A Modular Treatment of Blind Signatures from Identification Schemes

Eduard Hauck Eike Kiltz Julian Loss

October 10, 2019

Ruhr University Bochum
{eduard.hauck,eike.kiltz,julian.loss}@rub.de

Abstract

We propose a modular security treatment of blind signatures derived from linear identification schemes in the random oracle model. To this end, we present a general framework that captures several well known schemes from the literature and allows to prove their security.

Our modular security reduction introduces a new security notion for identification schemes called One-More-Man In the Middle Security which we show equivalent to the classical One-More-Unforgeability notion for blind signatures.

We also propose a generalized version of the Forking Lemma due to Bellare and Neven (CCS 2006) and show how it can be used to greatly improve the understandability of the classical security proofs for blind signatures schemes by Pointcheval and Stern (Journal of Cryptology 2000).

Keywords: Blind Signatures

1 Introduction

Blind Signatures are a fundamental cryptographic building block. Informally, a blind signature scheme is an interactive protocol between a signer and an user in which the signer issues signatures on messages chosen by the user. There are two security requirements: *blindness* ensures that the signer cannot link a signature to the run of the protocol in which it was created and *one-more unforgeability* that the user cannot forge a new signature. Originally proposed by Chaum [15] as the basis of his e-cash system, blind signatures have since found numerous applications including e-voting [32] and anonymous credentials [16, 26, 12, 14, 13, 7, 5]. Despite a flurry of schemes having been published over the past three and a half decades, only a handful of works has considered blind signature schemes which are mutually efficient, instantiable from standard assumptions, and remain secure even when executed in an arbitrarily concurrent fashion. The notoriously difficult task of constructing such schemes was first tackled by Pointcheval and Stern [29]. Their groundbreaking work introduces the well-known *forking lemma* and shows how it can be applied to prove security of the Okamoto-Schnorr blind signature scheme [25] under the discrete logarithm assumption in the random oracle model (ROM) [10]. Their proof technique was subsequently employed to prove the security of further schemes [28, 33, 6]. Unfortunately, due to the complexity and subtlety of the argument in [29], these works present either only proof sketches [28] or follow the proof of [29] almost verbatim.

1.1 Our Contribution: A Modular Framework for Blind Signatures

In this work, we propose a general framework which shows how to derive a blind signature scheme from any *linear function family* (with certain properties), as recently introduced by Backendal et al. [4]. Whereas blindness can be proved directly, one-more unforgeability is proved in two modular steps. In the first step, one builds a linear identification scheme from the linear function family. One-more unforgeability of the blind signature scheme in the random oracle model is shown to be tightly equivalent to a new and natural security notion of the linear identification scheme, which we call *one-more man-in-the-middle* security. In

Name	Type	Definition of linear function F		Collision resistance
OS	Group	$\mathsf{F}: \mathbb{Z}_q^2 \to \mathbb{G},$	$(x_1, x_2) \mapsto g_1^{x_1} g_2^{x_2}$	**DLP**
OGQ	RSA	$\mathsf{F}: \mathbb{Z}_\lambda \times \mathbb{Z}_N^* \to \mathbb{Z}_N^*,$	$(x_1, x_2) \mapsto a^{x_1} x_2^\lambda$	**RSA**
FS	Factoring	$\mathsf{F}: (\mathbb{Z}_N^*)^k \to (\mathbb{Z}_N^*)^k,$	$(x_1, \ldots, x_k) \mapsto (x_1^2, \ldots, x_k^2)$	**FAC**

Table 1: Examples of linear function families. Group type functions are defined over $\mathbb{G}$ of prime order q with generators g_1, g_2. RSA and factoring type functions are defined over an RSA modulus $N = P \cdot Q$ s.t. $\gcd(\lambda, \varphi(N)) = \gcd(\lambda, N) = 1$ and $a \in \mathbb{Z}_N^*$.

the second, technically involved, step it is shown that the latter is implied by collision resistance of the linear function family. Our framework captures several important schemes from the literature including the Okamoto-Schnorr (OS) [25], the Okamoto-GQ (OGQ) [25], and (a slightly modified version of) the Fiat-Shamir (FS) [28] blind signature schemes and offers, for the first time, a complete and formal proof for some of them. We now provide some details of our contributions.

Linear Function Families and Identification Schemes. In the following, we denote with LF a family of *linear (hash) functions*. An identification scheme $\mathsf{ID} = \mathsf{ID}[\mathsf{LF}]$ is called a *linear identification scheme* [4] if it follows a certain homomorphic structure induced by a linear (i.e., homomorphic) function F from the family LF. For the purpose of building blind signatures, we will require that $\mathsf{ID}[\mathsf{LF}]$ be perfectly correct and that LF satisfy collision resistance. We will also assume some additional algebraic properties about the functions in LF that will be introduced in further detail in Section 3. Example instantiations of (collision resistant) linear function families can be derived from OS, OGQ, and FS (Table 1).

OMMIM Security of Linear Identification Schemes. We introduce a natural new security notion for (arbitrary, not necessarily linear) canonical identification schemes called *One-More Man-in-the-Middle* (OMMIM) security. Informally, ID is OMMIM-secure if it is infeasible to complete $Q_\mathsf{P} + 1$ (or more) runs of ID in the role of prover P after completing at most Q_P runs of ID in the role of verifier Ver. Note that OMMIM is weaker than standard Man-in-the-Middle security [19] (which we show to be unachievable for linear identification schemes) but stronger than impersonation against active attacks [17, 9].

Our first main result can be stated as follows:

Theorem 4.3 (informal). If LF is collision resistant, then $\mathsf{ID}[\mathsf{LF}]$ is OMMIM-secure.

Our proof is based on a new Subset Forking Lemma that generalizes the one by Bellare and Neven [8] and contains many technical ingredients from [29] who prove the security of the Okamoto-Schnorr Blind Signature scheme. Unfortunately, the security bound from Theorem 4.3 is only meaningful if $Q_\mathsf{V}^{Q_\mathsf{P}+1} \le |\mathcal{C}| =: q$, where Q_V refers to the (potentially large) number of sessions with the verifier and challenge set $\mathcal{C}$ is a parameter of the identification scheme. We next show in Theorem 4.4 that a natural generalization of Schnorr's ROS-problem [34] to linear functions can be used to break the OMMIM security of $\mathsf{ID}[\mathsf{LF}]$. The ROS-problem (for the relevant parameters) becomes information theoretically hard when $Q_\mathsf{V}^{Q_\mathsf{P}+1} \le q$. For all other cases, it can be solved in sub-exponential time $(Q_\mathsf{Ch} + 1)\, 2^{\sqrt{\log q}/(1+\log(Q_\mathsf{Ch}+1))}$ using Wagner's k-List algorithm [35]. Our ROS-based attack works whenever $\mathcal{C}$ is a finite field, which is the case for OS and OGQ.

Canonical Blind Signature Schemes. We introduce the notion of *canonical blind signature schemes* (BS), which are three-move blind signature schemes of a specific form. In terms of security, we define *blindness* and *one-more unforgeability* (OMUF). Intuitively, OMUF states that the adversary can not produce more valid message-signatures pairs than it has completed successful sessions with the signer. (Note that each such session yields a valid message-signature pair.) Here we consider a natural and strong version of OMUF in which abandoned session with the signer (i.e., sessions that are started but never completed) are not counted as a successful sessions with the signer, as they do not yield a valid message-signature pair. We propose a general compiler to convert any linear identification scheme $\mathsf{ID}[\mathsf{LF}]$ and a hash function H into a canonical blind signature scheme $\mathsf{BS}[\mathsf{LF}, \mathsf{H}]$. Our second main result can be stated as follows:

Theorem 5.7 (informal). OMUF security of $\mathsf{BS}[\mathsf{LF}, \mathsf{H}]$ is tightly equivalent to OMMIM security of $\mathsf{ID}[\mathsf{LF}]$ in the random oracle model.

Theorem 5.8 (informal). $\mathsf{BS}[\mathsf{LF}, \mathsf{H}]$ is perfectly blind.

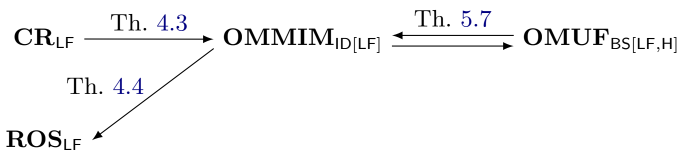

Figure 1: Overview of our modular security analysis for $\mathsf{BS}[\mathsf{LF},\mathsf{H}]$. The arrows denote security implications.

Figure 1.1 summarizes our modular security analysis of $\mathsf{BS}[\mathsf{LF},\mathsf{H}]$. Combining our main theorems, we obtain security proofs for the OS, OGQ, and FS blind signature schemes. Here, the number of random oracle queries Q_{H} corresponds to the number Q_{Ch} of open sessions with the verifier, whereas the number Q_{S} of signing sessions corresponds to the number of sessions Q_{P} with the prover. Hence, OMUF security of $\mathsf{BS}[\mathsf{LF},\mathsf{H}]$ is only guaranteed if $Q_{\mathsf{H}}^{Q_{\mathsf{S}}+1} \ll q$, i.e., for polylogarithmically parallel signing sessions Q_{S}. Our ROS-based attack demonstrates that this restriction is required.

1.2 Technical details

We now give an intuition for the proof of Theorem 4.3. Roughly, it states that one can reduce the OMMIM security of $\mathsf{ID}[\mathsf{LF}]$ from the problem of finding a non-trivial collision with respect to the linear function family LF. Our proof follows the ideas of Pointcheval and Stern [29], but uses as a key ingredient a novel forking lemma, which enables us to present the proof in [29] in a much more clean and general fashion. The main idea behind our reduction is to run the adversary M against OMMIM security twice, where the instance I and randomness ω in the second run are kept the same, and part of the oracle answers, denoted $\boldsymbol{h}, \boldsymbol{h'}$, are re-sampled uniformly. In this way, we hope to obtain from M two distinct values $\hat{\chi}, \hat{\chi}'$ which yield a collision with respect to LF. The main challenge in our setting is that $\hat{\chi}$ and $\hat{\chi}'$ depend on the internal state of M. To show that $\hat{\chi} \neq \hat{\chi}'$ with high probability, one requires an intricate argument that heavily builds upon a generalized version of Bellare and Neven's Forking Lemma [8]. Our lemma is tailored toward the ideas of the proof in [29] and allows for a more fine-grained replay strategy than the version of [8]. More precisely, our version of the forking lemma considers not only the probability of successfully running an algorithm twice with the same instance I, randomness ω, and (partially distinct) oracle answers $\boldsymbol{h}, \boldsymbol{h'}$, but also allows to analyze in more detail the properties of the triples $(I, \omega, \boldsymbol{h}), (I, \omega, \boldsymbol{h'})$.

1.3 Blind Signatures from Lattices?

We remark that our proof requires linear functions with perfect correctness. This leaves open the question of whether our framework can be extended to cover also the lattice-based identification scheme due to Lyubashevsky [23] and the resulting blind signature scheme due to Rückert [33]. At a technical level, imperfect correctness causes a problem in the proof of Theorem 5.7 which relates the OMMIM-security of $\mathsf{ID}[\mathsf{LF}]$ to OMUF-security of $\mathsf{BS}[\mathsf{LF},\mathsf{H}]$. If the adversary manages to abort even a single run of $\mathsf{BS}[\mathsf{LF},\mathsf{H}]$ in the simulated OMUF experiment, our reduction fails at simulating the necessary amount of completed runs of $\mathsf{BS}[\mathsf{LF},\mathsf{H}]$ to the adversary. This subtlety in the proof arises from the fact that in the OMMIM experiment, there is no way of telling whether a run of $\mathsf{ID}[\mathsf{LF}]$ with the adversary in the role of the verifier was completed. On the other hand, in $\mathsf{BS}[\mathsf{LF},\mathsf{H}]$, the user can prove to the signer that it obtained an invalid signature for a particular run of the protocol and hence force a restart. We leave it as an open problem to adapt our framework to linear functions with correctness errors.

1.4 History

This paper appeared at EUROCRYPT 2019 [20], this is the full version. Since publishing our paper, it underwent some changes. Most notably, as kindly pointed out to us by Jesse Selover, a previous version of our framework incorrectly stated that the algebraic structure defined by the Okamoto-Guillou-Quisquater and Fiat-Shamir linear function families defined modules rather than pseudo modules. Unfortunately, both our proofs of correctness and blindness relied on the distributive law over the resulting modules. To overcome this issue, we introduced in this version a further algorithm to the definition of a linear function family which we have called the *distributor function*, because it effectively acts as an error correction term whenever distributivity was previously required. In the process, we have also added a much more detailed description of these two linear function families. Further changes in the latest version include some simplifications and minor additions to some of our proofs. Most notably, the ROS problem is now parametrised by its dimension ℓ for convenience, whereas in a previous version, the adversary was free to choose the dimension of the equation system.

2 Preliminaries and Notation

In this section, we introduce notation, basic definitions, and recurring proof tools. We begin by defining (mathematical) notations for sets, vectors, sampling processes, and more. We also introduce security games [11] and the Random Oracle Model (ROM) [10]. Finally we state some technical lemmas that we will use in our analysis.

2.1 Notation

Sets and Vectors. For $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, \ldots, n\}$. We use bold-faced, lower case letters $\boldsymbol{h}$ to denote a vector of elements and denote the length of $\boldsymbol{h}$ as $|\boldsymbol{h}|$. For $j \geq 1$, we write $\boldsymbol{h}_j$ to denote the j-th element of $\boldsymbol{h}$ and we write $\boldsymbol{h}_{[j]}$ to refer to the first j entries of $\boldsymbol{h}$, i.e., the elements $\boldsymbol{h}_1, ..., \boldsymbol{h}_j$. We use boldface, upper case letters $\mathbf{A}$ to denote matrices. We denote the i-th row of $\mathbf{A}$ as $\mathbf{A}_i$ and the j-th entry of $\mathbf{A}_i$ as $\mathbf{A}_{i,j}$.

Sampling from Sets. We write $h \xleftarrow{\$} \mathcal{S}$ to denote that the variable h is uniformly sampled from the set $\mathcal{S}$. For $1 \leq j \leq Q$ and $\boldsymbol{g} \in \mathcal{S}^{j-1}$, we write $\boldsymbol{h'} \xleftarrow{\$} \mathcal{S}^Q|\boldsymbol{g}$ to denote that the vector $\boldsymbol{h'}$ is uniformly sampled from $\mathcal{S}^Q$, conditioned on $\boldsymbol{h'}_{[j-1]} = \boldsymbol{g}$. This sampling process can be implemented by copying vector $\boldsymbol{g}$ into the first $j-1$ entries of $\boldsymbol{h'}$ and next sampling the remaining $Q - j + 1$ entries of $\boldsymbol{h}$, i.e., $\boldsymbol{h}'_j, \ldots, \boldsymbol{h}'_Q \xleftarrow{\$} \mathcal{S}^{Q-j+1}$.)

Modular Arithmetic. Let $N \in \mathbb{Z}, N > 0$. Throughout this work, we write $\mathbb{Z}_N$ to denote the set of integers $\{0, \ldots, N-1\}$. We write $\mathbb{Z}_N^*$ to denote the set of integers $a \in \mathbb{Z}_N$ s.t. $\gcd(a, N) = 1$. For convenience, we will write $a \equiv_N b$ to denote that $a = b \bmod N$. We also sometimes denote the *remainder of* $a \in \mathbb{Z}$ *by division of* N as $[a]_N$.

Algorithms. We use uppercase, serif-free letters A, B to denote algorithms. Unless otherwise stated, algorithms are probabilistic and we write $(y_1, \ldots) \xleftarrow{\$} \mathsf{A}(x_1, \ldots)$ to denote that A returns $(y_1, \ldots)$ when run on input $(x_1, \ldots)$. We write A^B to denote that A has oracle access to B during its execution. To make the randomness ω of an algorithm A on input x explicit, we write $\mathsf{A}(x; \omega)$. Note that in this notation, A is deterministic. For a randomised algorithm A, we use the notation $y \in \mathsf{A}(x)$ to denote that y is a possible output of A on input x.

Security Games. We use standard code-based security games [11]. A *game* $\mathbf{G}$ is a probability experiment in which an adversary A interacts with an implicit challenger that answers oracle queries issued by A. $\mathbf{G}$ has one *main procedure* and an arbitrary amount of additional *oracle procedures* which describe how these oracle queries are answered. To distinguish game-related oracle procedures from algorithmic procedures more clearly, we denote the former using monospaced font, e.g., `Oracle`. We denote the (binary) output b of game $\mathbf{G}$ between a challenger and an adversary A as $\mathbf{G}^\mathsf{A} \Rightarrow b$. A is said to *win* $\mathbf{G}$ if $\mathbf{G}^\mathsf{A} \Rightarrow 1$. Unless otherwise stated, the randomness in the probability term $\Pr[\mathbf{G}^\mathsf{A} \Rightarrow 1]$ is over all the random coins in game $\mathbf{G}$.

The Random Oracle Model. A common approach to analyse the security of cryptographic schemes which internally use a hash function H is the random oracle model [10]. In this model, a hash function H is treated as an idealised random function. Concretely, H is modelled as an oracle H with the following properties. The oracle internally keeps a list H for bookkeeping purposes. Initially, all entries of H are set to $\perp$. On input x from the domain of H, the oracle first checks whether $H[x] \neq \perp$, i.e., whether it has already been defined via a prior query on the value x. If so, it returns $H[x]$. Otherwise, it sets $H[x]$ to a uniformly random value in the codomain of H and then returns $H[x]$. We write Q_H to denote the maximal number of allowed hash queries, i.e., the number of times that the adversary may call the oracle H. In this manner, Q_{H} becomes a parameter in our security notions.

Security Paramater. Throughout this work, we denote as κ the *security parameter*. We slightly abuse notation and refer to the security parameter's unary representation 1^κ as the security parameter indiscriminately.

2.2 Useful Lemmas

Splitting Lemmas. We first recall the well known splitting lemma [29]. This lemma is also sometimes referred to as the 'heavy row' lemma in the literature (e.g. [3]).

Lemma 2.1 (Splitting Lemma). *Let $\mathcal{X}, \mathcal{Y}$ be sets of finite size and $\mathcal{B} \subset \mathcal{X} \times \mathcal{Y}$ be such that*

$$\Pr_{(x,y) \xleftarrow{\$} \mathcal{X} \times \mathcal{Y}}[(x,y) \in \mathcal{B}] := \varepsilon.$$

For any $\alpha \leq \varepsilon$, define

$$\mathcal{B}_\alpha = \big\{(x,y) \in \mathcal{X} \times \mathcal{Y} \mid \Pr_{y' \xleftarrow{\$} \mathcal{Y}}[(x,y') \in \mathcal{B}] \geq \varepsilon - \alpha\big\}.$$

Then the following statements hold for any $\alpha \leq \varepsilon$:

(i) $\Pr_{(x,y) \xleftarrow{\$} \mathcal{X} \times \mathcal{Y}}[(x,y) \in \mathcal{B}_\alpha] \geq \alpha$

(ii) $\forall (x,y) \in \mathcal{B}_\alpha\colon \Pr_{y' \xleftarrow{\$} \mathcal{Y}}[(x,y') \in \mathcal{B}] \geq \varepsilon - \alpha$

(iii) $\Pr_{(x,y) \xleftarrow{\$} \mathcal{B}}[(x,y) \in \mathcal{B}_\alpha] = \Pr_{(x,y) \xleftarrow{\$} \mathcal{X} \times \mathcal{Y}}[(x,y) \in \mathcal{B}_\alpha \mid (x,y) \in \mathcal{B}] \geq \alpha/\varepsilon$

We refer to [29] for a proof of Theorem 2.1. For our purposes, the following version of Theorem 2.1 will actually be more convenient.

Lemma 2.2 (Subset Splitting Lemma). *Let sets $\mathcal{X}, \mathcal{Y}, \mathcal{B}, \mathcal{B}_\alpha$ be as in Theorem 2.1. Then*

$$\Pr_{y,y' \xleftarrow{\$} \mathcal{Y}, x \xleftarrow{\$} \mathcal{X}}[(x,y') \in \mathcal{B} \land (x,y) \in \mathcal{B}] \geq (\varepsilon - \alpha) \cdot \alpha.$$

Proof. For the conditional probability, we have that

$$\begin{aligned}
&\Pr_{y,y' \xleftarrow{\$} \mathcal{Y}, x \xleftarrow{\$} \mathcal{X}}[(x,y') \in \mathcal{B} \mid (x,y) \in \mathcal{B}] \\
&\geq \Pr_{y,y' \xleftarrow{\$} \mathcal{Y}, x \xleftarrow{\$} \mathcal{X}}[(x,y') \in \mathcal{B} \land (x,y) \in \mathcal{B}_\alpha \mid (x,y) \in \mathcal{B}] \\
&= \Pr_{y,y' \xleftarrow{\$} \mathcal{Y}, x \xleftarrow{\$} \mathcal{X}}[(x,y') \in \mathcal{B} \mid (x,y) \in \mathcal{B}_\alpha \cap \mathcal{B}] \cdot \Pr_{(x,y) \xleftarrow{\$} \mathcal{X} \times \mathcal{Y}}[(x,y) \in \mathcal{B}_\alpha \mid (x,y) \in \mathcal{B}] \\
&\geq (\varepsilon - \alpha) \cdot \frac{\alpha}{\varepsilon},
\end{aligned}$$

where the last inequality follows from (ii) and (iii) in Theorem 2.1. We conclude the proof by

$$\begin{aligned}
&\Pr_{y,y' \xleftarrow{\$} \mathcal{Y}, x \xleftarrow{\$} \mathcal{X}}[(x,y') \in \mathcal{B} \land (x,y) \in \mathcal{B}] \\
&= \Pr_{y,y' \xleftarrow{\$} \mathcal{Y}, x \xleftarrow{\$} \mathcal{X}}[(x,y') \in \mathcal{B} \mid (x,y) \in \mathcal{B}] \cdot \Pr_{(x,y) \xleftarrow{\$} \mathcal{X} \times \mathcal{Y}}[(x,y) \in \mathcal{B}] \\
&\geq (\varepsilon - \alpha) \cdot \frac{\alpha}{\varepsilon} \cdot \varepsilon = (\varepsilon - \alpha) \cdot \alpha.
\end{aligned}$$

■

(Simplified) Jensen Inequality. The following inequality can be inferred from Jensen's inequality [21] and was proven in [2].

