An Unconditionally Hiding Auditing Procedure for Multi-Party Computations

Lucas Schabhüser, Denise Demirel, and Johannes Buchmann

Technische Universität Darmstadt, Germany,
lschabhueser@cdc.informatik@tu-darmstadt.de,
ddemirel@cdc.informatik@tu-darmstadt.de,
buchmann@cdc.informatik@tu-darmstadt.de

Abstract. In this work an unconditionally hiding auditing procedure for computations on documents stored in distributed fashion is introduced. There is only one multi-party computation (MPC) scheme providing auditability which computationally protects the inputs of the computations. Building on this, we propose a computationally hiding solution that uses bilinear maps and therefore produces no additional overhead in the online phase. In addition, we introduce a second variation that is the first auditable MPC scheme providing unconditional (or information-theoretic) hidingness. We achieve this by combining bilinear maps with unconditionally hiding commitments leading to only a small overhead in the online phase. We prove our solutions secure and give arguments for practicability and efficiency. The auditing procedures presented here are an important contribution since distributed storage solutions, e.g. cloud of clouds, allow for information-theoretic confidentiality. Using our technique, they can be extended to perform auditable computations on the data stored.

Keywords: unconditional hidingness, auditable multi-party computations, cloud computing, distributed storage

1 Introduction

In this work an unconditionally hiding auditing procedure for computations on documents stored in distributed fashion is introduced. This combines two very important and up to date research directions, distributed (cloud) storage and auditable Multi Party Computations (MPC). The idea of distributed storage is to distribute documents among a set of parties, i.e. shareholders, using a secret sharing scheme. However, a shortcoming of all existing solutions is that they lack methods that allow for auditable computations on the data stored. Note that this is an important functionality for many use cases, e.g. to compute statistics on medical records. Instead of reconstructing the documents what exposes them to threats the computations should be performed in distributed fashion, i.e. each shareholder should evaluate the computations directly on its shares. Thus, applying existing MPC schemes is a promising approach to fill this gap. An alternative could be to use a verifiable computing scheme, such as [BCTV14] or [BBFR15], but the verifiable computing solutions existing in literature come with a significant computational overhead and the approaches providing unconditional hidingness rely on non-standard assumptions.

While performing MPC on shares is standard, there is only one approach that covers auditability [BDO14]. In our work we use this solution as a basis and present modifications that improve it with respect to efficiency and confidentiality to meet the requirements of distributed storage architectures. Like the solution presented in [BDO14] our variations consist of an input independent preprocessing phase and an online phase. While the original proposal produces an overhead factor of 2 to provide auditability of the online phase, we propose a computationally hiding solution that produces no additional overhead. Besides this speed up we introduce a second variation that is the first auditable MPC scheme providing unconditional (or information-theoretic) hidingness, i.e. that ensures confidentiality of the documents processed even in the presence of computationally unbounded adversaries. Note that we achieve this while producing only a small overhead in the online phase. This is an important contribution since unlike storage architectures using encryption, distributed storage solutions allows for information-theoretic confidentiality and are therefore suitable for use cases where the security of the documents stored must be ensured for several decades or even indefinitely. Note that if the audit trail hides the input documents only computationally all an attacker has to do is to store this data and to wait until it is able to break the underlying mathematical hardness assumption for the parameters chosen. Since processing power increases continually following Moore's law this attack will be successful sooner or later. There are estimations how long cryptographic primitives should be secure for certain parameter sets. However, there is no guarantee that new attacks or current trends in technology, like quantum computers, do not allow to break the security earlier than expected. In this case the audit data can be "decrypted" and confidentiality of the documents is irreversibly lost. Thus, for critical documents having a solution that ensures confidentiality information-theoretically is a clear advantage compared to merely computational secure solutions.

High Level Explanation of our Contribution. In the following we will briefly explain our approach for additions to give an intuition to our protocol. The case for multiplications and especially unconditional hidingness is a lot more complex and omitted here. Our scheme uses a distributed storage architecture, e.g. [BCQ$^+$11,BCQ$^+$13,Clo15,LHS15], containing of n shareholders s_j each having a unique ID $j \in \{1, \cdots, n\}$. To distribute a set of documents $m_1, \cdots, m_N$ Shamir Secret Sharing [Sha79] is used, where t shareholders, for $2t \leq n$, must participate in the reconstruction to reveal the document. More precisely, for each document m_i the document owner chooses a function $f_i(x) = m_i + a_{i,1} + a_{i,2}x + \cdots, a_{i,t-1}x^{t-1}$, where the document is the free coefficient and the other coefficients $a_{i,1}, \cdots, a_{i,t-1}$ are chosen uniformly at random. Then it sends to each shareholder s_j share $\sigma_{j,i} = f_i(j)$. Note that this construction allows to compute linear functions locally, e.g. shares to $f_{N+1}(x) = f_1(x) + f_2(x)$ can be computed by $\sigma_{j,1} + \sigma_{j,2} = f_1(j) + f_2(j) = f_{N+1}(j)$, for $j \in \{1, \cdots, n\}$. To ensure that the documents were computed correctly by the storage system we accompanied this process with an auditable MPC scheme. Thus, each data owner publishes a commitment $c(m_i)$ to each document m_i. Since also commitments are linearly homomorphic operations performed on the documents, or rather shares, can also be performed directly on the commitments, i.e. $c(m_{N+1}) = c(m_1) \cdot c(m_2)$. The hidingness (computational or unconditional) of the commitments generated ensure that the auditors learn nothing about the input documents m_1 and m_2 while the bindingness ensures that commitment $c(m_{N+1})$ can only be opened if m_{N+1} has been computed correctly. Building on this idea we provide the first protocol that allows for an audit trail for arbitrary computations that is information-theoretic secure, i.e. not even a computationally unbounded attacker having access to the audit data nor a subset with less than t shareholders is able to reveal the input documents. To achieve this we used Pedersen commitments [Ped91] and bilinear maps to prove the correctness of computations. Since bilinear maps also allow to improve the efficiency of computational hiding audits in this work we provide both variations, a computational hiding and a seconds unconditional hiding auditing procedure.

Structure The paper is structured as follows. In Section 2 we provide the related work followed by the preliminaries in Section 3. In Section 4 we first introduce the setup and security definitions used and then present our solution in detail. In Section 5 we present the assumptions made and prove their security. Furthermore, we discuss certain additional properties, i.e. practicability and efficiency. Finally, in Section 6 we provide a conclusion and present directions for future work.

