Unconditionally Secure Distributed Oblivious Polynomial Evaluation

Louis Cianciullo(✉) and Hossein Ghodosi

James Cook University, Townsville 4811, Australia
{louis.cianciullo,hossein.ghodosi}@jcu.edu.au

Abstract. Oblivious polynomial evaluation (OPE) was first introduced by Naor and Pinkas in 1999. An OPE protocol involves a receiver, R who holds a value, α and a sender, S with a private polynomial, $f(x)$. OPE allows R to compute $f(\alpha)$ without revealing either α or $f(x)$. Since its inception, OPE has been established as an important building block in many distributed applications.

In this article we investigate a method of achieving unconditionally secure distributed OPE (DOPE) in which the function of the sender is distributed amongst a set of n servers. Specifically, we introduce a model for DOPE based on the model for distributed oblivious transfer (DOT) described by Blundo et al. in 2002. We then describe a protocol that achieves the security defined by our model.

Our DOPE protocol is efficient and achieves a high level of security. Furthermore, our proposed protocol can also be used as a DOT protocol with little to no modification.

1 Introduction

Oblivious polynomial evaluation (OPE) was first introduced by Naor and Pinkas in 1999 [20]. An OPE protocol involves two parties, a receiver, R who holds a private value, α and a sender, S who holds a private polynomial, $f(x)$. Informally, an OPE protocol allows R to learn the evaluation of S's polynomial at his private value i.e. $f(\alpha)$, whilst keeping S from learning α and R from learning any more information about $f(x)$. A more formal definition, adapted from [7] is given below:

Definition 1 [7]. *An OPE protocol is composed of two parties, S who has a polynomial $f(x)$ over a finite field $\mathbb{F}$ and R who has an input value $\alpha \in \mathbb{F}$. Correctness is achieved if, at the end of the protocol, R learns $f(\alpha)$. Security is guaranteed if the following two conditions are met after the protocol has been executed:*

L. Cianciullo—This research is supported by an Australian Government Research Training Program (RTP) Scholarship.

K. Lee (Ed.): ICISC 2018, LNCS 11396, pp. 132–142, 2019.
https://doi.org/10.1007/978-3-030-12146-4_9

1. *S cannot reduce his uncertainty of α, i.e. the probability of S computing α is $1/|\mathbb{F}|$.*
2. *R does not learn any information relating to $f(x)$, other than $f(\alpha)$.*

OPE has been found to have a myriad of applications in such things as secure computation [12], oblivious neural learning [7], secure set intersection [15] and privacy preserving data mining [18]. As a result of this, an extensive amount of research has been conducted on this topic [7,13–15,17,19,24,25].

Within the literature, OPE protocols come in two flavours, (1) computationally (conditionally) secure protocols, which are secure against an adversary that is computationally bounded, and security is based on cryptographic assumptions; and (2) unconditionally (information theoretic) secure protocols, where the adversary is computationally unbounded. We limit the focus of this article to unconditionally secure OPE protocols.

To the best of the author's knowledge there exists only three unconditionally secure OPE protocols in the literature. The first unconditionally secure OPE was given by Chang and Lu [7]. To achieve information theoretic security they use a third party who takes an active role in the protocol execution. The second information theoretic secure OPE protocol was given by Hanaoka et al. in [14] (and was later expanded on in [24]). Their protocol also requires the use of a third party although, in their protocol the third party acts as an initialiser, in that he merely distributes some (unrelated, effectively random) information at the start of the protocol and then takes no further part in the protocol execution. The third OPE protocol that achieves information theoretic security was given by Li et al. [17]. Their protocol takes a different approach and instead utilises a set of servers to collectively implement the function of the sender. We denote such a scheme as a distributed oblivious polynomial evaluation (DOPE) protocol in order to differentiate this type of scheme from the other three-party protocols.

