Serdica J. Computing **1** (2007), 313–336

Serdica
Journal of Computing

Bulgarian Academy of Sciences
Institute of Mathematics and Informatics

ON DISTRIBUTED OBLIVIOUS TRANSFER*

Ventzislav Nikov, Svetla Nikova, Bart Preneel

Abstract. This paper is about unconditionally secure distributed protocols for oblivious transfer, as proposed by Naor and Pinkas and generalized by Blundo *et al.* In this setting a Sender has ζ secrets and a Receiver is interested in one of them. The Sender distributes the information about the secrets to n servers, and a Receiver must contact a threshold of the servers in order to compute the secret. We present a non-existence result and a lower bound for the existence of one-round, threshold, distributed oblivious transfer protocols, generalizing the results of Blundo *et al.* A threshold based construction implementing 1-out-of-ζ distributed oblivious transfer achieving this lower bound is described. A condition for existence of distributed oblivious transfer schemes based on general access structures is proven. We also present a general access structure protocol implementing 1-out-of-ζ distributed oblivious transfer.

ACM Computing Classification System (1998): D.4.6.
Key words: Cryptographic Protocols, Oblivious Transfer.

*The paper has been presented at the International Conference Pioneers of Bulgarian Mathematics, Dedicated to Nikola Obreshkoff and Lubomir Tschakaloff, Sofia, July, 2006.
The material in this paper was presented in part at INDOCRYPT 2002 [18]

1. Introduction. *Oblivious Transfer* (OT) refers to several types of two-party protocols where at the beginning of the protocol one party, the *Sender*, has an input, and at the end of the protocol the other party, the *Receiver* (sometimes called the chooser), learns some information about this input in a way that does not allow the Sender to figure out what the Receiver has learned. Introduced by M. Rabin in [20], and subsequently defined in different forms in [13, 1], oblivious transfer has found many applications in cryptographic schemes and protocol designs. A variety of slightly different definitions and implementations can be found in the literature as well as papers addressing issues such as the relation of OT with other cryptographic primitives (e. g., see [4, 6, 2, 12, 11, 17]).

The *Private Information Retrieval* (PIR) and *Symmetric Private Information Retrieval* (SPIR) Schemes, introduced in [7, 15], represent another closely related area. A PIR Scheme enables a user to retrieve an item of information from a public accessible database in such a way that the database manager cannot figure out from the query which item the user is interested in. However, the user can get information about more than one item. On the other hand, in SPIR schemes the user can get information about one and only one item, i. e., even the privacy of the database is considered. In PIR and SPIR schemes the emphasis is placed on the communication complexity of the interaction between user and servers. Other interesting PIR papers for the distributed OT scenario are [5, 9, 14].

Rivest's model given in [21] utilizes a trusted initializer, who participates only in an initial setup phase. The setting of the scheme is similar to the one described in [16] and considered in this paper. In [22] the author deals with distributed oblivious transfer implementations, similar to the settings in [16], but not unconditionally secure.

In this paper we are dealing with *unconditionally secure distributed oblivious transfer protocols*, as introduced by Naor and Pinkas in [16] and generalized by Blundo *et al.* in [3]. *Distributed Oblivious Transfer* (DOT) protocols distribute the task of the Sender between several servers. Security is ensured as long as a limited number of these servers collude (the threshold case). Since in many natural scenarios the assumption that trust is "uniformly distributed" over the players does not model the reality well we want the scheme to be secure in case of general access structures. We extend the threshold model to general access structure DOT scheme.

The paper is organized as follows. In Sect. 2 we introduce multiplicative linear secret sharing schemes and multi-party computation. Sect. 3 is devoted

to threshold case. We introduce the model, then prove a necessary condition for existence of unconditionally secure DOT. A protocol achieving the proven bound is proposed. Sect. 4 deals with the general case. Analogously we first generalize the model, second we prove necessary condition for existence and then a protocol is proposed. Finally we present a connection between multi-party computation and DOT.

2. Preliminaries.

2.1. Linear Secret Sharing Schemes. Denote the *participants* of a Secret Sharing Scheme (SSS) by P_i, $1 \leq i \leq n$, and the set of all *players* by $\mathcal{P} = \{P_1, \ldots, P_n\}$. Denote the *dealer* of the scheme by $\mathcal{D}$. The role of the dealer is to share a secret s to all participants in the scheme. The simplest access structure Γ is called a (k, n)-threshold: all subsets of players $\mathcal{P}$ with at least $k + 1$ participants are *qualified* to reconstruct the secret and any subset of up to k players are *forbidden* of doing it. Accordingly we will call a Secret Sharing Scheme (SSS) (k, n)-threshold if the access structure Γ associated with it is (k, n)-threshold. It is well known that all threshold SSS protocols can be generalized for general access structures using Monotone Span Programs (see Cramer et al. [8]). Denote the set of all subsets of $\mathcal{P}$ (i. e. the power set of $\mathcal{P}$) by $P(\mathcal{P})$. The set of qualified groups is denoted by Γ and the set of forbidden groups by Δ. The set Γ is called *monotone increasing* if for each set A in Γ each set containing A is also in Γ. Similarly, Δ is called *monotone decreasing* if for each set B in Δ each subset of B is also in Δ. The tuple (Γ, Δ) is called an *access structure* if $\Gamma \cap \Delta = \emptyset$. If the union of Γ and Δ is equal to $P(\mathcal{P})$ (so, Γ is equal to Δ^c, the complement of Δ), then we say that access structure (Γ, Δ) is *complete* and we denote it just by Γ. For a complete access structure the dual access structure could be defined as follows. The *dual access structure* $\Gamma^{\perp}$ of an access structure Γ, defined on $\mathcal{P}$, is the collection of sets $A \subseteq \mathcal{P}$ such that $\mathcal{P} \setminus A = A^c \notin \Gamma$.

It is common to model cheating by considering an *adversary* $\mathcal{A}$ who may corrupt some of the players *passively* and some *actively*. Passive corruption means that the adversary obtains the complete information held by the corrupt players, but the players execute the protocol correctly. Active corruption means that the adversary takes full control of the corrupt players. Thus the adversary $\mathcal{A}$ is characterized by a particular subset $\Delta_{\mathcal{A}}$ of Δ, which is itself a monotone decreasing structure. The set $\Delta_{\mathcal{A}}$ ($\Delta_{\mathcal{A}} \subseteq \Delta$) is called an *adversary structure* while the set Δ is called a *privacy structure*. The players which belong to Δ

are also called *curious* and the players which belong to $\Delta_{\mathcal{A}}$ are called *corrupt*. An $(\Delta, \Delta_{\mathcal{A}})$-adversary is an adversary who can (adaptively) corrupt some players passively and some players actively, as long as the set A of actively corrupt players and the set B of passively corrupt players satisfy both $A \in \Delta_{\mathcal{A}}$ and $(A \cup B) \in \Delta$.

Now we give a formal definition of a Monotone Span Program.

Definition 2.1 [8]. *A* Monotone Span Program *(MSP) $\mathcal{M}$ is a quadruple $(\mathbb{F}, M, \boldsymbol{\varepsilon}, \psi)$, where $\mathbb{F}$ is a finite field, M is a matrix (with m rows and $d \leq m$ columns) over $\mathbb{F}$, $\psi : \{1, \ldots, m\} \to \{1, \ldots, n\}$ is a surjective function and $\boldsymbol{\varepsilon} = (1, 0, \ldots, 0)^T \in \mathbb{F}^d$ is called the* target vector*.*

As ψ labels each row with an integer i from $[1, \ldots, m]$ that corresponds to player $P_{\psi(i)}$, we can think of each player as being the "owner" of one or more rows. Also consider a "function" φ from $[1, \ldots, n]$ to $[1, \ldots, m]$ which gives for every player P_i the set of rows owned by him (denoted by $\varphi(P_i)$). In some sense φ is the "inverse" of ψ. Let M_A denote the restriction of M to the rows i with $i \in A$. An MSP is said *to compute* a (complete) access structure Γ when $\boldsymbol{\varepsilon} \in \mathrm{im}(M_A^T)$ if and only if A is a member of Γ. We denote such an access structure by $\Gamma(\mathcal{M})$. We say that A is *accepted* by $\mathcal{M}$ if and only if $A \in \Gamma$, otherwise we say A is *rejected* by $\mathcal{M}$. In other words, the players in A can reconstruct the secret precisely if the rows they own contain in their linear span the target vector of $\mathcal{M}$, and otherwise they get no information about the secret. Hence when a set A is accepted by $\mathcal{M}$ there exists a so-called *recombination vector* (column) $\boldsymbol{\lambda}$ such that $M_A^T \boldsymbol{\lambda} = \boldsymbol{\varepsilon}$. Notice that the vector $\boldsymbol{\varepsilon} \notin \mathrm{im}(M_B^T)$ if and only if there exists a vector $\boldsymbol{k} \in \mathbb{F}^d$ such that $M_B \boldsymbol{k} = \mathbf{0}$ and $\boldsymbol{k}_1 = 1$. For any two monotone *decreasing* sets Δ_1, Δ_2 the operation *element-wise union* $\uplus$ is defined as follows: $\Delta_1 \uplus \Delta_2 = \{A = A_1 \cup A_2; A_1 \in \Delta_1, A_2 \in \Delta_2\}$. For any two monotone *increasing* sets Γ_1, Γ_2 the operation *element-wise union* $\uplus$ is defined as follows: $\Gamma_1 \uplus \Gamma_2 = \{A = A_1 \cup A_2; A_1 \notin \Gamma_1, A_2 \notin \Gamma_2\}^c$.

