

On Metering Schemes

Ventzislav Nikov

Department of Mathematics and Computing Science,
Eindhoven University of Technology
P.O. Box 513, 5600 MB, Eindhoven, the Netherlands
`v.nikov@tue.nl`

Svetla Nikova, Bart Preneel

Department Electrical Engineering, ESAT/COSIC,
Katholieke Universiteit Leuven, Kasteelpark Arenberg 10,
B-3001 Heverlee-Leuven, Belgium
`svetla.nikova, bart.preneel@esat.kuleuven.ac.be`

November 17, 2004

Abstract

In order to decide on advertisement fees for web servers, Naor and Pinkas introduced (threshold) metering schemes secure against coalitions of corrupt servers and clients. Several researchers have generalized the idea of Naor and Pinkas: first metering scheme with pricing and dynamic multi-threshold metering schemes have been proposed; later the solution has been extended to allow general access structures and an approach on linear algebra has been introduced. But all previous papers consider the following scenario - a general (or threshold) access structure for the clients and a threshold access structure for the servers. We generalize the model allowing also a *general access structure for the servers*, moreover we *strengthen* it limiting the possibility of corrupt servers to help each other. A more efficient protocol satisfying the requirements of the new model is described.

Naor and Pinkas show that one should be able also to detect illegal behavior of clients, i.e., one needs to verify the shares received from clients. Most metering schemes do not offer this feature. But Ogata and Kurosawa pointed out a minor flaw in the extension protocol by Naor and Pinkas providing detection of such illegal behavior and propose a correction. So, we extend the used linear algebra approach in order to build *robust* unconditionally secure general metering schemes. As a tool to achieve this goal we introduce *doubly-labelled* matrices and an operation on such matrices. Certain properties of this operation are proven.

1 Introduction

A metering scheme is a protocol to measure the interaction between clients and servers in a network. The time is divided into *time frames* and the audit agency is interested in counting the number of visits received by each server in any time frame. Metering schemes are useful in many applications, for instance to decide on the amount of money to be paid to web servers hosting adds, or for network accounting and electronic coupon management [13]. Franklin and Malkhi [6] were the first to consider a rigorous approach to the metering problem. Their solutions offer “lightweight security”, meaning that they are suitable if there are no

strong commercial interests to falsify the metering result. Naor and Pinkas [13], subsequently introduced metering schemes secure against fraud attempts by servers and clients. In their scheme any server which has been visited by any set of $k + 1$ or more clients in a time frame, where k is a fixed threshold, is able to compute a proof, whereas any server receiving visits from less than $k + 1$ clients has no information about the proof. By proof we mean a value computed by the server that substantiates the visits of a qualified set of clients. In this threshold case scenario for both clients and servers, the threshold refers to the maximum number of colluding players (server, clients). In order to have a more flexible payment system the authors of [1, 8] have introduced metering schemes with pricing. To be able to measure the number of visits in any granularity Blundo *et al.* in [2] have introduced dynamic multi-threshold metering schemes which are metering schemes with an associated threshold for any server and for any time frame. In [7], Masucci and Stinson consider general access structures for the clients and a threshold scheme for the servers, where the access structure is the family of all subsets of clients enabling a server to compute its proof. They also prove a lower bound on the communication complexity of metering schemes realizing such access structures. A linear algebra approach (i.e., applicable for any general monotone access structure) to metering schemes is presented in [3] by Blundo *et al.* More specifically, given any access structure for the clients, the authors propose a method to construct a metering scheme by means of linear secret sharing schemes with the same access structure. Besides, they prove some properties of the relationship between metering schemes and secret sharing schemes. They also present some new bounds on the information distributed to clients and servers in a metering scheme. The main difference between the scheme in [3] and the scheme in [7] is that the second one is worst with respect to the communication complexity.

We will consider only metering schemes that provide unconditional security. Computationally secure metering scheme based on the Decisional Diffie-Hellman assumption have been presented in [13]. Threshold-based secret sharing and metering make sense only in an environment where one assumes that trust is “uniformly distributed” over the players (clients and servers): any subset of players of a certain cardinality is equally likely (or unlikely) to cheat. In many natural scenarios this assumption is not very realistic; moreover, in a more realistic model no threshold solution will work. Why do we need to introduce a general access structure on the set of servers? In the model proposed by Naor and Pinkas the audit agency deals with servers, but in fact the servers are owned by companies, where each company possesses a different number of servers. In this scenario the uniformly distributed trust on the set of servers is not very realistic either. As Naor and Pinkas [13] pointed out one should be able to detect illegal behavior of clients by verifying their shares. This issue is not considered in [1, 2, 3, 8, 7], but in [14] a minor flaw in the extension protocol [13] providing detection of such illegal behavior was pointed out and a correction was proposed.

In this paper we make the following contributions:

- We distinguish between three types of general access structures: for clients, servers and an adversary. The access structure for clients consists of qualified and forbidden sets of clients, i.e., sets which allow or disallow the server visited by them in a given time frame to compute its proof. The corrupt clients structure gives us a possible distribution for the corrupt clients. Also a general access structure is considered for the set of servers. Until now authors have only considered the threshold case for them.
- Secondly, we propose a stronger model for metering schemes by adding a new security requirement.

- We propose a metering scheme, that is simpler and more efficient w.r.t. communication complexity and memory requirements than the scheme proposed by Blundo *et al.* [3]. The difference becomes clear in the public information used by clients and servers, which in our scheme is smaller.
- As noted above the basic model assumes that the clients do not present incorrect evidence to a server, thus preventing the server from constructing its proof. We build robust metering schemes by proposing another method than Ogata and Kurosawa' [14] method for the threshold case and then extending these methods to the general access structure case.
- We demonstrate that one can protect clients of the scheme against denied of service attacks of corrupt servers.

The paper is organized as follows. In Sect. 2 we introduce Linear Secret Sharing Schemes (LSSS) and Multiplicative Linear SSSs. In Sect. 3 *doubly-labelled* matrices are defined and certain properties of these matrices are proven. Sect. 4 focuses on the model of Metering Schemes and strengthen it. We propose a fully general access structure linear Metering Scheme in Sect. 5 and compare it with the previous results. Sect. 6 first discusses the known (threshold) solutions for Robust Metering and then shifts to the general case; two solutions are proposed and proved to be secure. Conclusions are presented in Sect. 7.

2 Preliminaries

2.1 Linear Secret Sharing Schemes

Denote the *participants* of the scheme by P_i , $1 \leq i \leq n$, and the set of all *players* by $\mathcal{P} = \{P_1, \dots, P_n\}$. 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 (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. [4]). 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 .

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. Both passive and active adversaries may be *static*, meaning that the set of corrupt players is chosen once and for all before the protocol starts, or *adaptive* meaning that the adversary can at any time during the protocol choose to corrupt a new player based on all the information he has at the time, as long as the total number of corrupt players is restricted. The adversary \mathcal{A}

is characterized by a particular subset $\Delta_{\mathcal{A}}$ of Δ , which is itself 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. [4] A Monotone Span Program (MSP) \mathcal{M} is a quadruple $(\mathbb{F}, M, \varepsilon, \psi)$, where \mathbb{F} is a finite field, M is a matrix (with m rows and $d \leq m$ columns) over \mathbb{F} , $\psi : \{1, \dots, m\} \rightarrow \{1, \dots, n\}$ is a surjective function and $\varepsilon = (1, 0, \dots, 0)^T \in \mathbb{F}^d$ is called the target vector.

As ψ labels each row with an integer i from $[1, \dots, m]$ 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 $[1, \dots, n]$ to $[1, \dots, m]$ 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 .