In the DOPE protocol of Li et al. [17] the sender initialises the protocol by distributing some information amongst a set of $n \geq 2$ servers. Following this, S takes no further part in the protocol. To compute his evaluation, R communicates with a subset composed of t amount of these servers where $t \leq n$ is known as the threshold. The sender's security is guaranteed against a coalition composed of $l-1$ servers and R; whilst the receiver's privacy is guaranteed against a subset of $b-1$ servers, where $b+l < t \leq n$. Li et al. also show how to improve this scheme allowing for the greater threshold of $b = l = t$ by introducing some publicly known information. However, we note that this increase in security comes at a cost. Namely, it increases the overall complexity of their protocol and it also allows both R and the servers to gain some extra information about $f(x)$. Since OPE protocols (and by extension DOPE protocols) are generally used as building blocks in larger multi-party protocols an OPE protocol that leaks information relating to $f(x)$ may result in security flaws in the overlying multi-party protocol.

As a result of this, an efficient DOPE protocol that does not leak any information and still achieves a high level of security is needed.

1.1 Our Contribution

In this paper we develop such a protocol by first describing a model of DOPE and then introducing an efficient DOPE protocol that achieves the security defined in our model. Specifically, our proposed protocol allows R to compute his evaluation by simply broadcasting some information and then receiving contact from t or more servers. The protocol achieves security for R against a coalition of $t-1$ servers and security for S against a coalition composed of $t-1$ servers and R and does not leak any information relating to either $f(x)$ or α.

To develop a model of DOPE we simply apply a slightly modified version of the already established and well studied security framework developed by Blundo et al. [4,5] for the purpose of distributed oblivious transfer (DOT) [1,8–10,21,22]. We then give the construction of a DOPE protocol that is secure under this model. An interesting property of our protocol is that it can also be utilised as a DOT protocol with little to no modification.

Our protocol achieves security equivalent to what Blundo et al. describe as a strong DOT protocol [4]. That is, our DOPE protocol is secure against a coalition composed of $t-1$ servers and R even after R has received $f(\alpha)$.

2 Model

Similar to a DOT protocol a DOPE protocol consists of a sender, S, the receiver, R and n servers, $s_1, \cdots, s_n$. As per Definition 1 the sender has a polynomial, $f(x)$ of degree $k \geq 1$ over $\mathbb{F}$, whilst the receiver has a point $\alpha \in \mathbb{F}$, such that $|\mathbb{F}| = q$ where q is a prime number and $q > \max(k, n)$. We assume a standard model of communication present in many multi-party protocols [2] i.e. a synchronous broadcast connection exists between the servers and R, such that R can privately and simultaneously send the same message to all of the servers. Additionally, we assume each server has a secure channel that allows them to send private messages to R. DOPE consists of two phases:

1. **Initialisation:** S privately distributes some information relating to $f(x)$ to each of the n servers. Following this S takes no further part in the protocol.
2. **Evaluation:** R broadcasts some information to all of the servers. A set of t or more servers send a response to R who then uses this information to compute $f(\alpha)$.

In order to achieve both correctness and security a DOPE protocol must satisfy the following security conditions, originally given by Blundo et al. [4] and informally stated by Corniaux and Ghodosi [9] for the purpose of DOT:

1. ***Correctness:*** 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 relating to α.

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

In our model we assume that all participants follow the protocol exactly, i.e. they are semi-honest. A benefit of our model is that the degree of $f(x)$ (given as k) is not related to the threshold parameter, t. This allows for a flexible and easily changeable level of security. For instance, even if the degree of k is small S can ensure security against a large number of servers by assigning a high value to t.

In regards to the security conditions given by Blundo et al. it was shown that a DOT protocol that achieves all four security conditions could only be achieved in two rounds of communication between the servers and R or by allowing S to contact R during the initialisation phase. This also proves true for our DOPE protocol which is given in the next section. We note that, similar to Blundo's "Strong DOT Protocol" [4] our protocol assumes that S correctly distributes the information to the servers and does not try to initiate any further contact with R or the servers after the initialisation phase.

3 DOPE Protocol

In this section we describe our DOPE protocol and then evaluate the security of the protocol against the security conditions given in the previous section.

In our proposed protocol S utilises Shamir's secret sharing scheme to securely distribute his polynomial among the n servers. For completeness, we will firstly review Shamir's secret sharing scheme.

3.1 Shamir's Secret Sharing Scheme

In a threshold secret sharing scheme a special participant, known as the dealer, distributes shares of his secret value, s, amongst n participants, in such a way that any t of these participants can reconstruct s. Whilst $t-1$ or fewer participants cannot compute any information relating to s. Secret sharing is a fundamental building block of many distributed protocols. The specific secret sharing scheme used in this article is Shamir's secret sharing scheme [23] which is briefly explained below.