2.2. Multiplicative Linear SSSs. Cramer *et al.* proposed in [8] an approach to build a Multi-Party Computation (MPC) protocol from any Linear SSS introducing so-called *(strongly) multiplicative* LSSS. The construction for multiplicative MSPs was extended in [19] by proposing the diamond operation $\diamond$. Next we provide the definition and some basic properties of this operation.

Let Γ_1 and Γ_2 be two access structures, computed by MSPs $\mathcal{M}_1 = (\mathbb{F}, M^{(1)}, \boldsymbol{\varepsilon^1}, \psi_1)$ and $\mathcal{M}_2 = (\mathbb{F}, M^{(2)}, \boldsymbol{\varepsilon^2}, \psi_2)$. Let $M^{(1)}$ be an $m_1 \times d_1$ matrix, $M^{(2)}$ be an $m_2 \times d_2$ matrix, and let φ_1, φ_2 be the "inverse" functions of ψ_1 and

ψ_2. Consider a vector $\mathbf{x}$. Let the coordinates in $\mathbf{x}$ which belong to the player P_j form a sub-vector $\mathbf{x^j} \in \mathbb{F}^{|\varphi(P_j)|}$ and let $\mathbf{x} = (\mathbf{x^1}, \ldots, \mathbf{x^n})$. Given an m_1-vector $\mathbf{x}$ and an m_2-vector $\mathbf{y}$, $\mathbf{x} \diamond \mathbf{y}$ will denote the vector containing all entries of the form $x_i y_j$, where $\psi_1(i) = \psi_2(j)$. Thus the *diamond* operation $\diamond$ for vectors can be defined as follows:

$$\mathbf{x} \diamond \mathbf{y} = (\mathbf{x^1} \otimes \mathbf{y^1}, \ldots, \mathbf{x^n} \otimes \mathbf{y^n}), \tag{1}$$

where $\otimes$ is the usual tensor vector product. So, $\mathbf{x} \diamond \mathbf{y}$ has $m = \sum_{P_u \in \mathcal{P}} |\varphi_1(u)||\varphi_2(u)|$ entries, and note that $m < m_1 m_2$. Let $M_u^{(1)}$ denote the matrix formed by the rows of $M^{(1)}$ owned by player P_u. Correspondingly, let $M_u^{(2)}$ denote the matrix formed by the rows of $M^{(2)}$ owned by player P_u. Then $M_u^{(1)}$ is an $|\varphi_1(u)| \times d_1$ matrix and $M_u^{(2)}$ is an $|\varphi_2(u)| \times d_2$ matrix. Now the diamond operation $\diamond$ for matrices can be defined as follows:

$$M^{(1)} = \begin{pmatrix} M_1^{(1)} \\ \ldots \\ M_n^{(1)} \end{pmatrix}, \quad M^{(2)} = \begin{pmatrix} M_1^{(2)} \\ \ldots \\ M_n^{(2)} \end{pmatrix}, \quad \text{and}$$

$$M^{(1)} \diamond M^{(2)} = \begin{pmatrix} M_1^{(1)} \otimes M_1^{(2)} \\ \ldots \\ M_n^{(1)} \otimes M_n^{(2)} \end{pmatrix}. \tag{2}$$

In other words, the diamond operation $\diamond$ for vectors (and analogously for matrices) is defined as the concatenation of vectors (matrices) which are the tensor ($\otimes$) multiplication of the sub-vectors (sub-matrices) belonging to a fixed player.

The *diamond* operation $\diamond$ for MSPs is defined as follows:

Definition 2.2 [19]**.** *Let MSPs* $\mathcal{M}_1 = (\mathbb{F}, M^{(1)}, \boldsymbol{\varepsilon^1}, \psi_1)$ *and* $\mathcal{M}_2 = (\mathbb{F}, M^{(2)},$
$\boldsymbol{\varepsilon^2}, \psi_2)$*. Define an MSP* $\mathcal{M}_1 \diamond \mathcal{M}_2 = (\mathbb{F}, M^{(1)} \diamond M^{(2)}, \boldsymbol{\varepsilon^1} \otimes \boldsymbol{\varepsilon^2}, \psi)$*, where* $\psi(i,j) = r$ *if and only if* $\psi_1(i) = \psi_2(j) = r$*.*

Lemma 2.3 [19]**.** *Let* $M^{(1)}$ *be an* $m_1 \times d_1$ *matrix and* $M^{(2)}$ *be an* $m_2 \times d_2$ *matrix,* $N^{(1)}$ *be an* $n_1 \times m_1$ *matrix and* $N^{(1)}$ *be* $n_2 \times m_2$ *matrix. Let* $\boldsymbol{a} \in \mathbb{F}^{d_1}$, $\boldsymbol{b} \in \mathbb{F}^{d_2}$ *be column vectors, then the following equalities hold:*

$$\begin{aligned} (M^{(1)} \diamond M^{(2)})\ (\boldsymbol{a} \otimes \boldsymbol{b}) &= (M^{(1)}\boldsymbol{a}) \diamond (M^{(2)}\boldsymbol{b}) \\ (N^{(1)}\ M^{(1)}) \diamond (N^{(2)}\ M^{(2)}) &= (N^{(1)} \diamond N^{(2)})(M^{(1)} \otimes M^{(2)}). \end{aligned}$$

Note that the diamond operation $\diamond$ confirms our intuitive expectations that the players can locally compute their new shares. Moreover, the next lemma shows that a (strongly) multiplicative product MSP computes the product of the secrets shared by the MSPs $\mathcal{M}_1$ and $\mathcal{M}_2$.

Theorem 2.4 [19]. *Let* $\mathbf{s^1} = M^{(1)}(s_1, \mathbf{a})$ *and* $\mathbf{s^2} = M^{(2)}(s_2, \mathbf{b})$ *be the shares distributed by the MSPs* $\mathcal{M}_1$ *and* $\mathcal{M}_2$*, for secrets* s_1 *and* s_2 *respectively. Then* $\mathbf{s^1} \diamond \mathbf{s^2}$ *are the shares distributed by the MSP* $\mathcal{M}_1 \diamond \mathcal{M}_2$ *for the secret* $s_1 s_2$*.*

2.3. Multi-Party Computation. The goal of *multi-party computation* (MPC) is to enable a set of players to evaluate an arbitrary function on their private inputs. The computation must guarantee the correctness of the result while preserving the privacy of the players' inputs, even if some of the players are corrupted by an adversary and misbehave in an arbitrary way. Consider n players, each player P_i holding an input x_i. The players want to compute a function $F(x_1, \ldots, x_n) = (y_1, \ldots, y_n)$ in a *secure* manner, which intuitively means that the adversary cannot disrupt the computation, i. e. the value computed is *correct*. Furthermore the adversary should not learn any information about the inputs of the good players (except for what is related to the function value).

We consider an adversary with two privacy structures Δ_1, Δ_2 and with one adversary structure $\Delta_{\mathcal{A}} \subseteq \Delta_1$, $\Delta_{\mathcal{A}} \subseteq \Delta_2$, let us call it a $(\Delta_1, \Delta_2, \Delta_{\mathcal{A}})$-adversary.

Definition 2.5. *We call an MSP* $\mathcal{M}$ trivial *if* $\Gamma(\mathcal{M})^- = \{\{P_1\}, \ldots, \{P_n\}\}$.

Thus a trivial $\mathcal{M}$ has an $n \times 1$ matrix $M = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$.

