10.1007/978-3-319-70694-8_22.

[7] Hazay C, Lindell Y. Efficient oblivious polynomial evaluation with simulation-based security. *IACR Cryptology ePrint Archive*, 2009, 2009: Article No. 459.

[8] Zhu H, Bao F. Augmented oblivious polynomial evaluation protocol and its applications. In *Proc. the 10th European Symposium on Research in Computer Security*, Sept. 2005, pp.222-230. DOI: 10.1007/11555827_13.

[9] Li H D, Yang X, Feng D G, Li B. Distributed oblivious function evaluation and its applications. *Journal of Computer Science and Technology*, 2004, 19(6): 942-947. DOI: 10.1007/BF02973458.

[10] Naor M, Pinkas B. Oblivious polynomial evaluation. *SIAM Journal on Computing*, 2006, 35(5): 1254-1281. DOI: 10.1137/S0097539704383633.

[11] Tonicelli R, Nascimento A C A, Dowsley R, Müller-Quade J, Imai H, Hanaoka G, Otsuka A. Information-theoretically secure oblivious polynomial evaluation in the commodity-based model. *International Journal of Information Security*, 2015, 14(1): 73-84. DOI: 10.1007/s10207-014-0247-8.

[12] Döttling N, Ghosh S, Nielsen J B, Nilges T, Trifiletti R. TinyOLE: Efficient actively secure two-party computation from oblivious linear function evaluation. In *Proc. the 2017 ACM SIGSAC Conference on Computer and Communications Security*, October 30-November 3, 2017, pp.2263-2276. DOI: 10.1145/3133956.3134024.

[13] Özarar M, Özgit A. Secure multiparty overall mean computation via oblivious polynomial evaluation. In *Proc. the 1st International Conference on Security of Information and Networks*, May 2007, pp.84-95.

[14] Chang Y C, Lu C J. Oblivious polynomial evaluation and oblivious neural learning. *Theoretical Computer Science*, 2005, 341(1/2/3): 39-54. DOI: 10.1016/j.tcs.2005.03.049.

[15] Ogata W, Kurosawa K. Oblivious keyword search. *Journal of Complexity*, 2004, 20(2/3): 356-371. DOI: 10.1016/j.jco.2003.08.023.

[16] Lindell P. Privacy preserving data mining. *Journal of Cryptology*, June 2002, 15(3): 177-206. DOI: 10.1007/s00145-001-0019-2.

[17] Damgård I, Haagh H, Nielsen M, Orlandi C. Commodity-based 2PC for arithmetic circuits. In *Proc. the 17th IMA International Conference on Cryptography and Coding*, Dec. 2019, pp.154-177. DOI: 10.1007/978-3-030-35199-1_8.

[18] Damgård I, Pastro V, Smart N, Zakarias S. Multiparty computation from somewhat homomorphic encryption. In *Proc. the 32nd Annual Cryptology Conference*, Aug. 2012, pp.643-662. DOI: 10.1007/978-3-642-32009-5_38.

[19] Keller M, Orsini E, Scholl P. MASCOT: Faster malicious arithmetic secure computation with oblivious transfer. In *Proc. the 2016 ACM SIGSAC Conference on Computer and Communications Security*, Oct. 2016, pp.830-842.

[20] Lindell Y, Pinkas B, Smart N P, Yanai A. Efficient constant round multi-party computation combining BMR and SPDZ. In *Proc. the 35th Annual Cryptology Conference*, Aug. 2015, pp.319-338. DOI: 10.1007/978-3-662-48000-7_16.

[21] Hazay C. Oblivious polynomial evaluation and secure set-intersection from algebraic PRFs. *Journal of Cryptology*, 2018, 31(2): 537-586. DOI: 10.1007/s00145-017-9263-y.

[22] Otsuka A, Imai H. Unconditionally secure electronic voting. In *Towards Trustworthy Elections: New Directions in Electronic Voting*, Chaum D, Jakobsson M, Rivest R, Ryan P, Benaloh J, Kutylowski M, Adida B (eds.), Springer, 2010, pp.107-123. DOI: 10.1007/978-3-642-12980-3_6.