Lemma 2.3 *Let $q \in \mathbb{N}, q > 0$ and let $x_1, \ldots, x_q \geq 0$ be real numbers. Then*

$$\sum_{i=1}^{q} x_i^2 \geq \frac{1}{q} \cdot \left(\sum_{i=1}^{q} x_i \right)^2 . \tag{1}$$

3 Linear Functions

In this section, we introduce modules, pseudomodules and linear function families. We give instantiations of linear function families in Section 8.

Basic Algebra. We say that sets $\mathcal{S}$ and $\mathcal{M}$ *form a module*, if $\mathcal{S}$ is a ring with multiplicative identity element $1_{\mathcal{S}}$ and $\mathcal{M}$ is an additive Abelian group, and there exists a mapping $\cdot : \mathcal{S} \times \mathcal{M} \to \mathcal{M}$, s.t. for all $r, s \in \mathcal{S}$ and $x, y \in \mathcal{M}$ we have (i) $r \cdot (x+y) = r \cdot x + r \cdot y$; (ii) $(r+s) \cdot x = r \cdot x + s \cdot x$; (iii) $(rs) \cdot x = r \cdot (s \cdot x)$; and (iv) $1_S \cdot x = x$. In an analogous fashion, we say that $\mathcal{S}$ and $\mathcal{M}$ *form a pseudo module*, if $\mathcal{S}, \mathcal{M}$ are additive Abelian groups and there exists a mapping $\cdot : \mathcal{S} \times \mathcal{M} \to \mathcal{M}$, s.t. for all $r \in \mathcal{S}$ and $x, y \in \mathcal{M}$ we have $r \cdot (x + y) = r \cdot x + r \cdot y$. Clearly, if $\mathcal{S}$ and $\mathcal{M}$ form a module, then they also form a pseudo module. Observe also that we denote an application of the operation $\cdot$ in an *inline* fashion, i.e., we write $r \cdot x$ instead of $\cdot(r, x)$. When convenient, we also sometimes disregard the order of arguments when applying $\cdot$, i.e, we sometimes write $r \cdot x = x \cdot r, x \in \mathcal{D}, r \in \mathcal{S}$.

3.1 Syntax of Linear Function Families

The notion of linear function families was introduced in [4]. We adapt their definitions for this work, so as to include some additional properties that we require.

Definition 3.1 (Linear Function Family). A *linear function family* LF is a tuple of algorithms $(\mathsf{PGen}, \mathsf{F}, \Psi)$ defined as follows.

- The randomized *parameter generation algorithm* PGen takes as input the security parameter 1^κ and returns system parameters *par* which implicitly define the sets $\mathcal{S}(par), \mathcal{D}(par)$ and $\mathcal{R}(par)$. $\mathcal{S}(par)$ is a set of scalars such that $\mathcal{D}(par)$ and $\mathcal{R}(par)$ form *pseudo modules* over $\mathcal{S}(par)$ and $|\mathcal{S}(par)| \geq 2^{2\kappa}$.
- The deterministic *evaluation function* F takes as input system parameters *par* and a point $x \in \mathcal{D}(par)$. It returns y, where $y \in \mathcal{R}(par)$. For all $par \in \mathsf{PGen}(1^\kappa)$, we require that the following properties are satisfied:
  - $\mathsf{F}(par, \cdot)$ is a *pseudo module homomorphism*: For all $x, y \in \mathcal{D}(par)$ and $s \in \mathcal{S}(par)$, we have that
  $$\mathsf{F}(par, s \cdot x + y) = s \cdot \mathsf{F}(par, x) + \mathsf{F}(par, y).$$
  - $\mathsf{F}(par, \cdot)$ has a *pseudo torsion-free element from the kernel*: There exist $z^* \in \mathcal{D}(par) \setminus \{0\}$ such that (i) $\mathsf{F}(par, z^*) = 0$; and (ii) for all $s, s' \in \mathcal{S}(par), s \neq s'$, we have $s \cdot z^* \neq s' \cdot z^*$. Note that this implies that $\mathsf{F}(par, \cdot)$ is a many-to-one mapping.
- The determinstic *distributor function* Ψ takes as input system parameters *par*, a point $y \in \mathcal{R}(par)$, and points $s, s' \in \mathcal{S}(par)$. It outputs a point $x \in \mathcal{D}(par)$. For all $par \in \mathsf{PGen}(1^\kappa)$ and points $x \in \mathcal{D}(par), s, s' \in \mathcal{S}(par)$, we require that $\Psi(par, \cdot)$ satisfy
  $$(s + s') \cdot \mathsf{F}(par, x) = s \cdot \mathsf{F}(par, x) + s' \cdot \mathsf{F}(par, x) + \mathsf{F}(\Psi(par, \mathsf{F}(par, x), s, s')).$$

Intuitively, the distributor function $\Psi(par, \cdot)$ can be thought of as a *correction term* that allows to treat a pseudo module as if the operation $+$ over $\mathcal{S}(par)$ distributes over $\mathcal{R}(par)$. In particular, the distributor function becomes the trivial zero function whenever $\mathcal{D}(par)$ and $\mathcal{R}(par)$ form full-fledged modules with $\mathcal{S}(par)$. Observe that the commutativity of the operation $+$ on $\mathcal{S}(par)$ implies that $\forall x \in \mathcal{D}(par), s, s' \in \mathcal{S}(par)$, we have that $\Psi(par, \mathsf{F}(par, x), s, s') = \Psi(par, \mathsf{F}(par, x), s', s)$. In the following, we simplify our notation by writing $\mathcal{S} = \mathcal{S}(par), \mathcal{D} = \mathcal{D}(par), \mathcal{R} = \mathcal{R}(par)$, as well as $\mathsf{F}(\cdot) = \mathsf{F}(par, \cdot), \Psi(\cdot) = \Psi(par, \cdot)$.

Game $\mathbf{CR}_{\mathsf{LF}}$:

$par \xleftarrow{\$} \mathsf{PGen}(1^\kappa)$
$(x_1, x_2) \xleftarrow{\$} \mathsf{A}(par)$
If $\mathsf{F}(x_1) = \mathsf{F}(x_2) \wedge x_1 \neq x_2$: Return 1
Return 0

Figure 2: Game $\mathbf{CR}_{\mathsf{LF}}$ with adversary A.

Game ℓ-$\mathbf{ROS}_{\mathsf{LF}}$:

$par \xleftarrow{\$} \mathsf{PGen}(1^\kappa)$
$\left(\boldsymbol{c} \in \mathcal{S}^{\ell+1}, \mathbf{A} \in \mathcal{S}^{(\ell+1)\times(\ell+1)}\right) \xleftarrow{\$} \mathsf{A}^{\mathsf{H}}(par)$
If $(\boldsymbol{c}_{\ell+1} = -1) \wedge (\mathbf{A}\boldsymbol{c} = 0) \wedge (\forall i, j \in [\ell+1] : \mathsf{H}(\mathbf{A}_{i,1}, \ldots, \mathbf{A}_{i,\ell}) = \mathbf{A}_{i,\ell+1}) \wedge (\mathbf{A}_i \neq \mathbf{A}_j)$: Return 1
Return 0

Figure 3: Game ℓ-$\mathbf{ROS}_{\mathsf{LF}}$ with adversary A. $\mathsf{H}\colon \{0,1\}^* \to \mathcal{S}$ is a random oracle.

3.2 Security Properties of Linear Function Families

We now define two security properties of a linear function family (collision resistance and ROS security) which will play a significant role in the subsequent sections. For the linear function family LF, we define its *collision resistance* via game $\mathbf{CR}_{\mathsf{LF}}$ which is depicted in Figure 2. We define A's advantage in $\mathbf{CR}_{\mathsf{LF}}$ as $\mathbf{Adv}^{\mathsf{A}}_{\mathsf{LF}} := \Pr[\mathbf{CR}^{\mathsf{A}}_{\mathsf{LF}} \Rightarrow 1]$ and denote its running time as $\mathbf{Time}^{\mathbf{CR}}_{\mathsf{LF}}(\mathsf{A})$.

Definition 3.2 (Collision Resistance). Let LF be a linear function family. LF is said to be (ε, t)-*collision resistant* if for all adversaries A satisfying $\mathbf{Time}^{\mathbf{CR}}_{\mathsf{LF}}(\mathsf{A}) \leq t$, we have that $\mathbf{Adv}^{\mathbf{CR}}_{\mathsf{LF}}(\mathsf{A}) \leq \varepsilon$. We say that A *breaks* (ε, t)-*collision resistance of* LF if $\mathbf{Time}^{\mathbf{CR}}_{\mathsf{LF}}(\mathsf{A}) \leq t$ and $\mathbf{Adv}^{\mathbf{CR}}_{\mathsf{LF}}(\mathsf{A}) > \varepsilon$.

Next, we define hardness of the *ROS problem* associated with linear function family LF. The ROS (Random inhomogenities in an Overdetermined, Solvable system of linear equations) problem was introduced by Schnorr [34] (also in the context of blind signatures). For the remainder of this chapter, we consider, for any choice of $par \in \mathsf{PGen}(1^\kappa)$, the hash function $\mathsf{H}(par, \cdot)\colon \{0,1\}^* \to \mathcal{C}(par)$. As above, we will (for simplicity of notation) henceforth omit par from H's input and simply write $\mathsf{H}(x) := \mathsf{H}(par, x)$. We now generalise Schnorr's formulation of the ROS problem to linear function families. For a linear function family LF and positive integer ℓ, the game ℓ-$\mathbf{ROS}_{\mathsf{LF}}$ is defined via Figure 3. The advantage of adversary A in ℓ-$\mathbf{ROS}_{\mathsf{LF}}$ is defined as $\mathbf{Adv}^{\ell\text{-}\mathbf{ROS}}_{\mathsf{LF}}(\mathsf{A}) := \Pr[\ell\text{-}\mathbf{ROS}^{\mathsf{A}}_{\mathsf{LF}} \Rightarrow 1]$ and its running time is denoted as $\mathbf{Time}^{\ell\text{-}\mathbf{ROS}}_{\mathsf{LF}}(\mathsf{A})$.

Definition 3.3 (ℓ-**ROS** Hardness). Let $\ell \in \mathbb{N}, \ell > 0$ and let LF be a linear function family. ℓ-$\mathbf{ROS}_{\mathsf{LF}}$ is said to be $(\varepsilon, t, Q_{\mathsf{H}})$-*hard in the random oracle model* if for all adversaries A satisfying $\mathbf{Time}^{\ell\text{-}\mathbf{ROS}}_{\mathsf{LF}}(\mathsf{A}) \leq t$ and making at most Q_{H} queries to H, we have that $\mathbf{Adv}^{\ell\text{-}\mathbf{ROS}}_{\mathsf{LF}}(\mathsf{A}) \leq \varepsilon$. We say that A $(\varepsilon, t, Q_{\mathsf{H}})$-*breaks* ℓ-$\mathbf{ROS}_{\mathsf{LF}}$ *in the random oracle model* if $\mathbf{Time}^{\ell\text{-}\mathbf{ROS}}_{\mathsf{LF}}(\mathsf{A}) \leq t$, A makes at most Q_{H} queries to H, and $\mathbf{Adv}^{\ell\text{-}\mathbf{ROS}}_{\mathsf{LF}}(\mathsf{A}) > \varepsilon$.

The following Lemma summarizes the known hardness results for the ℓ-ROS-Problem for the specific case in which $\mathcal{S}$ is a field of prime order and $\mathcal{D}$ and $\mathcal{R}$ form modules with $\mathcal{S}$.

Lemma 3.4 ([34, 35, 24]). *Let* LF *be a linear function family for which* $\mathcal{S}$ *is a field of prime order* $|\mathcal{S}| = O(2^{2\kappa})$ *and* $\mathcal{D}, \mathcal{R}$ *form* $\mathcal{S}$*-modules. For every* t, ℓ-$\mathbf{ROS}_{\mathsf{LF}}$ *is* $(\varepsilon = Q_{\mathsf{H}}^{\ell+1}/2^{2\kappa}, t, Q_{\mathsf{H}})$*-hard in the random oracle model. Conversely,* ℓ-$\mathbf{ROS}_{\mathsf{LF}}$ *is not* $(1/4, t, Q_{\mathsf{H}})$*-hard in the random oracle model for* $t \approx Q_{\mathsf{H}} = O\left((\ell+1) \cdot 2^{(2\kappa/(1+\log(\ell+1)))}\right)$.

4 Canonical Identification Schemes

In this section, we introduce the syntax and security of what we call *canonical identification schemes*. We first give the basic definitions for syntax and security. Then we give a generic construction that gives a canonical identification scheme $\mathsf{ID}[\mathsf{LF}]$ from any linear function family LF.

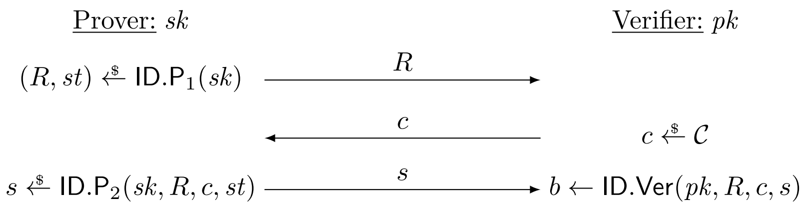

Figure 4: A canonical three-move identification scheme $\mathsf{ID} = (\mathsf{ID.PG}, \mathsf{ID.KG}, \mathsf{ID.P}_1, \mathsf{ID.P}_2, \mathsf{ID.Ver})$ and its transcript (R, c, s).

4.1 Syntax and Security

We now recall the definition of define canonical (three-move) identification schemes [1] and discuss their security notions.

Definition 4.1 (Canonical Three-Move Identification Scheme). A *canonical three-move identification scheme* is a tuple of algorithms $\mathsf{ID} = (\mathsf{ID.PG}, \mathsf{ID.KG}, \mathsf{ID.P} = (\mathsf{ID.P}_1, \mathsf{ID.P}_2), \mathsf{ID.Ver})$.

- The randomised *parameter generation algorithm* $\mathsf{ID.PG}$ takes as input the security parameter 1^κ and returns system parameters *par*.
- The randomised *key generation algorithm* $\mathsf{ID.KG}$ takes as input system parameters *par* and returns a public/secret key pair (pk, sk). We assume that pk implicitly defines a *challenge space* $\mathcal{C} := \mathcal{C}(pk)$ and that pk is distributed (and hence known) to all parties.
- The *prover algorithm* $\mathsf{ID.P}$ is split into two algorithms, i.e., $\mathsf{ID.P} := (\mathsf{ID.P}_1, \mathsf{ID.P}_2)$, where:
  - The randomised algorithm $\mathsf{ID.P}_1$ takes as input a secret key sk and returns a commitment R and a state st.
  - The deterministic algorithm $\mathsf{ID.P}_2$ takes as input a secret key sk, a commitment R, a challenge c, and a state st. It returns a response s.
- The deterministic *verification algorithm* $\mathsf{ID.Ver}$ takes as input a public key pk, a commitment R, a challenge c, and a response s. It returns 1 (accept) or 0 (reject).

We remark that modeling $\mathsf{ID.P}_2$ as a deterministic algorithm is w.l.o.g. since randomness can be transmitted through the state st. Figure 4 shows the interaction between algorithms $\mathsf{ID.P}_1, \mathsf{ID.P}_2$, and $\mathsf{ID.Ver}$.

Standard security notions for canonical identification schemes include impersonation security against passive and active attacks, and Man-in-the-Middle security [1, 9]. We now introduce a new security notion called *One-More Man-in-the-Middle* security. The One-More Man-in-the-Middle (**OMMIM**) security experiment for an identification scheme ID and an adversary A is defined in Figure 5. Adversary A simultaneously plays against a prover (modeled through oracles $\mathtt{P}_1$ and $\mathtt{P}_2$) and a verifier (modeled through oracles $\mathtt{V}_1$ and $\mathtt{V}_2$). Session identifiers *pSid* and *vSid* are used to model an interaction with the prover and the verifier, respectively. A call to $\mathtt{P}_1$ returns a new prover session identifier *pSid* and sets flag $\mathbf{pSess}_{pSid}$ to $\mathtt{open}$. A call to $\mathtt{P}_2(pSid, \cdot)$ with the same *pSid* sets the flag $\mathbf{pSess}_{pSid}$ to $\mathtt{closed}$. Similarly, a call to $\mathtt{V}_1$ returns a new verifier session identifier *vSid* and sets flag $\mathbf{vSess}_{vSid}$ to $\mathtt{open}$. A call to $\mathtt{V}_2(vSid, \cdot)$ with the same *pSid* sets the flag $\mathbf{vSess}_{vSid}$ to $\mathtt{closed}$. A closed verifier session *vSid* is successful if the oracle $\mathtt{V}_2(vSid, \cdot)$ returns 1. Lines 04-07 define several internal random variables for later references. Variable $Q_{\mathsf{P}_2}(\mathsf{A})$ counts the number of closed prover sessions and $Q_{\mathsf{P}_1}(\mathsf{A})$ counts the number of abandoned sessions (i.e., sessions that were opened but never closed). Most importantly, variable $\ell(\mathsf{A})$ counts the number of successful verifier sessions and variable $Q_{\mathsf{P}_2}(\mathsf{A})$ counts the number of closed sessions with the prover. Adversary A wins the $\mathbf{OMMIM}_{\mathsf{ID}}$ game, if $\ell(\mathsf{A}) \geq Q_{\mathsf{P}_2}(\mathsf{A}) + 1$, i.e., if A convinces the verifier in at least one more successful verifier sessions than there exist closed sessions with the prover. A's advantage in $\mathbf{OMMIM}_{\mathsf{ID}}$ is defined as $\mathbf{Adv}^{\mathbf{OMMIM}}_{\mathsf{ID}}(\mathsf{A}) := \Pr[\mathbf{OMMIM}^{\mathsf{A}}_{\mathsf{ID}} \Rightarrow 1]$ and we denote its running time as $\mathbf{Time}^{\mathbf{OMMIM}}_{\mathsf{ID}}(\mathsf{A})$.

Definition 4.2 (One-more man-in-the-middle security). We say that ID is $(\varepsilon, t, Q_{\mathsf{Ch}}, Q_{\mathsf{P}_1}, Q_{\mathsf{P}_2})$-*OMMIM-secure* if for all adversaries A satisfying $\mathbf{Time}^{\mathbf{OMMIM}}_{\mathsf{ID}}(\mathsf{A}) \leq t$ $Q_{\mathsf{Ch}}(\mathsf{A}) \leq Q_{\mathsf{Ch}}$, $Q_{\mathsf{P}_2}(\mathsf{A}) \leq Q_{\mathsf{P}_2}$, and $Q_{\mathsf{P}_1}(\mathsf{A}) \leq Q_{\mathsf{P}_1}$, we have $\mathbf{Adv}^{\mathbf{OMMIM}}_{\mathsf{ID}}(\mathsf{A}) \leq \varepsilon$. We say that A breaks $(\varepsilon, t, Q_{\mathsf{Ch}}, Q_{\mathsf{P}_1}, Q_{\mathsf{P}_2})$-*OMMIM security of* ID if $\mathbf{Time}^{\mathbf{OMMIM}}_{\mathsf{ID}}(\mathsf{A}) \leq t$, $Q_{\mathsf{Ch}}(\mathsf{A}) \leq Q_{\mathsf{Ch}}$, $Q_{\mathsf{P}_2}(\mathsf{A}) \leq Q_{\mathsf{P}_2}$, $Q_{\mathsf{P}_1}(\mathsf{A}) \leq Q_{\mathsf{P}_1}$, and we have $\mathbf{Adv}^{\mathbf{OMMIM}}_{\mathsf{ID}}(\mathsf{A}) > \varepsilon$.