2 Related Work

In the context of electronic voting several auditable tallying procedures providing unconditional confidentiality of the votes cast, such as [CRS05] or [DvdGdSA12], have been proposed. However, they do not provide solutions for arbitrary arithmetic circuits. In [AF07] a non-interactive zero-knowledge prove for *circuit-SAT*

has been proposed using bilinear maps to perform multiplications. However, they do not provide auditing nor do they perform computations on shares. In [SV15] it is shown how to construct zero-knowledge proofs for multi-party computations based on threshold encryption (such as [CDN01]) in the Random Oracle Model. Their solution allows for public auditing, but the overhead for the shareholders is not analysed. Furthermore, our solution is proven secure in the standard model. In [BDO14] the authors introduce the first auditable multi-party computation scheme. Their approach allows for a quite efficient computation process for a SPDZ type MPC (see [DPSZ12]). Their scheme however is not unconditionally hiding and is based on an (n, n) secret sharing scheme, which requires all shareholders to be present during reconstruction. Thus, if one server has a malfunction the documents cannot be accessed anymore. Our scheme provides an unconditional hiding audit trail and requires only a subset of shareholders to attend the computation and document reconstruction process. *Verifiable computing schemes* such as [PHGR13], [CFH$^+$15], [BCG$^+$13], [BCTV14] or [BBFR15] allow the verification of arbitrary arithmetic circuits. However, they demand a significant computational overhead from the shareholders and approaches providing unconditional hidingness come with non-falsifiable assumptions. Our setting has a somewhat more expensive verification phase, but minimises the shareholders overhead and only relies on standard assumptions.

3 Preliminaries

3.1 Secret Sharing

Shamir secret sharing allows a *data owner* to distribute a message among a set of *shareholders*, such that the message can only be reconstructed if a qualified subset of these shareholders collaborates. At the same time no other subset can learn any information about the message distributed. Let n be the total number of shareholders, $j \in \{1, \ldots, n\}$ be the unique ID of shareholder s_j, $t \leq n$ be the threshold required for reconstruction, and $\mathbb{F}_q$ be a field with $q > n$ elements. Then Shamir secret sharing can be defined via the following two algorithms.

SShare Given a message $m \in \mathbb{F}_q$ as input the algorithm chooses a function $f(x) = m + a_1 + a_2 x + \cdots, a_{t-1}x^{t-1}$, where the message m is the free coefficient and the other coefficients $a_1, \cdots, a_{t-1}$ are chosen uniformly at random. Then it computes n shares σ_j, for $j \in \{1, \ldots, n\}$, where $\sigma_j := f(j)$ is sent to shareholder s_j.

SReconstruct It takes as input a subset $\mathcal{B} \subset \{1, \ldots, n\}$ of shareholders and corresponding shares $\sigma_j, \forall j \in \mathcal{B}$. If $|\mathcal{B}| < t$ it outputs $\perp$. Otherwise it reconstructs the unique interpolation polynomial $f^*(x)$ of degree $t-1$ in $\mathbb{F}_q[x]$ and returns message $f^*(0) = m^*$.

Note that there exists a reconstruction vector $(\{w_j\}_{j\in\mathcal{B}})$ such that $\sum_{j\in\mathcal{B}} w_j \cdot \sigma_j = m$. Since for Shamir secret sharing the *Lagrange Interpolation* formula is used to reconstruct message m the reconstruction vector is defined as $w_j =$

$\prod_{i\in\mathcal{B},i\neq j} \frac{i}{i-j}$, for $j \in \mathcal{B}$. Furthermore, shares generated with polynomials of degree $t-1$ are called t-reconstructing shares.

3.2 Commitment Schemes

A commitment scheme allows a *committer* to commit to a message while keeping it hidden to others (*hidingness*) and without allowing the committer to change the message committed to at a later point in time (*bindingness*).

Definition 1 (Feldman Commitment [Fel87]). *The Feldman commitment scheme consists of the following algorithms.*

Setup *Given a security parameter λ choose a prime q, a group $\mathbb{G}$ of order q, and a generator g of $\mathbb{G}$.*
FCommit *Given a message $m \in \mathbb{F}_q$ it outputs a commitment $c = g^m$.*
FOpen *Given a commitment $c \in \mathbb{G}$ and a message $m \in \mathbb{F}_q$ it returns '1' if $c = g^m$ holds and '0' otherwise.*

Feldman commitments are unconditionally binding. They are computationally hiding, i.e. hidingness is provided as long as the *discrete logarithm problem* in $\mathbb{G}$ is hard.

Definition 2 (Pedersen Commitment [Ped91]). *The Pedersen commitment scheme consists of the following algorithms.*

Setup *Given a security parameter λ choose a prime q, a group $\mathbb{G}$ of order q, and distinct generators g, h of $\mathbb{G}$.*
PCommit *Given a message $m \in \mathbb{F}_q$ and randomness $r \in \mathbb{F}_q$ it outputs commitment $c = g^m h^r$.*
POpen *Given a message $m \in \mathbb{F}_q$, a randomness $r \in \mathbb{F}_q$, and a commitment $c \in \mathbb{G}$ it returns '1' if $c = g^m h^r$ holds and '0' otherwise.*

Unlike Feldman commitments a Pedersen commitment can be a commitment to any message with the same probability. Thus, an adversary that is able to break the *discrete logarithm problem* for the parameters chosen can open a commitment to any value. It follows that these commitments are only computationally binding. However, for the same reason they are unconditionally hiding. Since $m + x \cdot r$ is distributed uniformly at random in $\mathbb{F}_q$, where x is the unknown discrete logarithm $h = g^x$, not even a computationally unbounded attacker can identify the message initially committed to.

3.3 Pairings

Definition 3 (Pairing [BF03]). *A* bilinear map *is a tuple $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$ such that:*

1. *$\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_T$ are cyclic groups of prime order q.*
2. *The* Discrete Logarithm Problem *is hard to be computed in $\mathbb{G}_1, \mathbb{G}_2$, and $\mathbb{G}_T$.*

3. $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ *is bilinear, i.e.* $e({g_1}^a, {g_2}^b) = e(g_1, g_2)^{ab}$ *holds for all* $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$, *and* $a, b \in \mathbb{Z}$.
4. e *is non-degenerate, i.e.* $e(g_1, g_2) \neq 1$ *for all* $(g_1, g_2) \neq (1, 1)$
5. e *is efficiently computable.*

The function e *is called* bilinear map, *or* pairing.

In case of symmetric pairings where $\mathbb{G}_1 \simeq \mathbb{G}_2$ we write $\mathbb{G}$ for both $\mathbb{G}_1$ and $\mathbb{G}_2$ for the sake of convenience.

3.4 Computational Exponent Inversion Problem

Definition 4 (Computational Exponent Inversion Problem (CIE)). *Assume* $g, h = g^x \in \mathbb{G}$ *is given for some random* x, *then the* computational exponent inversion (CIE) problem *is to compute* $g^{\frac{1}{x}}$. *More precisely, for all uniform PPT adversaries* $\mathcal{A}$

$$\mathsf{Pr}[\mathsf{y} \leftarrow \mathcal{A}(\mathbb{G}, \mathsf{g}, \mathsf{h} = \mathsf{g}^{\mathsf{x}}) : \mathsf{y} \in \mathbb{G} \wedge \mathsf{y} = \mathsf{g}^{\frac{1}{\mathsf{x}}} \\ |\mathbb{G} \leftarrow \mathcal{G}(1^\lambda), g, h \leftarrow \mathbb{G}] \leq \mathsf{negl}(\lambda).$$

The *CIE* assumption is known to reduce to the *Computational Diffie Hellman Assumption* as has, for example, been shown in [SS01].