The *kernel* of a matrix M (denoted by $\ker(M)$) is the set of vectors $\mathbf{u} \in \mathbb{F}^d$, such that $M\mathbf{u} = \mathbf{0}$. The *image* of M (denoted by $\text{im}(M)$) is the set of vectors $\mathbf{v} \in \mathbb{F}^m$ such that $\mathbf{v} = M\mathbf{u}$ for some $\mathbf{u} \in \mathbb{F}^d$. 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 $\varepsilon \in \text{im}(M_A^T)$ if and only if A is a member of Γ . 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 $M_A^T \boldsymbol{\lambda} = \varepsilon$. Notice that the vector $\varepsilon \notin \text{im}(M_B^T)$ if and only if there exists a vector $\mathbf{k} \in \mathbb{F}^d$ such that $M_B \mathbf{k} = \mathbf{0}$ and $\mathbf{k}_1 = 1$.

Consider a vector $\mathbf{v} \in \mathbb{F}^m$. The coordinates in \mathbf{v} , which belong to player P_i are collected in a sub-vector denoted by \mathbf{v}^i , or in other words $\mathbf{v} = (\mathbf{v}^1, \dots, \mathbf{v}^n)$ where $\mathbf{v}^i \in \mathbb{F}^{|\varphi(P_i)|}$. The *p-support* of vector \mathbf{v} , denoted by $\text{sup}_P(\mathbf{v})$, is defined as the set of coordinates i , $1 \leq i \leq n$ for which $\mathbf{v}^i \neq \mathbf{0}$, i.e. $\text{sup}_P(\mathbf{v}) = \{i : \mathbf{v}^i \neq \mathbf{0}\}$.

Definition 2.2. [11] An MSP is called Δ -non-redundant (denoted by Δ -rMSP) when $v \neq 0 \in \ker(M^T) \iff \text{sup}_P(v) \in \Gamma$ ($\Gamma = \Delta^c$).

For the sake of simplicity we will call an Δ -rMSP simply an rMSP.

Remark 2.3. In [15] the authors consider (and prove the existence of) another class of MSPs called “monotone dependency program” (MDP). It has been proven in [11] that MDPs are even more restricted class of MSPs than rMSP. Thus the conjecture from [9] about existence of rMSP has a positive answer.

2.2 Multiplicative Linear SSSs

Cramer *et al.* proposed in [4] 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 [10] 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 $\mathcal{M}_1 = (\mathbb{F}, M^{(1)}, \varepsilon^1, \psi_1)$ and $\mathcal{M}_2 = (\mathbb{F}, M^{(2)}, \varepsilon^2, \psi_2)$. 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 $\mathbf{x} = (\mathbf{x}^1, \dots, \mathbf{x}^n)$. 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:

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

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 notice 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

$$\begin{aligned} M^{(1)} &= \begin{pmatrix} M_1^{(1)} \\ \dots \\ M_n^{(1)} \end{pmatrix}, \quad M^{(2)} = \begin{pmatrix} M_1^{(2)} \\ \dots \\ M_n^{(2)} \end{pmatrix}, \quad \text{and} \\ M^{(1)} \diamond M^{(2)} &= \begin{pmatrix} M_1^{(1)} \otimes M_1^{(2)} \\ \dots \\ M_n^{(1)} \otimes M_n^{(2)} \end{pmatrix}. \end{aligned} \quad (2)$$

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.

Remarks on the operation: The process is analogous to the Kronecker (tensor) product, with the *difference* that the tensor operation \otimes is replaced by the diamond operation \diamond for the columns. But note that the “symmetry” which the Kronecker product have between rows and columns is destroyed. This is illustrated by the following lemma.

Lemma 2.4. [10] *Let $M^{(1)}$ be an $m_1 \times d_1$ matrix, $M^{(2)}$ be an $m_2 \times d_2$ matrix, $N^{(1)}$ be an $n_1 \times m_1$ matrix and an $N^{(2)}$ be $n_2 \times m_2$ matrix. Then*

$$(N^{(1)} M^{(1)}) \diamond (N^{(2)} M^{(2)}) = (N^{(1)} \diamond N^{(2)}) (M^{(1)} \otimes M^{(2)}).$$

The *diamond* operation \diamond for MSPs is defined as follows.

Definition 2.5. [10] *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 $\psi_1(i) = \psi_2(j) = r$.*

3 Operations on Doubly-Labelled Matrices

Now let us consider a matrix A , which rows are labelled by a function ψ and the columns are labelled by a function $\bar{\psi}$. Denote the sub-matrices labelled with $\psi(i)$ and $\bar{\psi}(j)$ by $A_{i,j}$. We call such a matrix *doubly-labelled* [12]. Note that the block-diagonal matrix D from the Definition of multiplicative MSPs in [5] is in-fact doubly-labelled matrix with $\psi = \bar{\psi}$.

Let two matrices A and B be double-labelled by the functions ψ_1 and ψ_2 for the rows and by the functions $\overline{\psi_1}$ and $\overline{\psi_2}$ for the columns. Define $A \boxtimes B$ to be a $(\sum_i |\psi_1(i)||\psi_2(i)|) \times (\sum_i |\overline{\psi_1}(i)||\overline{\psi_2}(i)|)$ matrix consisting of sub-matrices $A_{i,j} \otimes B_{i,j}$:

$$\begin{aligned} A &= \begin{pmatrix} A_{1,1} & \dots & A_{1,n} \\ \vdots & \dots & \vdots \\ A_{n,1} & \dots & A_{n,n} \end{pmatrix}, \quad B = \begin{pmatrix} B_{1,1} & \dots & B_{1,n} \\ \vdots & \dots & \vdots \\ B_{n,1} & \dots & B_{n,n} \end{pmatrix}, \quad \text{and} \\ A \boxtimes B &= \begin{pmatrix} A_{1,1} \otimes B_{1,1} & \dots & A_{1,n} \otimes B_{1,n} \\ \vdots & \dots & \vdots \\ A_{n,1} \otimes B_{n,1} & \dots & A_{n,n} \otimes B_{n,n} \end{pmatrix}. \end{aligned} \quad (3)$$

Note that any MSP can be seen as a double-labelled matrix and that the operation we just introduced is a generalization of the operation diamond. Let $\mathcal{M}_1 = (\mathbb{F}, M^{(1)}, \boldsymbol{\varepsilon}^1, \psi_1)$ and $\mathcal{M}_2 = (\mathbb{F}, M^{(2)}, \boldsymbol{\varepsilon}^2, \psi_2)$ be MSPs, then it is easy to verify that $M^{(1)}(M^{(2)})^T$ is doubly-labelled matrix with labelled functions ψ_1 for the rows and ψ_2 for the columns. Now we prove the following properties of the new operation, which establish a relation between MSPs and diamond operation on the one hand and doubly-labelled matrices and the new operation on the other hand.