Remark 2.6. Let $\mathcal{M}$ be a trivial MSP and let $\widetilde{\mathcal{M}}$ be an MSP. It is easy to verify that $\widetilde{\mathcal{M}} = \mathcal{M} \diamond \widetilde{\mathcal{M}}$ and hence $\Gamma(\mathcal{M} \diamond \widetilde{\mathcal{M}}) = \Gamma(\widetilde{\mathcal{M}})$.

So, trivial MSPs play the role of a unit for diamond operation. In the next definition we require more interesting properties to hold.

Definition 2.7 [19]**.**

- *We call* $\mathcal{M}$ *a* product MSP *if there exist non-trivial MSPs* $\mathcal{M}_1$ *and* $\mathcal{M}_2$ *such that* $\mathcal{M} = \mathcal{M}_1 \diamond \mathcal{M}_2$*.*

- *We call* $\mathcal{M}$ *a* multiplicative product MSP *if* $\mathcal{M}$ *is product MSP and* $\Gamma(\mathcal{M}) = \{\mathcal{P}\}$.
- *We call* $\mathcal{M}$ *a* strongly multiplicative product MSP *if* $\mathcal{M}$ *is product MSP and* $\{\mathcal{P}\} \subsetneqq \Gamma(\mathcal{M})$.

We will not consider the case when the access structure $\Gamma(\mathcal{M}_1 \diamond \mathcal{M}_2) = \emptyset$. We call the access structure $\Gamma(\mathcal{M}_1 \diamond \mathcal{M}_2)$ *diamond product access structure.*

Proposition 2.8 [19]**.** *Let* $\mathcal{M}_1$ *and* $\mathcal{M}_2$ *be MSPs, then*

$$\Gamma(\mathcal{M}_1 \diamond \mathcal{M}_2) \subseteq \Gamma(\mathcal{M}_1) \uplus \Gamma(\mathcal{M}_2). \tag{3}$$

Define matrix $\overline{E}$ to be a zero matrix except for the entry in the upper left corner which is 1, or in other words $\overline{E} = \boldsymbol{\varepsilon}(\boldsymbol{\varepsilon}^*)^T$.

Theorem 2.9 [19]**.** *Let* $\mathcal{M}$ *and* $\mathcal{M}^\perp$ *be dual MSPs and let* $M^T M^\perp = \overline{E}$. *Then* $\Gamma(\mathcal{M} \diamond \mathcal{M}^\perp) = \{\mathcal{P}\}$, *i. e.* $\mathcal{M} \diamond \mathcal{M}^\perp$ *is a multiplicative product MSP.*

Theorem 2.10 [19]**.** *Let* $\mathcal{M} = \mathcal{M}_1 \diamond \mathcal{M}_2$. *Then there exists a MPC protocol unconditionally secure against a* $(\Delta_1, \Delta_2, \Delta_{\mathcal{A}})$*-adversary if*

$$\Gamma_{\mathcal{A}}^{\perp} \subseteq \Gamma(\mathcal{M}) \subseteq \Gamma(\mathcal{M}_1) \uplus \Gamma(\mathcal{M}_2), \quad (\Gamma_{\mathcal{A}} \uplus \Gamma_{\mathcal{A}})^{\perp} \subseteq \Gamma(\mathcal{M}_i), \quad \textit{for} \quad i = 1, 2$$

where $\Delta_i = \Gamma(\mathcal{M}_i)^c$.

Note that in case the adversary is passive, i. e., $\Delta_{\mathcal{A}} = \emptyset$, the theorem requires at least $\mathcal{M}$ to be the multiplicative product of $\mathcal{M}_1$ and $\mathcal{M}_2$.

3. Threshold DOT.

3.1. The Model. An r-out-of-n $DOT-\binom{\zeta}{1}$ protocol involves three types of parties:

- A *Sender* $\mathcal{D}$ which has ζ inputs (secrets) $s_0, s_1, \ldots, s_{\zeta-1}$. It is convenient to assume that these inputs are elements in a finite field $\mathbb{F}$.
- A *Receiver* $\mathcal{R}$ that has an input (index) $\sigma \in \{0, 1, \ldots, \zeta - 1\}$.
- Additional n *servers*, $P_1, P_2, \ldots, P_n$.

We assume that the Sender holds ζ secrets and the Receiver is interested in one of them. In the distributed setting the Sender $\mathcal{D}$ does not directly interact with the Receiver $\mathcal{R}$ in order to carry out the oblivious transfer. Rather, he *delegates* n servers to accomplish this task for him.

The protocol is composed of the following functional steps:

- *Initialization Phase.* Let $P_1, P_2, \ldots, P_n$ denote the servers. Sender $\mathcal{D}$ generates n programs $Prog_1, Prog_2, \ldots, Prog_n$ and, for $i = 1, \ldots, n$, sends in a *secure way*, program $Prog_i$ to server P_i. Each program $Prog_i$ depends on the secrets $s_0, s_1, \ldots, s_{\zeta-1}$ and on some random data.

- *Oblivious Transfer Phase.* The Receiver $\mathcal{R}$ holds a program $Prog_R$ which enables her to interact with a subset $\{P_{i_1}, \ldots, P_{i_r}\}$ of r servers at her choice. She sends to the server P_i a query $Quer_i$ which is a function of σ and i, and of some random data. The server answers the query with $Answ_i$. Using the answers that the Receiver $\mathcal{R}$ has collected, she is able to recover the secret of her choice, receiving no information about the other secrets. At the same time, any subset of k servers, say $\{P_{i_1}, \ldots, P_{i_k}\} \subseteq \{P_{i_1}, \ldots, P_{i_r}\}$, does not gain any information about the secret she has recovered.

A formal definition follows now.

Definition 3.1 [16, 10]**.** *An* $(r, n) - DOT - \binom{\zeta}{1}$ *must guarantee the following properties:*

- Reconstruction. *If the Receiver gets information from* r *out of the* n *servers, she can compute the secret* s_σ*.*

- Sender's Privacy. *Given the information presented by a group of* r *servers, the Receiver must gain information about a single secret, and no information about the other secrets.*

- Receiver's Privacy. *No coalition of up to* k *servers gains information about which secret the Receiver has recovered, except what could be implied by the inputs.*

- Receiver-servers collusion. *A coalition of the Receiver with up to* ℓ *corrupt servers cannot learn about the* ζ *secrets more than can be learned by the Receiver herself.*

We will follow the notations and the formal model given by Blundo *et al.* in [3]. Assume that $\mathcal{D}$ holds a program $Prog_D$ to generate n programs $Prog_1, \ldots, Prog_n$ enabling $P_1, \ldots, P_n$ and $\mathcal{R}$ to perform $(r,n) - DOT - \binom{\zeta}{1}$ protocol of his ζ secrets. $\mathcal{R}$ holds an associated program $Prog_R$ for interacting with the servers. The $n+1$ programs $Prog_1, \ldots, Prog_n$ and $Prog_R$, specify the computations to be performed to achieve $(r,n) - DOT - \binom{\zeta}{1}$. In order to model dishonest behavior, where a coalition of at most k servers tries to figure out which secret $\mathcal{R}$ has recovered from the transfer, we assume that cheating servers $P_{i_1}, \ldots, P_{i_k}$ hold a modified version of the programs, denoted by $\overline{Prog}_{i_1}, \ldots, \overline{Prog}_{i_k}$. These programs could have been generated either by a dishonest $\mathcal{D}$, who holds a cheating program $\overline{Prog}_D$, or could have been modified by the dishonest servers. Similarly, a cheating Receiver $\mathcal{R}$, who tries to gain some information about other secrets, holds a modified version of program $\overline{Prog}_R$. These programs can be described by random variables, denoted by $\widetilde{\mathcal{D}}, \widetilde{P}_1, \ldots, \widetilde{P}_n, \widetilde{\mathcal{R}}$ and the cheating programs with random variables $\widetilde{\overline{\mathcal{D}}}, \widetilde{\overline{P}}_1, \ldots, \widetilde{\overline{P}}_n, \widetilde{\overline{\mathcal{R}}}$.