Game $\mathbf{OMMIM}^{\mathsf{A}}_{\mathsf{ID}}$:

$par \xleftarrow{\$} \mathsf{ID.PG}(1^\kappa)$
$(sk, pk) \leftarrow \mathsf{ID.KG}(par)$
$pSid \leftarrow 0, vSid \leftarrow 0$
$\mathsf{A}^{\mathtt{P}_1,\mathtt{P}_2,\mathtt{Ch},\mathtt{Ver}}(pk)$
$Q_{\mathtt{Ch}}(\mathsf{A}) \leftarrow vSid$ // #total sessions with verifier
$Q_{\mathsf{P}_1}(\mathsf{A}) \leftarrow \#\{1 \leq k \leq pSid \mid \mathbf{pSess}_k = \mathtt{open}\}$ // #abandoned prover sessions
$Q_{\mathsf{P}_2}(\mathsf{A}) \leftarrow \#\{1 \leq k \leq pSid \mid \mathbf{pSess}_k = \mathtt{closed}\}$ // #closed prover sessions
$\ell(\mathsf{A}) \leftarrow \#\{1 \leq k \leq vSid \mid \mathbf{vSess}_k = \mathtt{closed} \land \boldsymbol{b}'_k = 1\}$ // #successful verifier sessions
If $\ell(\mathsf{A}) \geq Q_{\mathsf{P}_2}(\mathsf{A}) + 1$: Return 1 // A's winning condition
Return 0

Oracle $\mathtt{P}_1$:
$pSid \leftarrow pSid + 1$
$\mathbf{pSess}_{pSid} \leftarrow \mathtt{open}$
$(\boldsymbol{st}_{pSid}, \boldsymbol{R}_{pSid}) \xleftarrow{\$} \mathsf{ID.P}_1$
Return $(pSid, \boldsymbol{R}_{pSid})$

Oracle $\mathtt{P}_2(pSid, c)$:
If $\mathbf{pSess}_{pSid} \neq \mathtt{open}$: Return $\perp$
$\mathbf{pSess}_{pSid} \leftarrow \mathtt{closed}$
$s \leftarrow \mathsf{ID.P}_2(\boldsymbol{st}_{pSid}, sk, \boldsymbol{R}_{pSid}, c)$
Return s

Oracle $\mathtt{Ch}(R')$:
$vSid \leftarrow vSid + 1$
$\mathbf{vSess}_{vSid} \leftarrow \mathtt{open}$
$\boldsymbol{R}'_{vSid} \leftarrow R'; \boldsymbol{c}'_{vSid} \xleftarrow{\$} \mathcal{C}$
Return $(vSid, \boldsymbol{c}'_{vSid})$

Oracle $\mathtt{Ver}(vSid, s')$:
If $\mathbf{vSess}_{vSid} \neq \mathtt{open}$: Return $\perp$
$\mathbf{vSess}_{vSid} \leftarrow \mathtt{closed}$
$\boldsymbol{b}'_{vSid} \leftarrow \mathsf{ID.Ver}(pk, \boldsymbol{R}'_{vSid}, \boldsymbol{c}'_{vSid}, s')$
Return $\boldsymbol{b}'_{vSid}$

Figure 5: The One-More Man-in-the-Middle security game $\mathbf{OMMIM}^{\mathsf{A}}_{\mathsf{ID}}$

We remark that *security against impersonation under active and passive attacks* is a weaker notion than OMMIM security, whereas *man-in-the-middle security* is stronger. Concretely, in the standard man-in-the-middle experiment, the winning condition is relaxed in the sense that there only has to exist a successful session with the verifier with a transcript that does not result from a closed session with the prover.

4.2 Identification Schemes from Linear Function Families

As shown in [4], a linear function family LF directly implies a canonical three-move identification scheme $\mathsf{ID}[\mathsf{LF}]$. The construction is given in Figure 6. It is easy to verify that $\mathsf{ID}[\mathsf{LF}]$ satisfies perfect correctness.

Notation. To avoid too much notational overhead, we make the simple convention that if public key pk has the form $pk = (\mathsf{F}(sk), par)$, then we will instead write $pk = \mathsf{F}(sk)$. Note that using this notation, we can also write $c \cdot pk := c \cdot \mathsf{F}(sk)$.

We will prove later that $\mathsf{ID}[\mathsf{LF}]$ is OMMIM secure. This is the best we can hope for since by the linearity of LF, $\mathsf{ID}[\mathsf{LF}]$ can never be (fully) man-in-the-middle secure. (Concretely, an adversary receiving a commitment R from the prover can send $R' = \mathsf{F}(\hat{r}) + R$ for some $\hat{r} \neq 0$ to the verifier. After forwarding $c' = c$ from verifier to prover, it receives s from the prover and submits $s' = s + \hat{r}$ to the verifier. Since $(R, c, s) \neq (R', c', s')$, A wins the man-in-the-middle experiment with advantage 1.)

Theorem 4.3 *Let* LF *be a linear function family. If* LF *is* (ε', t')*-collision resistant then* $\mathsf{ID}[\mathsf{LF}]$ *is*

Algoriosthm $\mathsf{ID[LF].KG}(par)$:
$sk \xleftarrow{\$} \mathcal{D}$
$pk \leftarrow (\mathsf{F}(sk), par)$
$\mathcal{C} \leftarrow \mathcal{S}$
Return (sk, pk)

Algorithm $\mathsf{ID[LF].Ver}(pk, R, c, s)$:
$S \leftarrow \mathsf{F}(s)$
If $S = c \cdot pk + R$: Return 1
Return 0

Algorithm $\mathsf{ID[LF].P_1}(sk)$:
$r \xleftarrow{\$} \mathcal{D}$
$R \leftarrow \mathsf{F}(r)$
$st_{\mathsf{P}} \leftarrow r$
Return (st_{P}, R)

Algorithm $\mathsf{ID[LF].P_2}(sk, st_{\mathsf{P}}, c)$:
$r \leftarrow st_{\mathsf{P}}$
$s \leftarrow c \cdot sk + r$
Return s

Figure 6: Construction of $\mathsf{ID[LF]} := (\mathsf{ID[LF].PG} := \mathsf{PGen}, \mathsf{ID[LF].KG}, \mathsf{ID[LF].P}, \mathsf{ID[LF].Ver})$, where $\mathsf{LF} = (\mathsf{PGen}, \mathsf{F}, \Psi)$ is a linear function family and $\mathsf{ID[LF].P} := (\mathsf{ID[LF].P_1}, \mathsf{ID[LF].P_2})$.

Adversary $\mathsf{B}^{\mathtt{P_1},\mathtt{P_2},\mathtt{Ch},\mathtt{Ver}}(pk)$:
For $j \in [Q_{\mathsf{P}_2}]$ do:
  $(\mathbf{pSess}_j, \boldsymbol{R}_j) \xleftarrow{\$} \mathtt{P}_1$ //Start Q_{P_2} sessions with Prover
$(\boldsymbol{c} \in \mathcal{S}^{Q_{\mathsf{P}_2}+1}, \mathbf{A} \in \mathcal{S}^{(Q_{\mathsf{P}_2}+1)\times(Q_{\mathsf{P}_2}+1)}) \xleftarrow{\$} \mathsf{A}^{\mathsf{H}}(par)$
Parse $(\mathbf{Z} \in \mathcal{S}^{(Q_{\mathsf{P}_2}+1)\times Q_{\mathsf{P}_2}}, \boldsymbol{z} \in \mathcal{S}^{Q_{\mathsf{P}_2}+1}) \leftarrow \mathbf{A}$
For $j \in [Q_{\mathsf{P}_2}]$ do:
  $\boldsymbol{s}_j \leftarrow \mathtt{P}_2(\mathbf{pSess}_j, \boldsymbol{c}_j)$ //Close Q_{P_2} sessions with Prover
For $i \in [Q_{\mathsf{P}_2}+1]$ do:
  $\boldsymbol{s}'_i \leftarrow \sum_{j=1}^{Q_{\mathsf{P}_2}} \mathbf{A}_{i,j}\boldsymbol{s}_j$
  $\boldsymbol{b}_i \leftarrow \mathtt{Ver}(\mathbf{vSess}_{\mathbf{Z}_i}, \boldsymbol{s}'_i)$

Procedure $\mathsf{H}(\boldsymbol{a})$:
$\boldsymbol{R}'_{\boldsymbol{a}} \leftarrow \sum_{j=1}^{Q_{\mathsf{P}_2}} \boldsymbol{a}_j\boldsymbol{R}_j$
$(\mathbf{vSess}_{\boldsymbol{a}}, \boldsymbol{c}'_{\boldsymbol{a}}) \xleftarrow{\$} \mathtt{Ch}(\boldsymbol{R}'_{\boldsymbol{a}})$
Return $\boldsymbol{c}'_{\boldsymbol{a}}$

Figure 7: Adversary B in game $\mathbf{OMMIM}_{\mathsf{ID}}$.

$(\varepsilon, t, Q_{\mathsf{Ch}}, Q_{\mathsf{P}_1}, Q_{\mathsf{P}_2})$*-OMMIM-secure, where*

$$t' = 2t, \quad \varepsilon' = O\left(\left(\varepsilon - \frac{(Q_{\mathsf{Ch}}Q_{\mathsf{P}_1})^{Q_{\mathsf{P}_2}+1}}{2^{2\kappa}}\right)^2 \frac{1}{Q_{\mathsf{Ch}}^2 Q_{\mathsf{P}_2}^3}\right).$$

The proof of this theorem will be given in Section 7. The following theorem establishes a link between ℓ-$\mathbf{ROS}_{\mathsf{LF}}$ and $\mathbf{OMMIM}_{\mathsf{ID[LF]}}$

Theorem 4.4 *Let* $\mathsf{LF} = (\mathsf{PGen}, \mathsf{F}, \Psi)$ *be a linear function family and let* $\mathsf{ID} := \mathsf{ID[LF]}$*. Suppose that* ID *is* $(\varepsilon, t, Q_{\mathsf{Ch}}, Q_{\mathsf{P}_1} = 0, Q_{\mathsf{P}_2})$*-OMMIM-secure. Further, set* $\ell := Q_{\mathsf{P}_2}$ *and* $Q_{\mathsf{H}} := Q_{\mathsf{Ch}}$*. Then* ℓ-$\mathbf{ROS}_{\mathsf{LF}}$ *is* $(\varepsilon, t, Q_{\mathsf{H}})$*-hard.*

Proof. Let A be an adversary that $(\varepsilon, t, Q_{\mathsf{H}})$-breaks ℓ-$\mathbf{ROS}_{\mathsf{LF}}$. We assume w.l.o.g. that A only makes distinct queries to the random oracle H. In Figure 7, we show how to construct an adversary B that breaks $(\varepsilon, t, Q_{\mathsf{Ch}}, 0, Q_{\mathsf{P}_2})$-OMMIM security of ID and uses A as a subroutine. First, B starts Q_{P_2} sessions with the Prover oracle $\mathtt{P}_1$, receiving commitments $\boldsymbol{R}$. Next, A is executed, where B answers a query of the form $\mathsf{H}(\boldsymbol{a})$ from A as $\boldsymbol{c}'_{\boldsymbol{a}}$, where $\boldsymbol{c}'_{\boldsymbol{a}} := \mathtt{Ch}\left(\sum_{j=1}^{Q_{\mathsf{P}_2}} \boldsymbol{a}_j\boldsymbol{R}_j\right)$. Note that in this manner, each query to H prompts B to open a session with the verifier in $\mathbf{OMMIM}_{\mathsf{ID}}$. Once A returns a solution to $(\boldsymbol{c}, \mathbf{A})$ to ℓ-$\mathbf{ROS}_{\mathsf{LF}}$, B closes the Q_{P_2} opened sessions with the prover by calling $\mathtt{P}_2$ on input $(\mathbf{pSess}_j, \boldsymbol{c}_j)$ for all $j \in [Q_{\mathsf{P}_2}]$. We denote as $\boldsymbol{s}$ the vector of answers that $\mathtt{P}_2$ returns to these queries. Finally, from A's solution to ℓ-$\mathbf{ROS}_{\mathsf{LF}}$, B computes a vector $\boldsymbol{s}'$ of $Q_{\mathsf{P}_2}+1$ answers as described in Figure 7. If A is successful then $\boldsymbol{c}_{Q_{\mathsf{P}_2}+1} = -1$

and $\wedge$ $\mathbf{A}\boldsymbol{c} = 0$. Furthermore for all $i \in [Q_{\mathsf{P}_2} + 1]$, $\mathsf{H}(\mathbf{Z}_i) = \mathbf{A}_{i,Q_{\mathsf{P}_2}+1}$ and we have

$$\begin{aligned}\mathsf{F}(\boldsymbol{s}'_i) &= \mathsf{F}\left(\sum_{j=1}^{Q_{\mathsf{P}_2}} \mathbf{A}_{i,j}\boldsymbol{s}_j\right)\\ &= \sum_{j=1}^{Q_{\mathsf{P}_2}} \mathbf{A}_{i,j}(\boldsymbol{c}_j \cdot pk + \boldsymbol{R}_j) = pk \sum_{j=1}^{Q_{\mathsf{P}_2}} \mathbf{A}_{i,j}\boldsymbol{c}_j + \boldsymbol{R}'_{\mathbf{Z}_i} = pk \cdot \boldsymbol{c}'_{\mathbf{Z}_i} + \boldsymbol{R}'_{\mathbf{Z}_i},\end{aligned}$$

which is equivalent to $\mathsf{ID.Ver}(pk, \boldsymbol{R}'_{\mathbf{Z}_i}, \boldsymbol{c}'_{\mathbf{Z}_i}, \boldsymbol{s}'_i) = 1$. Observe that in the second to last step, we have used the distributive law over the pseudo module formed by $\mathcal{S}$ and $\mathcal{R}$. This shows $\boldsymbol{b}_i = 1$ for all $i \in [Q_{\mathsf{P}_2} + 1]$, which concludes the proof. ■

5 Canonical Blind Signature Schemes

In this section, we introduce the syntax and security of a special type of blind signature scheme, which we call *canonical three-move blind signature scheme*. In Section 5.1, we first introduce the syntax of such schemes and give the proper security definitions. Then, we give a generic construction that gives a canonical three-move blind signature scheme $\mathsf{BS}[\mathsf{LF}]$ from any linear function family LF.

5.1 Syntax and Correctness

We now introduce the syntax of a canonical three-move blind signature scheme.

Definition 5.1 (Canonical Three-Move Blind Signature Scheme). A *canonical three-move blind signature scheme* BS is a tuple of algorithms $\mathsf{BS} = (\mathsf{BS.PG}, \mathsf{BS.KG}, \mathsf{BS.S}, \mathsf{BS.U}, \mathsf{BS.Ver})$.

- The randomised *parameter generation algorithm* $\mathsf{BS.PG}$ takes as input the security parameter 1^κ and returns system parameters par.
- The randomised *key generation algorithm* $\mathsf{BS.KG}$ takes as input system parameters par and outputs a public key/secret key pair (pk, sk). We assume that pk defines a *challenge set* $\mathcal{C} := \mathcal{C}(pk)$ and that pk is known to all parties.
- The *signer algorithm* $\mathsf{BS.S}$ is split into two algorithms, i.e., $\mathsf{BS.S} := (\mathsf{BS.S}_1, \mathsf{BS.S}_2)$, where:
  - The randomised algorithm $\mathsf{BS.S}_1$ takes as input the secret key sk and returns a commitment R and the signer's state $st_{\mathsf{BS.S}}$.
  - The deterministic algorithm $\mathsf{BS.S}_2$ takes as input the signer's state $st_{\mathsf{BS.S}}$, a secret key sk, a commitment R, and a challenge $c \in \mathcal{C}$. It returns the response s.
- The *user algorithm* $\mathsf{BS.U}$ is split into two algorithms, i.e., $\mathsf{BS.U} := (\mathsf{BS.U}_1, \mathsf{BS.U}_2)$, where:
  - The randomised algorithm $\mathsf{BS.U}_1$ takes as input the public key pk, a commitment R, and a message m. It returns the user's state $st_{\mathsf{BS.U}}$ and a challenge $c \in \mathcal{C}$.
  - The deterministic algorithm $\mathsf{BS.U}_2$ takes as input the public key pk, a commitment R, a challenge $c \in \mathcal{C}$, a response s, a message m, and the user's state $st_{\mathsf{BS.U}}$. It returns a signature σ where, possibly, $\sigma = \perp$.
- The deterministic verification algorithm $\mathsf{BS.Ver}$ takes as input the public key pk, a signature σ, and a message m. It outputs 1 (accept) or 0 (reject). We make the convention that $\mathsf{BS.Ver}$ always outputs 0 on input a signature $\sigma = \perp$.

As usual, modelling $\mathsf{BS.S}_2$ and $\mathsf{BS.U}_2$ as deterministic algorithms is w.l.o.g. since randomness can be transmitted through the states.

The diagram below depicts an interaction between signer $\mathsf{BS.S}$ and user $\mathsf{BS.U}$.

Game $\mathbf{Blind}_{\mathsf{BS}}$:
$b \xleftarrow{\$} \{0,1\}$; $\boldsymbol{b}_1 \leftarrow b$; $\boldsymbol{b}_2 \leftarrow 1-b$
$(pk, sk) \xleftarrow{\$} \mathsf{BS.KG}(1^\kappa)$
$b' \xleftarrow{\$} \mathsf{A}^{\mathtt{Init},\mathtt{U}_1,\mathtt{U}_2}(pk, sk)$
Return $b = b'$

Oracle $\mathtt{Init}(\tilde{m}_0, \tilde{m}_1)$: // Only once
$\boldsymbol{m}_0 \leftarrow \tilde{m}_0, \boldsymbol{m}_1 \leftarrow \tilde{m}_1$
$\mathbf{sess}_1 \leftarrow \mathbf{sess}_2 \leftarrow \mathtt{init}$

Oracle $\mathtt{U}_1(sid, R)$:
If $sid \notin \{1,2\} \vee \mathbf{sess}_{sid} \neq \mathtt{init}$: Return $\bot$
$\mathbf{sess}_{sid} \leftarrow \mathtt{open}$
$\boldsymbol{R}_{sid} \leftarrow R$
$(\boldsymbol{st}_{sid}, \boldsymbol{c}_{sid}) \xleftarrow{\$} \mathsf{BS.U}_1(pk, \boldsymbol{R}_{sid}, \boldsymbol{m}_{\boldsymbol{b}_{sid}})$
Return $(sid, \boldsymbol{c}_{sid})$

Oracle $\mathtt{U}_2(sid, s)$:
If $\mathbf{sess}_{sid} \neq \mathtt{open}$: Return $\bot$
$\mathbf{sess}_{sid} \leftarrow \mathtt{closed}$
$\boldsymbol{s}_{sid} \leftarrow s$
$\boldsymbol{\sigma}_{\boldsymbol{b}_{sid}} \xleftarrow{\$} \mathsf{BS.U}_2(pk, \boldsymbol{R}_{sid}, \boldsymbol{c}_{sid}, \boldsymbol{s}_{sid}, \boldsymbol{st}_{sid})$
If $\mathbf{sess}_1 = \mathbf{sess}_2 = \mathtt{closed}$:
  If $\boldsymbol{\sigma}_0 = \bot \vee \boldsymbol{\sigma}_1 = \bot$: Return $(\bot, \bot)$
  Return $(\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1)$
Return $(sid, \mathtt{closed})$

Figure 8: Games defining $\mathbf{Blind}_{\mathsf{BS}}$ for a canonical three-move blind signature scheme BS, with the convention that adversary A makes exactly one query to $\mathtt{Init}$ at the beginning of its execution.

Signer $\mathsf{BS.S}(sk)$		User $\mathsf{BS.U}(pk, m)$
$(st_{\mathsf{BS.S}}, R) \xleftarrow{\$} \mathsf{BS.S}_1(sk)$	$\xrightarrow{R}$	
	$\xleftarrow{c}$	$(st_{\mathsf{BS.U}}, c) \xleftarrow{\$} \mathsf{BS.U}_1(pk, R, m)$
$s \leftarrow \mathsf{BS.S}_2(sk, R, c, st_{\mathsf{BS.S}})$	$\xrightarrow{s}$	$\sigma \leftarrow \mathsf{BS.U}_2(pk, R, c, s, m, st_{\mathsf{BS.U}})$
		Output σ

Definition 5.2 (Perfect Correctness). We say that $\mathsf{BS} = (\mathsf{BS.PG}, \mathsf{BS.KG}, \mathsf{BS.S}, \mathsf{BS.U}, \mathsf{BS.Ver})$ is *perfectly correct*, if for all $par \in \mathsf{BS.PG}(1^\kappa)$, $(pk, sk) \in \mathsf{BS.KG}(par)$, messages $m \in \{0,1\}^*$, and signatures σ that are a possible output of the interaction of $\mathsf{BS.S}(sk)$ and $\mathsf{BS.U}(pk, m)$, we have $\mathsf{BS.Ver}(pk, \sigma, m) = 1$.

5.2 Security Notions

Security of a Canonical Three-Move Blind Signature Scheme BS is captured by two security notions: *blindness* and *one-more unforgeability.*

Blindness. Intuitively, blindness ensures that a signer $\mathsf{BS.S}$ that issues signatures on two messages $(\boldsymbol{m}_0, \boldsymbol{m}_1)$ of its own choice to a user $\mathsf{BS.U}$, can not tell in what order it issues them. In particular, $\mathsf{BS.S}$ is given both resulting signatures $\boldsymbol{\sigma}_0, \boldsymbol{\sigma}_1$, and gets to keep the transcripts of both interactions with $\mathsf{BS.U}$. We remark that we consider for this work the weaker notion of blindness in the *honest signer model* [22] as compared to the *malicious signer model* [18]. The difference between these two models is that in the honest signer model, the adversary obtains the keys from the experiment, whereas in the malicious signer model, the adversary gets to choose its own keys. We formalize the notion of blindness (for a canonical three-move blind signature scheme BS) via game $\mathbf{Blind}_{\mathsf{BS}}$ depicted in Figure 8. In $\mathbf{Blind}_{\mathsf{BS}}$, the game takes the role of the user and A takes the role of the signer. First, the game selects a random bit b which determines the order of adversarially chosen messages in both transcripts. It then runs A on a freshly generated key pair (pk, sk). A is given access to the three oracles $\mathtt{Init}, \mathtt{U}_1$ and $\mathtt{U}_2$. By convention, A first has to query oracle $\mathtt{Init}$. Subsequently, A may open at most two sessions. For each of these two sessions, A obtains corresponding transcripts $\boldsymbol{T}_1 = (\boldsymbol{R}_1, \boldsymbol{c}_1, \boldsymbol{s}_1)$ and $\boldsymbol{T}_2 = (\boldsymbol{R}_2, \boldsymbol{c}_2, \boldsymbol{s}_2)$. The game uses $\boldsymbol{m}_b$ and $\boldsymbol{m}_{1-b}$ to generate the transcripts $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$, respectively. If A honestly completes both sessions with the game, it obtains signatures $\boldsymbol{\sigma}_b$ and $\boldsymbol{\sigma}_{1-b}$ on messages $\boldsymbol{m}_b$ and $\boldsymbol{m}_{1-b}$. Note that A obtains $\boldsymbol{\sigma}_b$ and $\boldsymbol{\sigma}_{1-b}$ by calling $\mathtt{U}_2$ twice. More precisely, the first call to $\mathtt{U}$ closes the first session and the second call closes the second session. Once both sessions are closed, the game checks if A acted honestly in both of them and if so, returns the signatures $(\boldsymbol{\sigma}_b, \boldsymbol{\sigma}_{1-b})$. If instead A has behaved dishonestly and, as a result, $\boldsymbol{\sigma}_b = \bot$ or $\boldsymbol{\sigma}_{1-b} = \bot$ at the time of closing the second session, $\mathtt{U}_2$ returs $(\bot, \bot)$. At the end of the experiment, A has to guess the bit b. We define the advantage of adversary A in $\mathbf{Blind}_{\mathsf{BS}}$ as $\mathbf{Adv}^{\mathbf{Blind}}_{\mathsf{BS}}(\mathsf{A}) := \left|\Pr[\mathbf{Blind}^{\mathsf{A}}_{\mathsf{BS}} \Rightarrow 1] - \frac{1}{2}\right|$.