4 Auditing of Computations Performed in Distributed Storage Systems

4.1 Setup and Security Definitions

In this work we provide an auditing mechanism for computations on data stored in distributed fashion. More precisely, our distributed storage system contains of the algorithms Setup, Distribute, Compute, Audit, Reconstruct, and Verify. Furthermore, three parties are involved. (1) *Data owners* distributing their documents, called *messages*, to a set of shareholders by running Distribute. (2) A set of *shareholders* holding shares to messages and performing computations on them by running Compute, and (3) *auditors* verifying the computations performed on the messages by running Audit and Verify. Messages stored in the storage system can be reconstructed by authorized parties (e.g. document owners or third parties when only computation results are requested) using algorithm Reconstruct. In the following we describe the algorithms for the computationally hiding audit with the modifications for the unconditionally hiding audit in brackets.

Setup Given a security parameter λ as input this algorithm chooses a prime q and a threshold t, such that $t \leq n < q$ ($2t \leq n < q$), where q, t, n are the parameters for the secret sharing scheme. It chooses a pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ of groups $\mathbb{G}, \mathbb{G}_T$ of order q and generators $g \in \mathbb{G}$ ($g, h \in \mathbb{G}$), which are the parameters of the commitment scheme.

Distribute In this algorithm a data owner distributes a message m among the n shareholders by calling SShare. Note that each message gets a unique ID $i \in \mathcal{I}$ that is equal for all shareholders. This is necessary to refer to the shares of messages stored when performing computations on them. Additionally, the data owner computes a commitment c to m by calling FCommit (PCommit). (Pedersen commitments are computed using a random value r chosen by the data owner. Thus, to be able to open c also r must be stored among the shareholders by calling SShare.) The commitment c is made public.

Compute On input a circuit $\mathcal{C}$ and the IDs of the input messages each shareholder evaluates $\mathcal{C}$ using its shares. In addition, each shareholder publishes some data for auditing.

Audit The auditor computes a commitment c_{out} to the outcome of circuit $\mathcal{C}$ using the data published by the data owner and the shareholders.

Reconstruct It takes as input a subset $\mathcal{B} \subset \{1, \ldots, n\}$ of shareholders and a corresponding subset of shares σ_j^*, $\forall j \in \mathcal{B}$. Then it calls SReconstruct and returns the output.

Verify The auditor takes as input the outcome of algorithm Reconstruct and the commitment c assigned to the message, either computed by the data owner during Share or by the auditor during Audit. It computes the corresponding Feldman (Pedersen) commitment c^* to the reconstructed message. If $c^* = c$ it outputs '1'otherwise it outputs '0'.

To allow for auditing the dealer and shareholders must publish certain audit data during Distribute and Compute. In our construction we use for this a *public bulletin board* which is common practice. Informally speaking a public bulletin board allows to make data publicly available in a tamper proof way. For a more rigorous definition we refer to [HL08].

In the following we formally define *Auditable Correctness* for operations performed on data stored in distributed fashion (a similar definition can be found in [BDO14]). In this work we only describe the MPC protocol secure against passive adversaries. However our scheme can easily be extended to address active adversaries by using standard techniques, e.g. [DPSZ12].

Definition 5 (Auditable Correctness). *Let $\mathcal{C}$ be a circuit, $\{m_i\}_{i\in\mathcal{I}}$ be the inputs to $\mathcal{C}$ where i is their unique ID and $\mathcal{I}$ the set of IDs of the input messages, m^* be a potential output of $\mathcal{C}$, and τ be a protocol transcript for the evaluation of $\mathcal{C}$ containing of all published information. An* auditing scheme *containing of the algorithms* Audit *and* Verify *is* correct *if it has the following two properties.*

Soundness *The algorithm* Verify *on input m^* outputs '1' with probability* 1 *if $\mathcal{C}$ on inputs $\{m_i\}_{i\in\mathcal{I}}$ produces the output m^* and τ is a valid transcript.*

Unforgeability *The algorithm* Verify *will return '0' (except with negligible probability) if $\mathcal{C}(\{m_i\}_{i\in\mathcal{I}}) \neq m^*$ or τ is not a valid transcript.*

4.2 Protocols

In this section we will show how auditable computations on data stored in distributed fashion can be performed. We will start by introducing our framework for arbitrary computations. We will first describe the preprocessing process and the circuit evaluation, i.e. containing of addition and multiplication gates. Afterwards, we show in detail how additions of messages can be performed by the shareholders without reconstructing the messages and how an auditor is able to compute the corresponding audit data using the commitments to these messages as input. Following, we present a corresponding protocol for multiplication gates. We show that also for this operation audit data can be computed on the basis of the commitments to the input messages and additional commitments generated and published by the shareholders. The security proof and details regarding the assumptions made and attackers addressed follow in Section 5. In the following we describe the algorithms for the computationally hiding audit with the modifications for the unconditionally hiding audit in brackets.

Preprocessing Stage In the preprocessing stage triples are generated. Note that this has been a standard tool since its introduction in [Bea91], because it allows to avoid a quadratic communication complexity during multiplication.

Preprocessing Each shareholder s_j generates M triples $l \in \{1, \cdots, M\}$ of t-reconstructing shares $\sigma_j(\alpha_l), \sigma_j(\beta_l), \sigma_j(\gamma_l)$ to $(\alpha_l, \beta_l, \gamma_l) \subset \mathbb{F}_q$ such that for each the equation $\alpha_l \cdot \beta_l = \gamma_l$ holds (see, e.g. [DN07]). (It also chooses t-reconstructing shares $\rho_j(\zeta_l), \rho_j(\eta_l), \rho_j(\theta_l) \in \mathbb{F}_q$. Since they do not have to fulfill any restrictions they can be chosen uniformly at random.) Each shareholder runs FCommit (PCommit) to compute commitments $c_j(\alpha_l), c_j(\beta_l), c_j(\gamma_l)$ (to compute commitments $c_j(\alpha_l, \zeta_l), c_j(\beta_l, \eta_l), c_j(\gamma_l, \theta_l), \mu_{j,l}$) that are published on the bulletin board.

PreAudit The auditor checks whether each shareholder published commitments to M triples on the bulletin board. If not it returns $\perp$. Else it checks the well-formedness by taking the reconstruction vector $(w_1, \ldots w_n)$ and performing the following steps.

1. For each $l \in \{1 \cdots, M\}$ it computes

$$c(\alpha_l) = \prod_{j \in \mathcal{B}} c_j(\alpha_l)^{w_j};\ c(\beta_l) = \prod_{j \in \mathcal{B}} c_j(\beta_l)^{w_j};\ c(\gamma_l) = \prod_{j \in \mathcal{B}} c_j(\gamma_l)^{w_j}$$

(In the unconditionally hiding case it additionally computes

$$\mu_l = \prod_{j \in \mathcal{B}} (\mu_{j,l})^{w_j}.)$$

2. For each $l \in \{1, \cdots M\}$ it checks whether $e(c(\alpha_l), c(\beta_l)) = e(c(\gamma_l), g)$ (checks whether $e(c(\alpha_l), c(\beta_l)) = e(c(\gamma_l), g) \cdot e(\mu_l, h)$) holds. If it does it outputs '1'. Else it outputs '0'.