An execution of the protocol can be described by using the following additional random variables: for $j = 1, \ldots, n$ let C_j be the transcript of the communication between $\mathcal{R}$ and P_j. Moreover, let $\mathcal{W}$ be the set of all length ζ sequences of secrets, and, for any $w \in \mathcal{W}$, let w_i be the i-th secret of the sequence. Let $\widetilde{W}$ be the random variable that represents the choice of an element in $\mathcal{W}$, $\widetilde{T}$ be the random variable representing the choice of an index σ in $\mathcal{T} = \{0, 1, \ldots, \zeta - 1\}$ and $\widetilde{C}_j$ be the random variable corresponding to C_j. One can define (as in [3]) the conditions that an $(r,n) - DOT - \binom{\zeta}{1}$ protocol has to satisfy as follows:

Definition 3.2 [3]**.** *For an* $(r,n) - DOT - \binom{\zeta}{1}$ *the sequence of programs* $[Prog_D, Prog_1, \ldots, Prog_n, Prog_R]$ *is* correct *if for any* $i \in \mathcal{T}$ *and for any group* $A = \{P_{i_1}, \ldots, P_{i_r}\} \subseteq \mathcal{P} = \{P_1, \ldots, P_n\}$,

$$H(\widetilde{C}_A \mid \widetilde{P}_A \ \widetilde{T} \ \widetilde{R}) = 0 \tag{4}$$

and for any $w \in \mathcal{W}$

$$H(\widetilde{W}_T \mid \widetilde{C}_A \ \widetilde{T} \ \widetilde{R}) = 0. \tag{5}$$

The definition above means that the transcript of the communication is completely determined by the program of the server P_j and the program of the Receiver and her choices. Moreover, after interacting with r servers, an honest Receiver always recovers the secret in which she is interested.

Assuming that both the Sender $\mathcal{D}$ and the Receiver $\mathcal{R}$ are aware of the joint probability distribution $\mathcal{P}_{\mathcal{W},\mathcal{T}}$ on $\mathcal{W}$ and $\mathcal{T}$, the probability with which $\mathcal{D}$ chooses the secrets in $\mathcal{W}$ and $\mathcal{R}$ chooses an index $i \in T$, the privacy property of $(r,n) - DOT - \binom{\zeta}{1}$ can be defined as follows:

Definition 3.3 [3]. *For an* $(r,n) - DOT - \binom{\zeta}{1}$ *the sequence of programs* $[Prog_D, Prog_1, \ldots, Prog_n, Prog_R]$ *is* private *if*

- *for any group* $B_1 = \{P_{i_1}, \ldots, P_{i_k}\} \subseteq \mathcal{P}$

$$H(\widetilde{T} \mid \widetilde{\overline{P}}_{B_1}\ \widetilde{C}_{B_1}) = H(\widetilde{T}), \tag{6}$$

- *for any program* $\overline{Prog}_R$*, any group* $A = \{P_{i_1}, \ldots, P_{i_r}\} \subseteq \mathcal{P}$*, and any* $i \in \mathcal{T}$

$$H(\widetilde{W} \setminus \widetilde{W}_T \mid \widetilde{T}\ \widetilde{\overline{R}}\ \widetilde{C}_A\ \widetilde{W}_T) = H(\widetilde{W} \setminus \widetilde{W}_T), \tag{7}$$

- *for any group* $B_2 = \{P_{i_1}, \ldots, P_{i_\ell}\} \subseteq \mathcal{P}$*, for any* $i \in \mathcal{T}$*, and for any* $\widetilde{\overline{R}}$*,*

$$H(\widetilde{W} \mid \widetilde{T}\ \widetilde{\overline{R}}\ \widetilde{C}_{B_2}\ \widetilde{\overline{P}}_{B_2}) = H(\widetilde{W}), \tag{8}$$

- *for any groups* $B_2 = \{P_{i_1}, \ldots, P_{i_\ell}\} \subseteq \mathcal{P}$ *and* $A = \{P_{i_1}, \ldots, P_{i_r}\} \subseteq \mathcal{P}$*, for any* $i \in \mathcal{T}$*, and for any program* $\overline{Prog}_R$*,*

$$H(\widetilde{W} \setminus \widetilde{W}_T \mid \widetilde{T}\ \widetilde{\overline{R}}\ \widetilde{\overline{P}}_{B_2}\ \widetilde{C}_A\ \widetilde{W}_T) = H(\widetilde{W} \setminus \widetilde{W}_T). \tag{9}$$

Conditions (6), (7) ensure that a dishonest coalition of k servers does not gain information about the index of the Receiver $\mathcal{R}$: a dishonest Receiver $\mathcal{R}$ infers at most one secret among the ones held by $P_1, \ldots, P_n$. Condition (8) takes into account the possibility of an attack against $\mathcal{D}$ performed either by at most ℓ servers alone or with the cooperation of the Receiver $\mathcal{R}$. The condition states that such kind of coalitions do not gain any information about the secrets held by the Sender $\mathcal{D}$. Finally, Condition (9) states that a coalition of ℓ servers and the Receiver cannot compute any information about the other secrets, once the Receiver has obtained a secret.

3.2. Conditions for Existence. Using some tools from Information Theory (see Appendix) and the ideas in [3] we can show that for the one round DOT

protocol a non-existence result holds for the parameters r, k, and ℓ. Consequently we will prove a lower bound for the existence of a DOT with these parameters.

First of all, notice that if the protocol is one round, then $C_j = (Quer_j, Answ_j)$, where $Quer_j$ is the query of the Receiver and $Answ_j$ is the answer of the server. Therefore, Condition (4) can be re-phrased by saying that

$$H(\widetilde{Q}_A \mid \widetilde{R}\ \widetilde{T}) = 0 \qquad \text{and} \qquad H(\widetilde{A}_A \mid \widetilde{Q}_A\ \widetilde{P}_A) = 0. \tag{10}$$

With this notation, we can prove the following non-existence result:

Theorem 3.4 [18]**.** *Let consider an* $(r,n) - DOT - \binom{\varsigma}{1}$ *scheme with parameters* k*, and* ℓ*. If* $r \leq k + \ell$*, then once the Receiver has legally recovered a secret, a coalition of* ℓ *corrupt servers and the Receiver can recover all the other secrets.*

Proof. Let $r = \ell + k$ i. e. $\ell = r - k$. Denote by $A = \{P_{i_1}, \ldots, P_{i_\ell}\}$ and by $B = \{P_{i_{\ell+1}}, \ldots, P_{i_r}\}$. Let $q_1, \ldots, q_r$ be the queries sent by the Receiver when $T = i$, and let $a_1, \ldots, a_r$ be the answers that the servers $P_1, \ldots, P_r$ send back to $\mathcal{R}$. The Receiver's security property (6) with respect to k servers, say $P_{\ell+1}, \ldots, P_r$, implies that there exist queries q_A^σ and answers a_A^σ for any $\sigma \neq i$, such that if

$$H(\widetilde{W}_i \mid \widetilde{Q}_{A\cup B} = q_{A\cup B}\ \widetilde{A}_{A\cup B} = a_{A\cup B}) = 0$$

then

$$H(\widetilde{W}_\sigma \mid \widetilde{Q}_A = q_A^\sigma\ \widetilde{Q}_B = q_B\ \widetilde{A}_A = a_A^\sigma\ \widetilde{A}_B = a_B) = 0.$$

Since the answers given by the servers in A depend only on their own programs $Prog_1, \ldots, Prog_\ell$ and on the received queries (i. e. $H(\widetilde{A}_A \mid \widetilde{Q}_A\ \widetilde{P}_A) = 0$) it follows that

$$H(\widetilde{W} \mid \widetilde{P}_A\ \widetilde{A}_B\ \widetilde{Q}_B\ \widetilde{R}) = 0.$$

Indeed

$$H(\widetilde{W} \mid \widetilde{P}_A\ \widetilde{A}_B\ \widetilde{Q}_B\ \widetilde{R}) \leq \sum_{t\in\mathcal{T}} H(\widetilde{W}_t \mid \widetilde{P}_A\ \widetilde{A}_B\ \widetilde{Q}_B\ \widetilde{R}\ \widetilde{T} = t)$$

and

$$\begin{aligned} H(\widetilde{W}_t \mid \widetilde{P}_A\ \widetilde{A}_B\ \widetilde{Q}_B\ \widetilde{R}\ \widetilde{T} = t) &\leq H(\widetilde{W}_t \mid \widetilde{P}_A\ \widetilde{A}_B\ \widetilde{Q}_{A\cup B}) \\ &\leq H(\widetilde{W}_t \mid \widetilde{A}_{A\cup B}\ \widetilde{Q}_{A\cup B}) = 0. \end{aligned}$$

Therefore the Receiver and a coalition of ℓ servers can recover all the secrets and the bound holds. □

The last theorem is a natural extension of [3, Theorem 3.5], where the case $r = k - 1$, $\ell = 1$ is considered.