Definition 5.3 (Perfect Blindness). Let BS be a canonical three-move blind signature scheme. We say that BS is *perfectly blind* if for all adversaries A, $\mathbf{Adv}^{\mathbf{Blind}}_{\mathsf{BS}}(\mathsf{A}) = 0$.

Game $\mathbf{OMUF}_{\mathsf{BS}}$:
$par \xleftarrow{\$} \mathsf{BS.PG}(1^\kappa)$
$(sk, pk) \xleftarrow{\$} \mathsf{BS.KG}(par)$
$sid \leftarrow 0$ //initialize signer session id
$((\boldsymbol{m}_1, \boldsymbol{\sigma}_1), ..., (\boldsymbol{m}_{\ell(\mathsf{A})}, \boldsymbol{\sigma}_{\ell(\mathsf{A})})) \leftarrow \mathsf{A}^{\mathtt{S}_1, \mathtt{S}_2}(pk)$
If $\exists i \neq j : \boldsymbol{m}_i = \boldsymbol{m}_j$: Return 0 //all messages have to be distinct
If $\exists i \in [\ell(\mathsf{A})] : \mathsf{BS.Ver}(pk, \boldsymbol{m}_i, \boldsymbol{\sigma}_i) = 0$: Return 0 //All signatures have to be valid
$Q_{\mathtt{S}_1}(\mathsf{A}) \leftarrow \#\{k \mid \mathbf{sess}_k = \mathtt{open}\}$ //#abandoned signer sessions
$Q_{\mathtt{S}_2}(\mathsf{A}) \leftarrow \#\{k \mid \mathbf{sess}_k = \mathtt{closed}\}$ //#closed signer sessions
If $\ell(\mathsf{A}) \geq Q_{\mathtt{S}_2}(\mathsf{A}) + 1$: Return 1
Return 0

Oracle $\mathtt{S}_1$:
$sid \leftarrow sid + 1$
$\mathbf{sess}_{sid} \leftarrow \mathtt{open}$
$(\boldsymbol{st}_{sid}, \boldsymbol{R}_{sid}) \xleftarrow{\$} \mathsf{BS.S}_1(sk)$
Return $(sid, \boldsymbol{R}_{sid})$

Oracle $\mathtt{S}_2(sid, c)$:
If $\mathbf{sess}_{sid} \neq \mathtt{open}$: Return $\perp$
$\mathbf{sess}_{sid} = \mathtt{closed}$
$\boldsymbol{s}_{sid} \leftarrow \mathsf{BS.S}_2(sk, \boldsymbol{st}_{sid}, \boldsymbol{R}_{sid}, c)$
Return $\boldsymbol{s}_{sid}$

Figure 9: Game $\mathbf{OMUF}_{\mathsf{BS}}$ with adversary A.

OMUF Security of Blind Signature Schemes. We now define the standard unforgeability notion for blind signatures, namely *one-more unforgeability.* Intuitively, one-more unforgeability ensures that a user $\mathsf{BS.U}$ can not produce even a single signature more than it should be able to learn from its interactions with the signer $\mathsf{BS.S}$. We formalize the notion of one-more unforgeability (for a canonical three-move blind signature scheme BS) via game $\mathbf{OMUF}_{\mathsf{BS}}$ as depicted in Figure 9. In $\mathbf{OMUF}_{\mathsf{BS}}$, an adversary A in the role of $\mathsf{BS.U}$ is run on input the public key of the signer $\mathsf{BS.S}$ and subsequently interacts with oracles that imitate the behaviour of $\mathsf{BS.S}$. A call to $\mathtt{S}_1$ returns a new session identifier sid and sets flag $\mathbf{sess}_{sid}$ to $\mathtt{open}$. A call to $\mathtt{S}_2(sid, \cdot)$ with the same sid sets the flag $\mathbf{sess}_{sid}$ to $\mathtt{closed}$. The closed sessions result in (at most) $Q_{\mathtt{S}_2}$ transcripts $(\boldsymbol{R}_k, \boldsymbol{c}_k, \boldsymbol{s}_k)$, where the challenges $\boldsymbol{c}$ are chosen by A. (The remaining (at most) $Q_{\mathtt{S}_1}$ abandoned sessions are of the form $(\boldsymbol{R}_k, \perp, \perp)$ and hence do not contain a complete transcript.) A wins the experiment, if it is able to produce $\ell(\mathsf{A}) \geq Q_{\mathtt{S}_2}(\mathsf{A}) + 1$ signatures (on distinct messages) after having closed $Q_{\mathtt{S}_2}(\mathsf{A}) \leq Q_{\mathtt{S}_2}$ signer sessions (from which it should be able to compute $Q_{\mathtt{S}_2}(\mathsf{A})$ signatures). We define the advantage of adversary A in $\mathbf{OMUF}_{\mathsf{BS}}$ as $\mathbf{Adv}^{\mathbf{OMUF}}_{\mathsf{BS}}(\mathsf{A}) := \Pr[\mathbf{OMUF}^{\mathsf{A}}_{\mathsf{BS}} \Rightarrow 1]$ and denote its running time as $\mathbf{Time}^{\mathbf{OMUF}}_{\mathsf{BS}}(\mathsf{A})$.

Definition 5.4 (One-More Unforgeability). Let BS be a canonical three-move blind signature scheme. We say that BS is $(\varepsilon, t, Q_{\mathtt{S}_1}, Q_{\mathtt{S}_2})$*-OMUF-secure* if for all adversaries A satisfying

$$\mathbf{Time}^{\mathbf{OMUF}}_{\mathsf{BS}}(\mathsf{A}) \leq t, \quad Q_{\mathtt{S}_1}(\mathsf{A}) \leq Q_{\mathtt{S}_1}, \quad Q_{\mathtt{S}_2}(\mathsf{A}) \leq Q_{\mathtt{S}_2}, \tag{2}$$

we have $\mathbf{Adv}^{\mathbf{OMUF}}_{\mathsf{BS}}(\mathsf{A}) \leq \varepsilon$. We say that A *breaks* $(\varepsilon, t, Q_{\mathtt{S}_1}, Q_{\mathtt{S}_2})$*-OMUF security of* BS if it satisfies 2 and $\mathbf{Adv}^{\mathbf{OMUF}}_{\mathsf{BS}}(\mathsf{A}) > \varepsilon$.

5.3 Blind Signature Schemes from Linear Function Families

Let LF be a linear function family and $\mathsf{H}\colon \{0,1\}^* \to \mathcal{S}$ be a hash function. Figure 10 shows how to construct a canonical three-move blind signature scheme $\mathsf{BS}[\mathsf{LF}, \mathsf{H}]$.

Correctness of $\mathsf{BS}[\mathsf{LF}, \mathsf{H}]$. We begin by proving correctness of $\mathsf{BS}[\mathsf{LF}, \mathsf{H}]$.

Lemma 5.5 *Let* $\mathsf{LF} = (\mathsf{PGen}, \mathsf{F}, \Psi)$ *be a linear function family, let* $\mathsf{H}\colon \{0,1\}^* \to \mathcal{S}$ *be a hash function, and* $\mathsf{BS} := \mathsf{BS}[\mathsf{LF}, \mathsf{H}]$. *Then* BS *has perfect correctness.*

Proof. Consider a signature $\sigma = (s', c')$ that is the result of an interaction between an honestly behaving signer $\mathsf{BS.S}$ holding the secret key $sk \in \mathcal{D}$ and an honestly behaving user $\mathsf{BS.U}$ holding the public key $pk = \mathsf{F}(sk) \in \mathcal{R}$. We denote with (R, c, s) the transcript resulting from this interaction and with $\alpha \in \mathcal{D}, \beta \in \mathcal{S}$ the associated blinding parameters that $\mathsf{BS.U}_1$ samples. Finally, let $r \in \mathcal{D}$ be the value chosen by $\mathsf{BS.S}_1$ s.t. $\mathsf{F}(r) = R$. To ensure that $\mathsf{BS.Ver}(pk, \sigma, m) = 1$, we need to show that

$$R + \mathsf{F}(\alpha) + \beta \cdot pk = R' = \mathsf{F}(s') - c' \cdot pk. \tag{3}$$

Algorithm $\mathsf{BS.S}_1(sk)$:

$r \xleftarrow{\$} \mathcal{D}, R \leftarrow \mathsf{F}(r)$
$st_{\mathsf{BS.S}} \leftarrow r$
Return $(st_{\mathsf{BS.S}}, R)$

Algorithm $\mathsf{BS.S}_2(sk, st_{\mathsf{BS.S}}, c)$:

$r \leftarrow st_{\mathsf{BS.S}}$
$s \leftarrow c \cdot sk + r$
Return s

Algorithm $\mathsf{BS.U}_1(pk, R, m)$:

$\alpha \xleftarrow{\$} \mathcal{D}, \beta \xleftarrow{\$} \mathcal{S}$
$R' \leftarrow R + \mathsf{F}(\alpha) + \beta \cdot pk$
$c' \leftarrow \mathsf{H}(R', m)$
$c \leftarrow c' + \beta$
$st_{\mathsf{BS.U}} \leftarrow (\alpha, \beta, c)$
Return $(c, st_{\mathsf{BS.U}})$

Algorithm $\mathsf{BS.U}_2(pk, R, c, s, m, st_{\mathsf{BS.U}})$:

$S \leftarrow \mathsf{F}(s)$
If $S \neq c \cdot pk + R$: Return $\perp$
$(\alpha, \beta, c) \leftarrow st_{\mathsf{BS.U}}$
$R' \leftarrow R + \mathsf{F}(\alpha) + \beta \cdot pk$
$c' \leftarrow \mathsf{H}(R', m)$
$s' \leftarrow s + \alpha + \Psi(pk, -c', c)$
$\sigma \leftarrow (c', s')$
Return σ

Algorithm $\mathsf{BS.Ver}(pk, \sigma, m)$:

$(c', s') \leftarrow \sigma$
$R' \leftarrow \mathsf{F}(s') - c' \cdot pk$
If $c' \neq \mathsf{H}(R', m)$: Return 0
Return 1

Figure 10: Let LF be a linear function and $\mathsf{H}\colon \{0,1\}^* \to \mathcal{S}$ be a hash function. This figure shows the construction of the canonical three-move blind signature scheme $\mathsf{BS} := \mathsf{BS}[\mathsf{LF}, \mathsf{H}]$ where $\mathsf{BS} := (\mathsf{BS.PG} = \mathsf{ID}[\mathsf{LF}].\mathsf{PG}, \mathsf{BS.KG} := \mathsf{ID}[\mathsf{LF}].\mathsf{KG}, \mathsf{BS.S} = (\mathsf{BS.S}_1, \mathsf{BS.S}_2), \mathsf{BS.U} = (\mathsf{BS.U}_1, \mathsf{BS.U}_2), \mathsf{BS.Ver})$. Note that we again implicitly set $\mathcal{C} := \mathcal{S}$.

Writing $\beta = (c - c')$, we obtain

$$R + \mathsf{F}(\alpha) + \beta \cdot pk = R + \mathsf{F}(\alpha) + (c - c') \cdot pk$$

for the left hand side of Equation (3). Expanding s' as

$$s' = s + \alpha + \Psi(pk, -c', c) = r + c \cdot sk + \alpha + \Psi(pk, -c', c),$$

the right hand side becomes:

$$\begin{aligned} \mathsf{F}(s') - c' \cdot pk &= \mathsf{F}(r + c \cdot sk + \alpha + \Psi(pk, -c', c)) - c' \cdot pk \\ &= \mathsf{F}(r) + c \cdot \mathsf{F}(sk) + \mathsf{F}(\alpha) + \mathsf{F}(\Psi(pk, -c', c)) - c' \cdot pk \\ &= R + (c \cdot \mathsf{F}(sk) + \mathsf{F}(\Psi(\mathsf{F}(sk), -c', c)) - c' \cdot \mathsf{F}(sk)) + \mathsf{F}(\alpha) \\ &= R + (c - c') \cdot \mathsf{F}(sk) + \mathsf{F}(\alpha) \\ &= R + (c - c') \cdot pk + \mathsf{F}(\alpha). \end{aligned}$$

The proof is complete since both sides of Equation (3) are equal (by commutativity of $+$ on $\mathcal{R}$). ∎

One-More Unforgeability of $\mathsf{BS}[\mathsf{LF}, \mathsf{H}]$. In this subsection, we show that $\mathbf{OMUF}_{\mathsf{BS}[\mathsf{LF},\mathsf{H}]}$ is equivalent (in the ROM) to $\mathbf{OMMIM}_{\mathsf{ID}[\mathsf{LF}]}$.

Theorem 5.6 *Let* LF *be a linear function family, let* $\mathsf{H}\colon \{0,1\}^* \to \mathcal{S}$ *be a hash function, and let* $\mathsf{ID} := \mathsf{ID}[\mathsf{LF}], \mathsf{BS} := \mathsf{BS}[\mathsf{LF}, \mathsf{H}]$. *If* ID *is* $(\varepsilon', t', Q_{\mathsf{Ch}}, Q_{\mathsf{P}_1}, Q_{\mathsf{P}_2})$*-OMMIM-secure then* BS *is* $(\varepsilon, t, Q_{\mathsf{S}_1}, Q_{\mathsf{S}_2}, Q_{\mathsf{H}})$*-OMUF-secure in the random oracle model, where*

$$t' = t, \quad \varepsilon' = \varepsilon, \quad Q_{\mathsf{Ch}} = Q_{\mathsf{H}} + Q_{\mathsf{S}_2} + 1, \quad Q_{\mathsf{P}_1} = Q_{\mathsf{S}_1}, \quad Q_{\mathsf{P}_2} = Q_{\mathsf{S}_2}.$$

Proof. Let A be an adversary that breaks $(\varepsilon, t, Q_{\mathsf{S}_1}, Q_{\mathsf{S}_2}, Q_{\mathsf{H}})$-one-more-unforgeability of $\mathsf{BS}[\mathsf{LF}, \mathsf{H}]$ in the random oracle model. In Figure 11 we construct an adversary B that runs in the $\mathbf{OMMIM}_{\mathsf{ID}}$ experiment and perfectly simulates A's oracles $\mathtt{S}_1$, $\mathtt{S}_2$ and H via its own oracles $\mathtt{P}_1$, $\mathtt{P}_2$, and $\mathtt{Ch}$, respectively. Note that B calls P_2 at most $Q_{\mathsf{P}_2} = Q_{\mathsf{S}_2}$ many times over the course of its simulation and moreover, $Q_{\mathsf{P}_2}(\mathsf{B}) = Q_{\mathsf{S}_2}(\mathsf{A})$. We show that B breaks $(\varepsilon', t', Q_{\mathsf{Ch}}, Q_{\mathsf{P}_1}, Q_{\mathsf{P}_2})$-OMMIM security of ID. Suppose that A is successful, i.e., it outputs $\ell(\mathsf{A}) \geq Q_{\mathsf{S}_2}(\mathsf{A}) + 1 = Q_{\mathsf{P}_2}(\mathsf{B}) + 1$ valid signatures on distinct messages and the number of closed sessions with the signer is at most $Q_{\mathsf{S}_2}(\mathsf{A}) = Q_{\mathsf{P}_2}(\mathsf{B})$. Since all messages in $\boldsymbol{m}$ are distinct, each

Adversary $\mathsf{B}^{\mathtt{P}_1,\mathtt{P}_2,\mathtt{Ch},\mathtt{Ver}}(pk)$:

00 $\big((\boldsymbol{m}_1,\boldsymbol{\sigma}_1),...,(\boldsymbol{m}_{\ell(\mathsf{A})},\boldsymbol{\sigma}_{\ell(\mathsf{A})})\big) \leftarrow \mathsf{A}^{\mathsf{S}_1,\mathsf{S}_2,\mathsf{H}}(pk)$
01 For $i \in [\ell(\mathsf{A})]$ do:
02 $\quad (\boldsymbol{c}'_i, \boldsymbol{s}'_i) \leftarrow \boldsymbol{\sigma}_i$
03 $\quad \boldsymbol{R}'_i \leftarrow \mathsf{F}(\boldsymbol{s}'_i) - \boldsymbol{c}'_i \cdot pk$
04 $\quad \mathsf{H}(\boldsymbol{R}'_i, \boldsymbol{m}_i)$
05 $\quad vSid \leftarrow \mathbf{vSess}_{\boldsymbol{R}'_i,\boldsymbol{m}_i}$
06 $\quad \boldsymbol{b}_i \leftarrow \mathtt{Ver}(vSid, \boldsymbol{s}'_i)$

Procedure S_1:

07 $(pSid, \boldsymbol{R}_{pSid}) \xleftarrow{\$} \mathtt{P}_1$
08 Return $(pSid, \boldsymbol{R}_{pSid})$

Procedure $\mathsf{S}_2(pSid, c)$:

09 $\boldsymbol{s}_{pSid} \leftarrow \mathtt{P}_2(pSid, c)$
10 Return $\boldsymbol{s}_{pSid}$

Procedure $\mathsf{H}(R', m)$:

11 if $H[R',m] \neq \bot$: Return $H[R',m]$
12 $(vSid, c') \xleftarrow{\$} \mathtt{Ch}(R')$
13 $\mathbf{vSess}_{R',m} \leftarrow vSid$
14 $H[R',m] \leftarrow c'$
15 Return $H[R',m]$

Figure 11: Reduction from $\mathbf{OMMIM}_{\mathsf{ID}}$ to $\mathbf{OMUF}_{\mathsf{BS[LF,H]}}$

signature corresponds to a distinct session with the oracle $\mathtt{Ch}$ via the relation $\mathsf{H}(R', \boldsymbol{m}_i) = \mathtt{Ch}(R')$. Since also $\boldsymbol{\sigma}_i = (\boldsymbol{c}'_i, \boldsymbol{s}'_i)$ is a valid signature on $\boldsymbol{m}_i$, we know that $\mathsf{H}(\mathsf{F}(\boldsymbol{s}'_i) - \boldsymbol{c}'_i \cdot pk, m) = \mathsf{H}(\boldsymbol{R}'_i, m) = \mathtt{Ch}(\boldsymbol{R}'_i)$. Therefore, B can make a successful query to oracle $\mathtt{Ver}(vSid, \boldsymbol{s}'_i)$ in line 06 resulting in $\boldsymbol{b}_i = 1$ for every valid signature. Since overall, B makes $\ell(\mathsf{B}) = Q_{\mathsf{P}_2}(\mathsf{B}) + 1$ successful queries to $\mathtt{Ver}$, B wins $\mathbf{OMMIM}_{\mathsf{ID}}$ whenever A wins $\mathbf{OMUF}_{\mathsf{BS}}$. This proves $\varepsilon' \geq \varepsilon$. Moreover, the number of abandoned sessions (denoted as $Q_{\mathsf{S}_1}(\mathsf{A})$) in the $\mathbf{OMUF}_{\mathsf{BS}}$ experiment equals the number of abandoned sessions (denoted as $Q_{\mathsf{P}_1}(\mathsf{B})$) in the $\mathbf{OMMIM}_{\mathsf{ID}}$ experiment and the number $Q_{\mathsf{Ch}}(\mathsf{B})$ of calls to oracle $\mathtt{Ch}$ is bounded by Q_{H} (for the simulation of H) plus additional $Q_{\mathsf{P}_2}(\mathsf{A}) + 1$ calls in Line 04 (the latter calls are necessary in case A guesses the output of $\mathtt{Ch}$ on some points). Finally, the running times of A and B are roughly the same, i.e. $t \approx t'$.