[23] Corniaux C L F, Ghodosi H. An information-theoretically secure threshold distributed oblivious transfer protocol. In *Proc. the 15th International Conference on Information Security and Cryptology*, Nov. 2012, pp.184-201. DOI: https://doi.org/10.1007/978-3-642-37682-5_14.

[24] Crépeau C, Morozov K, Wolf S. Efficient unconditional oblivious transfer from almost any noisy channel. In *Proc. the 4th International Conference on Security in Communication Networks*, Sept. 2004, pp.47-59. DOI: 10.1007/978-3-540-30598-9_4.

[25] Rivest R L. Unconditionally secure commitment and oblivious transfer schemes using private channels and a trusted initializer. http://people.csail.mit.edu/rivest/Rivest-commitment.pdf, Nov. 2021.

[26] Bo Y, Wang Q, Cao Y. An efficient and unconditionally-secure oblivious polynomial evaluation protocol. In *Proc. the 1st International Symposium on Data, Privacy, and E-Commerce*, Nov. 2007, pp.181-184. DOI: 10.1109/ISDPE.2007.60.

[27] Chor B, Kushilevitz E. A zero-one law for Boolean privacy. *SIAM Journal on Discrete Mathematics*, 1991, 4(1): 36-47. DOI: 10.1137/0404004.

[28] Cramer R, Damgård I B, Nielsen J B. Secure Multiparty Computation and Secret Sharing. Cambridge University Press, 2015. DOI: 10.1017/CBO9781107337756.

[29] Corniaux C L F, Ghodosi H. A verifiable distributed oblivious transfer protocol. In *Proc. the 16th Australasian Conference on Information Security and Privacy*, July 2011, pp.444-450. DOI: 10.1007/978-3-642-22497-3_33.

[30] Blundo C, D'Arco P, De Santis A, Stinson D. On unconditionally secure distributed oblivious transfer. *Journal of Cryptology*, 2007, 20(3): 323-373. DOI: 10.1007/s00145-007-0327-2.

[31] Shamir A. How to share a secret. *Commun. ACM*, 1979, 22(11): 612-613. DOI: 10.1145/359168.359176.

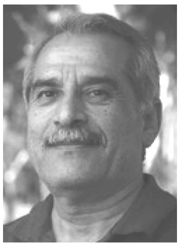
[32] Cheong K Y, Koshiba T, Nishiyama S. Strengthening the security of distributed oblivious transfer. In *Proc. the 14th Australasian Conference on Information Security and Privacy*, July 2009, pp.377-388. DOI: 10.1007/978-3-642-02620-1_26.

[33] Naor M, Pinkas B. Distributed oblivious transfer. In *Proc. the 6th International Conference on the Theory and Application of Cryptology and Information Security*, Dec. 2000, pp.205-219. DOI: 10.1007/3-540-44448-3_16.

[34] Hanaoka G, Imai H, Mueller-Quade J, Nascimento A C A, Otsuka A, Winter A. Information theoretically secure oblivious polynomial evaluation: Model, bounds, and constructions. In *Proc. the 9th Australasian Conference on Information Security and Privacy*, July 2004, pp.62-73. DOI: 10.1007/978-3-540-27800-9_6.

[35] Beaver D. Commodity-based cryptography (extended abstract). In *Proc. the 29th Annual ACM Symposium on Theory of Computing*, May 1997, pp.446-455. DOI: 10.1145/258533.258637.

Louis Cianciullo received his Bachelor's degree (Hons) in information technology from James Cook University (JCU), Townsville, in 2016. He is now employed as a software engineer and is also working on completing his Ph.D. degree in computer science from JCU, Townsville. His research focuses on multi-party computation and information theoretic cryptography protocols.



Hossein Ghodosi completed his Ph.D. degree in computer science under the supervision of Professor Josef Pieprzyk at University of Wollongong, Wollongong (1998). He is an associate professor at James Cook University, Townsville, with research interests in multi-party computation, oblivious transfer, and secret sharing schemes.