Performing Arbitrary Functions/Evaluating Arbitrary Circuits In the following we introduce the algorithms needed to perform arbitrary, auditable computations on messages stored in distributed fashion. The inputs are a circuit and the unique IDs of the input messages. The circuit consists of a list of *addition* and *multiplication* operations each to be performed on two inputs $\mathsf{id_a}$ and $\mathsf{id_b}$. During the circuit evaluation each shareholder computes a share to the output message using the shares it holds from the input messages. During the individual addition and multiplication operations each input can either be the share of a message received from the data owner or an intermediate value that has been computed by the shareholder while evaluating the circuit. Since each message that got distributed among the shareholders is accompanied with a publicly available commitment an auditor is able to compute a commitment to the expected output message using these commitments together with additional audit data published by the shareholders. This commitment is published on the bulletin board to allow the outcome of the computation to be input to other circuits.

ComputationSetup This algorithm receives a circuit $\mathcal{C}$ as input (and a random linear function $\mathcal{L} : \mathbb{F}_q^M \to \mathbb{F}_q$, where M is the number of multiplication gates in $\mathcal{C}$) and forwards it to each shareholder and the bulletin board. (It additionally receives $\hat{m}, \hat{r}$ jointly generated by the data owners and runs Distribute returning commitment $\hat{c}$.) A qualifying subset $\mathcal{B}$ of size t (of size $2t$) is chosen. If there is no such subset return $\perp$. Otherwise the shareholders compute the reconstruction vector $(w_1, \ldots w_n)$ and run Compute.

Compute Each shareholder goes gate by gate through the circuit $\mathcal{C}$ and calls either Add or Multiply and finally calls Result.

Audit It takes the circuit $\mathcal{C}$ from the bulletin board, follows it gate by gate calling either AddAudit or MulAudit, and finally calls Verify.

Add On input IDs $\mathsf{id_a}, \mathsf{id_b}$ each shareholder $j \in \mathcal{B}$ uses its shares to compute shares to the sum of the input messages. The result receives the ID $\mathsf{id_c}$.

AddAudit On input IDs $\mathsf{id_a}, \mathsf{id_b}, \mathsf{id_c}$ the auditor checks whether it has commitments to the input messages. If not it takes them from the bulletin board. If they do not exist on the bulletin board it outputs $\perp$. Otherwise it computes a commitment to the expected result.

Multiply On input IDs $\mathsf{id_a}, \mathsf{id_b}$ each shareholder $j \in \mathcal{B}$ uses its shares to compute shares to the product of the input messages. The result receives the ID $\mathsf{id_c}$.

MultAudit On input IDs $\mathsf{id_a}, \mathsf{id_b}, \mathsf{id_c}$ the auditor checks whether it has commitments to the input messages. If not it takes them from the bulletin board. If they do not exist on the bulletin board it outputs $\perp$. Otherwise it computes a commitment to the expected result.

Result It takes the output $\sigma_j(m^*)$ (and $\rho_j(r^*)$) of the final gate and sends it (them) to the bulletin board. (Each shareholder s_j computes $\sigma_j(\tilde{m}) = \mathcal{L}(\sigma_j(\gamma_1), \ldots, \sigma_j(\gamma_M)) + \sigma_j(\hat{m})$ and $\rho_j(\tilde{r}) = \mathcal{L}(\rho_j(\theta_1), \ldots, \rho_j(\theta_M)) + \rho_j(\hat{r})$ and sends them to the bulletin board, where $\mathcal{L}, \hat{m}, \hat{r}$ was input to the ComputationSetup.)

Verify It takes c^* as the output of the final call of either AddAudit or Multaudit as well as $\sigma_j(m^*)$ (and $\rho_j(r^*), \sigma_j(\tilde{m}), \rho_j(\tilde{r}), \hat{c}$) from the bulletin board. If it does not exist it outputs '0' else it runs Reconstruct to compute m^* (and $r^*, \tilde{m}, \tilde{r}$). Then it checks whether $c^* = g^{m^*}$ (checks whether $c^* = g^{m^*} h^{r^*}$). (It additionally checks whether $\hat{c} \cdot \prod_{l=1}^{M} c(m_{\gamma_l})^{\lambda_l} = g^{\tilde{m}} h^{\tilde{r}}$ where λ_l is the l-th coefficient of $\mathcal{L}$.) If the conditions do not hold it outputs '0', else it outputs '1'.

Add Each shareholder $j \in \mathcal{B}$ takes the input shares $\sigma_j(m_{\mathsf{id_a}})$ and $\sigma_j(m_{\mathsf{id_b}})$ to message $m_{\mathsf{id_a}}$ and $m_{\mathsf{id_b}}$ respectively and outputs a new share $\sigma_j^* = \sigma_j(m_{\mathsf{id_a}}) + \sigma_j(m_{\mathsf{id_b}})$. (Each shareholder $j \in \mathcal{B}$ additionally takes the input shares $\rho_j(r_{\mathsf{id_a}})$ and $\rho_j(r_{\mathsf{id_b}})$ to message $r_{\mathsf{id_a}}$ and $r_{\mathsf{id_b}}$ respectively and outputs a new share $\rho_j^* = \sigma_j(m_{\mathsf{id_a}}) + \sigma_j(m_{\mathsf{id_b}})$.)

AddAudit The auditor takes as inputs the commitments $c(m_{\mathsf{id_a}})$ and $c(m_{\mathsf{id_b}})$ and returns $c^* = c(m_{\mathsf{id_a}}) \cdot c(m_{\mathsf{id_b}})$

For the multiplication we assume that $\mathcal{B}$ is the subset of shareholders participating in this process, with $|\mathcal{B}| \geq t$ (with $|\mathcal{B}| \geq 2t$), where t is the threshold required during reconstruction.

Multiply The shareholders choose a triple (α, β, γ) from the preprocessing stage. Each shareholder s_j, $j \in \mathcal{B}$, takes the input shares $\sigma_j(m_{\mathsf{id_a}})$ and $\sigma_j(m_{\mathsf{id_b}})$ as well as the shares $\sigma_j(\alpha), \sigma_j(\beta), \sigma_j(\gamma)$ to the triple. Then the following steps are performed.

1. Each shareholder s_j computes $\sigma_j(\delta) = \sigma_j(m_{\mathsf{id_a}}) - \sigma_j(\alpha), \sigma_j(\epsilon) = \sigma_j(m_{\mathsf{id_b}}) - \sigma_j(\beta)$.