■

Theorem 5.7 *Let* LF *be a linear function family, let* $\mathsf{H}\colon \{0,1\}^* \to \mathcal{S}$ *be a hash function, and let* $\mathsf{BS} := \mathsf{BS[LF,H]}, \mathsf{ID} := \mathsf{ID[LF]}$. *If* BS *is* $(\varepsilon, t, Q_{\mathsf{S}_1}, Q_{\mathsf{S}_2}, Q_{\mathsf{H}})$*-OMUF-secure in the random oracle model then* ID *is* $(\varepsilon', t', Q_{\mathsf{Ch}}, Q_{\mathsf{P}_1}, Q_{\mathsf{P}_2})$*-OMMIM-secure, where*

$$t' = t, \quad \varepsilon' = \varepsilon, \quad Q_{\mathsf{Ch}} = 2 \cdot Q_{\mathsf{H}}, \quad Q_{\mathsf{P}_1} = Q_{\mathsf{S}_1}, \quad Q_{\mathsf{P}_2} = Q_{\mathsf{S}_2}.$$

Proof. Let B be an adversary that breaks $(\varepsilon', t', Q_{\mathsf{Ch}}, Q_{\mathsf{P}_1}, Q_{\mathsf{P}_2})$-OMMIM security of ID. In Figure 12 we construct an adversary A that is executed in game $\mathbf{OMUF}_{\mathsf{BS}}$. A perfectly simulates B's oracles $\mathtt{P}_1, \mathtt{P}_2$ and $\mathtt{Ch}$ via its own oracles $\mathsf{S}_1, \mathsf{S}_2$ and H, respectively. We show that A breaks $(\varepsilon, t, Q_{\mathsf{S}_1}, Q_{\mathsf{S}_2}, Q_{\mathsf{H}})$-one-more-unforgeability of BS. To simulate oracle $\mathtt{Ver}$, A executes the same code as specified in the $\mathbf{OMMIM}_{\mathsf{ID}}$ experiment, with the only difference being line 20. This additional line does not change the behavior of $\mathtt{Ver}$ and is thus not detectable by B. Suppose that B is successful, i.e., it completes $Q_{\mathsf{P}_2}(\mathsf{B})$ sessions with P_2 and at least $Q_{\mathsf{P}_2}(\mathsf{B}) + 1$ sessions with $\mathtt{Ver}$ (denoted as $\ell(\mathsf{B})$ in the $\mathbf{OMMIM}_{\mathsf{ID}}$ experiment). From the $\ell(\mathsf{B})$ successful calls of B to $\mathtt{Ver}$, it follows that A learns $\ell(\mathsf{B}) \geq Q_{\mathsf{S}_2}(\mathsf{A}) + 1$ transcripts $(\boldsymbol{R}, \boldsymbol{c}, \boldsymbol{s})$. The messages $\boldsymbol{m}$ are defined by executing $\mathsf{BS.U}_1$ in Line 12. Note that each execution of $\mathsf{BS.U}_1$ internally entails a call to H. Furthermore, each call to $\mathsf{BS.Ver}$ on Line 18 entails a further call to H. Thus, for each session that B initiates with the verifier in $\mathbf{OMMIM}_{\mathsf{ID}}$, A calls H at most twice, implying that $Q_{\mathsf{Ch}} \leq 2 \cdot Q_{\mathsf{H}}$. It is easy to see that A creates $\ell(\mathsf{B})$ valid signatures after learning values $\boldsymbol{s}$ by simply following the protocol specification of $\mathsf{BS.U}_2$ as done in Line 05. This proves $\varepsilon' = \varepsilon$. Moreover the number of abandoned sessions (denoted as $Q_{\mathsf{P}_1}(\mathsf{B})$) in the $\mathbf{OMMIM}_{\mathsf{ID}}$ experiment equals the number of abandoned sessions (denoted as $Q_{\mathsf{S}_1}(\mathsf{A})$) in the $\mathbf{OMUF}_{\mathsf{BS}}$ experiment. Finally, the running times of A and B are roughly the same, i.e. $t \approx t'$. ■

Theorem 5.8 *Let* $\mathsf{LF} = (\mathsf{PGen}, \mathsf{F}, \Psi)$ *be a linear function family. Then* $\mathsf{BS[LF,H]}$ *is perfectly blind.*

Proof. Let A be an adversary playing in game $\mathbf{Blind}^{\mathsf{A}}_{\mathsf{BS[LF,H]}}$. A obtains as input a valid pair of keys $(pk, sk) \in \mathsf{BS.KG}(1^\kappa)$. Note that this ensures that we can write $pk = \mathsf{F}(sk)$.[1] After its execution, A

[1] Indeed, it is here that we require the keys to be honestly generated by the experiment.

Adversary $\mathsf{A}^{\mathsf{S}_1,\mathsf{S}_2,\mathsf{H}}(pk)$:

$vSid \leftarrow 0$
$\mathsf{B}^{\mathtt{P}_1,\mathtt{P}_2,\mathtt{Ch},\mathtt{Ver}}(pk)$
$i \leftarrow 1$
For $1 \leq k \leq vSid$ where $(\mathbf{vSess}_k = \mathtt{closed}) \wedge (\boldsymbol{b}_k = 1)$ do:
  $\boldsymbol{c}'_k := \boldsymbol{c}_k - \boldsymbol{\beta}_k$
  $\boldsymbol{m}_i \leftarrow k, \boldsymbol{\sigma}_i \leftarrow (\boldsymbol{c}'_k, \boldsymbol{s}'_k := \boldsymbol{s}_k + \boldsymbol{\alpha}_k + \Psi(pk, -\boldsymbol{c}'_k, \boldsymbol{c}_k))$
  $i \leftarrow i + 1$
Return $(\boldsymbol{m}_1, \boldsymbol{\sigma}_1), \ldots, (\boldsymbol{m}_{i-1}, \boldsymbol{\sigma}_{i-1})$

Procedure P_1:
$(pSid, \boldsymbol{R}_{pSid}) \xleftarrow{\$} \mathsf{S}_1$
Return $(pSid, \boldsymbol{R}_{pSid})$

Procedure $\mathsf{Ch}(R)$:
$vSid \leftarrow vSid + 1$
$\mathbf{vSess}_{vSid} \leftarrow \mathtt{open}$
$(\boldsymbol{c}_{vSid}, \boldsymbol{st}_{vSid}) \leftarrow \mathsf{BS.U}_1(pk, R, m := vSid)$
$(\boldsymbol{\alpha}_{vSid}, \boldsymbol{\beta}_{vSid}, \boldsymbol{c}_{vSid}) \leftarrow \boldsymbol{st}_{vSid}$
Return $(vSid, \boldsymbol{c}_{vSid})$

Procedure $\mathsf{P}_2(pSid, c)$:
$\boldsymbol{s}_{pSid} \leftarrow \mathsf{S}_2(pSid, c)$
Return $\boldsymbol{s}_{pSid}$

Procedure $\mathsf{Ver}(vSid, s)$:
If $\mathbf{vSess}_{vSid} \neq \mathtt{open}$: Return $\perp$
$\boldsymbol{b}_{vSid} \leftarrow \mathsf{BS.Ver}(pk, \boldsymbol{R}_{vSid}, \boldsymbol{c}_{vSid}, s)$
$\mathbf{vSess}_{vSid} \leftarrow \mathtt{closed}$
$\boldsymbol{s}_{vSid} \leftarrow s$
Return $\boldsymbol{b}_{vSid}$

Figure 12: Reduction from $\mathbf{OMUF}_{\mathsf{BS[LF,H]}}$ to $\mathbf{OMMIM}_{\mathsf{ID[LF]}}$

holds $(\boldsymbol{m}_0, \boldsymbol{\sigma}_0)$, $(\boldsymbol{m}_1, \boldsymbol{\sigma}_1)$ where $\boldsymbol{\sigma}_0$ is a signature on $\boldsymbol{m}_0$ and $\boldsymbol{\sigma}_1$ is a signature on $\boldsymbol{m}_1$. (Here we assume without loss of generality that both signatures are valid as otherwise A obtains $\boldsymbol{\sigma}_0 = \boldsymbol{\sigma}_1 = \perp$ and thus $\mathbf{Adv}^{\mathsf{A}}_{\mathbf{Blind},\mathsf{BS[LF,H]}} = 0$.) Adversary A furthermore learns two transcripts $\boldsymbol{T}_1 = (\boldsymbol{R}_1, \boldsymbol{c}_1, \boldsymbol{s}_1)$ and $\boldsymbol{T}_2 = (\boldsymbol{R}_2, \boldsymbol{c}_2, \boldsymbol{s}_2)$ from its interaction with the first and the second signer session, respectively. The goal of A is to match the message/signature pairs with the two transcripts.

We show that there exists no adversary which is able to distinguish, whether the message $\boldsymbol{m}_0$ was used by the experiment to create Transcript $\boldsymbol{T}_1$ or $\boldsymbol{T}_2$. We argue that for all sessions $1 \leq i \leq 2$ and indexes $0 \leq j \leq 1$, the tuple $(\boldsymbol{T}_i, \boldsymbol{m}_j, \boldsymbol{\sigma}_j)$ completely determines a properly distributed state $\boldsymbol{st}_j = (\boldsymbol{\alpha}_{i,j}, \boldsymbol{\beta}_{i,j}, \boldsymbol{c}_j)$ of $\mathsf{BS.U}_1$. This implies that given A's view, it is equally likely that the experiment was executed with $b = 0$ or $b = 1$ since for both choices $b \in \{0, 1\}$ there exist properly distributed states $(\boldsymbol{st}_0, \boldsymbol{st}_1)$ that would have resulted in A's view.

It remains to argue that $\boldsymbol{T}_i = (\boldsymbol{R}_i, \boldsymbol{c}_i, \boldsymbol{s}_i)$, $\boldsymbol{m}_j$, and $\boldsymbol{\sigma}_j = (\boldsymbol{c}'_j, \boldsymbol{s}'_j)$ determine values $\boldsymbol{\alpha}_{i,j}, \boldsymbol{\beta}_{i,j}$ such that $\boldsymbol{c}'_j = \mathsf{H}(\boldsymbol{R}_i + \beta_{i,j} \cdot pk + \mathsf{F}(\boldsymbol{\alpha}_{i,j}), \boldsymbol{m}_j)$ and $\boldsymbol{\alpha}_{i,j} = \boldsymbol{s}'_j - \boldsymbol{s}_i - \Psi(pk, -c'_j, c_i), \boldsymbol{\beta}_{i,j} = \boldsymbol{c}_i - \boldsymbol{c}'_j$. Uniformity of $(\boldsymbol{\alpha}_{i,j}, \boldsymbol{\beta}_{i,j})$ is implied by uniformity of $(\boldsymbol{s}'_j, \boldsymbol{c}'_j)$, which come from the experiment.

Since $\boldsymbol{T}_i$ is a valid transcript, we have $\mathsf{F}(\boldsymbol{s}_i) = \boldsymbol{R}_i + \boldsymbol{c}_i \cdot pk$. Therefore,

$$\begin{aligned}
\boldsymbol{R}_i + \beta_{i,j} \cdot pk + \mathsf{F}(\boldsymbol{\alpha}_{i,j}) &= \boldsymbol{R}_i + (\boldsymbol{c}_i - \boldsymbol{c}'_j) \cdot pk + \mathsf{F}(\boldsymbol{s}'_j - \boldsymbol{s}_i - \Psi(pk, -c'_j, c_i)) \\
&= \boldsymbol{R}_i + (\boldsymbol{c}_i - \boldsymbol{c}'_j) \cdot \mathsf{F}(sk) + \mathsf{F}(\boldsymbol{s}'_j - \boldsymbol{s}_i - \Psi(\mathsf{F}(sk), -c'_j, c_i)) \\
&= \boldsymbol{R}_i + \boldsymbol{c}_i \cdot \mathsf{F}(sk) - \boldsymbol{c}'_j \cdot \mathsf{F}(sk) + \mathsf{F}(\Psi(\mathsf{F}(sk), -c'_j, c_i)) + \mathsf{F}(\boldsymbol{s}'_j - \boldsymbol{s}_i - \Psi(\mathsf{F}(sk), -c'_j, c_i)) \\
&= (\boldsymbol{R}_i + \boldsymbol{c}_i \cdot pk - \mathsf{F}(\boldsymbol{s}_i)) + \mathsf{F}(\boldsymbol{s}'_j) - \boldsymbol{c}'_j \cdot pk + \big(\mathsf{F}(\Psi(pk, -c'_j, c_i)) - \mathsf{F}(\Psi(pk, -c'_j, c_i))\big) \\
&= \mathsf{F}(\boldsymbol{s}'_j) - \boldsymbol{c}'_j \cdot pk \; .
\end{aligned}$$

Since $\boldsymbol{\sigma}_j$ is a valid signature on $\boldsymbol{m}_j$ we have $\mathsf{H}(\mathsf{F}(\boldsymbol{s}'_j) - \boldsymbol{c}'_j \cdot pk, \boldsymbol{m}_j) = \boldsymbol{c}'_j$ which concludes the proof. ■

Corollary 5.9 *Let* LF *be a linear function family. If* LF *is* (ε', t')*-collision resistant, then* BS[LF, H] *is* $(\varepsilon, t, Q_{\mathsf{S}_1}, Q_{\mathsf{S}_2}, Q_{\mathsf{H}})$*-OMUF-secure where*

$$t' = 2t, \quad \varepsilon' = O\left(\left(\varepsilon - \frac{(Q \cdot Q_{\mathsf{S}_1})^{Q_{\mathsf{S}_2}+1}}{2^{2\kappa}}\right)^2 \frac{1}{Q^2 Q_{\mathsf{S}_2}^3}\right),$$

and $Q = Q_{\mathsf{H}} + Q_{\mathsf{S}_2} + 1$*. Moreover,* BS[LF, H] *is perfectly blind.*

Proof. The proof of the one-more unforgability security follows from combining Theorems 4.3 and 5.7. Perfect blindness follows directly from Theorem 5.8. ■

6 The Subset Forking Lemma

In this section, we prove a further generalization of the forking lemma, which was first introduced in the groundbreaking work of Pointcheval and Stern [29]. The original version of the forking lemma was stated in a specialized form that only applied to some *specific* signature schemes and blind signature schemes. In later work, Bellare and Neven [8] generalised the forking lemma by rephrasing it a as purely probabilistic statement. Very roughly, it states that one can run the algorithm $\mathcal{C}$ twice on the same instance I and randomness ω, but different challenge values h, h' to obtain (with non-negligible probability) two different, but related answers σ, σ', from which it is possible to solve the instance I.[2] Due to the generality of this statement, the forking lemma (in the version of [8]) is widely applicable and has become an indispensable tool for cryptographic proofs. As we observe in this work, the lemma in [8], in spite of its general nature, is insufficient to prove certain types of statements that attempt to use the above rewinding strategy. As we will see, the reason for this is that [8] only talks about the probability that one obtains from $\mathcal{C}$ two successful runs and related answers σ, σ'. However, it says nothing about the distribution of the answers σ, σ'. As a simple example, one might be interested in the probability that $\mathcal{C}$ returns in both runs the most likely answer $\hat{\sigma}$ corresponding to a successful run of $\mathcal{C}$ (where the probability is over I, ω, and h). There seems to be no way to infer such a statement from [8]. For this reason, we now prove a more general version of the forking lemma called *subset forking lemma*, which is closer to the proof strategy of [29]. At a high level, the subset forking lemma considers an adversary $\mathcal{C}$ that obtains Q challenges $\boldsymbol{h}$ in addition to the instance I and randomness ω. It produces an answer σ that we refer to as the *side output* below. If $\mathcal{C}$ is successful, then one can relate σ to one of the Q challenges that $\mathcal{C}$ has obtained over the course of its run, say $\boldsymbol{h}_j$. Therefore, we can associate any successful run of $\mathcal{C}$ with the corresponding $j \in [Q]$ and an unsuccessful run of $\mathcal{C}$ with the value $j = 0$. We denote below as $\mathcal{W}_j$ the set of inputs $(I, \omega, \boldsymbol{h})$ to $\mathcal{C}$ for which $\mathcal{C}$ produces an output of the form $(j, \sigma), j \geq 1$. One can view the set $\mathcal{W} := \bigcup_j \mathcal{W}_j$ as the set of all such triples for which C is successful.

Lemma 6.1 (Subset Forking Lemma). *Fix any integer $Q \geq 1$ and a set $\mathcal{H}$ of size ≥ 2 as well as a set of side outputs Σ, instances $\mathcal{I}$, and a randomness space Ω. Let C be an algorithm that on input $(I, \boldsymbol{h}) \in \mathcal{I} \times \mathcal{H}^Q$ and randomness $\omega \in \Omega$ returns a tuple (j, σ), where $0 \leq j \leq Q$ and $\sigma \in \Sigma$. We partition its input space $\mathcal{I} \times \Omega \times \mathcal{H}^Q$ into sets $\mathcal{W}_1, \dots, \mathcal{W}_Q$ where for fixed $1 \leq j \leq Q$, $\mathcal{W}_j$ is the set of all $(I, \omega, \boldsymbol{h})$ that result in $(j, \sigma) \leftarrow \mathsf{C}(\boldsymbol{h}, I; \omega)$ for some arbitrary side output σ.*

For any $1 \leq j \leq Q$ and $\mathcal{B} \subseteq \mathcal{W}_j$ define

$$\mathtt{acc}(\mathcal{B}) := \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} \mathcal{I} \times \Omega \times \mathcal{H}^Q} [(I, \omega, \boldsymbol{h}) \in \mathcal{B}]$$

$$\mathtt{frk}(\mathcal{B}, j) := \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} \mathcal{I} \times \Omega \times \mathcal{H}^Q, \boldsymbol{h}' \xleftarrow{\$} \mathcal{H}^Q | \boldsymbol{h}_{[j-1]}} \left[\begin{array}{c} (\boldsymbol{h}_j \neq \boldsymbol{h}'_j) \wedge \\ ((I, \omega, \boldsymbol{h}) \in \mathcal{B}) \wedge ((I, \omega, \boldsymbol{h}') \in \mathcal{B}) \end{array} \right].$$

Then

$$\mathtt{frk}(\mathcal{B}, j) \geq \mathtt{acc}(\mathcal{B}) \cdot \left(\frac{\mathtt{acc}(\mathcal{B})}{4} - \frac{1}{|\mathcal{H}|} \right).$$

Proof. By applying *Theorem* 2.2 to $\varepsilon = \mathtt{acc}(B)$, $\alpha := \varepsilon/2$, and to the two sets $\mathcal{X} = \mathcal{I} \times \Omega \times \mathcal{H}^{j-1}$ and $\mathcal{Y} = \mathcal{H}^{Q-j+1}$, we obtain

$$\Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} \mathcal{I} \times \Omega \times \mathcal{H}^Q, \boldsymbol{h}' \xleftarrow{\$} \mathcal{H}^Q | \boldsymbol{h}_{[j-1]}} [(I, \omega, \boldsymbol{h}) \in \mathcal{B} \wedge (I, \omega, \boldsymbol{h}') \in \mathcal{B}] \geq \frac{\mathtt{acc}^2(\mathcal{B})}{4}.$$

Next, we observe that

$$\begin{aligned}
&\mathtt{frk}(\mathcal{B}, j) = \Pr[(I, \omega, \boldsymbol{h}) \in \mathcal{B} \wedge (I, \omega, \boldsymbol{h}') \in \mathcal{B} \wedge \boldsymbol{h}_j \neq \boldsymbol{h}'_j] \\
&= \Pr[(I, \omega, \boldsymbol{h}) \in \mathcal{B} \wedge (I, \omega, \boldsymbol{h}') \in \mathcal{B}] - \Pr[(I, \omega, \boldsymbol{h}) \in \mathcal{B} \wedge (I, \omega, \boldsymbol{h}') \in \mathcal{B} \wedge \boldsymbol{h}_j = \boldsymbol{h}'_j] \\
&\geq \Pr[(I, \omega, \boldsymbol{h}) \in \mathcal{B} \wedge (I, \omega, \boldsymbol{h}') \in \mathcal{B}] - \Pr[(I, \omega, \boldsymbol{h}) \in \mathcal{B} \wedge \boldsymbol{h}_j = \boldsymbol{h}'_j] \\
&= \Pr[(I, \omega, \boldsymbol{h}) \in \mathcal{B} \wedge (I, \omega, \boldsymbol{h}') \in \mathcal{B}] - \frac{\Pr[(I, \omega, \boldsymbol{h}) \in \mathcal{B}]}{|\mathcal{H}|},
\end{aligned}$$

[2]This discussion is (intentionally) somewhat simplified; the lemma in [8] actually considers an algorithm $\mathcal{C}$ that obtains partially identical *vectors* of challenges $\boldsymbol{h}, \boldsymbol{h}'$.

where the last equation follows from independence and uniformity of $\boldsymbol{h}_j$ and $\boldsymbol{h}'_j$. We continue with

$$
\begin{aligned}
&= \Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{B} \wedge (I,\omega,\boldsymbol{h}') \in \mathcal{B}] - \frac{\Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{B}]}{|\mathcal{H}|} \\
&\geq \frac{\mathtt{acc}^2(\mathcal{B})}{4} - \frac{\Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{B}]}{|\mathcal{H}|} = \frac{\mathtt{acc}^2(\mathcal{B})}{4} - \frac{\mathtt{acc}(\mathcal{B})}{|\mathcal{H}|} \\
&= \mathtt{acc}(\mathcal{B}) \cdot \left(\frac{\mathtt{acc}(\mathcal{B})}{4} - \frac{1}{|\mathcal{H}|} \right),
\end{aligned}
$$

which completes the proof. ■

We remark that Theorem 6.1 implies the version of the Forking Lemma in [8]. Below, we consider $\mathtt{frk} := \sum_{j=1}^{Q} \mathtt{frk}(\mathcal{W}_j, j)$ which can be seen as the probability of simply running $\mathcal{C}$ successfully twice and obtaining two outputs of the form $(j,\sigma), (j,\sigma')$, where $j \in [Q]$. We also write $\mathtt{acc}$ to denote the probability of $\mathcal{C}$ producing a successful output $(j,\sigma), j \in [Q]$. Using the notation from Theorem 6.1, we obtain:

Corollary 6.2 ([8]). *Let $\mathcal{W} = \bigcup_j \mathcal{W}_j$. Define*

$$
\mathtt{acc} := \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} \mathcal{I} \times \Omega \times \mathcal{H}^Q} [(I,\omega,\boldsymbol{h}) \in \mathcal{W}]
$$

and

$$
\mathtt{frk} := \sum_{j=1}^{Q} \mathtt{frk}(\mathcal{W}_j, j).
$$

Then

$$
\mathtt{frk} \geq \mathtt{acc} \cdot \left(\frac{\mathtt{acc}}{4Q} - \frac{1}{|\mathcal{H}|} \right).
$$

Proof. Note that $\mathtt{acc} = \sum_{j=1}^{Q} \mathtt{acc}(\mathcal{W}_j)$. It follows that

$$
\begin{aligned}
\mathtt{frk} = \sum_{j=1}^{Q} \mathtt{frk}(\mathcal{W}_j, j) &\geq \sum_{j=1}^{Q} \mathtt{acc}(\mathcal{W}_j) \cdot \left(\frac{\mathtt{acc}(\mathcal{W}_j)}{4} - \frac{1}{|\mathcal{H}|} \right) \\
&= \left(\sum_{j=1}^{Q} \frac{\mathtt{acc}^2(\mathcal{W}_j)}{4} \right) - \frac{\mathtt{acc}}{|\mathcal{H}|} \geq \frac{1}{4Q} \left(\sum_{j=1}^{Q} \mathtt{acc}(\mathcal{W}_j) \right)^2 - \frac{\mathtt{acc}}{|\mathcal{H}|} \\
&= \frac{1}{4Q} \mathtt{acc}^2 - \frac{\mathtt{acc}}{|\mathcal{H}|} = \mathtt{acc} \cdot \left(\frac{\mathtt{acc}}{4Q} - \frac{1}{|\mathcal{H}|} \right),
\end{aligned}
$$

where the first inequality follows from Theorem 6.1 and the second inequality follows from Theorem 2.3. ■

7 Proof of Theorem 4.3

Before we give the proof of Theorem 4.3, we provide some intuition about the difficulty that arises in the context of proving the **OMMIM**-security of $\mathsf{ID}[\mathsf{LF}]$ and how our proof overcomes it. The main issue is that the adversary M in **OMMIM** can interleave sessions between the oracles $\mathsf{P}_1, \mathsf{P}_2$ and $\mathtt{Ch}, \mathtt{Ver}$. This gives M strong adaptive capabilities which lead to the ROS-attack described in Section 4.2. The ROS-attack is reflected in Corollary 7.6, which can be translated into an upper bound on M's success probability of providing our reduction with two identical values $\hat{\chi}, \hat{\chi}'$ that result from running the adversary twice with fixed public key pk and randomness ω, but (partially) different replies $\boldsymbol{h}, \boldsymbol{h}'$ to $\mathtt{Ch}$. If the adversary succeeds in setting $\hat{\chi} = \hat{\chi}'$, the reduction fails in recovering values $\hat{\chi} \neq \hat{\chi}'$ s.t. $\mathsf{F}(\hat{\chi}) = \mathsf{F}(\hat{\chi}')$.