Lemma 3.1. [12] *Let MSPs $\mathcal{M}_i = (\mathbb{F}, M^{(i)}, \boldsymbol{\varepsilon}^i, \psi_i)$ for $i = 1, 2, 3, 4$ be such that $M^{(i)}$ are $m_i \times d_i$ matrices and let $d_1 = d_3$, $d_2 = d_4$. Then the following equality holds*

$$(M^{(1)} \diamond M^{(2)})(M^{(3)} \diamond M^{(4)})^T = (M^{(1)}(M^{(3)})^T) \boxtimes (M^{(2)}(M^{(4)})^T).$$

Proof. Let $M^{(i)} = \begin{pmatrix} M_1^{(i)} \\ \dots \\ M_n^{(i)} \end{pmatrix}$. Then we need to show that the following holds:

$$\begin{aligned} & \begin{pmatrix} M_1^{(1)} \otimes M_1^{(2)} \\ \dots \\ M_n^{(1)} \otimes M_n^{(2)} \end{pmatrix} \begin{pmatrix} M_1^{(3)} \otimes M_1^{(4)} \\ \dots \\ M_n^{(3)} \otimes M_n^{(4)} \end{pmatrix}^T \\ &= \begin{pmatrix} M_1^{(1)}(M_1^{(3)})^T \otimes M_1^{(2)}(M_1^{(4)})^T & \dots & M_1^{(1)}(M_n^{(3)})^T \otimes M_1^{(2)}(M_n^{(4)})^T \\ \vdots & \dots & \vdots \\ M_n^{(1)}(M_1^{(3)})^T \otimes M_n^{(2)}(M_1^{(4)})^T & \dots & M_n^{(1)}(M_n^{(3)})^T \otimes M_n^{(2)}(M_n^{(4)})^T \end{pmatrix}. \end{aligned}$$

From the properties of the tensor product we have

$$(M_i^{(1)} \otimes M_i^{(2)})(M_j^{(3)} \otimes M_j^{(4)})^T = (M_i^{(1)}(M_j^{(3)})^T) \otimes (M_i^{(2)}(M_j^{(4)})^T),$$

which concludes the proof. \square

By applying Lemma 2.4 it is easy to verify that the following relation is satisfied.

Lemma 3.2. [12] *Let MSPs $\mathcal{M}_i = (\mathbb{F}, M^{(i)}, \boldsymbol{\varepsilon}^i, \psi_i)$ for $i = 1, 2, 3, 4$ be such that $M^{(i)}$ are $m_i \times d_i$ matrices. Let $R^{(1)}$ be a $d_1 \times d_3$ matrix and let $R^{(2)}$ be a $d_2 \times d_4$ matrix. Then the following equality holds*

$$(M^{(1)} \diamond M^{(2)})(R^{(1)} \otimes R^{(2)})(M^{(3)} \diamond M^{(4)})^T = (M^{(1)}R^{(1)}(M^{(3)})^T) \boxtimes (M^{(2)}R^{(2)}(M^{(4)})^T).$$

Note that $M^{(1)}R^{(1)}(M^{(3)})^T$ is also a doubly-labelled matrix.

4 Modelling Metering Schemes

4.1 A Naive Model

In this section we will follow the model of [13, 3], but replacing the threshold access structures (for the servers and/or for the clients) with general ones.

Consider the following scenario: there are n clients, \tilde{n} servers and an audit agency \mathbb{A} which is interested in counting the client visits to the servers in τ different time frames. For any $i = 1, \dots, n$ and $j = 1, \dots, \tilde{n}$, we denote the i -th client (user) by \mathcal{U}_i and the j -th server (player) by P_j .

We consider an *access structure* Γ of qualified and its complement $\Delta = \Gamma^c$ of forbidden groups for the set of clients $\mathcal{U} = \{\mathcal{U}_1, \dots, \mathcal{U}_n\}$.

In a metering scheme realizing the clients access structure Γ any server which has been visited by at least a qualified subset of clients in Γ in a fixed time frame is able to provide the audit agency with a proof for the visits it has received.

A second access structure Γ_S for the set of servers $\{P_1, \dots, P_{\tilde{n}}\}$ can be considered. We call a subset of servers *corrupt* if they are not in Γ_S , i.e. if they are in $\Delta_S = \Gamma_S^c$. We also denote, as usual, the set of possible subsets of *corrupt clients* by $\Delta_{\mathcal{A}}$; note that $\Delta_{\mathcal{A}} \subseteq \Delta$. So, in this model we consider a $(\Delta_{\mathcal{A}}, \Delta_S)$ -adversary \mathcal{A} .

A corrupt server can be assisted by corrupt clients and other corrupt servers in computing its proof without receiving visits from qualified subsets. A corrupt client can donate to a corrupt server all the private information received by the audit agency during the initialization phase. A corrupt server can donate to another corrupt server the private information received from clients in the previous time frames and in the actual time frame.

Several phases can be defined in the Metering scheme.

- a) There is an *initialization phase* in which the audit agency \mathbb{A} chooses the access structures, computes the corresponding matrices, makes them public and distributes some information to each client \mathcal{U}_i through a private channel. For any $i = 1, \dots, n$ we denote by $V_i^{(t)}$ the shares that the audit agency \mathbb{A} gives to the client \mathcal{U}_i for time frames $t = 1, \dots, \tau$.
- b) A *regular operation* consists of a client's visit to a server during a time frame. During such a visit the client gives to the visited server a piece of information which depends on the private information, on the identity of the server and on the time frame during which the client visits the server. For any $i = 1, \dots, n$; $j = 1, \dots, \tilde{n}$ and $t = 1, \dots, \tau$, we denote by $C_{i,j}^{(t)}$ the information that the client \mathcal{U}_i sends to the server P_j when visiting him in time frame t .
- c) During the *proof computation phase* any server P_j which has been visited by at least a subset of qualified clients in time frame t is able to compute its proof. For any $j = 1, \dots, \tilde{n}$ and $t = 1, \dots, \tau$ we denote by $p_j^{(t)}$ the proof computed by server P_j at time frame t when it has been visited by a qualified set of clients.
- d) During the *proof verification phase* audit agency \mathbb{A} verifies the proofs received by the servers and decides on the amount of money to be paid to the servers. If the proof received from a server at the end of a time frame is correct, then \mathbb{A} pays the server for its services.

Definition 4.1. [3] An (n, \tilde{n}, τ) metering scheme realizing the access structures Γ, Γ_S and secure against an (Δ_A, Δ_S) -adversary \mathcal{A} , is a protocol to measure the interaction between clients $\mathcal{U}_1, \dots, \mathcal{U}_n$ with an access structure Γ and servers $P_1, \dots, P_{\tilde{n}}$ with an access structure Γ_S during τ time frames in such a way that the following properties are satisfied:

- For any time frame t any client is able to compute the information needed to visit any server.
- Correctness. For any time frame t any server P_j which has been visited by a qualified subset of clients $A \in \Gamma$ in time frame t can compute its proof for time frame t .
- Privacy. Let B_2 be a coalition of corrupt servers, i.e., $B_2 \in \Delta_S$ and let B_1 be a coalition of corrupt clients, i.e., $B_1 \in \Delta_A$. Assume that in some time frame t each server in the coalition has been visited by a subset of forbidden clients B_3 , i.e., $B_3 \in \Delta$, such that we still have $B_3 \cup B_1 \in \Delta$. Then the servers in coalition B_2 have no information about their proofs for a time frame t , even if they are helped by the corrupt clients in B_1 .

4.2 A Strong Model

Actually the model presented in Section 4.1 has the following weakness: there is no requirements for the corrupt server which has been visited by a qualified set of clients. This means that the naive model does not specify what happens when a corrupt server which has been visited by a qualified set of clients helps other corrupt servers. To fix this weakness we propose a strengthened model.