2. The shareholders jointly run SReconstruct on $\sigma_j(\delta), \sigma_j(\epsilon)$ to publicly reconstruct δ, ϵ using the bulletin board.
3. Each shareholder s_j computes $\sigma_j^* = \sigma_j(\gamma) + \epsilon \cdot \sigma_j(m_{\mathsf{id_a}}) + \delta \cdot \sigma_j(m_{\mathsf{id_b}}) - \delta\epsilon$ and outputs σ_j^*.

(In addition the following steps are performed.

4. Each shareholder s_j computes $\rho_j(\tilde{\delta}) = \rho_j(\zeta) - \rho_j(r_{\mathsf{id_a}}), \rho_j(\tilde{\epsilon}) = \rho_j(\eta) - \rho_j(r_{\mathsf{id_b}})$.
5. The shareholders jointly run SReconstruct on $\rho_j(\tilde{\delta}), \rho_j(\tilde{\epsilon})$ to publicly reconstruct $\tilde{\delta}, \tilde{\epsilon}$ using the bulletin board.
6. Each shareholder s_j computes $\rho_j^* = \rho_j(\theta) + \epsilon \cdot \rho_j(r_{\mathsf{id_a}}) + \delta \cdot \rho_j(r_{\mathsf{id_b}}) - \delta\epsilon$ and outputs ρ_j^*.)

MultAudit The auditor takes as inputs the commitments $c(m_1)$ and $c(m_2)$ to both input messages $m_{\mathsf{id_a}}$ and $m_{\mathsf{id_b}}$. It takes δ, ϵ and $c(\alpha), c(\beta), c(\gamma)$ (and additionally $\tilde{\delta}, \tilde{\epsilon}$) from the bulletin board. If they do not exist it outputs $\perp$. Else it checks whether $c(\alpha)^{-1} \cdot c(m_{\mathsf{id_a}}) = g^{\delta}, c(\beta)^{-1} \cdot c(m_{\mathsf{id_b}}) = g^{\epsilon}$ (checks whether $c(\alpha)^{-1} \cdot c(m_{\mathsf{id_a}}) = g^{\delta} h^{\tilde{\delta}}, c(\beta)^{-1} \cdot c(m_{\mathsf{id_b}}) = g^{\epsilon} h^{\tilde{\epsilon}}$). If the equations do not hold it outputs $\perp$ else it computes a commitment $c^* = c(m_\gamma) \cdot c(m_{\mathsf{id_a}})^{\epsilon} \cdot c(m_{\mathsf{id_b}})^{\delta} \cdot g^{-\delta\epsilon}$ and outputs c^*.

5 Properties

5.1 Auditable Correctness

For the correctness of our solution we make the following assumptions: (Ass.1) The parameters for the bilinear map are chosen, such that the CIE problem in $\mathbb{G}$ (see Section 3.4) holds. Note that from this it follows that we also assume the discrete logarithm problem to hold. (Ass.2) Audit data can be published in a tamper proof way, e.g. using a secure bulletin board. (Ass.3) The commitments published by the data owners can be opened using the messages and random values distributed. (Ass.4) The linear function $\mathcal{L}$ is chosen at random and not known to the shareholders before the preprocessing phase. (Ass.5) We assume that the commitments published by the data owners have been computed correctly[1].

Theorem 1. *Under Assumption Ass.1, Ass.2, Ass.3, Ass.4, and Ass.5 the computational and unconditional hiding audit protocols presented in Section 4.2 provide* auditable correctness *as defined in Definition 5.*

Proof. To provide auditable correctness the protocols must provide *Soundness* and *Unforgeability*.

[1] Note that for applications where this is not a reasonable assumption additional measures can be used to ensure consistency between the shares and the commitments. See, for instance, corresponding techniques developed for eVoting schemes [MN10].

Soundness First, we show that if the algorithms Preprocessing, Compute, Reconstruction, and Audit were performed correctly, then the algorithm Verify will accept the result with probability '1', i.e. *Soundness* is provided. In other words, the audit for each gate must, with overwhelming probability, output a commitment for which the output of the gate is a valid opening. In the following we will only show soundness of the unconditional hiding audit protocol. A corresponding proof for the merely computational hiding protocol can be obtained by setting all random values r_i to be 0, i.e. $h^{r_i} = 1$. Assume we have messages $m_{\mathsf{id_a}}$, $m_{\mathsf{id_b}}$ with randomness $r_{\mathsf{id_a}}, r_{\mathsf{id_b}}$ and their corresponding commitments $c_{\mathsf{id_a}} = g^{m_{\mathsf{id_a}}} \cdot h^{r_{\mathsf{id_a}}}$ and $c_{\mathsf{id_b}} = g^{m_{\mathsf{id_b}}} \cdot h^{r_{\mathsf{id_b}}}$. Furthermore, the shareholders hold shares $\sigma_j(m_{\mathsf{id_a}}), \sigma_j(m_{\mathsf{id_b}})$ of the messages and shares $\rho_j(r_{\mathsf{id_a}}), \rho_j(r_{\mathsf{id_b}})$ of the random values. In the following we will first look at the addition case and then discuss the multiplication case by first showing the correctness of the preprocessing stage followed by the multiplication performed online.

Additions: If all shareholders followed the protocol correctly and computed $\sigma_j(m_{\mathsf{id_c}}) = \sigma_j(m_{\mathsf{id_a}}) + \sigma_j(m_{\mathsf{id_b}})$ and $\rho_j(r_{\mathsf{id_c}}) = \rho_j(r_{\mathsf{id_a}}) + \rho_j(r_{\mathsf{id_b}})$ these shares would reconstruct to $m_{\mathsf{id_a}} + m_{\mathsf{id_b}}$ and $r_{\mathsf{id_a}} + r_{\mathsf{id_b}}$ respectively. Therefore,

$$g^{m_{\mathsf{id_c}}} h^{r_{\mathsf{id_c}}} = g^{m_{\mathsf{id_a}}+m_{\mathsf{id_b}}} h^{r_{\mathsf{id_a}}+r_{\mathsf{id_b}}} = g^{m_{\mathsf{id_a}}} h^{r_{\mathsf{id_a}}} \cdot g^{m_{\mathsf{id_b}}} h^{r_{\mathsf{id_b}}} = c_{\mathsf{id_a}} \cdot c_{\mathsf{id_b}} = c_{\mathsf{id_c}}$$

holds and the result will be accepted by Verify.