To prove the bound in Corollary 7.6, our proof follows the ideas of [29], but takes into account also the abandoned sessions with P_1, which [29] does not consider.[3] The intuitive idea behind ensuring $\hat{\chi} \neq \hat{\chi}'$ is to run M on input pk that could be the result of applying F to either sk or $\hat{sk} = sk + z^*$ from the domain $\mathcal{D}$ of F. One can show that from M's perspective, the resulting view is identical in both cases (Lemma 7.4). On the other hand, since $\hat{\chi}$ depends non-trivially on sk (or $\hat{sk}$, respectively), it should take (with high probability) different values from the reduction's point of view, depending on whether the reduction used sk or $sk + z^*$ as a preimage to pk. Indeed, this intuition is supported by Corollary 7.6. However, Corollary 7.6 can only be translated into an upper bound on the probability that $\hat{\chi}$ takes the same *particular* value $C(sk, \omega, \boldsymbol{h})$, regardless of whether sk or $\hat{sk}$ was used by the reduction. Intuitively, $C(sk, \omega, \boldsymbol{h})$ is the value that is most likely taken by the random variable $\hat{\chi}'$, which occurs as the result of rewinding M with the same sk, ω, but a partially different set of $\mathtt{Ch}$-replies $\boldsymbol{h}'$ (i.e., the probability is over the resampled values in $\boldsymbol{h}'$). To ensure that $\hat{\chi} \neq \hat{\chi}'$, the analysis first defines the set $\mathcal{B}$ of tuples $(sk, \omega, \boldsymbol{h})$ which yield a successful run of M, but for which $\hat{\chi}(sk, \omega, \boldsymbol{h}) \neq C(sk, \omega, \boldsymbol{h})$. It then estimates the probability that both tuples $(sk, \omega, \boldsymbol{h}), (sk, \omega, \boldsymbol{h}')$ that are used to run M, are tuples from the set $\mathcal{B}$. The final step of the proof is to leverage this fact to obtain a lower bound on the success probability of the reduction, i.e., to ensure that $\hat{\chi} \neq \hat{\chi}'$ (Lemma 7.1). To argue that not only both runs of M are successful, but yield tuples in $\mathcal{B}$, we require the subset forking lemma introduced in Section 6.

7.1 The Reduction Algorithm

Let M be an adversary that breaks $(\varepsilon, t, Q_{\mathsf{Ch}}, Q_{\mathsf{P}_1}, Q_{\mathsf{P}_2})$-OMMIM security of $\mathsf{ID}[\mathsf{LF}]$. We show an adversary B that breaks (ε', t')-collision resistance of LF.

Without loss of generality, we will assume throughout the proof that $Q_{\mathsf{P}_1}(\mathsf{M}) = Q_{\mathsf{P}_1}, Q_{\mathsf{P}_2}(\mathsf{M}) = Q_{\mathsf{P}_2}, Q_{\mathsf{Ch}}(\mathsf{M}) = Q_{\mathsf{Ch}}, \ell(\mathsf{M}) = Q_{\mathsf{P}_2} + 1$, as well as $Q_{\mathsf{P}_1} \geq Q_{\mathsf{P}_2}$. For $1 \leq i \leq Q_{\mathsf{P}_2} + 1$, we define an auxiliary algorithm A_i which 'sandboxes' M and that will be used later by adversary B to break collision resistance of LF. More concretely, A_i obtains as input an instance $I = (sk, par)$, runs M on random tape ω and uses vector $\boldsymbol{h} \in \mathcal{C}^{Q_{\mathsf{Ch}}}$ to answer M's Q_{Ch} queries to $\mathtt{Ch}$. Throughout the proof, we will denote with $q = |\mathcal{C}| \geq 2^{2\kappa}$ the size of the challenge space $\mathcal{C} = \mathcal{C}(par)$. The description of algorithm A_i is given in Figure 13. Note that A_i is deterministic for fixed randomness ω.

Analysis of A_i. To analyse A_i, we now introduce some notation. First, consider the variables $\hat{\boldsymbol{J}}_i, \hat{\boldsymbol{\chi}}_i, \hat{\boldsymbol{s}}'$, and $\hat{\boldsymbol{h}}_i$ defined on Lines 30 through 33 of Figure 13. These variables are introduced to simplify the referencing of values associated with successful calls to the verification oracle $\mathtt{Ver}(vSid, \cdot)$ over the course of the proof. Concretely, the variable

$$\hat{\boldsymbol{\chi}}_i = \hat{\boldsymbol{s}}'_i - \hat{\boldsymbol{h}}_i \cdot sk$$

results from the *i-th successful call* to the verification oracle $\mathtt{Ver}(vSid, \cdot)$, whereas the index $\hat{\boldsymbol{J}}_i$ indicates which session identity $vSid$ corresponds to this call.

We will fix an execution of A_i via the tuples $I = (sk, par)$, $\boldsymbol{h}$, and A_i's randomness ω. We define the set $\mathcal{W}$ of *successful inputs of* A_i as the set of all tuples $(I, \omega, \boldsymbol{h})$ which lead to a successful run of A_i, i.e.,

$$\mathcal{W} := \{(I, \omega, \boldsymbol{h}) \mid \hat{\boldsymbol{J}}_i \neq 0; (\hat{\boldsymbol{J}}_i, \hat{\boldsymbol{\chi}}_i) \leftarrow \mathsf{A}_i(I, \boldsymbol{h}; \omega)\}$$

Note that $\mathcal{W}$ is independent of i and, by construction of A_i,

$$\Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I} \times \Omega \times \mathcal{C}^{Q_{\mathsf{V}}})}[(I, \omega, \boldsymbol{h}) \in \mathcal{W}] = \mathbf{Adv}^{\mathbf{OMMIM}}_{\mathsf{ID}[\mathsf{LF}]}(\mathsf{M}) = \varepsilon.$$

We can view $\hat{\boldsymbol{J}}_i, \hat{\boldsymbol{\chi}}_i, \hat{\boldsymbol{s}}'$, and $\hat{\boldsymbol{h}}_i$ as random variables whose distribution is induced by the the uniform distribution on $(\mathcal{I} \times \Omega \times \mathcal{C}^{Q_{\mathsf{V}}})$. Furthermore, their outcome is uniquely determined given $(I, \omega, \boldsymbol{h}) \in \mathcal{W}$, so let us write

$$\left(\hat{\boldsymbol{J}}_i(I, \omega, \boldsymbol{h}), \hat{\boldsymbol{\chi}}_i(I, \omega, \boldsymbol{h})\right) \leftarrow \mathsf{A}_i(I, \boldsymbol{h}; \omega).$$

In the following, when stating probability distributions over I, ω, and $\boldsymbol{h}$, unless specified differently, we will always refer to the uniform distributions. That is, $(I, \omega, \boldsymbol{h}) \xleftarrow{\$} (\mathcal{I} \times \Omega \times \mathcal{C}^{Q_{\mathsf{V}}})$. We consider the following probability for fixed $(I, \omega, \boldsymbol{h}), j, c$ and i:

$$\Pr_{\boldsymbol{h}' \xleftarrow{\$} \mathcal{C}^{Q_{\mathsf{V}}} | \boldsymbol{h}_{[j-1]}}[\hat{\boldsymbol{J}}_i(I, \omega, \boldsymbol{h}') = j \wedge \hat{\boldsymbol{\chi}}_i(I, \omega, \boldsymbol{h}') = c], \tag{4}$$

[3] However, the same bound was given (without proof) by Pointcheval in Theorem 5 of [27].

Adversary $\mathsf{A}_i(I = (sk, par), \boldsymbol{h}; \omega)$:

Parse $(\omega_{\mathsf{M}}, \boldsymbol{r}) \leftarrow \omega$
$\boldsymbol{R} \leftarrow \mathsf{F}(\boldsymbol{r})$
$pk \leftarrow (\mathsf{F}(sk), par)$
$ctr \leftarrow 0; pSid \leftarrow 0; vSid \leftarrow 0$
$\mathsf{M}^{\mathsf{P}_1,\mathsf{P}_2,\mathsf{Ch},\mathsf{Ver}}(pk)$
$\ell(\mathsf{M}) \leftarrow \#\{k \mid \mathbf{vSess}_k = \texttt{closed} \wedge \boldsymbol{b}'_k = 1\}$
$Q_{\mathsf{P}_2}(\mathsf{M}) \leftarrow \#\{k \mid \mathbf{pSess}_k = \texttt{closed}\}$
$Q_{\mathsf{P}_1}(\mathsf{M}) \leftarrow \#\{k \mid \mathbf{pSess}_k = \texttt{open}\}$
$Q_{\mathsf{Ch}}(\mathsf{M}) \leftarrow vSid$
If $(\ell(\mathsf{M}) \geq Q_{\mathsf{P}_2}(\mathsf{M}) + 1)$: Return $(\hat{\boldsymbol{J}}_i, \hat{\boldsymbol{\chi}}_i)$
Return $(\hat{\boldsymbol{J}}_i, \hat{\boldsymbol{\chi}}_i) \leftarrow (0, 0)$

Procedure P_1

$pSid \leftarrow pSid + 1$
$\mathbf{pSess}_{pSid} \leftarrow \texttt{open}$
$\boldsymbol{c}_{pSid} \leftarrow \bot$
Return $(pSid, \boldsymbol{R}_{pSid})$

Procedure $\mathsf{P}_2(pSid, c)$

If $\mathbf{pSess}_{pSid} \neq \texttt{open}$: Return $\bot$
$\mathbf{pSess}_{pSid} \leftarrow \texttt{closed}$
$\boldsymbol{s}_{pSid} \leftarrow c \cdot sk + \boldsymbol{r}_{pSid}$
$\boldsymbol{c}_{pSid} \leftarrow c$
Return $\boldsymbol{s}_{pSid}$

Procedure $\mathsf{Ch}(R')$

$vSid \leftarrow vSid + 1$
$\boldsymbol{R}'_{vSid} \leftarrow R'$
$\mathbf{vSess}_{pSid} \leftarrow \texttt{open}$
Return $(vSid, \boldsymbol{h}_{vSid})$

Procedure $\mathsf{Ver}(vSid, s')$

If $\mathbf{vSess}_{vSid} \neq \texttt{open}$: Return $\bot$
$\boldsymbol{S}'_{vSid} \leftarrow \mathsf{F}(s')$
$\mathbf{vSess}_{vSid} \leftarrow \texttt{closed}$
$\boldsymbol{b}'_{vSid} \leftarrow 0$
If $\boldsymbol{S}'_{vSid} = \boldsymbol{h}_{vSid} \cdot pk + \boldsymbol{R}'_{vSid}$:
    $ctr \leftarrow ctr + 1$
    $\hat{\boldsymbol{s}}'_{ctr} \leftarrow s'$
    $\hat{\boldsymbol{h}}_{ctr} \leftarrow \boldsymbol{h}_{vSid}$
    $\hat{\boldsymbol{\chi}}_{ctr} \leftarrow \hat{\boldsymbol{s}}'_{ctr} - \hat{\boldsymbol{h}}_{ctr} \cdot sk$
    $\hat{\boldsymbol{J}}_{ctr} \leftarrow vSid$
    $\boldsymbol{b}'_{vSid} \leftarrow 1$
Return $\boldsymbol{b}'_{vSid}$

Figure 13: Wrapping adversaries A_i for $1 \leq i \leq Q_{\mathsf{P}_2} + 1$

Adversary $\mathsf{B}(par)$:

$i^* \xleftarrow{\$} [Q_{\mathsf{P}_2} + 1]$
$\boldsymbol{h} \xleftarrow{\$} \mathcal{C}^{Q_{\mathsf{Ch}}}$
$\omega \xleftarrow{\$} \Omega$
$sk \xleftarrow{\$} \mathcal{D}$
$(\hat{\boldsymbol{J}}_{i^*}, \hat{\boldsymbol{\chi}}_{i^*}) \leftarrow \mathsf{A}_{i^*}(I = (sk, par), \boldsymbol{h}; \omega)$ //First execution of A_{i^*}
If $\hat{\boldsymbol{J}}_{i^*} = 0$: Return $\bot$
$\boldsymbol{h}' \xleftarrow{\$} \mathcal{C}^{Q_{\mathsf{V}}} | \boldsymbol{h}_{[\hat{\boldsymbol{J}}_{i^*} - 1]}$ //Conditionally resample $\boldsymbol{h}'$
$(\hat{\boldsymbol{J}}'_{i^*}, \hat{\boldsymbol{\chi}}'_{i^*}) \leftarrow \mathsf{A}_{i^*}(I = (sk, par), \boldsymbol{h}'; \omega)$ //Second execution of A_{i^*}
If $(\hat{\boldsymbol{J}}'_{i^*} = \hat{\boldsymbol{J}}_{i^*}) \wedge (\hat{\boldsymbol{\chi}}_{i^*} \neq \hat{\boldsymbol{\chi}}'_{i^*})$: Return $(\hat{\boldsymbol{\chi}}_{i^*}, \hat{\boldsymbol{\chi}}'_{i^*})$
Return $\bot$

Figure 14: Adversary B against collision resistance of LF.

where the conditional probability $\boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}}|\boldsymbol{h}_{[j-1]}$ was introduced in Section 2.

We denote by $c_{i,j}(I,\omega,\boldsymbol{h})$ the lexicographically first value c s.t. the probability in (4) is maximized when $(I,\omega,\boldsymbol{h}), j, i$ are fixed. We then write $C_i(I,\omega,\boldsymbol{h}) = c_{i,\hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h})}(I,\omega,\boldsymbol{h})$. For fixed i,j, let us define $\mathcal{B}_{i,j} \subset \mathcal{W}$ as

$$\mathcal{B}_{i,j} := \{(I,\omega,\boldsymbol{h}) \in \mathcal{W} \mid \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h}) = j \wedge \hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h}) \neq C_i(I,\omega,\boldsymbol{h})\}.$$

and

$$\beta_{i,j} = \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_\mathsf{V}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{B}_{i,j}]$$

$$\delta_{i,j} = \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_\mathsf{V}}), \boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}}|\boldsymbol{h}_{[j-1]}}\left[\begin{array}{l}\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) \neq \hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h}) \\ \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h}) = \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j\end{array}\right].$$

Lemma 7.1 *For all i,j: $\delta_{i,j} \geq \beta_{i,j}\left(\frac{\beta_{i,j}}{8} - \frac{1}{2q}\right)$.*

The proof of this lemma is postponed to Section 7.2.

Lemma 7.2 *There exist $i \in [Q_{\mathsf{P}_2}+1], j \in [Q_\mathsf{Ch}]$ such that* $\beta_{i,j} > \left(\varepsilon - \frac{Q_\mathsf{V}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}\right) \cdot \frac{1}{2Q_\mathsf{V}(Q_{\mathsf{P}_2}+1)}$.

The proof of this lemma is postponed to Section 7.3.

Adversary B against Collision Resistance of LF. We are now ready to describe our adversary B depicted in Figure 14, which plays in the collision resistance game of LF. B works roughly as follows. On input $par \xleftarrow{\$} \mathsf{PGen}(1^\kappa)$, It first samples randomness $\omega \xleftarrow{\$} \Omega$, a secret key $sk \xleftarrow{\$} \mathcal{D}$, a vector $\boldsymbol{h} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{Ch}}$, and an index $i^* \xleftarrow{\$} [Q_{\mathsf{P}_2}+1]$ and runs A_{i^*} on input $(I = (sk, par), \boldsymbol{h}; \omega)$. It samples a second random vector $\boldsymbol{h'}$ as $\boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}}|\boldsymbol{h}_{[\hat{\boldsymbol{J}}_{i^*}-1]}$ and runs A_{i^*} a second time with the same randomness ω and the same instance I, but replacing $\boldsymbol{h}$ by $\boldsymbol{h'}$. In the case that B does not abort, note that by definition of A_{i^*},

$$\begin{aligned}\mathsf{F}(\hat{\boldsymbol{\chi}}_{i^*}) &= \mathsf{F}(\hat{\boldsymbol{s}}'_{i^*} - \hat{\boldsymbol{h}}_{i^*}\cdot sk) \\ &= \boldsymbol{S}'_{\hat{\boldsymbol{J}}_{i^*}} - \boldsymbol{h}_{\hat{\boldsymbol{J}}_{i^*}} \cdot pk = \boldsymbol{R}'_{\hat{\boldsymbol{J}}_{i^*}}\end{aligned}$$

Because A_{i^*} sees identical answers for the first $\hat{\boldsymbol{J}}_{i^*}-1$ queries to $\mathtt{Ch}$, it behaves identically in both runs until it receives the answer to the $\hat{\boldsymbol{J}}_{i^*}$-th query to $\mathtt{Ch}$. In particular, A_{i^*} poses the same $\hat{\boldsymbol{J}}_{i^*}$-th query to $\mathtt{Ch}$ which means that $\mathsf{F}(\hat{\boldsymbol{\chi}}'_{i^*}) = \boldsymbol{R}'_{\hat{\boldsymbol{J}}_{i^*}}$ and therefore also $\mathsf{F}(\hat{\boldsymbol{\chi}}_{i^*}) = \mathsf{F}(\hat{\boldsymbol{\chi}}'_{i^*})$. We now consider

$$\begin{aligned}\varepsilon' = \mathbf{Adv}^{\mathbf{CR}}_{\mathsf{LF}}(\mathsf{B}) &= \Pr_{par \xleftarrow{\$} \mathsf{PGen}(1^\kappa), (\hat{\boldsymbol{\chi}}_{i^*}, \hat{\boldsymbol{\chi}}'_{i^*}) \xleftarrow{\$} \mathsf{B}(par)}[\hat{\boldsymbol{\chi}}_{i^*} \neq \hat{\boldsymbol{\chi}}'_{i^*} \wedge \mathsf{F}(\hat{\boldsymbol{\chi}}_{i^*}) = \mathsf{F}(\hat{\boldsymbol{\chi}}'_{i^*})] \\ &= \sum_{j=1}^{Q_\mathsf{Ch}} \Pr[\hat{\boldsymbol{\chi}}_{i^*} \neq \hat{\boldsymbol{\chi}}'_{i^*} \wedge \mathsf{F}(\hat{\boldsymbol{\chi}}_{i^*}) = \mathsf{F}(\hat{\boldsymbol{\chi}}'_{i^*}) \wedge \hat{\boldsymbol{J}}_{i^*} = \hat{\boldsymbol{J}}'_{i^*} = j] \\ &= \sum_{j=1}^{Q_\mathsf{Ch}} \Pr[\hat{\boldsymbol{\chi}}_{i^*} \neq \hat{\boldsymbol{\chi}}'_{i^*} \wedge \hat{\boldsymbol{J}}_{i^*} = \hat{\boldsymbol{J}}'_{i^*} = j] = \sum_{j=1}^{Q_\mathsf{Ch}} \delta_{i^*,j} \\ &\geq \frac{1}{Q_{\mathsf{P}_2}+1} \cdot \max_{i\in[Q_{\mathsf{P}_2}+1]} \sum_{j=1}^{Q_\mathsf{Ch}} \delta_{i,j} \\ &\geq \max_{i,j} \frac{\beta_{i,j}}{2(Q_{\mathsf{P}_2}+1)}\left(\frac{\beta_{i,j}}{4} - \frac{1}{q}\right),\end{aligned}$$

where for the first inequality we used that $\sum \delta_{i^*,j} = \max_i \sum \delta_{i,j}$ with probability at least $1/(Q_{\mathsf{P}_2}+1)$ and in the last step we applied Lemma 7.1. By Lemma 7.2 we finally obtain

$$\begin{aligned}\varepsilon' &\geq \frac{\varepsilon - \frac{Q_\mathsf{V}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}}{32Q_\mathsf{Ch}^2(Q_{\mathsf{P}_2}+1)^3} \cdot \left(\varepsilon - \frac{Q_\mathsf{V}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q} - \frac{16Q_\mathsf{Ch}^2(Q_{\mathsf{P}_2}+1)^2}{q}\right) \\ &= O\left(\left(\varepsilon - \frac{(Q_\mathsf{Ch}Q_{\mathsf{P}_1})^{Q_{\mathsf{P}_2}+1}}{q}\right)^2 \frac{1}{Q_\mathsf{Ch}^2 Q_{\mathsf{P}_2}^3}\right) = O\left(\left(\varepsilon - \frac{(Q_\mathsf{Ch}Q_{\mathsf{P}_1})^{Q_{\mathsf{P}_2}+1}}{2^{2\kappa}}\right)^2 \frac{1}{Q_\mathsf{Ch}^2 Q_{\mathsf{P}_2}^3}\right),\end{aligned}$$

where the second-to-last equality holds for $Q_{\mathsf{P}_1} \geq Q_{\mathsf{P}_2}$.