Denote the n participants as $P_1, \cdots, P_n$, the dealer as $\mathcal{D}$ and let all calculations take place in the finite field $\mathbb{F}$ where $|\mathbb{F}| = q$ such that $q > n$ is a prime number. The scheme consists of two phases, the sharing phase and the reconstruction phase.

Sharing Phase

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

Reconstruction Phase

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

3.2 The Proposed DOPE Protocol

The underlying idea behind our protocol is similar to the protocol given by Li et al. [17], in that we have S utilise Shamir's secret sharing scheme to distribute shares of the coefficients of $f(x)$ to each server.

To achieve privacy for R we have S distribute some semi-random information along with the shares of the coefficients. Each server receives shares of this information whilst R receives the information in its entirety. Using the distributed information R can then easily distribute his value α among the servers, who then perform a computation and send the output back to R. Following this, R computes a polynomial of which the free term is his desired evaluation.

The actual method utilised to distribute shares of α was originally given in [11] as a means to securely introduce input values under a shared MAC key in multi-party computation. We specifically use it to allow the contacted servers to efficiently compute a share of α multiplied by a given coefficient of $f(x)$. The full OPE protocol is given in Fig. 1.

In Sect. 2 we stated the result of Blundo et al. [4] which proved that a strong DOT protocol can only be achieved in two rounds. The same is true for our DOPE protocol, we merely circumvented this limitation by allowing S to contact R in the initialisation phase. Specifically, in our protocol we have S directly send the values $r_1, \cdots, r_k$ to R in the initialisation phase. This is actually not strictly necessary, and to limit direct contact between S and R we could instead have S distribute shares of $r_1, \cdots, r_k$ to each server. At the start of the evaluation phase a set of t or more servers would then send R their shares of these values. This results in a two round protocol in which R only has to be present during the evaluation phase. This is, of course, the exact same approach taken by Blundo et al. [4] for their strong DOT protocol.

In fact, due to the similarity of the models our DOPE protocol can easily be converted to a strong $\binom{1}{m}$ DOT protocol. In a $\binom{1}{m}$ DOT protocol the receiver wishes to learn 1 of m secrets held by S. If we define S's secrets as $\omega_1, \cdots, \omega_m$ then we can achieve DOT by having S compute $f(x)$ so that $f(i) = \omega_i$ for $i = 1, \cdots, m$. In this case the degree of the polynomial is then $k = m - 1$. To learn the i^{th} secret R sets $\alpha = i$ and then executes the rest of the protocol as before.

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

Initialisation

1. S creates a set of random values $r_1, \cdots, r_k$ and computes k values of the form $\gamma_i = r_i \cdot a_i$ for $i = 1, \cdots, k$.
2. For each coefficient, a_h $(h = 0, \cdots, k)$, S computes a random polynomial, $A_h(x)$ of degree at most $t-1$ such that $A_h(0) = a_h$. He 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 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. S privately sends to R the values $r_1, \cdots, r_k$ and then takes no further part in the protocol.

Evaluation

1. 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 R's broadcast values. Each server, $s_j \in \mathcal{W}$, computes and sends to R the share:

$$z_j = a_{0_j} + \sum_{i=1}^{k} (a_{i_j} \cdot \epsilon_i + \gamma_{i_j})$$

3. R performs Lagrange interpolation across each z_j value to compute the polynomial $Z(x)$ with free term $Z(0) = f(\alpha)$.

Fig. 1. The proposed DOPE protocol

Where our protocol differs from many DOT protocols [5] however, is that our proposed DOPE protocol allows the receiver to contact more than the threshold amount of t servers. In fact, in our protocol R actually contacts all n servers, and we require t or more servers to respond to R. The specific servers that do respond to R can be chosen in any arbitrary fashion, as long as there are t or more of them. This allows for a fairly robust protocol, in that the protocol can tolerate up to $n - t$ servers not responding to R.

3.3 Evaluation

In this section we evaluate the security of the proposed DOPE protocol by proving that it meets the conditions given in Sect. 2.