Definition 4.2. [9] An (n, \tilde{n}, τ) metering scheme realizing the access structures Γ, Γ_S and secure against an (Δ_A, Δ_S) -adversary \mathcal{A} , is a protocol to measure the interaction between clients $\mathcal{U}_1, \dots, \mathcal{U}_n$ with an access structure Γ and servers $P_1, \dots, P_{\tilde{n}}$ with an access structure Γ_S during τ time frames in such a way that the following properties are satisfied:

- For any time frame t any client is able to compute the information needed to visit any server.
- Correctness. For any time frame t any server P_j which has been visited by a qualified subset of clients $A \in \Gamma$ in time frame t can compute its proof for time frame t .
- Privacy. Let B_2 be a coalition of corrupt servers, i.e., $B_2 \in \Delta_S$ and let B_1 be a coalition of corrupt clients, i.e., $B_1 \in \Delta_A$. Assume that in some time frame t each server in the coalition has been visited by a subset of forbidden clients B_3 , i.e., $B_3 \in \Delta$, such that we still have $B_3 \cup B_1 \in \Delta$. Then the servers in coalition B_2 have no information about their proofs for a time frame t , even if they are helped by the corrupt clients in B_1 .
- (Stronger) Privacy. Let B_2 be a coalition of corrupt servers, i.e., $B_2 \in \Delta_S$ and let B_1 be a coalition of corrupt clients, i.e., $B_1 \in \Delta_A$. Assume that in some time frame t a fixed server in the coalition (e.g. $P_j \in B_2$) has been visited by a subset of forbidden clients B_3 , i.e., $B_3 \in \Delta$, and $B_3 \cup B_1 \in \Delta$. Assume that in the same time frame t any other server in the coalition B_2 has been visited by a subset of qualified clients B_4 . Then the servers in the coalition $B_2 \setminus \{P_j\}$ are able to compute their proofs for a time frame t , but they are unable to “help” the server P_j with the computation of its proofs, even if they are helped by the corrupt clients in B_1 .

5 A Linear Metering Scheme

5.1 Threshold Scheme

First we will present the threshold metering scheme described in [13].

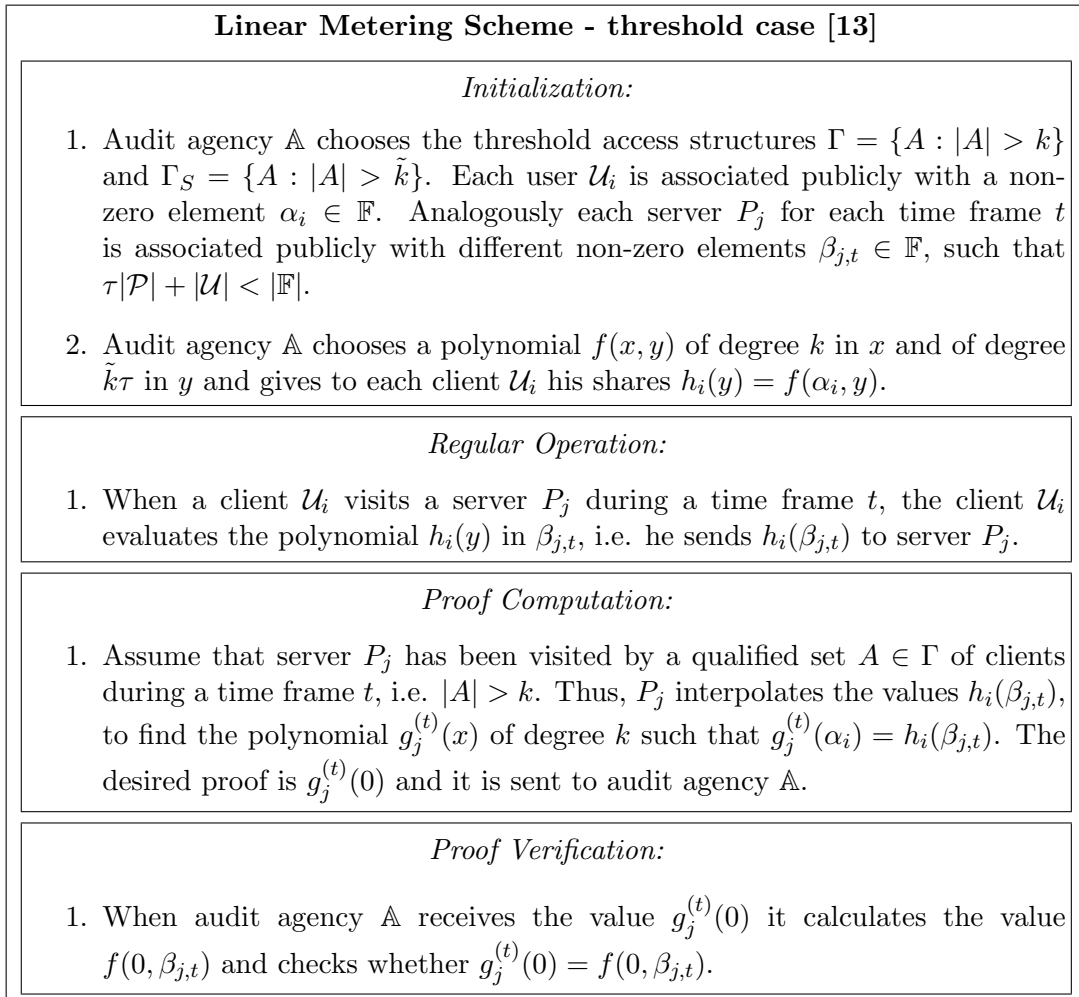


Figure 1: Linear Metering Scheme - threshold case [13]

Note that $g_j^{(t)}(\alpha_i) = h_i(\beta_{j,t}) = f(\alpha_i, \beta_{j,t})$, thus $g_j^{(t)}(x) = f(x, \beta_{j,t})$. Hence the proof is $g_j^{(t)}(0) = f(0, \beta_{j,t})$, which prove the correctness of the scheme.

5.2 General Access Structure Scheme

We use linear algebra, MSP based approach to build a general linear metering scheme. The scheme rely on LSSS as a primitive and on the linearity.