7.2 Proof of Lemma 7.1

We will show in the following that for all $(I, \omega, \boldsymbol{h}) \xleftarrow{\$} (\mathcal{I} \times \Omega \times \mathcal{C}^{Q_\mathsf{V}}), d \in \mathcal{D}$:

$$\begin{aligned}\alpha_{i,j}(I,\omega,\boldsymbol{h},d) &:= \Pr_{\boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}} | \boldsymbol{h}_{[j-1]}}[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) \neq d \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j] \\ &\geq \mu_{i,j}(I,\omega,\boldsymbol{h})/2, \end{aligned} \tag{5}$$

where

$$\mu_{i,j}(I,\omega,\boldsymbol{h}) := \Pr_{\boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}} | \boldsymbol{h}_{[j-1]}}[(I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j} \wedge \boldsymbol{h}_j \neq \boldsymbol{h}'_j].$$

For a true/false statement s, define $B(s)$ as 1 if s is true and 0 otherwise. It is easy to see that (5) implies the theorem statement since

$$\begin{aligned}\delta_{i,j} &= \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_\mathsf{V}}), \boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}} | \boldsymbol{h}_{[j-1]}}\left[\begin{array}{l}\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) \neq \hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h}) \\ \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h}) = \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j\end{array}\right] \\ &= \sum_d \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_\mathsf{V}}), \boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}} | \boldsymbol{h}_{[j-1]}}\left[\begin{array}{l}\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) \neq d \wedge \hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h}) = d \\ \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h}) = \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j\end{array}\right] \\ &= \sum_d \mathbf{E}_{I,\omega,\boldsymbol{h}}[B(\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h}) = d \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h}) = j) \cdot \alpha_{i,j}(I,\omega,\boldsymbol{h},d)] \\ &\geq \frac{1}{2}\sum_d \mathbf{E}_{I,\omega,\boldsymbol{h}}[B(\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h}) = d \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h}) = j) \cdot \mu_{i,j}(I,\omega,\boldsymbol{h})],\end{aligned}$$

where in the last step, we have applied linearity and monotonicity of the expectation and the fact that due to (5), for all $I, \omega, \boldsymbol{h} \in \mathcal{C}^{Q_{\mathsf{Ch}}}, d$, we have $\alpha_{i,j}(I,\omega,\boldsymbol{h},d) \geq \mu_{i,j}(I,\omega,\boldsymbol{h})/2$. We continue with

$$\begin{aligned}&\frac{1}{2}\sum_d \mathbf{E}_{I,\omega,\boldsymbol{h}}[B(\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h}) = d \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h}) = j) \cdot \mu_{i,j}(I,\omega,\boldsymbol{h})] \\ &= \frac{1}{2}\cdot\sum_d \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_\mathsf{V}}), \boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}} | \boldsymbol{h}_{[j-1]}}\left[\begin{array}{l}\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h}) = d \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h}) = j \\ \wedge (I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j} \wedge \boldsymbol{h}_j \neq \boldsymbol{h}'_j\end{array}\right] \\ &= \frac{1}{2}\cdot \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_\mathsf{V}}), \boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}} | \boldsymbol{h}_{[j-1]}}\left[\begin{array}{l}\hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h}) = j \\ \wedge (I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j} \wedge \boldsymbol{h}_j \neq \boldsymbol{h}'_j\end{array}\right] &(6) \\ &\geq \frac{1}{2}\cdot \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_\mathsf{V}}), \boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}} | \boldsymbol{h}_{[j-1]}}[(I,\omega,\boldsymbol{h}) \in \mathcal{B}_{i,j} \wedge (I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j} \wedge \boldsymbol{h}_j \neq \boldsymbol{h}'_j] &(7) \\ &= \frac{1}{2}\cdot \mathtt{frk}(\mathcal{B}_{i,j}, j) &(8) \\ &\geq \beta_{i,j}\left(\beta_{i,j}/8 - \frac{1}{2q}\right), &(9)\end{aligned}$$

where from (6) to (7), we have used the fact that $(I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j}$ implies $\hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j$. The inequality from (8) to (9) follows directly from Lemma 6.1. We prove (5) by analyzing two cases. For all $I, \omega, \boldsymbol{h}, d$, we define

$$\gamma_{i,j}(I,\omega,\boldsymbol{h},d) := \Pr_{\boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_\mathsf{V}} | \boldsymbol{h}_{[j-1]}}[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) = d \wedge (I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j} \wedge h_j \neq h'_j].$$

Case 1: $\gamma_{i,j}(I,\omega,\boldsymbol{h},d) \geq \mu_{i,j}(I,\omega,\boldsymbol{h})/2$. Note that in this case we can assume $d \neq c_{i,j}(I,\omega,\boldsymbol{h})$. This is because if $d = c_{i,j}(I,\omega,\boldsymbol{h})$, then

$$\begin{aligned}\gamma_{i,j}(I,\omega,\boldsymbol{h},d) &\leq \Pr[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) = c_{i,j}(I,\omega,\boldsymbol{h}) \wedge (I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j}] \\ &= \Pr[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) = c_{i,j}(I,\omega,\boldsymbol{h'}) \wedge (I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j}] \\ &= \Pr[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) = C_i(I,\omega,\boldsymbol{h'}) \wedge (I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j}] = 0,\end{aligned}$$

which would trivialize the claim. For the first equality, we have used the fact that $c_{i,j}(I,\omega,\boldsymbol{h}) = c_{i,j}(I,\omega,\boldsymbol{h'})$ for any $\boldsymbol{h} = \boldsymbol{h}'$ that agree on the first $j-1$ entries. For the second equality, we have again used the fact that $(I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j}$ implies $\hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j$. Assuming that $d \neq c_{i,j}(I,\omega,\boldsymbol{h})$, we continue with

$$\begin{aligned}
\alpha_{i,j}(I,\omega,\boldsymbol{h},d) &= \Pr_{\boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_V}|\boldsymbol{h}_{[j-1]}}[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) \neq d \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j] \\
&\geq \Pr[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) = c_{i,j}(I,\omega,\boldsymbol{h}) \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j] \\
&\geq \Pr[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) = d \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j].
\end{aligned}$$

Using once more that $(I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j}$ implies $\hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j$, we obtain

$$\begin{aligned}
&\Pr[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) = d \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j] \geq \Pr[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) = d \wedge (I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j}] \\
&\geq \gamma_{i,j}(I,\omega,\boldsymbol{h},d) \geq \mu_{i,j}(I,\omega,\boldsymbol{h})/2.
\end{aligned}$$

Case 2: $\gamma_{i,j}(I,\omega,\boldsymbol{h},d) < \mu_{i,j}(I,\omega,\boldsymbol{h})/2$. Now,

$$\begin{aligned}
\alpha_{i,j}(I,\omega,\boldsymbol{h},d) &= \Pr_{\boldsymbol{h'} \xleftarrow{\$} \mathcal{C}^{Q_V}|\boldsymbol{h}_{[j-1]}}[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) \neq d \wedge \hat{\boldsymbol{J}}_i(I,\omega,\boldsymbol{h'}) = j] \\
&\geq \Pr[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) \neq d \wedge (I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j} \wedge \boldsymbol{h}_j \neq \boldsymbol{h}'_j] \\
&= \Pr[(I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j} \wedge \boldsymbol{h}_j \neq \boldsymbol{h}'_j] \\
&\quad - \Pr[\hat{\boldsymbol{\chi}}_i(I,\omega,\boldsymbol{h'}) = d \wedge (I,\omega,\boldsymbol{h'}) \in \mathcal{B}_{i,j} \wedge \boldsymbol{h}_j \neq \boldsymbol{h}'_j] \\
&= \mu_{i,j}(I,\omega,\boldsymbol{h}) - \gamma_{i,j}(I,\omega,\boldsymbol{h},d) > \mu_{i,j}(I,\omega,\boldsymbol{h})/2.
\end{aligned}$$

This proves (5) and hence the lemma.

7.3 Proof of Lemma 7.2

Consider again the algorithm A_i in Figure 13 and its internal variables. On input $(I = (sk, par), \omega = (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h})$, A_i invokes M on $pk = \mathsf{F}(sk)$ and randomness ω_{M} and answers its queries using the values in $\boldsymbol{r}, \boldsymbol{h}$. Similarly as before, this allows us to fix an execution of M (within A_i) via a tuple of the form $(I,\omega,\boldsymbol{h}) = (I,(\omega_{\mathsf{M}},\boldsymbol{r}),\boldsymbol{h})$. Let $\boldsymbol{c}(I,\omega,\boldsymbol{h})$ denote the vector of challenge values as defined in Line 18 of Figure 13.

Recall that we have assumed that $\mathsf{F} : \mathcal{D} \longrightarrow \mathcal{R}$ and the existence of a pseudo torsion-free element $z^* \in \mathcal{D} \setminus \{0\}$ such that (i) $\mathsf{F}(z^*) = 0$; and (ii) $\forall s, s' \in \mathcal{C} : s \neq s' \implies s \cdot z^* \neq s' \cdot z^*$.

Lemma 7.3 *Consider the mapping*

$$\begin{aligned}
&\Phi : \mathcal{W} \longrightarrow (\mathcal{I} \times \Omega \times \mathcal{C}^{Q_V}) \\
&((sk, par), (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \mapsto ((sk + z^*, par), (\omega_{\mathsf{M}}, \boldsymbol{r} - z^* \cdot \boldsymbol{c}(I,\omega,\boldsymbol{h})), \boldsymbol{h}),
\end{aligned}$$

where we make the convention that for $v \in \mathcal{D} \cup \mathcal{C} \cup \mathcal{R}, v \cdot \perp := 0$. Then Φ is a permutation on $\mathcal{W}$.

For the proof we require the following claim.

Claim 7.4 Let $(I,\omega,\boldsymbol{h}) \in \mathcal{W}$. Then the tuples $(I,\omega,\boldsymbol{h})$ and $\Phi(I,\omega,\boldsymbol{h})$ fix the same execution of M.

Proof. We show that M sees identical values in both executions corresponding to $(I,\omega,\boldsymbol{h})$ and $\Phi(I,\omega,\boldsymbol{h})$. To this end we consider all values in the view of M.

- **Initial input to M.** Since Φ does not alter the values of ω_{M} and par, we only need to verify that M obtains the same public key in both executions. This is ensured via $\mathsf{F}(sk + z^*) = \mathsf{F}(sk) + \mathsf{F}(z^*) = \mathsf{F}(sk) = pk$

- **Outputs of oracle P_1.** Oracle P_1 consecutively returns the values from $\boldsymbol{R} = \mathsf{F}(\boldsymbol{r})$, as defined in Line 01 of Figure 13. They remain the same in both executions since $\mathsf{F}(\boldsymbol{r}) = \boldsymbol{R} = \boldsymbol{R} - 0 \cdot \boldsymbol{c}(I,\omega,\boldsymbol{h}) = \mathsf{F}(\boldsymbol{r}) - \mathsf{F}(z^*) \cdot \boldsymbol{c}(I,\omega,\boldsymbol{h}) = \mathsf{F}(\boldsymbol{r} - z^* \cdot \boldsymbol{c}(I,\omega,\boldsymbol{h}))$.

- **Outputs of oracle Ver.** Oracle Ver consecutively returns the values from $\boldsymbol{b}'$. They remain the same in both executions since they depend on $\boldsymbol{R}$, $\boldsymbol{h}$, and the randomness ω_{M}.
- **Outputs of oracle $\mathtt{P}_2$.** Oracle $\mathtt{P}_2$ consecutively returns the values from $\boldsymbol{s} = \boldsymbol{c} \cdot sk + \boldsymbol{r}$, as defined in Line 17 of Figure 13. Note that the first value $\boldsymbol{c}_1$ in both executions is the same (as it only depends on values that we have already argued to remain the same in both executions), i.e., $\boldsymbol{c}_1 = \boldsymbol{c}_1(I, \omega, \boldsymbol{h}) = \boldsymbol{c}_1(\Phi(I, \omega, \boldsymbol{h}))$. Thus, $\boldsymbol{s}_1(I, \omega, \boldsymbol{h}) = \boldsymbol{r}_1 + sk \cdot \boldsymbol{c}_1(I, \omega, \boldsymbol{h}) = \boldsymbol{r}_1 - z^* \cdot \boldsymbol{c}_1(I, \omega, \boldsymbol{h}) + z^* \cdot \boldsymbol{c}_1(I, \omega, \boldsymbol{h}) + sk \cdot \boldsymbol{c}_1(I, \omega, \boldsymbol{h}) = (\boldsymbol{r}_1 - z^* \cdot \boldsymbol{c}_1(\Phi(I, \omega, \boldsymbol{h}))) + (sk + z^*) \cdot \boldsymbol{c}_1(\Phi(I, \omega, \boldsymbol{h})) = \boldsymbol{s}_1(\Phi(I, \omega, \boldsymbol{h}))$, where in the second to last step, we have used the distributive law over the pseudo module formed by $\mathcal{C}$ and $\mathcal{D}$. By a simple inductive argument, it now follows that $\boldsymbol{s}(I, \omega, \boldsymbol{h}) = \boldsymbol{s}(\Phi(I, \omega, \boldsymbol{h}))$.

Thus, $(I, \omega, \boldsymbol{h})$ and $\Phi(I, \omega, \boldsymbol{h})$ fix the same executions of M. ∎

Proof of Lemma 7.3. First note that Lemma 7.4 implies that Φ maps to $\mathcal{W}$. It remains to prove that Φ is also a bijection. Suppose Φ is not injective. Thus, for distinct tuples $(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \neq (I', (\omega'_{\mathsf{M}}, \boldsymbol{r}'), \boldsymbol{h}')$, $\Phi\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = \Phi(I', (\omega'_{\mathsf{M}}, \boldsymbol{r}'), \boldsymbol{h}')$. This implies $par = par', \omega_{\mathsf{M}} = \omega'_{\mathsf{M}}$ and $\boldsymbol{h} = \boldsymbol{h}'$. Similarly, $sk + z^* = sk' + z^*$, which implies that $sk = sk'$. Lastly, $\boldsymbol{r} - z^* \cdot \boldsymbol{c}\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = \boldsymbol{r}' - z^* \cdot \boldsymbol{c}(I', (\omega'_{\mathsf{M}}, \boldsymbol{r}'), \boldsymbol{h}')$. Since $\Phi\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = \Phi(I', (\omega'_{\mathsf{M}}, \boldsymbol{r}'), \boldsymbol{h}')$, by Claim 7.4, $(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h})$ and $(I', (\omega'_{\mathsf{M}}, \boldsymbol{r}'), \boldsymbol{h}')$ fix the same execution and therefore also $\boldsymbol{c}\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = \boldsymbol{c}(I', (\omega'_{\mathsf{M}}, \boldsymbol{r}'), \boldsymbol{h}')$. This implies $\boldsymbol{r} = \boldsymbol{r}'$, leading to the contradiction $(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = (I', (\omega'_{\mathsf{M}}, \boldsymbol{r}'), \boldsymbol{h}')$.

To prove that Φ is surjective, we consider the function $\Phi^{-1} : (\mathcal{I} \times \Omega \times \mathcal{C}^{Q_{\mathsf{V}}}) \longrightarrow (\mathcal{I} \times \Omega \times \mathcal{C}^{Q_{\mathsf{V}}})$, defined as $\Phi^{-1}((sk, par), (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = ((sk - z^*, par), (\omega_{\mathsf{M}}, \boldsymbol{r} + z^* \cdot \boldsymbol{c}(I, \omega, \boldsymbol{h})), \boldsymbol{h})$, which is the inverse of Φ. With the same argument as above, one can also prove that Φ^{-1} is injective which implies the surjectivity of Φ. ∎

We now introduce the following notation. Let $\mathcal{B} = \bigcup_{i,j} \mathcal{B}_{i,j}$ and let $\mathcal{G} = \mathcal{W} \setminus \mathcal{B}$. That is, for all $(I, \omega, \boldsymbol{h}) \in \mathcal{G}$, we have $\forall k \in [Q_{\mathsf{P}_2} + 1] : \hat{\boldsymbol{\chi}}_k(I, \omega, \boldsymbol{h}) = C_k(I, \omega, \boldsymbol{h})$. The following combinatorial lemma will help to lower bound the probability that $\hat{\boldsymbol{\chi}}$ takes different values (i.e., differs in at least one component) as a result of distinct instances $I = (sk, par), I' = (sk + z^*, par)$.

Lemma 7.5 *For any fixed $(I, (\omega_{\mathsf{M}}, \boldsymbol{r})) \in \mathcal{I} \times \Omega$,*

$$\Pr_{\boldsymbol{h} \xleftarrow{\$} \mathcal{C}^{Q_{\mathsf{Ch}}}} [(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \in \mathcal{G} \wedge \Phi\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \in \mathcal{G}] \leq \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1} \cdot \binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}.$$

Proof. We argue by contradiction. Thus, assume that for some $(I, (\omega_{\mathsf{M}}, \boldsymbol{r})) \in \mathcal{I} \times \Omega$,

$$\Pr_{\boldsymbol{h} \xleftarrow{\$} \mathcal{C}^{Q_{\mathsf{Ch}}}} [(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \in \mathcal{G} \wedge \Phi\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \in \mathcal{G}] > \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1} \cdot \binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}.$$

Then there exist a set $\{u_1, ..., u_{Q_{\mathsf{P}_2}+1}\}$ of $Q_{\mathsf{P}_2} + 1$ distinct indices from $[Q_{\mathsf{Ch}}]$ such that

$$\Pr_{\boldsymbol{h} \xleftarrow{\$} \mathcal{C}^{Q_{\mathsf{Ch}}}} \left[\begin{array}{l} ((I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \in \mathcal{G}) \wedge \ (\Phi\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \in \mathcal{G}) \\ \wedge \forall j : \hat{\boldsymbol{J}}_j\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = u_j \end{array} \right] > \frac{\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}.$$

Similarly, there exists a vector $\boldsymbol{d} \in (\mathcal{C} \cup \{\perp\})^{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}$ of challenges such that $\boldsymbol{d}$ has exactly Q_{P_1} entries which are $\perp$ and furthermore has the property that

$$\Pr_{\boldsymbol{h} \xleftarrow{\$} \mathcal{C}^{Q_{\mathsf{Ch}}}} \left[\begin{array}{l} ((I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \in \mathcal{G}) \wedge \ (\Phi\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \in \mathcal{G}) \\ \wedge\,(\boldsymbol{c}\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = \boldsymbol{d}) \wedge \Big(\forall j : \hat{\boldsymbol{J}}_j\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = u_j\Big) \end{array} \right] > \frac{1}{q^{Q_{\mathsf{P}_2}+1}}.$$

Lastly, there exists a set $\{v_1, ..., v_{Q_{\mathsf{Ch}}-Q_{\mathsf{P}_2}-1}\}$ of $Q_{\mathsf{Ch}} - Q_{\mathsf{P}_2} - 1$ distinct indices from $[Q_{\mathsf{Ch}}] \setminus \{u_1, ..., u_{Q_{\mathsf{P}_2}+1}\}$ and a vector $(\tilde{\boldsymbol{h}}_{v_1}, ..., \tilde{\boldsymbol{h}}_{v_{Q_{\mathsf{Ch}}-Q_{\mathsf{P}_2}-1}}) \in \mathcal{C}^{Q_{\mathsf{Ch}}-Q_{\mathsf{P}_2}-1}$ such that

$$\Pr_{\boldsymbol{h} \xleftarrow{\$} \mathcal{C}^{Q_{\mathsf{Ch}}}} \left[\begin{array}{l} ((I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \in \mathcal{G}) \wedge (\Phi\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) \in \mathcal{G}) \wedge \ (\boldsymbol{c}\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = \boldsymbol{d}) \\ \wedge \Big(\forall j : (\hat{\boldsymbol{J}}_j\,(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{h}) = u_j\Big) \wedge \big(\forall j : \boldsymbol{h}_{v_j} = \tilde{\boldsymbol{h}}_{v_j}\big) \end{array} \right]$$
$$> \frac{1}{q^{Q_{\mathsf{P}_2}+1} q^{Q_{\mathsf{Ch}}-Q_{\mathsf{P}_2}-1}} = \frac{1}{q^{Q_{\mathsf{Ch}}}}.$$

Since the random variable $\boldsymbol{h}$ takes a particular value $\boldsymbol{k} \in \mathcal{C}^{Q_{\mathsf{Ch}}}$ with probability exactly $q^{-Q_{\mathsf{Ch}}}$, the statement inside the probability term above must be true for at least two distinct vectors $\boldsymbol{k}, \boldsymbol{k'} \in \mathcal{C}^{Q_{\mathsf{Ch}}}$. Furthermore, since the condition in the probability term above fixes all but the $Q_{\mathsf{P}_2} + 1$ components $\{u_1, ..., u_{Q_{\mathsf{P}_2}+1}\}$ of $\boldsymbol{k}$ and $\boldsymbol{k'}$, there exists an index $i \in [Q_{\mathsf{P}_2} + 1]$ s.t. $\boldsymbol{k}_{u_i} \neq \boldsymbol{k}'_{u_i}$.