Correctness

Theorem 1. *If all participants follow the protocol correctly the receiver obtains $f(\alpha)$ by contacting t or more servers.*

Proof. At the end of the evaluation phase R will have received t or more (up to n) shares of the form:

$$z_j = a_{0_j} + \sum_{i=1}^{k} (a_{i_j} \cdot \epsilon_i + \gamma_{i_j})$$

Where the share z_j is from server s_j. Due to the homomorphic nature of Shamir's secret sharing scheme linear operations performed on shares also correspond to the secrets and polynomials these shares are computed from [3]. In other words the shares correspond to the polynomial:

$$Z(x) = A_0(x) + \sum_{i=1}^{k} (A_i(x) \cdot \epsilon_i + \Gamma_i(x))$$

The free term of each $A_i(x)$ is $A_i(0) = a_i$, similarly $\Gamma_i(0) = r_i \cdot a_i$, therefore:

$$Z(0) = a_0 + \sum_{i=1}^{k} (a_i \cdot \epsilon_i + r_i \cdot a_i)$$

Since $\epsilon_i = \alpha^i - r_i$ this becomes:

$$\begin{aligned} Z(0) &= a_0 + \sum_{i=1}^{k} (a_i \cdot \alpha^i - a_i \cdot r_i + r_i \cdot a_i) \\ &= a_0 + \sum_{i=1}^{k} a_i \cdot \alpha^i \\ &= a_0 + a_1 \cdot \alpha + a_2 \cdot \alpha^2 + \cdots + a_k \cdot \alpha^k \\ &= f(\alpha) \end{aligned}$$

Receiver's Privacy

Theorem 2. *A coalition of $t-1$ servers cannot compute any information relating to α.*

Proof. Suppose that a set of $t-1$ servers, who were all contacted by R, form a coalition. The goal of this coalition is to compute some information relating to α. Collectively the servers have a set of $t-1$ shares relating to each coefficient of $f(x)$, (i.e. $a_0, \cdots, a_k$) as well as $t-1$ shares relating to the product of each random value and a coefficient, i.e. $\gamma_i = a_i \cdot r_i$ for $i = 1, \cdots, k$. Additionally, the servers also have k values of the form $\epsilon_i = \alpha^i - r_i$ which gives the following system of equations:

$$\begin{aligned} \epsilon_1 &= \alpha - r_1 \\ \epsilon_2 &= \alpha^2 - r_2 \\ &\vdots \\ \epsilon_k &= \alpha^k - r_k \end{aligned}$$

From the above system we can see that to compute α the coalition would first need to compute a given r_i value. However, due to the perfectly secure nature of Shamir's secret sharing scheme [6,23], $t-1$ shares does not reveal any information relating to a given secret. As a result of this, the coalition of servers cannot compute any information relating to any of the coefficients of $f(x)$, the γ_i or the r_i values. Since each r_i value is chosen at random, and they cannot compute any information relating to these values the above system is composed of k independent equations and $k+1$ unknowns (each r_i value in addition to α) which results in every possible value of α being equally likely.

Sender's Privacy

Theorem 3. *A coalition composed of $t-1$ servers and R cannot compute any information relating to $f(x)$ during initialisation.*

Proof. At the end of the initialisation phase a coalition of $t-1$ servers and R will have the following information:

1. The values $r_1, \cdots, r_k$.
2. $t-1$ shares corresponding to each coefficient polynomial $(A_0(x), \cdots, A_k(x))$, which gives $(k+1)(t-1)$ shares.
3. $t-1$ shares relating to the each of other set of polynomials $(\Gamma_1(x), \cdots, \Gamma_k(x))$, giving $k(t-1)$ collective shares.

As per the proof of Theorem 1 it is impossible to compute any information about a given polynomial, of degree $t-1$, with only $t-1$ shares. However, the free term of each polynomial of the form $\Gamma_i(x)$ for $1 = 1 \cdots k$ is $\Gamma_i(0) = r_i a_i$ where r_i is a known quantity. The coalition can use this knowledge to compute a polynomial with free term a_i. This allows them to hold two polynomials with the free term a_i.

We note that even with this extra knowledge they cannot achieve anything as a_i is unknown to them and furthermore, holding two sets of $t-1$ shares relating to two different polynomials with the same free term does not actually reveal any information [16,23].