A consequence of this non-existence result for one-round protocols is the following lower bound for the existence of a DOT with parameters r, k, and ℓ.

Corollary 3.5 [18]. *A necessary and sufficient condition for the existence of an* $(r, n) - DOT - \binom{\zeta}{1}$ *scheme with parameters* k *and* ℓ *is*

$$r > k + \ell.$$

Proof. The necessary condition follows directly from Theorem 3.4. In the next section the protocol implementing $(r, n) - DOT - \binom{\zeta}{1}$ scheme with parameters k, ℓ and satisfying $r = k + \ell + 1$ will be presented in Fig. 2, which proves the sufficient condition. □

Note that two-round protocols, as for example the one proposed in [3], satisfy the same bound, because contacting η servers twice can be viewed as contacting 2η servers once. Hence $r = 2\eta$, $\ell = \eta$ and $k = \eta - 1$ are the appropriate parameters for the existence of the DOT.

3.3. A Protocol. Two protocols for $(r, n) - DOT - \binom{2}{1}$ have been proposed by Naor and Pinkas in [16]. Recently Blundo *et al.* in [3] generalized the idea of Naor and Pinkas and proposed several protocols for $(r, n) - DOT - \binom{\zeta}{1}$. The protocols proposed by Naor and Pinkas and two of the protocols in [3] are based on polynomial interpolation. Combinatorial constructions are presented in [3] as well.

First we present the protocol proposed by Blundo *et al.* in [3]. Then we propose a protocol, based also on a polynomial interpolation, which is a generalization of the protocols of Naor and Pinkas and Blundo *et al.* The protocols are described in Fig. 1 and Fig. 2.

As we have noted before the protocol proposed by Blundo *et al.* in [3] is $(r, n) - DOT - \binom{\zeta}{1}$ with parameters $k = r - 1$ and $\ell = 1$.

Theorem 3.6 [18]. *The protocol described in Fig.* 2 *implements an* $(r, n) - DOT - \binom{\zeta}{1}$ *scheme with parameters* k, ℓ.

Proof. The *correctness* of the proposed protocol: The degree of polynomial $V(x)$ is $r - 1$, hence after receiving r values in Step 3 the Receiver is able to recover $V(x)$ correctly and to calculate $V(0)$. On the other hand, assuming

that $(D_1(0), \ldots, D_{\zeta-1}(0)) = (0, \ldots, 0, 1, 0, \ldots, 0)$ (i. e., at most a 1 in position σ), then

$$V(0) = Q(0, D_1(0), \ldots, D_{\zeta-1}(0)) = Q(0, 0, \ldots, 0, 1, 0, \ldots, 0) = s_\sigma.$$

Now we will see that the proposed protocol for $(r, n) - DOT - \binom{\zeta}{1}$ satisfies the four properties of Definition 3.1. The *Reconstruction* follows from the *Correctness*.

The *Receiver's Privacy* is guaranteed against coalitions of at most k servers, because $\mathcal{R}$ herself chooses polynomials $D_1(x), \ldots, D_{\zeta-1}(x)$ to have degree k. Again using the proof for correctness of the proposed protocol it follows that the *Sender's Privacy* is guaranteed. And finally, the *Receiver-servers collusion* is guaranteed assuming that the Receiver has already calculated one secret and that a coalition of at most ℓ corrupt servers helps her to discover others. Because the Sender $\mathcal{D}$ chooses the polynomials $B_1(x), \ldots, B_{\zeta-1}(x)$ of degree ℓ and a polynomial $B_0(x)$ of degree $r - 1 \geq \ell + k$, the information that these ℓ corrupt servers possess (i. e. $B_0(\alpha_{i_j}), B_1(\alpha_{i_j}), \ldots, B_{\zeta-1}(\alpha_{i_j})$ for $j = 1, \ldots, \ell$) is insufficient to recover any of the polynomials $B_0(x), B_1(x), \ldots, B_{\zeta-1}(x)$, hence it is insufficient to find any of the values $B_0(0), B_1(0), \ldots, B_{\zeta-1}(0)$. □

Remark 3.7 [18]**.** The proposed protocol in Fig. 2 satisfies $r = \ell + k + 1$, which proves the "sufficient" part in the proof of Corollary 3.5.

Comparing our scheme with the polynomial scheme of Blundo *et al.* we note that both are equal w.r.t. the following parameters: the memory storage of the servers, the complexity of each interaction, the required randomness to set up the scheme and the randomness of the whole communication. The scheme proposed here achieves the bounds of [3, Theorems 3.1, 3.2, 3.3, 3.4]. The memory storage requirement for the Sender and the Receiver is higher in our scheme, due to its better security.

One of the questions that Naor and Pinkas have posed is how the scheme will ensure that a Receiver does not obtain more than r shares. It is clear that in our scheme the Sender can choose $n = r$, and solve this problem thus providing the desired security.

4. General Access Structure DOT. Recall that threshold-based schemes make sense only in an environment where one assumes that any subset of players of a certain cardinality is equally likely (or unlikely) to cheat (or to be

$(r,n) - DOT - \binom{\zeta}{1}$ **[3]**

Initialization Phase:

Let $s_0, s_1, \ldots, s_{\zeta-1} \in \mathbb{F}$ be the secrets of Sender $\mathcal{D}$. Each server P_i is associated publicly with a non-zero element $\alpha_i \in \mathbb{F}$, so $|\mathcal{P}| < |\mathbb{F}|$.

1. Sender $\mathcal{D}$ generates a random polynomial $A(x)$ of degree $(r-1)$ with values in $\mathbb{F}$ such that $A(0) = s_0$.
2. Then, $\mathcal{D}$ constructs an ζ-variate polynomial $Q(x, y_1, \ldots, y_{\zeta-1})$ with values in $\mathbb{F}$ such that $Q(0,0,\ldots,0) = s_0$, $Q(0,1,0,\ldots,0) = s_1, \ldots, Q(0,0,\ldots,1) = s_{\zeta-1}$. More precisely,
$$Q(x, y_1, \ldots, y_{\zeta-1}) = A(x) + \sum_{j=1}^{\zeta-1} b_j y_j,$$
where $b_i = s_i - s_0$.
3. Finally, for $i = 1, \ldots, n$, he sends the $(\zeta-1)$-variate polynomial $Q(\alpha_i, y_1, \ldots, y_{\zeta-1})$ to server P_i.

Oblivious Transfer Phase:

Let $\sigma \in \{0, 1, \ldots, \zeta-1\}$ be the index of Receiver $\mathcal{R}$.

1. Receiver $\mathcal{R}$ generates $(\zeta-1)$ random polynomials $D_1(x), \ldots, D_{\zeta-1}(x)$ of degree $(r-1)$ such that $(D_1(0), \ldots, D_{\zeta-1}(0))$ is an $(\zeta-1)$-tuple of only zeroes except possibly for a 1 at position σ, the position corresponding to the secret in which she is interested. Define a univariate polynomial $V(x)$ by $V(x) = Q(x, D_1(x), \ldots, D_{\zeta-1}(x))$. The degree of $V(x)$ is $(r-1)$.
2. Then, R asks r servers P_{i_j} for $j = 1, \ldots, r$, by sending a query of the form $(D_1(\alpha_{i_j}), \ldots, D_{\zeta-1}(\alpha_{i_j}))$.
3. Server P_{i_j} calculates the value $Q(\alpha_{i_j}, D_1(\alpha_{i_j}), \ldots, D_{\zeta-1}(\alpha_{i_j})) = V(\alpha_{i_j})$ and sends it back to Receiver $\mathcal{R}$.
4. After receiving r values of $V(x)$, say $V(\alpha_{i_1}), \ldots, V(\alpha_{i_r})$, $\mathcal{R}$ interpolates $V(x)$ and computes $V(0)$.

Fig. 1. $(r,n) - DOT - \binom{\zeta}{1}$ [3]

$(r,n) - DOT - \binom{\zeta}{1}$ **[18]**

Initialization Phase:

Let $s_0, s_1, \ldots, s_{\zeta-1} \in \mathbb{F}$ be the secret of Sender $\mathcal{D}$. Each server P_i is associated publicly with a non-zero element $\alpha_i \in \mathbb{F}$, so $|\mathcal{P}| < |\mathbb{F}|$.