Preprocessing: Assume M sets of triples $(\alpha, \beta, \gamma), (\zeta, \eta, \theta)$, such that $\alpha \cdot \beta = \gamma$ were generated during preprocessing. Furthermore, assume we have commitments $c_\alpha = g^\alpha h^\zeta$,$c_\beta = g^\beta h^\eta$ and $c_\gamma = g^{\alpha\cdot\beta} h^\theta$. It follows that

$$\begin{aligned}
&e(c_\alpha, c_\beta) = e(g^\alpha h^\zeta, g^\beta h^\eta) = e(g^{\alpha+x\cdot\zeta}, g^{\beta+x\cdot\eta}) = g_T^{\alpha\cdot\beta+x(\alpha\eta+\beta\zeta)+x^2\cdot\zeta\cdot\eta} \\
&= g_T^{\alpha\cdot\beta+x(\alpha\eta+\beta\zeta-\theta+\theta)+x^2\cdot\zeta\cdot\eta} = g_T^{\alpha\cdot\beta+x\cdot\theta} \cdot g_T^{x(\alpha\eta+\beta\zeta-\theta+x\cdot\zeta\cdot\eta)} \\
&= e(g^{\alpha\cdot\beta+x\cdot\theta}, g) \cdot e(g^{\alpha\eta+\beta\zeta-\theta+x\cdot\zeta\cdot\eta}, h) = e(g^{\alpha\cdot\beta} h^\theta, g) \cdot e(g^{\alpha\eta+\beta\zeta-\theta} h^{\zeta\cdot\eta}, h) \\
&= e(c_\gamma, g) \cdot e(\mu, h)
\end{aligned}$$

holds and any honestly generated set will be accepted. Note that we can recover μ since a qualified subset of $2t$ shareholders participate in the protocol. Furthermore, note that if we use Feldman commitments we have $\mu = 1$ which is why the equation $e(g^\alpha, g^\beta) = g_T^{\alpha\cdot\beta} = e(g^{\alpha\cdot\beta}, g)$ holds.

Online Multiplication: If all shareholders followed Multiply correctly each shareholder will hold a share $\sigma_j(m_{\mathsf{id_c}})$ such that $\sigma_j(m_{\mathsf{id_c}}) = \sigma_j(\gamma) + \epsilon \cdot \sigma_j(m_{\mathsf{id_a}}) + \delta \cdot \sigma_j(m_{\mathsf{id_b}}) + \delta\epsilon$. Assume the preprocessing stage was performed correctly, then these shares would reconstruct to

$$\begin{aligned}
&\gamma_{\mathsf{id_c}} + \epsilon_{\mathsf{id_c}} m_{\mathsf{id_a}} + \delta m_{\mathsf{id_b}} - \epsilon_{\mathsf{id_c}}\delta_{\mathsf{id_c}} \\
&= \gamma_{\mathsf{id_c}} + (m_{\mathsf{id_b}} - \beta_{\mathsf{id_c}}) m_{\mathsf{id_a}} + (m_{\mathsf{id_a}} - \alpha_{\mathsf{id_c}}) m_{\mathsf{id_b}} - (m_{\mathsf{id_b}} - \beta_{\mathsf{id_c}})(m_{\mathsf{id_a}} - \alpha_{\mathsf{id_c}}) \\
&= m_{\mathsf{id_a}} m_{\mathsf{id_b}} + \gamma_{\mathsf{id_c}} - \alpha_{\mathsf{id_c}}\beta_{\mathsf{id_c}} = m_{\mathsf{id_a}} m_{\mathsf{id_b}}.
\end{aligned}$$

During MultAudit the auditor computes

$$c_j(m_{\mathsf{id_c}}) = c_j(\gamma_{\mathsf{id_c}}) \cdot c_j(m_{\mathsf{id_a}})^{\beta_{\mathsf{id_c}}} \cdot c_j(m_{\mathsf{id_b}})^\delta \cdot g^{-\delta\cdot\epsilon}$$

which can be opened to $m_{\mathsf{id_a}} m_{\mathsf{id_b}}$ using the computed random value.

Unforgeability Now we will show that if the algorithm Compute was not performed correctly, then the algorithm Verify will reject the result, i.e. *Unforgeability* is provided, even if all $j \in \mathcal{B}$ are malicious. For the proof we will assume that all shareholders are cooperating, as they could obviously simulate any honest party. Note that if the shareholders do not provide the commitments necessary for Audit it will output $\perp$ and guarantee that Verify returns '0'. Since the auditor follows the online phase by using the homomorphic properties of Feldman or Pedersen commitments respectively, a successful forgery would necessarily imply a forgery during the preprocessing stage. In the following we will first discuss unforgeability for the computationally hiding audit and afterwards for the unconditionally hiding audit.

Computationally Hiding Audit: According to the bilinearity of e, protocol MutlAudit only returns a commitment c^* if $e(c^*, g) = e(c(m_{\mathsf{id_a}}), c(m_{\mathsf{id_b}}))$ $= e(g^{m_{\mathsf{id_a}}}, g^{m_{\mathsf{id_b}}}) = e(g^{m_{\mathsf{id_a}} \cdot m_{\mathsf{id_b}}}, g)$ holds. As $|\mathbb{G}| = |\mathbb{G}_T|$ and e is non degenerate, the map $e(\cdot, g) : \mathbb{G} \to \mathbb{G}_T$ is an isomorphism which is why a result is only accepted when $c^* = g^{m_{\mathsf{id_a}} \cdot m_{\mathsf{id_b}}}$ is ensured.

Unconditionally Hiding Audit: Given a random linear function $\mathcal{L} : \mathbb{F}_q^M \to \mathbb{F}_q$ the shareholders are asked to publish the function evaluated on $\rho_j(\theta_1), \ldots, \rho_j(\theta_M)$ and $\sigma_j(\gamma_1), \ldots, \sigma_j(\gamma_M)$, where γ is supposed to be the product $\alpha \cdot \beta$. This implies a joint computation of $\mathcal{L}(\gamma_1, \ldots \gamma_M) + \hat{m}$ and $\mathcal{L}(\theta_1, \ldots \theta_M) + \hat{r}$. If these are valid opening values of $\hat{c} \cdot \tilde{\mathcal{L}}(c(\gamma_1), \ldots c(\gamma_M))$ this proves that they jointly know the opening values of $\tilde{\mathcal{L}}(c(\gamma_1), \ldots c(\gamma_M))$. Let $\Omega \subset \{1, \ldots, M\}$ indicate the set $\{c(\gamma_\omega)\}_{\omega \in \Omega}$ of all commitments the shareholders cannot open. Then $\tilde{\mathcal{L}}$ would induces a random linear function on $\{c(\gamma_\omega)\}_{\omega \in \Omega}$, whose result the shareholders are able to open. However, this can only happen with negligible probability. Thus, we can assume the shareholders jointly know openings of all $c(\gamma)$. Since all δ, ϵ are published and the auditors checked whether they are valid openings, the shareholders have to know openings to $c(\alpha) \cdot c(m_{\mathsf{id_a}})^{-1}$ for some input $m_{\mathsf{id_a}}$ (see the check performed in MulAudit). For symmetric reasons the same holds for $c(\beta)$. If the shareholders cannot open $c(\alpha), c(\beta)$ they can provide valid δ, ϵ only with negligible probability due to the bindingness of the commitment scheme used. From this it follows that the shareholders produced a tuple $c(\alpha), c(\beta), c(\gamma), \mu$ from which they know the openings $(\alpha, \zeta), (\beta, \eta), (\gamma, \theta)$ such that $e(c(\alpha), c(\beta)) = e(c(\gamma), \mu)$ holds.