Sender's Privacy After Protocol Execution

Theorem 4. *A coalition composed of $t-1$ servers and R cannot compute any information relating to $f(x)$ after the execution of the protocol, other than what the evaluation of R's chosen value, $f(\alpha)$, gives them.*

Proof. The proof of this is analogous to the previous proof with the addition of some extra information, namely the information given to R by the other servers who contacted him. For the sake of the proof we will assume the worst, i.e. that all n servers contact R. Without loss of generality and for the sake of convenience, assume that the coalition is composed of R and the first $t-1$ servers, $s_1, \cdots, s_{t-1}$. This coalition has the exact same information as before, this time however, they also have the added knowledge of the other $n-t$ server's responses to R. That is:

$$\begin{aligned} z_t &= a_{0_t} + \sum_{i=1}^{k} (a_{i_t} \cdot \epsilon_i + \gamma_{i_t}) \\ z_{t+1} &= a_{0_{t+1}} + \sum_{i=1}^{k} (a_{i_{t+1}} \cdot \epsilon_i + \gamma_{i_{t+1}}) \\ &\vdots \\ z_n &= a_{0_n} + \sum_{i=1}^{k} (a_{i_n} \cdot \epsilon_i + \gamma_{i_n}) \end{aligned}$$

If the coalition are able to compute any of the polynomials used to distribute the coefficients of the senders polynomial, $A_0(x), \cdots, A_k(x)$, or even the polynomials used to distribute the product of the random values and the coefficients, $\Gamma_1(x), \cdots, \Gamma_k(x)$, then they can easily compute the value of a given coefficient of $f(x)$. We must therefore prove that this is not possible.

First, let $h = 0, \cdots, k$ and let $i = 1, \cdots, k$ then any given server, s_j, contacted by R has $k+1$ shares of the form a_{h_j} corresponding to $A_h(x)$ and k shares of the form γ_{i_j} corresponding to $\Gamma_h(x)$. We can write these polynomials as:

$$\begin{aligned} A_h(x) &= a_h + A_{h_1} x + A_{h_2} 2x^2 + \cdots + A_{h_{t-1}} x^{t-1} \\ \Gamma_i(x) &= r_i a_i + G_{i_1} x + G_{i_2} x^2 + \cdots + G_{i_{t-1}} x^{t-1} \end{aligned}$$

Using this notation the response of each server, z_j for $j = 1, \cdots n$, can be written as:

$$z_j = \sum_{y=1}^{k} a_y \alpha^y + \sum_{h=0}^{k} \left(\epsilon_h \Big(\sum_{v=1}^{t-1} A_{h_v} j^v \Big) \right) + \sum_{i=1}^{k} \left(\sum_{v=1}^{t-1} G_{i_v} j^v \right)$$

Therefore, from n responses R obtains a system composed of n equations and $t(k+1)+k(t-1)$ unknowns, specifically, t unknowns from each of the $k+1$ polynomials of the form $A_h(x)$ and $t-1$ unknowns from each of the k amount of polynomials of the form $\Gamma_i(x)$.

However, we note that each z_j is composed of a linear combination of polynomials of degree $t-1$. Therefore, the system that R constructs is only composed of, at most, t independent equations. We note that $t \geq 2$ and $k \geq 1$, meaning that the amount of unknowns will always be greater than the amount of independent equations. As a result of this, R and the coalition of $t-1$ servers cannot compute anything from just the responses.

In fact, even with the direct shares of each of the $t-1$ servers in the coalition they still cannot compute any information. This is because the equation used to describe a given share is not linearly independent to the equation used for a given z_j i.e. each z_j is simply a linear combination of a given participant's share and thus, is not a separate (independent) equation.

The net result for the coalition is a system composed of only t independent equations and $t(k+1)+k(t-1)$ unknowns, resulting in each value of a given coefficient of $f(x)$ being equally likely.