1. Sender $\mathcal{D}$ generates $(\zeta - 1)$ random polynomials $B_1(x), \ldots, B_{\zeta-1}(x)$ of degree ℓ and one random polynomial $B_0(x)$ of degree $r - 1 \geq \ell + k$ with values in $\mathbb{F}$ such that $B_0(0) = s_0$ and, for $i = 1, \ldots, \zeta - 1$, $s_i = B_0(0) + B_i(0)$.
2. Then, $\mathcal{D}$ constructs an ζ-variate polynomial $Q(x, y_1, \ldots, y_{\zeta-1})$ with values in $\mathbb{F}$ such that $Q(0, 0, \ldots, 0) = s_0$, $Q(0, 1, 0, \ldots, 0) = s_1, \ldots, Q(0, 0, \ldots, 1) = s_{\zeta-1}$. More precisely,
$$Q(x, y_1, \ldots, y_{\zeta-1}) = B_0(x) + \sum_{j=1}^{\zeta-1} B_j(x) y_j.$$
3. Finally, for $i = 1, \ldots, n$, he sends the $(\zeta - 1)$-variate polynomial $Q(\alpha_i, y_1, \ldots, y_{\zeta-1})$ to server P_i.

Oblivious Transfer Phase:

Let $\sigma \in \{0, 1, \ldots, \zeta - 1\}$ be the index of Receiver $\mathcal{R}$.

1. Receiver $\mathcal{R}$ generates $(\zeta - 1)$ random polynomials $D_1(x), \ldots, D_{\zeta-1}(x)$ of degree k such that $(D_1(0), \ldots, D_{\zeta-1}(0))$ is an $(\zeta - 1)$-tuple of only zeroes except possibly for a 1 at position σ, the position corresponding to the secret in which she is interested. Define a univariate polynomial $V(x)$ by $V(x) = Q(x, D_1(x), \ldots, D_{\zeta-1}(x))$. The degree of $V(x)$ is $(r - 1)$.
2. Then, R asks r servers P_{i_j} for $j = 1, \ldots, r$, by sending a query of the form $(D_1(\alpha_{i_j}), \ldots, D_{\zeta-1}(\alpha_{i_j}))$.
3. Server P_{i_j} calculates the value $Q(\alpha_{i_j}, D_1(\alpha_{i_j}), \ldots, D_{\zeta-1}(\alpha_{i_j})) = V(\alpha_{i_j})$ and sends it back to Receiver $\mathcal{R}$.
4. After receiving r values of $V(x)$, say $V(\alpha_{i_1}), \ldots, V(\alpha_{i_r})$, $\mathcal{R}$ interpolates $V(x)$ and computes $V(0)$.

Fig. 2. $(r,n) - DOT - \binom{\zeta}{1}$ [18]

corrupt). The well known drawback of using a general access structure approach rather than the threshold one is that the memory storage and the complexity of each interaction could not be optimal. In this section we will apply a general access structure method for building a $DOT - \binom{\zeta}{1}$.

4.1. The Model. A General Access Structure $DOT - \binom{\zeta}{1}$ protocol involves the same three types of parties as in the threshold case: Sender, Receiver and servers.

The protocol is now composed in nearly the same way with a few changes in the *Oblivious Transfer Phase*: The Receiver $\mathcal{R}$ holds a program $Prog_R$ which enables her to interact with a subset of qualified servers in Γ at her choice. At the same time, any subset in Δ_k of servers corrupted by the Sender, does not gain any information about the secret she has recovered. Also the coalition between $\mathcal{R}$ and a subset in Δ_ℓ of servers corrupted by the Receiver cannot learn about the secrets more than can be learned by the Receiver herself.

A formal definition follows:

Definition 4.1 [18]**.** *A General Access Structure* $DOT - \binom{n}{1}$ *must guarantee the following properties:*

- Reconstruction. *If the Receiver gets information from a set of qualified servers* $\mathcal{G} \in \Gamma$*, she can compute the secret* s_σ*.*
- Sender's Privacy. *Given the information presented by a qualified group of servers* $\mathcal{G} \in \Gamma$*, the Receiver gains information about a single secret, but no information about the other secrets.*
- Receiver's Privacy. *No coalition of servers corrupted by the Sender* $B \in \Delta_k$ *gains information about which secret the Receiver has recovered.*
- Receiver-servers collusion. *A coalition of the Receiver with a set* $A \in \Delta_\ell$ *of servers corrupted by her cannot learn information about the* ζ *secrets more than can be learned by the Receiver herself.*

The set of n servers is divided in three sets of subsets: Γ—the set of qualified serves, Δ_k—the set of servers corrupted by the Sender and Δ_ℓ—the set of severs corrupted by the Receiver. The set Γ is monotone increasing and the sets Δ_k and Δ_ℓ are monotone decreasing.

4.2. Conditions for Existence. Now, using some Information Theory tools we can prove (in the same way as in the threshold case (see Theorem 3.4)) a

necessary condition for the existence of the one-round General Access Structure DOT protocol.

Theorem 4.2 [18]**.** *Let consider a General Access Structure DOT* $-\binom{\zeta}{1}$ *scheme with set of qualified servers* Γ*, set* Δ_k *of servers corrupted by the Sender and set* Δ_ℓ *of servers corrupted by the Receiver. If* $\Gamma \cap (\Delta_k \uplus \Delta_\ell) \neq \emptyset$*, then once the Receiver has legally recovered a secret, a coalition of corrupt servers from* Δ_ℓ *and the Receiver can recover all the other secrets.*

A consequence of this existence condition for the one-round protocols is the following Corollary.

Corollary 4.3 [18]**.** *A necessary condition for the existence of a General Access Structure DOT* $-\binom{\zeta}{1}$ *scheme with sets* $\Gamma, \Delta_k, \Delta_\ell$ *of qualified servers, servers corrupted by the Sender and servers corrupted by the Receiver is the tuple* $(\Gamma, \Delta_k \uplus \Delta_\ell)$ *to be an access structure.*

4.3. A Protocol. Denote by $\Gamma_k = \Delta_k^c$ and $\Gamma_\ell = \Delta_\ell^c$. We are now ready to describe the protocol for General Access Structure $DOT - \binom{\zeta}{1}$ scheme with set Γ of qualified servers, set Δ_k of servers corrupted by the Sender and set Δ_ℓ of servers corrupted by the Receiver, and the corresponding three access structures $\Gamma, \Gamma_k, \Gamma_\ell$.

Let Γ_k, Γ_ℓ be the access structures computed by MSPs $\mathcal{M}_k$ and $\mathcal{M}_\ell$, and let $\mathcal{M}$ be *(strongly) multiplicative product* MSP of $\mathcal{M}_k$ and $\mathcal{M}_\ell$ (see Definition 2.7). Hence $\mathcal{M} = \mathcal{M}_k \diamond \mathcal{M}_\ell$ and assume that MSP $\mathcal{M}$ computes Γ. Thus, a necessary condition for the existence of General Access Structure $DOT - \binom{\zeta}{1}$ scheme, which turns out to be also a sufficient condition, is the following.

Theorem 4.4 [18]**.** *Let consider a General Access Structure DOT* $-\binom{\zeta}{1}$ *scheme with set* Γ *of qualified servers, set* Δ_k *of servers corrupted by the Sender, and set* Δ_ℓ *of servers corrupted by the Receiver, and the corresponding three access structures* $\Gamma, \Gamma_k, \Gamma_\ell$*. A necessary and sufficient condition for the existence of the scheme is the MSP* $\mathcal{M}$ *to be a (strongly) multiplicative product MSP of* $\mathcal{M}_k$ *and* $\mathcal{M}_\ell$*.*

P r o o f. The necessary condition follows directly from Corollary 4.3. A protocol implementing $DOT - \binom{\zeta}{1}$ scheme will be presented in Fig. 3 together with Lemma 4.6 they prove the sufficient conditions. □

Remark 4.5. It is easy to observe now that DOT can be considered as a MPC protocol. The similarity is not only in the usage of (strongly) multiplicative

General Access Structure $DOT - \binom{\zeta}{1}$ **[18]**

Initialization Phase:

Let $s_0, s_1, \ldots, s_{\zeta-1} \in \mathbb{F}$ be the secrets of the Sender $\mathcal{D}$.