In the following we show by contraction that if the shareholders produced an incorrect triple, i.e. $\alpha \cdot \beta \neq \gamma$, then they are able to break Assumption Ass.1. Having $(\alpha, \zeta), (\beta, \eta), (\gamma, \theta)$ it is possible to compute $g^{\frac{1}{\alpha \cdot \beta - \gamma}(\theta - \alpha \cdot \eta - \beta \cdot \zeta)} h^{-\zeta \cdot \eta} \mu^{\frac{1}{\alpha \cdot \beta - \gamma}}$. We see that

$$e(g^{\frac{1}{\alpha \cdot \beta - \gamma}(\theta - \alpha \cdot \eta - \beta \cdot \zeta)} h^{-\zeta \cdot \eta} \mu^{\frac{1}{\alpha \cdot \beta - \gamma}}, h) = (e(g^{\theta - \alpha \cdot \eta - \beta \cdot \zeta} h^{-\zeta \cdot \eta}, h) \cdot e(\mu, h))^{\frac{1}{\alpha \cdot \beta - \gamma}}$$
$$= g_T^{x((\theta - \alpha \cdot \eta - \beta \cdot \zeta) - x \cdot \zeta \cdot \eta) \frac{1}{\alpha \cdot \beta - \gamma}} \cdot e(\mu, h)^{\frac{1}{\alpha \cdot \beta - \gamma}} = g_T^{(\alpha \cdot \beta - \gamma) \cdot \frac{1}{\alpha \cdot \beta - \gamma}} = g_T = e(g, g)$$

Therefore $g^{\frac{1}{\alpha \cdot \beta - \gamma}(\theta - \alpha \cdot \eta - \beta \cdot \zeta)} h^{-\zeta \cdot \eta} \mu^{\frac{1}{\alpha \cdot \beta - \gamma}} = g^{\frac{1}{x}}$ and the shareholders would have solved the CIE problem.

5.2 Hidingness

For hidingness we assume that (Ass.6) if Feldman commitment are used the commitments are computationally hiding for the parameters chosen.

From the definition of Shamir secret sharing (see Section 3) it follows that any unqualified subset of $t' \leq t$ shareholders cannot reveal any information about the messages distributed. So all is left to show is that the data published for auditing does not reveal any additional information allowing to violate confidentiality.

Proposition 1. *Under Assumption Ass.6 the Feldman based audit scheme presented in Section 4 ensures computational hidingness of the input messages processed.*

Proof. Only Feldman commitments and unconditionally hiding sums, e.g. $\alpha - m_{\mathsf{id_a}}$, are published on the bulletin board. For a proof regarding the hidingness of Feldman commitments we refer to [Fel87]. Regarding the sums they are unconditionally hiding since α is random and not known to the auditor.

Proposition 2. *The Pedersen based audit scheme presented in Section 4 ensures unconditional confidentiality of the input messages processed.*

Proof. Only unconditionally hiding Pedersen commitments and unconditionally hiding sums, e.g. $\alpha - m_{\mathsf{id_a}}$, are published on the bulletin board. For a proof on the hidingness of Pedersen commitments we refer to [Ped91]. To complete the proof we will show that the commitments generated during our protocol do not reveal additional information about the messages processed that can be exploited to break the unconditional hidingness.

Additions: In AddAudit the public information processed and revealed consists of the tuple $(c(m_{\mathsf{id_a}}), c(m_{\mathsf{id_b}}), c^*)$=$(g^{m_{\mathsf{id_a}}} h^{r_{\mathsf{id_a}}}, g^{m_{\mathsf{id_b}}} h^{r_{\mathsf{id_b}}}, g^{m_{\mathsf{id_a}}+m_{\mathsf{id_b}}} h^{r_{\mathsf{id_a}}+r_{\mathsf{id_b}}})$. Since this tuple has the same distribution as $(C_1, C_2, C \cdot D)$ for random $C_1, C_2 \in \mathbb{G}$ it is unconditionally hiding.

Online Multiplications: In MultAudit the only public information processed and revealed consists of the tuple

$$(c(m_{\mathsf{id_a}}), c(m_{\mathsf{id_b}}), \delta, \tilde{\delta}, \epsilon, \tilde{\epsilon}, c(\alpha), c(\beta), c(\gamma), c^* = c(\gamma) c(m_{\mathsf{id_a}})^{\epsilon} c(m_{\mathsf{id_b}})^{\delta} g^{\delta\epsilon})$$

which has the same distribution as

$$(C_1, C_2, M_1, R_1, M_1, R_2, g^{M_1} h^{R_1} C_1^{-1}, g^{M_2} h^{R_2} C_2^{-1}, C_3, C_3 C_1^{M_1} C_2^{M_1} g^{M_1 \cdot M_2})$$

for random $C_1, C_2, C_3 \in \mathbb{G}, M_1, R_1, M_2, R_2 \in \mathbb{F}_q$.

Preprocessing: In PreAudit we reconstruct $c(\alpha), c(\beta), c(\gamma), \mu$ satisfying $e(c(\alpha), c(\beta)) = e(c(\gamma), g) \cdot e(\mu, h)$ Note that any random triple $C_1, C_2, C_3 \in \mathbb{G}$ uniquely define a μ such that $e(C_1, C_2) = e(C_3, g) \cdot e(\mu, h)$, namely the preimage of $e(C_1, C_2) \cdot e(C_3, g)^{-1}$ under the isomorphism $e(\cdot, h) : \mathbb{G} \to \mathbb{G}_T$. Therefore $(c(\alpha), c(\beta), c(\gamma), \mu)$ has the same distribution as (C_1, C_2, C_3, μ).

Result: In Result only $\tilde{m} = \hat{m} + \mathcal{L}(\gamma_1, \ldots, \gamma_M)$ and $\tilde{r} = \hat{r} + \mathcal{L}(\theta_1, \ldots, \theta_M)$ can be reconstructed. As $\hat{m}, \hat{r}$ were chosen uniformly at random this is unconditionally hiding for the γ_l.

5.3 Practicability and Efficiency

In this section we discuss several questions raising when implementing and running our scheme. We will concentrate here on the biliear maps and the random linear function $\mathcal{L}$, because all other components, e.g. the commitment schemes and the bulletin board, are fairly standard.

Bilinear Maps: Our scheme makes use of symmetric, i.e. type I pairings which can be implemented using supersingular elliptic curves over some prime field $\mathbb{F}_q$. This scheme naturally transforms to asymmetric, i. e. type II pairings by mapping $g_1, h_1 \in \mathbb{G}_1$ to $\mathbb{G}_2$ via the efficiently computable isomorphism $\Psi : \mathbb{G}_1 \to \mathbb{G}_2$. To use type III pairings we have to generate generators $g_1, h_1 = g_1^x$ of $\mathbb{G}_1$ and $g_2, h_1 = g_2^x$ of $\mathbb{G}_2$. In this case providing commitments (with the same randomness) they have to be computed in both $\mathbb{G}_1$ and $\mathbb{G}_2$. For a detailed description of type I, II, and III pairings and their respective differences we refer to [GPS08]. It remains to be evaluated, whether the use of fast asymmetric pairings, for example on *Barreto - Naehrig curves* [BN05], offset the additional cost of computing further commitments during the preprocessing.