References

1. Beimel, A., Chee, Y.M., Wang, H., Zhang, L.F.: Communication-efficient distributed oblivious transfer. J. Comput. Syst. Sci. **78**(4), 1142–1157 (2012)
2. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing, STOC 1988. ACM, New York (1988)
3. Benaloh, J.C.: Secret sharing homomorphisms: keeping shares of a secret secret (extended abstract). In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 251–260. Springer, Heidelberg (1987). https://doi.org/10.1007/3-540-47721-7_19
4. Blundo, C., D'Arco, P., De Santis, A., Stinson, D.: On unconditionally secure distributed oblivious transfer. J. Cryptol. **20**(3), 323–373 (2007)
5. Blundo, C., D'Arco, P., De Santis, A., Stinson, D.R.: New results on unconditionally secure distributed oblivious transfer. In: Nyberg, K., Heys, H. (eds.) SAC 2002. LNCS, vol. 2595, pp. 291–309. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36492-7_19
6. Corniaux, C.L.F., Ghodosi, H.: An entropy-based demonstration of the security of Shamir's secret sharing scheme. In: 2014 International Conference on Information Science, Electronics and Electrical Engineering, vol. 1, pp. 46–48, April 2014
7. Chang, Y.-C., Lu, C.-J.: Oblivious polynomial evaluation and oblivious neural learning. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 369–384. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_22
8. Cheong, K.Y., Koshiba, T., Nishiyama, S.: Strengthening the security of distributed oblivious transfer. In: Boyd, C., González Nieto, J. (eds.) ACISP 2009. LNCS, vol. 5594, pp. 377–388. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02620-1_26
9. Corniaux, C.L.F., Ghodosi, H.: A verifiable distributed oblivious transfer protocol. In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 444–450. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22497-3_33
10. Corniaux, C.L.F., Ghodosi, H.: An information-theoretically secure threshold distributed oblivious transfer protocol. In: Kwon, T., Lee, M.-K., Kwon, D. (eds.) ICISC 2012. LNCS, vol. 7839, pp. 184–201. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37682-5_14

11. Damgård, I., Pastro, V., Smart, N., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Safavi-Naini, R., Canetti, R. (eds.) CRYPTO 2012. LNCS, vol. 7417, pp. 643–662. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32009-5_38
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: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, pp. 2263–2276. ACM, New York (2017)
13. Ghosh, S., Nielsen, J.B., Nilges, T.: Maliciously secure oblivious linear function evaluation with constant overhead. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017. LNCS, vol. 10624, pp. 629–659. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70694-8_22
14. 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: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 62–73. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27800-9_6
15. Hazay, C.: Oblivious polynomial evaluation and secure set-intersection from algebraic PRFs. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9015, pp. 90–120. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_4
16. Ito, M., Saito, A., Nishizeki, T.: Secret sharing scheme realizing general access structure. Electron. Commun. Jpn. (Part III: Fundam. Electron. Sci.) **72**(9), 56–64 (1989)
17. Li, H.D., Yang, X., Feng, D.G., Li, B.: Distributed oblivious function evaluation and its applications. J. Comput. Sci. Technol. **19**(6), 942–947 (2004)
18. Lindell, Y., Pinkas, B.: Privacy preserving data mining. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 36–54. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_3
19. Naor, M., Pinkas, B.: Oblivious polynomial evaluation. SIAM J. Comput. **35**(5), 1254–1281 (2006)
20. Naor, M., Pinkas, B.: Oblivious transfer and polynomial evaluation. In: Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing, STOC 1999, pp. 245–254. ACM, New York (1999)
21. Naor, M., Pinkas, B.: Distributed oblivious transfer. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 205–219. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44448-3_16
22. Nikov, V., Nikova, S., Preneel, B., Vandewalle, J.: On unconditionally secure distributed oblivious transfer. In: Menezes, A., Sarkar, P. (eds.) INDOCRYPT 2002. LNCS, vol. 2551, pp. 395–408. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-36231-2_31
23. Shamir, A.: How to share a secret. Commun. ACM **22**(11), 612–613 (1979)
24. Tonicelli, R., et al.: Information-theoretically secure oblivious polynomial evaluation in the commodity-based model. Int. J. Inf. Secur. **14**(1), 73–84 (2015)
25. Zhu, H., Bao, F.: Augmented oblivious polynomial evaluation protocol and its applications. In: di Vimercati, S.C., Syverson, P., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 222–230. Springer, Heidelberg (2005). https://doi.org/10.1007/11555827_13