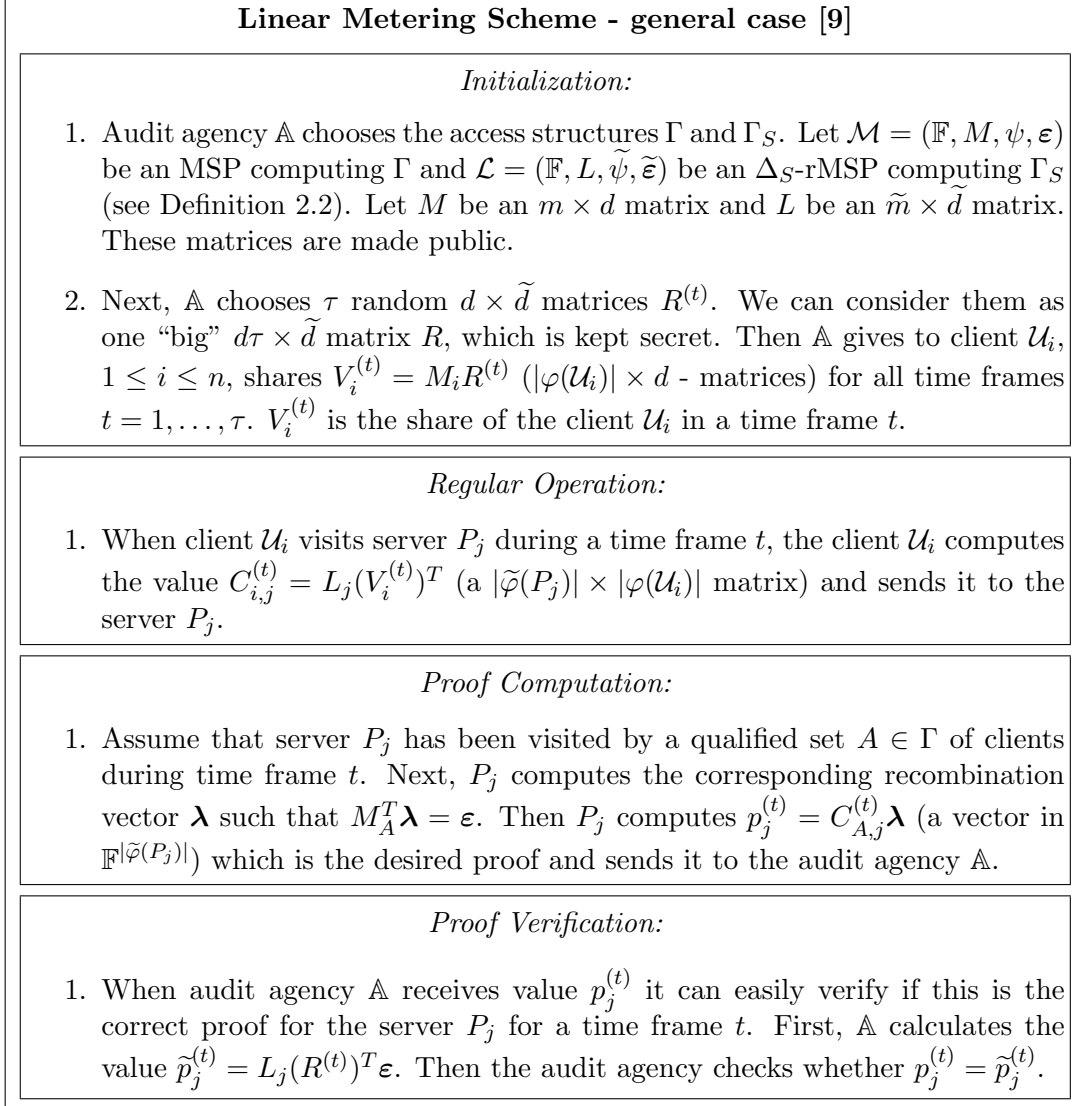


Figure 2: Linear Metering Scheme - general case [9]

Theorem 5.1. [9] *The linear Metering Scheme described in Fig. 2 is perfectly secure against a (Δ_A, Δ_S) -adversary.*

Proof. The *Correctness* follows from the following observation. Assume that server P_j has been visited by a qualified set of clients $A \in \Gamma$ during time frame t . Then

$$\begin{aligned}
 p_j^{(t)} &= C_{A,j}^{(t)} \lambda = L_j (V_A^{(t)})^T \lambda = L_j (M_A R^{(t)})^T \lambda \\
 &= L_j (R^{(t)})^T M_A^T \lambda = L_j (R^{(t)})^T \varepsilon = \tilde{p}_j^{(t)}.
 \end{aligned}$$

The *Privacy* can be shown as follows. Consider the worst possible case, namely a subset of corrupt clients $B_1 \in \Delta_A$ helps a coalition of corrupt servers $B_2 \in \Delta_S$ in computing their

proofs for time frame τ , where we assume that during this time frame each server in the coalition has been visited by a subset of forbidden clients B_3 , i.e., $B_3 \in \Delta$ (so that we still have $B_3 \cup B_1 \in \Delta$). The total information known to the coalition of corrupt servers is constituted by the maximum information collected in time frames $1, \dots, \tau - 1$. That is, we assume that each server in the coalition has been visited by all clients $\mathcal{U}_1, \dots, \mathcal{U}_n$ in these time frames plus the information received in time frame τ .

Since the audit agency \mathbb{A} chooses the matrices $R^{(t)}$ randomly and keeps them secret, the clients have different shares for different time frames, so the information they give to server P_j is different. Hence all information collected during the previous visits is not consistent with the current information and the coalition of corrupt servers cannot use it.

Consider the value $p_j^{(t)} = C_{A,j}^{(t)} \boldsymbol{\lambda}$. By using the standard argument for proving privacy in LSSSs (e.g. see the proof for privacy in [4]) we see that from the point of view of the clients in $D = B_1 \cup B_3$, the information $C_{D,j}^{(t)}$ can be consistent with any secret matrix $\tilde{R}^{(t)}$. Indeed, since $D \in \Delta$ there exists a vector $\mathbf{k} \in \ker(M_B)$ such that $\mathbf{k}_1 = 1$. Then for any element $\alpha \in \mathbb{F}$ the matrix $\tilde{R}^{(t)} = R^{(t)} + \alpha(\mathbf{k} \otimes \mathbf{k})$ is consistent with $C_{D,j}^{(t)}$. So, the clients in D have no information about the secret matrix $R^{(t)}$ and about the value $C_{A,j}^{(t)}$ for some $A \in \Gamma$.

Finally, we shall show that the *Strong Privacy* property holds too. Consider the value $\tilde{p}_j^{(t)} = L_j(R^{(t)})^T \boldsymbol{\varepsilon}$. The coalition of corrupt servers $B_2 \in \Delta_S$ can try to guess $(R^{(t)})^T \boldsymbol{\varepsilon}$ or, if there is a linear dependence between the row-vectors L_j for servers $P_j \in B_2$, to compute $\tilde{p}_j^{(t)}$ provided that they already know all values $\tilde{p}_{j_1}^{(t)}$ for $P_{j_1} \in B_2 \setminus \{P_{j_2}\}$. Set $B_4 = B_2 \setminus \{P_{j_2}\}$. Consider the second possibility for the server P_{j_2} which is visited only by the clients in B_1 and is helped by the corrupt clients in B_3 . Even if all the servers in the corrupt coalition B_4 , have been visited by a qualified subset of clients $A \in \Gamma$ during that time frame (i.e., they are able to compute their proofs), P_{j_2} still cannot compute its proofs by finding a linear combination of their proofs $p_{j_1}^{(t)}$ for $P_{j_1} \in B_4$. This is true since from one side B_2 is in Δ_S and by the properties of Δ_S -rMSP (see Definition 2.2) does not exist a linear combination between the row vectors L_j for $P_j \in B_2$. On the other hand the group of bad servers can not calculate $(V_A^{(t)})^T \boldsymbol{\lambda}$ from $p_{B_4}^{(t)} = C_{A,B_4}^{(t)} \boldsymbol{\lambda}$ (for some $A \in \Gamma$) again by the properties of rMSP and thus can not get the proof $p_{j_2}^{(t)} = L_{j_2}(V_A^{(t)})^T \boldsymbol{\lambda}$. \square

5.3 Efficiency of the Scheme

Let $|\mathbb{F}| = q$ and denote by $\dim E_i$ the dimension of the vector space generated by the vectors M_i of client \mathcal{U}_i over \mathbb{F} , i.e., $\dim E_i = |\varphi(\mathcal{U}_i)|$. We denote by E_0 the set of secrets and by $\dim E_0$ the dimension of E_0 . It is well known that the information rate of a LSSS is $\rho = \dim E_0 / (\max_{1 \leq i \leq n} \dim E_i)$ and this rate is optimal (e.g. $\rho = 1$) in the threshold case. Assume that matrix M (built by means of an MSP) has the maximum possible information rate for the given access structure Γ . In order to be able to compare our result with the result of Blundo *et al.* in [3] we need to consider Γ_S as a threshold (k, \tilde{n}) access structure. In this case the matrix L is an $(\tilde{n}, k + 1)$ -Vandermonde matrix (i.e., $\tilde{m} = \tilde{n}$, $\tilde{d} = k + 1$ and $\tilde{\psi}, \tilde{\varphi}$ are bijections). Note that for the threshold access structures the MDP is the usual MSP with a Vandermonde matrix.

In [3] the audit agency broadcasts two types of public information: one is the linear mapping M_χ that enables the clients in $\chi \in \Gamma$ to compute the secret. The second is the linear mapping $\Pi_j^{(t)}$, i.e., the numbers $\lambda_{j,i}^{(t)}$ for $j = 1, \dots, \tilde{n}$; $i = 1, \dots, (k + 1)\tau$; and $t = 1, \dots, \tau$.

The amount of information that client \mathcal{U}_i receives from the audit agency during the initialization phase is equal to $(k + 1) \tau \log(q) \dim E_i$ and it is the same as in [3].

The amount of information that a client sends to a server during a visit is equal to $\log(q) \dim E_i$, which is again the same as in [3].