1. Sender $\mathcal{D}$ generates ζ random vectors $\mathbf{b^0} \in \mathbb{F}^d$ and $\mathbf{b^1}, \ldots, \mathbf{b^{\zeta-1}} \in \mathbb{F}^{d_2}$, such that $\langle \mathbf{b^0}, \boldsymbol{\varepsilon} \rangle = s_0$ and, for $i = 1, \ldots, \zeta - 1$; $s_i - s_0 = \langle \mathbf{b^i}, \boldsymbol{\varepsilon^2} \rangle$.
2. Then, for $i = 1, \ldots, n$, he sends the ζ packets of shares (vectors) $\mathbf{v^{j,i}} \in \mathbb{F}^{|\varphi_2(P_i)|}$, for $j = 1, \ldots, \zeta - 1$ and $\mathbf{v^{0,i}} \in \mathbb{F}^{|\varphi(P_i)|}$ to server P_i. Here $\mathbf{v^{0,i}} = M_i \mathbf{b^0}$ and $\mathbf{v^{j,i}} = M_i^{(2)} \mathbf{b^j}$ for $j = 1, \ldots, \zeta - 1$.

Oblivious Transfer Phase:

Let $\sigma \in \{0, 1, \ldots, \zeta - 1\}$ be the index of Receiver $\mathcal{R}$.

1. Receiver $\mathcal{R}$ generates $\zeta - 1$ random vectors $\mathbf{c^1}, \ldots, \mathbf{c^{\zeta-1}} \in \mathbb{F}^{d_1}$ such that $(\langle \mathbf{c^1}, \boldsymbol{\varepsilon^1} \rangle, \ldots, \langle \mathbf{c^{\zeta-1}}, \boldsymbol{\varepsilon^1} \rangle)$ is an $(\zeta - 1)$-tuple of zeroes with at most a 1 in position σ, the position corresponding to the secret in which she is interested.
2. Then Receiver $\mathcal{R}$ asks a set of qualified servers P_i, by sending a query of $(\zeta - 1)$ packets of temporary shares (vectors) $\mathbf{u^{j,i}} \in \mathbb{F}^{|\varphi_1(P_i)|}$, for $j = 1, \ldots, \zeta - 1$, where $\mathbf{u^{j,i}} = M_i^{(1)} \mathbf{c^j}$.
3. Server P_i calculates the values (vector)

$$\mathbf{w^i} = \mathbf{v^{0,i}} + \sum_{j=1}^{\zeta-1} \mathbf{u^{j,i}} \diamond \mathbf{v^{j,i}}$$

   and sends it back to $\mathcal{R}$.
4. After having received the values $\mathbf{w^i}$ for a set of qualified servers (i. e. $P_i \in \mathcal{G}$ and $\mathcal{G} \in \Gamma$) the Receiver is able to recover the secret s_σ. First she computes the recombination vector $\boldsymbol{\lambda}$, such that $M_{\mathcal{G}}^T \boldsymbol{\lambda} = \boldsymbol{\varepsilon}$ and then she computes $s_\sigma = \langle \mathbf{w}^{\mathcal{G}}, \boldsymbol{\lambda} \rangle$.

Fig. 3. General Access Structure $DOT - \binom{\zeta}{1}$ [18]

product MSPs, but also that Corollary 4.3 corresponds to Proposition 2.8, since they both give necessary conditions for existence of DOT and MPC protocols. In the same way the existence of (strongly) multiplicative product MSP is a sufficient condition for both DOT and MPC. We will elaborate further on the connection between DOT and MPC.

So, there are three access structures $\Gamma, \Gamma_k, \Gamma_\ell$ and corresponding to them three MSPs $\mathcal{M}_k = (\mathbb{F}, M^{(1)}, \boldsymbol{\varepsilon}^{\mathbf{1}}, \psi_1)$, $\mathcal{M}_\ell = (\mathbb{F}, M^{(2)}, \boldsymbol{\varepsilon}^{\mathbf{2}}, \psi_2)$ and $\mathcal{M} = (\mathbb{F}, M, \boldsymbol{\varepsilon}, \psi)$ as well as the "reverse" functions φ_1, φ_2 and φ. Let $M^{(1)}$ be an $m_1 \times d_1$ matrix, $M^{(2)}$ be an $m_2 \times d_2$ matrix and M be an $m \times d$ matrix. Now we are ready to present the following protocol for a General Access Structure $DOT - \binom{\zeta}{1}$ scheme, assuming that MSP $\mathcal{M}$ is a (strongly) multiplicative product MSP of $\mathcal{M}_k$ and $\mathcal{M}_\ell$.

Lemma 4.6 [18]**.** *The protocol described in Fig.* 3 *implements general access structure* $DOT - \binom{\zeta}{1}$ *scheme with set* Γ *of qualified servers, set* Δ_k *of servers corrupted by the Sender and set* Δ_ℓ *of servers corrupted by the Receiver.*

P r o o f. The *Correctness* of the proposed protocol can be proved as follows. We have $\langle \mathbf{b^0}, \boldsymbol{\varepsilon} \rangle = s_0$ and, $s_j - s_0 = \langle \mathbf{b^j}, \boldsymbol{\varepsilon}^{\mathbf{2}} \rangle$ for $j = 1, \ldots, \zeta - 1$. Denote by $(d_1, \ldots, d_{\zeta-1}) = (\langle \mathbf{c^1}, \boldsymbol{\varepsilon}^{\mathbf{1}} \rangle, \ldots, \langle \mathbf{c}^{\boldsymbol{\zeta-1}}, \boldsymbol{\varepsilon}^{\mathbf{1}} \rangle)$. So, $\mathbf{u^{j,i}} \in \mathbb{F}^{|\varphi_1(P_i)|}$ is the share of P_i corresponding to d_j and $\mathbf{v^{j,i}} \in \mathbb{F}^{|\varphi_2(P_i)|}$ is the share of P_i corresponding to $s_j - s_0$. Thus $\mathbf{u^{j,i}} \diamond \mathbf{v^{j,i}} \in \mathbb{F}^{|\varphi(P_i)|}$ is the share of P_i corresponding to the share of a (strongly) multiplicative resulting MSP $\mathcal{M}$ computing Γ, i. e. the share for the secret $d_j(s_j - s_0)$. Hence share $\mathbf{w^i}$ corresponds to the share of the same (strongly) multiplicative MSP $\mathcal{M}$ with a shared secret $s_0 + \sum_{j=1}^{\zeta-1} d_j(s_j - s_0)$. Since $(d_1, \ldots, d_{\zeta-1}) = (0, \ldots, 0, 1, 0, \ldots, 0)$ is a $(\zeta - 1)$-tuple of zeroes with at most a 1 in the position σ (the position corresponding to the secret in which $\mathcal{R}$ is interested), we have $s_\sigma = s_0 + \sum_{j=1}^{\zeta-1} d_j(s_j - s_0)$. Having received all this information the Receiver recovers (using $\langle \mathbf{w}^{\mathcal{G}}, \boldsymbol{\lambda} \rangle$) the secret, which is exactly s_σ.

Now we will see that the proposed General Access Structure protocol for $DOT - \binom{\zeta}{1}$ satisfies the four properties from Definition 4.1.

The *Reconstruction* follows from the *Correctness*. The *Receiver's Privacy* is guaranteed against coalitions Δ_k of forbidden servers, because $\mathcal{R}$ herself chooses vectors $\mathbf{c^1}, \ldots, \mathbf{c}^{\boldsymbol{\zeta-1}}$ with values $d_1, \ldots, d_{\zeta-1}$ and from the standard arguments for privacy in LSSS. Again using the proof for correctness of the proposed protocol it follows that the *Sender's Privacy* is guaranteed. And finally the *Receiver-servers collusion* is guaranteed assuming that the Receiver has already calculated one secret and that a coalition of Δ_ℓ corrupt servers helps her to discover the others.

Because the Sender $\mathcal{D}$ chooses the vectors $\mathbf{b^0}, \mathbf{b^1}, \ldots, \mathbf{b^{\zeta-1}}$ the information these Δ_ℓ corrupt servers posses (i. e. their collected shares) is insufficient to recover any of the secrets $s_0, s_1 - s_0, \ldots, s_{\zeta-1} - s_0$. $\square$

Remark 4.7. Observe that a $DOT - \binom{\zeta}{1}$ scheme can be considered as a MPC protocol computing the function

$$f(\mathbf{s}, \boldsymbol{\eta}) = s_0.(1 - \sum_{i=1}^{\zeta-1} \eta_i) + \sum_{i=1}^{\zeta-1} s_i \eta_i.$$

The inputs are the secrets of the Sender $s_0, s_1, \ldots, s_{\zeta-1}$ and a tuple $(\eta_1, \ldots, \eta_{\zeta-1})$, of only zeroes except possibly for a 1, of the Receiver.