Linear Function $\mathcal{L}$: The random linear function $\mathcal{L}$ given by the client can be provided in the form of a random seed K of a *pseudorandom number generator* $\mathcal{G}\{0,1\}^k \to \mathbb{F}_q^M$, such as the ones standardized in [BK12]. The client then only has to check that commitments $c(\alpha_l), c(\alpha_l), c(\alpha_l), \mu_l$ have been sent to the bulletin board and the auditor has to check whether $\mathcal{G}(K) = \mathcal{L}$.

Efficiency: This works seeks to minimize the shareholders computational overhead during the online phase. Our computationally hiding audit scheme demands no extra computations during the online phase and is therefore optimal. Our unconditionally hiding audit scheme is computationally dominated by evaluating the gates on both the actual messages and the randomness and has therefore an overhead of factor 2.

6 Conclusion

In this work we introduced a computationally hiding and an unconditionally hiding auditing procedure for computations on documents stored in distributed fashion. While the first one improves the state of the art with respect to efficiency the latter one is the first to provide unconditional hidingness for the documents processed. In addition, we proved that our solution is secure under well studied standard assumptions and gave arguments for practicability and efficiency. For future work we aim at a more thorough audit that not only detects an incorrect solution, but also identifies which shareholders deviate from the protocol.

Acknowledgments

This work has been co-funded by the DFG as part of project "Long-Term Secure Archiving" within the CRC 1119 CROSSING. In addition, it has received funding from the European Union's Horizon 2020 research and innovation program under Grant Agreement No 644962.

References

[AF07] Masayuki Abe and Serge Fehr. Perfect NIZK with adaptive soundness. In *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, pages 118–136, 2007.

[BBFR15] Michael Backes, Manuel Barbosa, Dario Fiore, and Raphael M. Reischuk. ADSNARK: nearly practical and privacy-preserving proofs on authenticated data. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 271–286, 2015.

[BCG+13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. Snarks for C: verifying program executions succinctly and in zero knowledge. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, pages 90–108, 2013.

[BCQ+11] Alysson Neves Bessani, Miguel Correia, Bruno Quaresma, Fernando André, and Paulo Sousa. DepSky: Dependable and Secure Storage in a Cloud-of-Clouds. In Christoph M. Kirsch and Gernot Heiser, editors, *European Conference on Computer Systems - EuroSys 2011*, pages 31–46. ACM, 2011.

[BCQ+13] Alysson Neves Bessani, Miguel Correia, Bruno Quaresma, Fernando André, and Paulo Sousa. DepSky: Dependable and Secure Storage in a Cloud-of-Clouds. *ACM Transactions on Storage - TOS*, 9(4):12, 2013.

[BCTV14] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Succinct non-interactive zero knowledge for a von neumann architecture. In *Proceedings of the 23rd USENIX Security Symposium, San Diego, CA, USA, August 20-22, 2014.*, pages 781–796, 2014.

[BDO14] Carsten Baum, Ivan Damgård, and Claudio Orlandi. Publicly auditable secure multi-party computation. In *Security and Cryptography for Networks - 9th International Conference, SCN 2014, Amalfi, Italy, September 3-5, 2014. Proceedings*, pages 175–196, 2014.

[Bea91] Donald Beaver. Efficient multiparty protocols using circuit randomization. In *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, pages 420–432, 1991.

[BF03] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003.

[BK12] Elaine B. Barker and John M. Kelsey. Sp 800-90a. recommendation for random number generation using deterministic random bit generators. Technical report, Gaithersburg, MD, United States, 2012.

[BN05] Paulo S. L. M. Barreto and Michael Naehrig. Pairing-friendly elliptic curves of prime order. In *Selected Areas in Cryptography, 12th International Workshop, SAC 2005, Kingston, ON, Canada, August 11-12, 2005, Revised Selected Papers*, pages 319–331, 2005.

[CDN01] Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. Multiparty computation from threshold homomorphic encryption. In *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, pages 280–299, 2001.

[CFH+15] Craig Costello, Cédric Fournet, Jon Howell, Markulf Kohlweiss, Benjamin Kreuter, Michael Naehrig, Bryan Parno, and Samee Zahur. Geppetto: Versatile verifiable computation. In *2015 IEEE Symposium on Security and Privacy, SP 2015, San Jose, CA, USA, May 17-21, 2015*, pages 253–270, 2015.

[Clo15] Cloud-of-clouds. DepSky by cloud-of-clouds. `http://cloud-of-clouds.github.io/depsky/`, 2015. Accessed: 2015-11-06.

[CRS05] David Chaum, Peter Y. A. Ryan, and Steve A. Schneider. A practical voter-verifiable election scheme. In *Computer Security - ESORICS 2005, 10th European Symposium on Research in Computer Security, Milan, Italy, September 12-14, 2005, Proceedings*, pages 118–139, 2005.

[DN07] Ivan Damgård and Jesper Buus Nielsen. Scalable and unconditionally secure multiparty computation. In *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, pages 572–590, 2007.

[DPSZ12] Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, pages 643–662, 2012.

[DvdGdSA12] Denise Demirel, Jeroen van de Graaf, and Roberto Samarone dos Santos Araújo. Improving helios with everlasting privacy towards the public. In *2012 Electronic Voting Technology Workshop / Workshop on Trustworthy Elections, EVT/WOTE '12, Bellevue, WA, USA, August 6-7, 2012*, 2012.

[Fel87] Paul Feldman. A practical scheme for non-interactive verifiable secret sharing. In *28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA, 27-29 October 1987*, pages 427–437, 1987.

[GPS08] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.

[HL08] James Heather and David Lundin. The append-only web bulletin board. In *Formal Aspects in Security and Trust, 5th International Workshop, FAST 2008, Malaga, Spain, October 9-10, 2008, Revised Selected Papers*, pages 242–256, 2008.

[LHS15] Thomas Lorünser, Andreas Happe, and Daniel Slamanig. ARCHISTAR: towards secure and robust cloud based data sharing. In *7th IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2015, Vancouver, BC, Canada, November 30 - Dec. 3, 2015*, pages 371–378, 2015.

[MN10] Tal Moran and Moni Naor. Split-ballot voting: Everlasting privacy with distributed trust. *ACM Trans. Inf. Syst. Secur.*, 13(2), 2010.

[Ped91] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings*, pages 129–140, 1991.

[PHGR13] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*, pages 238–252, 2013.

[Sha79] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[SS01] Ahmad-Reza Sadeghi and Michael Steiner. Assumptions related to discrete logarithms: Why subtleties make a real difference. In *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, pages 244–261, 2001.

[SV15] Berry Schoenmakers and Meilof Veeningen. Universally verifiable multiparty computation from threshold homomorphic cryptosystems. In *Applied Cryptography and Network Security - 13th International Conference, ACNS 2015, New York, NY, USA, June 2-5, 2015, Revised Selected Papers*, pages 3–22, 2015.