In our scheme the public information broadcast by the audit agency \mathbb{A} given by the matrices M and L is equal to $d \log(q) \sum_{i=1}^n \dim E_i = m d \log(q)$ and $\tilde{n} (k + 1) \log(q)$, respectively. Note also that in order to perform their duties the clients only need to know the matrix L , and the servers only need to know the matrix M .

On the other hand, the amount of public information in [3] is the linear mapping M_χ , which corresponds to our matrix M , and the numbers $\lambda_{j,i}^{(t)}$ from the second linear map $\Pi_j^{(t)}$. Hence the amount of information for the second mapping is $\tau^2 \tilde{n} (k + 1) \log(q)$. Note also that both the clients and the servers need to know these numbers $\lambda_{j,i}^{(t)}$. The public information is either stored in a central trusted server (e.g. audit agency), so every user can access it whenever need it, or it is spread (broadcast) to all users which store it locally. Thus in the first case our scheme has less communication complexity, or in the second case the memory usage is less.

Therefore our scheme has the following contributions compared to [3].

- It is more efficient w.r.t. the communication complexity compared to the scheme proposed in [3], since it uses less public information to the clients and servers ($\tilde{n} (k + 1) \log(q)$ versus $\tau^2 \tilde{n} (k + 1) \log(q)$).
- The memory storage required in our scheme is smaller since the public information has to be stored by the participants.
- Moreover, we proved that it satisfies stronger security requirements.
- Our scheme consider more general scenario where we consider general access structures both for the clients and for the servers.

6 Robust Metering Schemes

As noted before the basic model assumes that the clients do not present incorrect evidence to a server, thus preventing the server from constructing its proof. In order to prevent such a behavior robust Metering Schemes were proposed in [13, 14].

6.1 Threshold case

Naor and Pinkas [13] proposed the following robust metering scheme. We will give only the *Initialization* and *Regular Operation* phases, because the *Proof Computation* and *Proof Verification* phases are the same as in the linear metering scheme described in Fig. 1.

Robust Metering Scheme - threshold case [13]

Initialization:

1. Audit agency \mathbb{A} chooses the threshold access structures $\Gamma = \{A : |A| > k\}$ and $\Gamma_S = \{A : |A| > \tilde{k}\}$. Each user \mathcal{U}_i is associated publicly with a non-zero element $\alpha_i \in \mathbb{F}$. Analogously each server P_j for each time frame t is associated publicly with different non-zero elements $\beta_{j,t} \in \mathbb{F}$, such that $\tau|\mathcal{P}| + |\mathcal{U}| < |\mathbb{F}|$.
2. Audit agency \mathbb{A} chooses a polynomials $f_1(x, y)$ of degree k in x and of degree $\tau\tilde{k}$ in y , $f_2(x, y)$ of degree k_d in x and of degree $\tau\tilde{k}_d$ in y , and $f_3(y)$ of degree $\tau\tilde{k}_d$ in y . Then it computes $f_4(x, y)$ such that $f_4(x, y) = f_1(x, y)f_2(x, y) + f_3(y)$.
3. Audit agency \mathbb{A} gives: to each client \mathcal{U}_i his shares $f_4(\alpha_i, y)$ and $f_1(\alpha_i, y)$; and to each server P_j his shares $f_2(x, \beta_{j,t})$ and $f_3(\beta_{j,t})$.

Regular Operation:

1. When a client \mathcal{U}_i visits a server P_j during a time frame t , the client \mathcal{U}_i evaluates the polynomials in $\beta_{j,t}$, i.e. he sends $f_4(\alpha_i, \beta_{j,t})$ and $f_1(\alpha_i, \beta_{j,t})$ to server P_j .
2. The server evaluates the polynomial $f_2(x, \beta_{j,t})$ in α_i and verifies that $f_4(\alpha_i, \beta_{j,t}) = f_1(\alpha_i, \beta_{j,t})f_2(\alpha_i, \beta_{j,t}) + f_3(\beta_{j,t})$ holds.
3. If the check succeed server P_j offers a service to client \mathcal{U}_i , otherwise rejects. Next P_j stores the value $f_1(\alpha_i, \beta_{j,t})$ which will be used in the proof computation phase.

Figure 3: Robust Metering Scheme - threshold case [13]

But this scheme is subject to the following attack [14]. Assume that for a server P_j and for some time frame t there exists two clients \mathcal{U}_{i_1} and \mathcal{U}_{i_2} such that $f_1(\alpha_{i_1}, \beta_{j,t}) = 0$ and $f_1(\alpha_{i_2}, \beta_{j,t}) \neq 0$. Then they can compute $f_3(\beta_{j,t}) = f_4(\alpha_{i_1}, \beta_{j,t})$ and hence $f_2(\alpha_{i_2}, \beta_{j,t}) = \frac{f_4(\alpha_{i_2}, \beta_{j,t}) - f_3(\beta_{j,t})}{f_1(\alpha_{i_2}, \beta_{j,t})}$. Next client \mathcal{U}_{i_2} computes a random \bar{f}_1 and \bar{f}_4 such that $\bar{f}_4 = \bar{f}_1 f_2(\alpha_{i_2}, \beta_{j,t}) - f_3(\beta_{j,t})$ and $\bar{f}_1 \neq f_1(\alpha_{i_2}, \beta_{j,t})$. Finally, client \mathcal{U}_{i_2} can fool server P_j at time frame t to get a service without providing correct information. Then the authors of [14] propose the following modification (see Fig. 4).

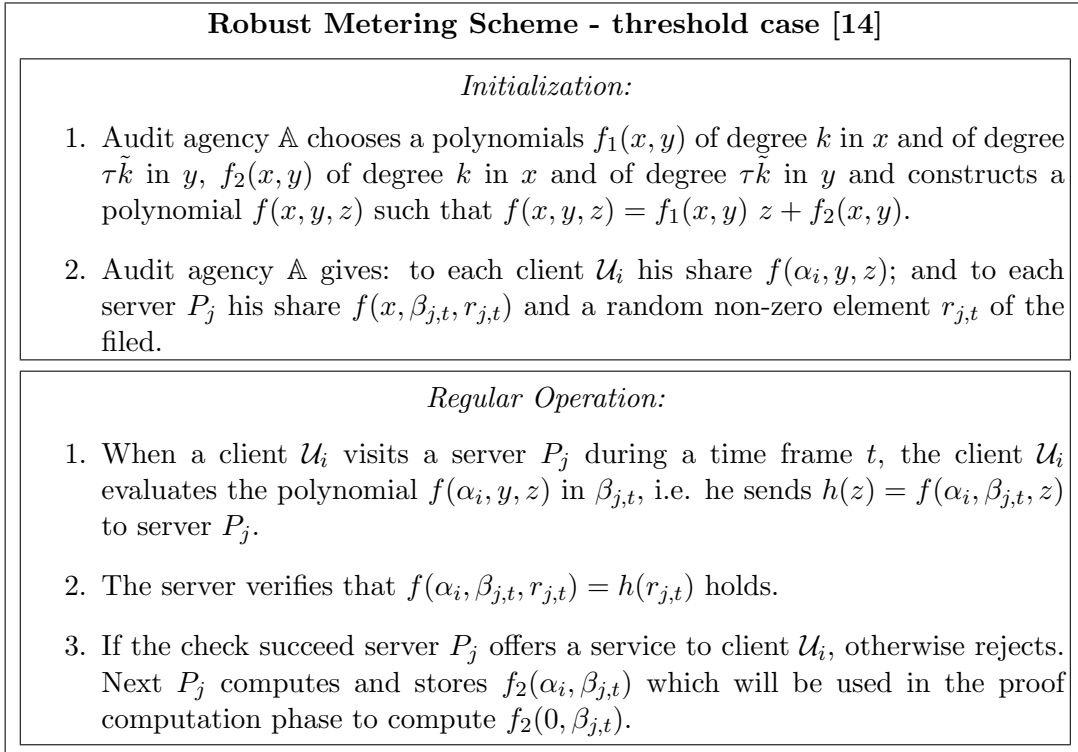


Figure 4: Robust Metering Scheme - threshold case [14]

The server security relies on the well known fact that for any two field elements a and c there are $|\mathbb{F}|$ tuples (b, d) such that $d = a b + c$.