Comparing to the general MPC protocols a DOT can be implemented efficiently because of the linearity of the computed function, which allow to reduce the interaction between the servers. Indeed when Receiver $\mathcal{R}$ sends a query to a server they perform together a MPC computing a share of $f(\mathbf{s}, \boldsymbol{\eta})$. Therefore the interaction with a qualified group of servers allow the Receiver to compute the function and to receive the secret which she is interested.

Another important difference is that all participants in a DOT protocol are assumed to be possibly curious but never corrupt. Thus DOT settings correspond to passive multi-party protocol settings, hence Theorem 2.10 corresponds to Theorem 4.4.

REFERENCES

[1] Brassard G., C. Crepeau, J.-M. Roberts. All-or-Nothing Disclosure of Secrets. *CRYPTO'86*, LNCS 263, Springer-Verlag, 1987, 234–238.

[2] Brassard G., C. Crepeau, M. Santha. Oblivious Transfer and Intersecting Codes. *IEEE Trans. on Inf. Th.*, **42**, No 6 (1996) 1769–1780.

[3] *Blundo C., P. D'Arco, A. De Santis, D. Stinson.* New Results on Unconditionally Secure Distributed Oblivious Transfer. *SAC'02*, LNCS 2595, Springer-Verlag, 2002, 291–309.

[4] Beaver D., J. Feigenbaum, J .Kilian, P. Rogaway. Locally Random Reductions: Improvements and Applications. *Journal of Cryptology* **10**, No 1 (1997), 17–36.

[5] Beimel A., Y. Ishai, T. Malkin. Reducing the Servers Computation in Private Information Retrieval: PIR with Preprocessing, *CRYPTO'00*, LNCS 1880, Springer-Verlag, 2000, 55–73.

[6] Bellare M., S. Micali. Non-interactive Oblivious Transfer and Applications. *CRYPTO'89*, LNCS 435, Springer-Verlag, 1990, 547–559.

[7] Chor B., O. Goldreich, E. Kushilevitz, M. Sudan. Private Information Retrieval. *FOCS'95*, 41–50.

[8] Cramer R., I. Damgard, U. Maurer. General Secure Multi-Party Computation from any Linear Secret Sharing Scheme. *EUROCRYPT'00*, LNCS 1807, Springer-Verlag, 2000, 316–334.

[9] Di Crescenzo G., Y. Ishai, R. Ostrtovski. Universal Service-Providers for Database Private Information Retrieval. *PODC'98*.

[10] Crepeau C., G. Savvides, C. Schaffner, J. Wullschleger. Information-Theoretic Conditions for Two-Party Secure Function Evaluation. *EUROCRYPT'06*, LNCS 4004, Springer-Verlag, 2006, 538–554.

[11] Dodis Y., S. Micali. Lower bounds for Oblivious Transfer Reduction., *EUROCRYPT'1999*, LNCS 1592, Springer-Verlag, 1999, 42–54.

[12] D'Arco P., D. Stinson. Generalized Zig-zag Functions and Oblivious Transfer Reductions. *SAC'01*, LNCS 2259, Springer-Verlag, 2001, 87–103.

[13] Even S., O. Goldreich, A. Lempel. A Randomized Protocol for Signing Contracts. *Communications of the ACM* **28** (1985), 637–647.

[14] Gertner Y., S. Goldwasser, T. Malkin. A Random Server Model for Private Information Retrieval or How to Achieve Information Theoretic PIR Avoiding Database Replication. *RANDOM'98*, LNCS 1518, Springer-Verlag. 1998, 200–217.

[15] Gertner Y., Y. Ishai, E. Kushilevitz, T. Malkin. Protecting Data Privacy in Private Information Retrieval Schemes. *STOC'98*, 151–160.

[16] Naor M., B. Pinkas. Distributed Oblivious Transfer. *ASIACRYPT'00*, LNCS 1976, Springer-Verlag, 2000, 205–219.

[17] Naor M., B. Pinkas, R. Sumner. Privacy Preserving Auctions and Mechanism Design. *ACM Conference on Electronic Commerce*, 1999.

[18] Nikov V., S. Nikova, B. Preneel, J. Vandewalle. On Unconditionally Secure Distributed Oblivious Transfer. *INDOCRYPT'02*, LNCS 2551, Springer-Verlag. 2002, 395–409.

[19] Nikov V., S. Nikova, B. Preneel. On Multiplicative Linear Secret Sharing Schemes. *INDOCRYPT'03*, LNCS 2904, Springer-Verlag. 2003, 135–147.

[20] Rabin M. How to Exchange Secrets by Oblivious Transfer. *Technical Memo TR-81*, Aiken Computation Laboratory, Harvard University, 1981.

[21] Rivest R. Unconditionally Secure Commitment and Oblivious Transfer Schemes Using Private Channels and a Trusted Initializer. *Manuscript*.

[22] Tzeng W. Efficient 1-out-of-n Oblivious Transfer Schemes. *PKC'02*, LNCS 2274, Springer-Verlag, 2002, 159–171.

5. Appendix. Here we provide some Information Theoretic definitions and results. Let X and Y be (possibly dependent) random variables with probability measures P_X and P_Y respectively. We define $P \log_2 P$ evaluated in $P = 0$ to be equal to 0. The *entropy* $H(X)$ is defined by

$$H(X) = -\sum_x P_X(x) \log_2 P_X(x).$$

Equivalently $H(X) = -E(\log_2 P_X(x))$, where E denotes the *expectation*. $H(X)$ is also called the *uncertainty* about X. Its interpretation is as follows. $H(X)$ measures the average amount of bits needed to describe a realization of random variable X, or in other words $H(X)$ measures the amount of information contained in X. So, $H(X) = 0$ if and only if one is certain about X. The *conditional entropy* $H(X|Y = y)$ is defined by

$$H(X|Y = y) = -\sum_x P_{X|Y}(x|y) \log_2 P_{X|Y}(x|y).$$

Obviously $H(X|Y = y) = -E(\log_2 P_{X|Y}(X|y))$. The *conditional entropy* $H(X|Y)$ is now defined by

$$H(X|Y) = -\sum_y P_Y(y)H(X|Y = y).$$

Hence $H(X|Y) = -E(\log_2 P_{X|Y}(X|Y))$ and it measures the average amount of bits needed to describe X given the knowledge of Y. Equations and inequalities with natural interpretations can be derived now. For example

$$H(XY) = H(X) + H(Y|X), \tag{11}$$

i. e., the information needed to describe both X and Y equals the information needed to describe X together with the information needed to describe Y given the knowledge (information) of X. Another example is the following inequality:

$$0 \leq H(X|Y) \leq H(X) \leq \log_2 |\{x : P_X(x) > 0\}|.$$

The difference between $H(X)$ and $H(X|Y)$ gives another useful notion. The *mutual information* between X and Y is defined as

$$I(X;Y) = H(X) - H(X|Y).$$

The mutual information between X and Y measures the amount of information X and Y have in common (it is easy using the equation (11) to see that $I(X;Y) = I(Y;X)$). Given $n+1$ random variables $X_1, \ldots, X_n, Y$, the entropy of $X_1 \ldots X_n$ given Y can be written as

$$H(X_1 \ldots X_n|Y) = H(X_1|Y) + H(X_2|X_1Y) + \cdots + H(X_n|X_1 \ldots X_{n-1}Y).$$

Therefore, for any sequence of n random variables $X_1, \ldots, X_n$ it holds that

$$H(X_1 \ldots X_n) = \sum_{i=1}^{n} H(X_i|X_1 \ldots X_{i-1}) \leq \sum_{i=1}^{n} H(X_i).$$

Moreover, the above relation implies that, for any $j \leq n$,

$$H(X_1 \ldots X_n) \geq H(X_1 \ldots X_j).$$

Let Z be another random variable. The *conditional mutual information* between X and Y given Z is defined as

$$I(X;Y|Z) = H(X|Z) - H(X|YZ) = H(Y|Z) - H(Y|ZX) = I(Y;X|Z).$$

Since the conditional mutual information $I(X;Y|Z)$ is non-negative we get

$$H(X|Z) \geq H(X|YZ).$$

Ventzislav Nikov
Innovation and Development Center Leuven
NXP Semiconductors, Belgium
e-mail: venci.nikov@gmail.com

Svetla Nikova, Bart Preneel
Department Electrical Engineering
ESAT/COSIC
Katholieke Universiteit Leuven
Kasteelpark Arenberg 10
B-3001 Heverlee-Leuven, Belgium
e-mail: svetla.nikova
e-mail: bart.preneel@esat.kuleuven.be

Received March 30, 2007
Final Accepted September 13, 2007