We propose another way to fix the metering scheme from Fig. 3. The key for the attack proposed by Ogata and Kurosawa is that the function $f_3(y)$ does not depend on x , recall that $f_4(x, y) = f_1(x, y)f_2(x, y) + f_3(y)$. Thus the same value $f_3(\beta_{j,t})$ is used for verification of the shares of all players, which allow the attackers to compute it and then to mount an attack. Hence a way to avoid the described attack is to replace $f_3(y)$ with $f_3(x, y)$. The rest of the protocol described in Fig. 3 stays the same.

Note that in the considered model the clients are not protected against corrupt servers which deny to offer their services. One way to protect the clients is to allow them to complain to the audit agency against the server's behavior: if after Step 3 in the Regular Operation Phase, the server P_j denies service to client \mathcal{U}_i the latter will broadcast $(h(z), \mathcal{U}_i, P_j, t)$ as an accusation against the server. Note that if the client is honest then the server is corrupt or the information is broadcast by a corrupt client. The audit agency rejects payment to server P_j if in a given time frame t there is a qualified set of clients which complains (with correct information $h(z)$) against P_j .

6.2 General Access Structure case

In this section we propose two ways to construct robust metering scheme in the general case. First we will generalize the approach proposed in [14].

Robust Metering Scheme - general case I [12]

Initialization:

1. Audit agency \mathbb{A} chooses the access structures Γ and Γ_S . Let $\mathcal{M} = (\mathbb{F}, M, \psi, \varepsilon)$ be an MSP computing Γ and $\mathcal{L} = (\mathbb{F}, L, \tilde{\psi}, \tilde{\varepsilon})$ be an Δ_S -rMSP computing Γ_S (see Definition 2.2). Let M be an $m \times d$ matrix and L be an $\tilde{m} \times \tilde{d}$ matrix. These matrices are made public.
2. Next, \mathbb{A} chooses τ random $d \times \tilde{d}$ matrices $R^{(t,1)}$ and $R^{(t,2)}$. We can consider them as two “big” $d\tau \times \tilde{d}$ matrices $R^{(1)}$ and $R^{(2)}$, which are kept secret. Then \mathbb{A} gives to client \mathcal{U}_i shares $V_i^{(t,1)} = M_i R^{(t,1)}$ and $V_i^{(t,2)} = M_i R^{(t,2)}$ ($|\varphi(\mathcal{U}_i)| \times d$ - matrices) for all time frames $t = 1, \dots, \tau$. Also \mathbb{A} gives to server P_j share $U_j^{(t)} = L_j((R^{(t,1)} \otimes r_{j,t}) + R^{(t,2)})^T$ and a randomly chosen field element $r_{j,t}$.

Regular Operation:

1. When client \mathcal{U}_i visits server P_j during a time frame t , the client \mathcal{U}_i computes the values $C_{i,j}^{(t,1)} = L_j(V_i^{(t,1)})^T$ and $C_{i,j}^{(t,2)} = L_j(V_i^{(t,2)})^T$ ($|\tilde{\varphi}(P_j)| \times |\varphi(\mathcal{U}_i)|$ matrices) and sends them to the server P_j .
2. The server verifies that $(C_{i,j}^{(t,1)} \otimes r_{j,t}) + C_{i,j}^{(t,2)} = U_j^{(t)} M_i^T$.
3. If the check succeed server P_j offers a service to client \mathcal{U}_i , otherwise rejects. Next P_j stores $C_{i,j}^{(t,2)}$ which will be used in the proof computation phase to compute $p_j^{(t)} = C_{A,j}^{(t,2)} \lambda$ for some qualified set of clients $A \in \Gamma$.

Figure 5: Robust Metering Scheme - general case I [12]

Theorem 6.1. [12] *The Metering Scheme described in Fig. 5 is robust and perfectly secure against a (Δ_A, Δ_S) -adversary.*

Proof. The correctness and (strong) privacy follows directly from Theorem 5.1. What we need to prove in addition is that the scheme is *robust*, i.e. the server security. Using the properties of Kronecker (tensor) product it is easy to check that the following relations hold.

$$\begin{aligned}
 U_j^{(t)} M_i^T &= L_j((R^{(t,1)} \otimes r_{j,t}) + R^{(t,2)})^T M_i^T = L_j(M_i((R^{(t,1)} \otimes r_{j,t}) + R^{(t,2)}))^T \\
 &= L_j(M_i(R^{(t,1)} \otimes r_{j,t}) + M_i R^{(t,2)})^T \\
 &= L_j((M_i \otimes 1)(R^{(t,1)} \otimes r_{j,t}) + M_i R^{(t,2)})^T \\
 &= L_j((M_i R^{(t,1)}) \otimes (1 r_{j,t}) + M_i R^{(t,2)})^T = L_j((V_i^{(t,1)} \otimes r_{j,t}) + V_i^{(t,2)})^T \\
 &= L_j(((V_i^{(t,1)})^T \otimes r_{j,t}) + (V_i^{(t,2)})^T) \\
 &= ((L_j(V_i^{(t,1)})^T) \otimes r_{j,t}) + L_j(V_i^{(t,2)})^T = (C_{i,j}^{(t,1)} \otimes r_{j,t}) + C_{i,j}^{(t,2)}.
 \end{aligned}$$

Thus we prove the correctness of the server verification. The robustness of the scheme follows from Ogata and Kurosawa' arguments. \square

Now we generalize the method we proposed in Section 6.1.

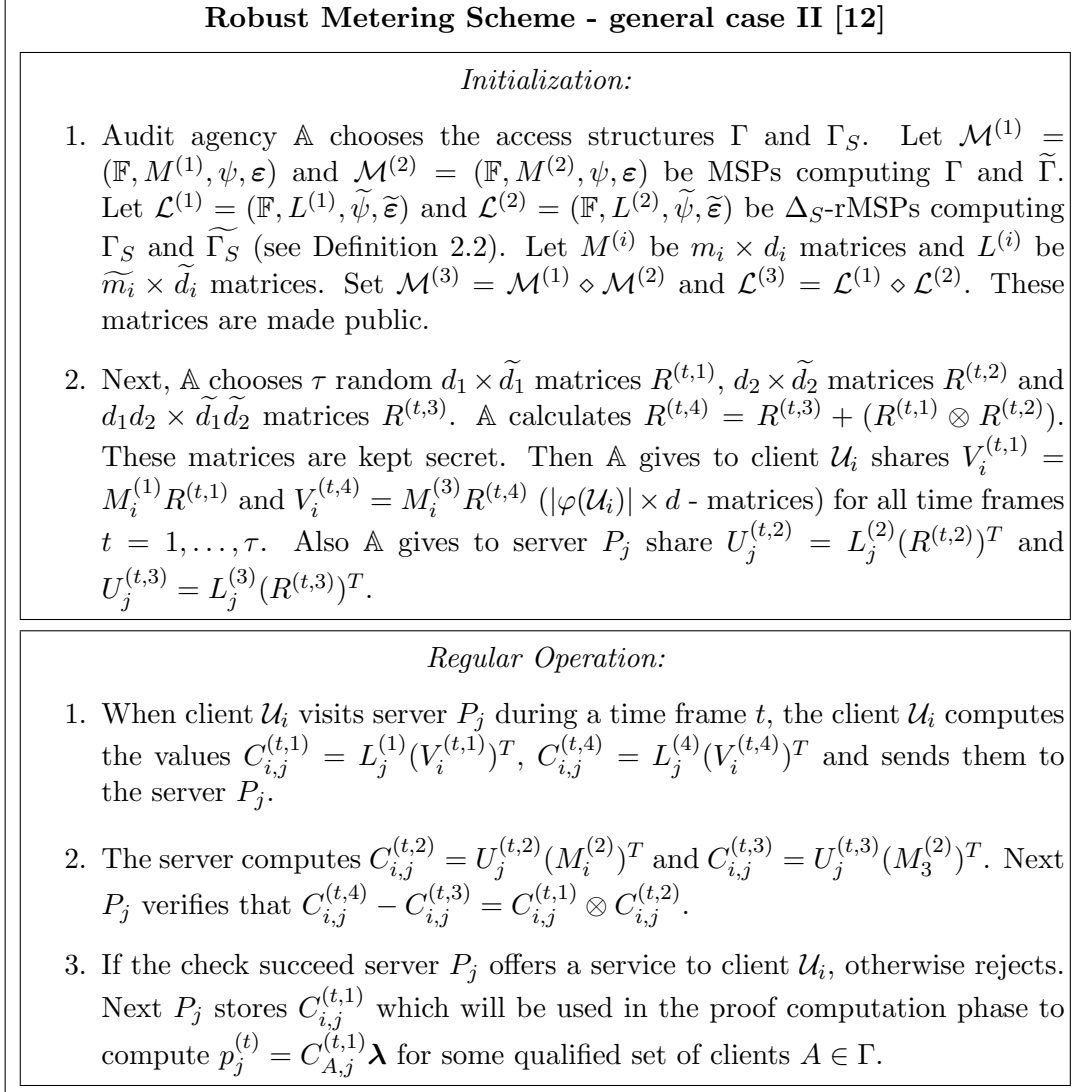


Figure 6: Robust Metering Scheme - general case II [12]

Theorem 6.2. [12] *The Metering Scheme described in Fig. 6 is robust and perfectly secure against a (Δ_A, Δ_S) -adversary.*

Proof. Again we need to prove only that the scheme is *robust*. First note that

$$L_j^{(z)} (V_i^{(t,z)})^T = L_j^{(z)} (M_i^{(z)} R^{(t,z)})^T = L_j^{(z)} (R^{(t,z)})^T (M_i^{(z)})^T = U_j^{(t,z)} (M_i^{(z)})^T.$$

Thus the verification check $C_{i,j}^{(t,4)} - C_{i,j}^{(t,3)} = C_{i,j}^{(t,1)} \otimes C_{i,j}^{(t,2)}$ actually rely on the equations

$$\begin{aligned} L^{(3)}(R^{(t,4)} - R^{(t,3)})^T(M^{(3)})^T &= (L^{(1)} \diamond L^{(2)})(R^{(t,4)} - R^{(t,3)})^T(M^{(1)} \diamond M^{(2)})^T \\ &= (L^{(1)} \diamond L^{(2)})(R^{(t,1)} \otimes R^{(t,2)})^T(M^{(1)} \diamond M^{(2)})^T \\ &= (L^{(1)}(R^{(t,1)})^T(M^{(1)})^T) \boxtimes (L^{(2)}(R^{(t,2)})^T(M^{(2)})^T), \end{aligned}$$

which are satisfied due to Lemma 3.2. This proves the correctness of the server verification. \square

7 Conclusions

Metering schemes can be considered as *two-level* SSSs. In an SSS the players which get the shares from the dealer reconstruct the secret themselves. In two-level structures the players in the first level get the shares from the dealer and later provide some information to the players in the second level who compute certain value related to the secret.

Earlier works on this topic considered general access structures for clients and threshold access structure for servers. In this paper we extend the model for metering schemes to *fully general access structure* – for clients, corrupt clients and servers. Next we *strengthened* the security model. Then we proposed a scheme, which is simpler, with more efficient communication complexity and reduced memory requirements compared to earlier works. We also have revisited the robust threshold metering schemes from [13, 14]; we have proposed two solutions to build *robust* general metering schemes based on the linear algebra approach. As a tool to achieve this goal we have introduced *doubly-labelled* matrices and an operation on such matrices. We have established a relation between MSPs and the diamond operation on the one hand and doubly-labelled matrices and the new operation on the other hand. Since MSPs and doubly-labelled matrices are used for multiplicative linear SSSs [4, 5] their relations are of independent interest. Finally we have demonstrated that one can protect clients against denied of service attacks of corrupt servers.

References

- [1] C. Blundo, A. De Bonis, B. Masucci, Metering Schemes with Pricing, *DISC'00*, LNCS 1914, Springer-Verlag 2000, pp. 194-208.
- [2] C. Blundo, A. De Bonis, B. Masucci, D. Stinson, Dynamic Multi-Threshold Metering Schemes, *SAC'00*, LNCS 2012, Springer-Verlag 2001, pp. 130-144.
- [3] C. Blundo, S. Martin, B. Masucci, C. Padro, A Linear Algebraic Approach to Metering Schemes, *Cryptology ePrint Archive: Report 2001/087*.
- [4] R. Cramer, I. Damgard, U. Maurer, General Secure Multi-Party Computation from any Linear Secret Sharing Scheme, *EUROCRYPT'00*, LNCS 1807, Springer-Verlag 2000, pp. 316-334.
- [5] R. Cramer, S. Fehr, Y. Ishai, E. Kushilevitz, Efficient Multi-Party Computation over Rings, *EUROCRYPT'03*, LNCS 2656, Springer-Verlag 2003, pp. 596-613.

- [6] M. K. Franklin, D. Malkhi, Auditable Metering with Lightweight Security, *Financial Cryptography'97*, LNCS 1318, Springer-Verlag 1997, pp. 151-160.
- [7] B. Masucci, D. Stinson, Metering Schemes for General Access Structures, *ESORICS'00*, LNCS 1895, Springer-Verlag 2000, pp. 72-87.
- [8] B. Masucci, D. Stinson, Efficient Metering Schemes with Pricing, *IEEE Transactions on Information Theory*, 47 (7), 2001, pp. 2835-2844.
- [9] V. Nikov, S. Nikova, B. Preneel, J. Vandewalle, Applying General Access Structure to Metering Schemes, *WCC'03*, 2003, *Cryptology ePrint Archive*: Report 2002/102.
- [10] V. Nikov, S. Nikova, B. Preneel, On Multiplicative Linear Secret Sharing Schemes, *INDOCRYPT'2003*, LNCS 2904, Springer-Verlag 2003, pp. 135-147.
- [11] V. Nikov, S. Nikova, On the size of Monotone Span Programs, *SCN'2004*, to appear in LNCS, Springer-Verlag.
- [12] V. Nikov, S. Nikova, B. Preneel, Robust Metering Schemes for General Access Structures, *ICICS'2004*, to appear in LNCS, Springer-Verlag.
- [13] M. Naor, B. Pinkas, Secure and Efficient Metering, *EUROCRYPT'98*, LNCS 1403, Springer-Verlag 1998, pp. 576-590.
- [14] W. Ogata, K. Kurosawa, Provably Secure Metering Scheme, *ASIACRYPT'00*, LNCS 1976, Springer-Verlag 2000, pp. 388-398.
- [15] P. Pudlak, J. Sgall, Algebraic models of computation and interpolation for algebraic proof systems, *Proc. Feasible Arithmetic and Proof Complexity*, LNCS, Springer-Verlag 1998, pp. 279-295.