W.l.o.g., let i be the smallest such index. This implies that $\forall j < u_i : \boldsymbol{k}_j = \boldsymbol{k}'_j$ and $\boldsymbol{k}_{u_i} \neq \boldsymbol{k}'_{u_i}$. Therefore,

$$
\begin{aligned}
&C_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) = c_{i,\hat{\boldsymbol{J}}_i(I,(\omega_{\mathsf{M}},\boldsymbol{r}),\boldsymbol{k})}(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) \\
&= c_{i,u_i}(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) = c_{i,u_i}(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'}) \\
&= c_{i,\hat{\boldsymbol{J}}_i(I,(\omega_{\mathsf{M}},\boldsymbol{r}),\boldsymbol{k'})}(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'}) = C_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'}). \qquad (10)
\end{aligned}
$$

By Lemma 7.4, $\hat{\boldsymbol{J}}_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) = \hat{\boldsymbol{J}}_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})) = u_i$ and $\hat{\boldsymbol{s}}'_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) = \hat{\boldsymbol{s}}'_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}))$. Furhtermore, since also $(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) \in \mathcal{G}$ and $\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) \in \mathcal{G}$, we know that $\hat{\boldsymbol{\chi}}_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) = C_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) = \hat{\boldsymbol{s}}'_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) - sk \cdot \boldsymbol{k}_{u_i}$ and $\hat{\boldsymbol{\chi}}_i\left(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})\right) = C_i\left(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})\right) = \hat{\boldsymbol{s}}'_i\left(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})\right) - (sk + z^*) \cdot \boldsymbol{k}_{u_i}$. Putting things together, we obtain

$$
\begin{aligned}
&C_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) = \hat{\boldsymbol{s}}'_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) - sk \cdot \boldsymbol{k}_{u_i} \\
&= \hat{\boldsymbol{s}}'_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})) - sk \cdot \boldsymbol{k}_{u_i} \\
&= \hat{\boldsymbol{s}}'_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})) - sk \cdot \boldsymbol{k}_{u_i} + z^* \cdot \boldsymbol{k}_{u_i} - z^* \cdot \boldsymbol{k}_{u_i} \\
&= \hat{\boldsymbol{s}}'_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})) - (sk + z^*) \cdot \boldsymbol{k}_{u_i} + z^* \cdot \boldsymbol{k}_{u_i} \\
&= C_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})) + z^* \cdot \boldsymbol{k}_{u_i}, \qquad (11) \\
& \qquad (12)
\end{aligned}
$$

where we have again applied the distributive law over the pseudo module formed by $\mathcal{C}$ and $\mathcal{D}$. Analogously, we infer

$$
C_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'}) = C_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'})) + z^* \cdot \boldsymbol{k}'_{u_i}. \qquad (13)
$$

Combining (in this order) equations 11, 10, and 13, we obtain:

$$
\begin{aligned}
&C_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})) + z^* \cdot \boldsymbol{k}_{u_i} \\
&= C_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) = C_i(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'}) \\
&= C_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'})) + z^* \cdot \boldsymbol{k}'_{u_i}. \qquad (14)
\end{aligned}
$$

Denote again $I' = (sk + z^*, par)$. Since above we have fixed $\boldsymbol{c}(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}) = \boldsymbol{c}(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'}) = \boldsymbol{d}$, we also know that

$$
\begin{aligned}
&C_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})) \qquad (15) \\
&= C_i(I', \omega_{\mathsf{M}}, \boldsymbol{r} - z^* \cdot \boldsymbol{c}(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k}), \boldsymbol{k}) \\
&= C_i(I', \omega_{\mathsf{M}}, \boldsymbol{r} - z^* \cdot \boldsymbol{d}, \boldsymbol{k}) \\
&= C_i(I', \omega_{\mathsf{M}}, \boldsymbol{r} - z^* \cdot \boldsymbol{d}, \boldsymbol{k'}) \qquad (16) \\
&= C_i(I', \omega_{\mathsf{M}}, \boldsymbol{r} - z^* \cdot \boldsymbol{c}(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'}), \boldsymbol{k'}) \\
&= C_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'})), \qquad (17)
\end{aligned}
$$

where 16 follows again from the fact that $\forall j < u_i : \boldsymbol{k}_j = \boldsymbol{k}'_j$ and $\hat{\boldsymbol{J}}_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k})) = \hat{\boldsymbol{J}}_i(\Phi(I, (\omega_{\mathsf{M}}, \boldsymbol{r}), \boldsymbol{k'})) = u_i$. By combining 14 and 17, it now follows that $z^* \cdot \boldsymbol{k}_{u_i} = z^* \cdot \boldsymbol{k}'_{u_i}$. Thus, pseudo torsion-freeness of z^* implies that $\boldsymbol{k}_{u_i} = \boldsymbol{k}'_{u_i}$ which contradicts the assumption that $\boldsymbol{k}_{u_i} \neq \boldsymbol{k}'_{u_i}$. This completes the proof. ■

Corollary 7.6 $\Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I} \times \Omega \times \mathcal{C}^{Q_{\mathsf{V}}})} [(I, \omega, \boldsymbol{h}) \in \mathcal{G} \wedge \Phi(I, \omega, \boldsymbol{h}) \in \mathcal{G}] \leq \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1} \cdot \binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}$.

Discussion. The lower bound in Theorem 7.6 exponentially depreciates with the number Q_{P_2} of parallel sessions allowed in the **OMMIM** experiment. Unfortunately, the ROS-attack in 4.2 shows that the bound in Theorem 7.6 can not be improved beyond a factor of $\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}$. The reason for this is that our attacker computes $\hat{\boldsymbol{\chi}}$ in a manner that does not depend on $\boldsymbol{h}$, but only on ω, I (more precisely, any contribution of $\boldsymbol{h}$–and therefore of *sk*– 'cancels out' in the values returned by the attacker). Therefore, $\hat{\boldsymbol{\chi}}$ *always* takes the 'most likely' value according to 4 in the sense that, regardless of $\boldsymbol{h}$, the attacker can force $(\omega, I, \boldsymbol{h}) \in \mathcal{G}$ *and* $\Phi(\omega, I, \boldsymbol{h}) \in \mathcal{G}$.

Lemma 7.7 $\Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_{\mathsf{V}}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{B}] \geq \frac{1}{2}\left(\varepsilon - \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}\right)$.

Proof. We partition $\mathcal{G}$ into subsets $\mathcal{G}_g, \mathcal{G}_b$ such that all elements in $\mathcal{G}_g$ are mapped into $\mathcal{G}$ via Φ and all elements in $\mathcal{G}_b$ are mapped into $\mathcal{B}$ via Φ. It follows that

$$\begin{aligned}&\Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_{\mathsf{V}}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{G}]\\ &= \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_{\mathsf{V}}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{G}_g] + \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_{\mathsf{V}}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{G}_b]. \end{aligned} \tag{18}$$

By Corollary 7.6 and because Φ is a bijection, we can infer that

$$\Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_{\mathsf{V}}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{G}_g] \leq \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}, \tag{19}$$

$$\Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_{\mathsf{V}}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{G}_b] \leq \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_{\mathsf{V}}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{B}]. \tag{20}$$

It follows from 18,19, 20 that

$$\Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{G}] \leq \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q} + \Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{B}]. \tag{21}$$

From 21, we can bound $\Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{B}]$ as

$$\begin{aligned}&\Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{B}] = \Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{W}] - \Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{G}]\\ &\geq \Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{W}] - \Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{B}] - \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}.\end{aligned}$$

Since $\varepsilon = \Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_{\mathsf{V}}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{W}]$, we finally obtain

$$\Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_{\mathsf{V}}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{B}] \geq \frac{1}{2}\left(\varepsilon - \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}\right).$$

■

We are now ready to prove Lemma 7.2, i.e., we show that there exist $i \in [Q_{\mathsf{P}_2}+1], j \in [Q_{\mathsf{Ch}}]$ such that $\beta_{i,j} > \left(\varepsilon - \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}\right)\cdot\frac{1}{2Q_{\mathsf{V}}(Q_{\mathsf{P}_2}+1)}$. Toward a contradiction, suppose instead that for all $i \in [Q_{\mathsf{P}_2}+1], j \in [Q_{\mathsf{Ch}}]$, we have that

$$\Pr_{(I,\omega,\boldsymbol{h}) \xleftarrow{\$} (\mathcal{I}\times\Omega\times\mathcal{C}^{Q_{\mathsf{V}}})}[(I,\omega,\boldsymbol{h}) \in \mathcal{B}_{i,j}] < \left(\varepsilon - \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}\right)\cdot\frac{1}{2Q_{\mathsf{V}}(Q_{\mathsf{P}_2}+1)}.$$

By Lemma 7.7,

$$\begin{aligned}&\frac{1}{2}\left(\varepsilon - \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}\right) \leq \Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{B}] = \Pr[(I,\omega,\boldsymbol{h}) \in \bigcup_{i,j}\mathcal{B}_{i,j}]\\ &\leq \sum_{i,j}\Pr[(I,\omega,\boldsymbol{h}) \in \mathcal{B}_{i,j}] < \frac{1}{2}\left(\varepsilon - \frac{Q_{\mathsf{V}}^{Q_{\mathsf{P}_2}+1}\cdot\binom{Q_{\mathsf{P}_2}+Q_{\mathsf{P}_1}}{Q_{\mathsf{P}_1}}}{q}\right).\end{aligned}$$

This is a contradiction.

Game $\mathbf{DLP}_{\mathsf{GGen}}$:
00 $(\mathbb{G}, q, g) \xleftarrow{\$} \mathsf{GGen}(1^\kappa)$
01 $x \xleftarrow{\$} \mathbb{Z}_q$
02 $X \leftarrow g^x$
03 $z \xleftarrow{\$} \mathsf{A}(X, \mathcal{G})$
04 If $z \equiv_q x$: Return 1
05 Return 0

Figure 15: Game $\mathbf{DLP}_{\mathsf{GGen}}$ with adversary A.

8 Instantiations of Linear Function Families

In this section we first give three hardness assumptions and three instatiations of linear function families basing their security notion on the hardness of these assumptions.

8.1 Hardness Assumptions

In the following, let GGen denote a group generating algorithm which on input the security parameter 1^κ outputs the description of a group $\mathcal{G} := (\mathbb{G}, q, g)$, where $\mathbb{G}$ is a cyclic group of prime order q with generator g and we assume that q has bit length $2\kappa + 1$. This way, when instantiated over a suitable elliptic curve, the discrete logarithm problem has κ bits of security, given current knowledge. More concretely, the best known algorithm, i.e., Pollard's rho algorithm [30], requires (heuristically) an expected running time of at least 2^κ in order to achieve constant success probability in this regime.

Discrete Logarithm Problem. The *discrete logarithm problem* relative to GGen is defined via game $\mathbf{DLP}_{\mathsf{GGen}}$ (Figure 15). We define A's advantage in $\mathbf{DLP}_{\mathsf{GGen}}$ as $\mathbf{Adv}^{\mathbf{DLP}}_{\mathsf{GGen}}(\mathsf{A}) := \Pr[\mathbf{DLP}^{\mathsf{A}}_{\mathsf{GGen}} \Rightarrow 1]$ and denote its running time as $\mathbf{Time}^{\mathbf{DLP}}_{\mathsf{GGen}}(\mathsf{A})$.

Definition 8.1 (**DLP** Security). We say that $\mathbf{DLP}_{\mathsf{GGen}}$ *is (ε, t)-hard* if for all adversaries A satisfying $\mathbf{Time}^{\mathbf{DLP}}_{\mathsf{GGen}}(\mathsf{A}) \leq t$, we have that $\mathbf{Adv}^{\mathbf{DLP}}_{\mathsf{GGen}}(\mathsf{A}) \leq \varepsilon$. We say that A *(ε, t)-breaks* $\mathbf{DLP}_{\mathsf{GGen}}$ if $\mathbf{Time}^{\mathbf{DLP}}_{\mathsf{GGen}}(\mathsf{A}) \leq t$ and $\mathbf{Adv}^{\mathbf{DLP}}_{\mathsf{GGen}}(\mathsf{A}) > \varepsilon$.

Factoring Problem. Let PG denote a parameter generating algorithm which on input the security parameter 1^κ outputs $par := (P, Q)$. Here, P and Q are random primes of bit length $\kappa \cdot \tau(\kappa)$, where, throughout this work, $\tau(\kappa) \in \mathbb{N}$ denotes the smallest natural number such that for $n = 2^{2\kappa \cdot \tau(\kappa)}$,

$$\log\left(L_n\left[\frac{1}{3}, \sqrt[3]{\frac{64}{9}}\right]\right) \geq \kappa$$

and where

$$L_n[c, \alpha] := e^{(\alpha + o(1))(\ln n)^c (\ln \ln n)^{1-c}}.$$

More concretely, $L_n\left[\frac{1}{3}, \sqrt[3]{\frac{64}{9}}\right]$ denotes the sub-exponential (heuristic) complexity of the *general number field sieve algorithm* [31], which is the best known factoring algorithm. As in the previous subsection, this choice of size for P and Q guarantees an expected running time of at least 2^κ for the best known factoring algorithm to factor the modulus $N = P \cdot Q$ into its prime components P, Q with constant success probability. The *factoring problem* relative to PG is defined via game $\mathbf{FAC}_{\mathsf{PG}}$ (Figure 16). We define A's advantage in $\mathbf{FAC}_{\mathsf{PG}}$ as $\mathbf{Adv}^{\mathbf{FAC}}_{\mathsf{PG}}(\mathsf{A}) := \Pr[\mathbf{FAC}^{\mathsf{A}}_{\mathsf{PG}} \Rightarrow 1]$ and denote its running time as $\mathbf{Time}^{\mathbf{FAC}}_{\mathsf{PG}}(\mathsf{A})$.

Definition 8.2 (**FAC** Security). We say that $\mathbf{FAC}_{\mathsf{PG}}$ *is (ε, t)-hard* if for all adversaries A satisfying $\mathbf{Time}^{\mathbf{FAC}}_{\mathsf{PG}}(\mathsf{A}) \leq t$, we have that $\mathbf{Adv}^{\mathbf{FAC}}_{\mathsf{PG}}(\mathsf{A}) \leq \varepsilon$. We say that A *(ε, t)-breaks* $\mathbf{FAC}_{\mathsf{PG}}$ if $\mathbf{Time}^{\mathbf{FAC}}_{\mathsf{PG}}(\mathsf{A}) \leq t$ and $\mathbf{Adv}^{\mathbf{FAC}}_{\mathsf{PG}}(\mathsf{A}) > \varepsilon$.

RSA Problem. We now state the RSA problem over $\mathbb{Z}_N^*$. In the following, let us denote with $\varphi(N)$ Euler's totient function. If $N = P \cdot Q$, for primes P, Q, then $\varphi(N) = (P-1)(Q-1)$. We assume a

Game $\mathbf{FAC}_{\mathsf{PG}}$:
$(P, Q) \xleftarrow{\$} \mathsf{PG}(1^\kappa)$
$N = P \cdot Q$
$(P', Q') \xleftarrow{\$} \mathsf{A}(N)$
If $(P', Q' \neq N) \wedge N = P' \cdot Q'$: Return 1
Return 0

Figure 16: Game $\mathbf{FAC}_{\mathsf{PG}}$ with adversary A.

Game $\mathbf{RSA}_{\mathsf{PG}}$:
$(N, e) \xleftarrow{\$} \mathsf{PG}(1^\kappa)$
$x \xleftarrow{\$} \mathbb{Z}_N^*$
$y \leftarrow x^e$
$z \xleftarrow{\$} \mathsf{A}(N, e, y)$
If $z^e = y$: Return 1
Return 0

Figure 17: Game $\mathbf{RSA}_{\mathsf{PG}}$ with adversary A.

slightly modified parameter generating algorithm PG that on input the security parameter 1^κ outputs $par := (N = P \cdot Q, e)$ where P and Q are random primes of bit length $\kappa \cdot \tau(\kappa)$ and e is positive integer satisfying $\gcd(e, \varphi(N)) = 1$. The *RSA problem* relative to PG is defined via game $\mathbf{RSA}_{\mathsf{PG}}$ (Figure 17). We define A's advantage in $\mathbf{RSA}_{\mathsf{PG}}$ as $\mathbf{Adv}_{\mathsf{PG}}^{\mathbf{RSA}}(\mathsf{A}) := \Pr[\mathbf{RSA}_{\mathsf{PG}}^{\mathsf{A}} \Rightarrow 1]$ and denote its running time as $\mathbf{Time}_{\mathsf{PG}}^{\mathbf{RSA}}(\mathsf{A})$.

Definition 8.3 (**RSA** Security). We say that $\mathbf{RSA}_{\mathsf{PG}}$ *is* (ε, t)*-hard* if for all adversaries A satisfying $\mathbf{Time}_{\mathsf{PG}}^{\mathbf{RSA}}(\mathsf{A}) \leq t$, we have that $\mathbf{Adv}_{\mathsf{PG}}^{\mathbf{RSA}}(\mathsf{A}) \leq \varepsilon$. We say that A (ε, t)*-breaks* $\mathbf{RSA}_{\mathsf{PG}}$ if $\mathbf{Time}_{\mathsf{PG}}^{\mathbf{RSA}}(\mathsf{A}) \leq t$ and $\mathbf{Adv}_{\mathsf{PG}}^{\mathbf{RSA}}(\mathsf{A}) > \varepsilon$.

8.2 Instantiations

Okamoto-Schnorr. On input the security parameter 1^κ, PGen returns parameters $par := (\mathbb{G}, g_1, g_2, q)$, where $\mathbb{G}$ is a cyclic group of prime order q and q has bit length $2\kappa + 1$. Furthermore, $g_1, g_2 \in \mathbb{G}$. *par* defines the sets $\mathcal{S}, \mathcal{D}, \mathcal{R}$, as well as the homomorphic evaluation function F as follows:

$$\mathcal{S} := \mathbb{Z}_q; \quad \mathcal{D} := \mathbb{Z}_q^2; \quad \mathcal{R} := \mathbb{G}; \quad \mathsf{F} : \mathbb{Z}_q^2 \to \mathbb{G}, (x_1, x_2) \mapsto g_1^{x_1} g_2^{x_2}.$$

For $s \in \mathcal{S}$, $x \in \mathcal{D}$ and $Z \in \mathcal{R}$, we define $s \cdot x = x^s, s \cdot Z = Z^s$, i.e., as the s-fold component wise application of $+$ (modulo q) to x with itself when applied to $x \in \mathcal{D}$ and the s-fold application of the group operation in $\mathbb{G} = \mathcal{R}$ to Z with itself when applied to $Z \in \mathcal{R}$. It is easy to verify that $\mathcal{D}, \mathcal{R}$ form $\mathcal{S}$-modules (and therefore $\mathcal{S}$-pseudo modules), that F is a pseudo module homomorphism with the trivial distributor function $\Psi \equiv 0$, and that LF is (ε, t)-collision resistant if $\mathbf{DLP}_{\mathsf{GGen}}$ (defined as usual) is (ε, t)-hard. Furthermore, for $x := \mathsf{dlog}_{g_1}(g_2)$, F has a pseudo torsion-free element from the kernel with $\boldsymbol{z}^* := (x, -1)$ satisfying the required properties.

Okamoto-Guillou-Quisquater. On input 1^κ, PGen returns system parameters $par := (N = P \cdot Q, \lambda, a)$, where P, Q are primes, $a \in \mathbb{Z}_N^*$ and λ is a prime number of bit length $2\kappa + 1$ that satisfies $\gcd(\varphi(N), \lambda) = \gcd(N, \lambda) = 1$. The parameters *par* define

$$\mathcal{S} := \mathbb{Z}_\lambda; \quad \mathcal{R} := \mathbb{Z}_N^*; \quad \mathcal{D} = \mathcal{S} \times \mathcal{R},$$

where the group operation on $\mathcal{S}$ is the addition modulo λ, the group operation on $\mathcal{R}$ is the multiplication modulo N and $\mathcal{D}$ is an abelian group with the group operation

$$(x_1, x_2) \circ (y_1, y_2) = \left([x_1 + y_1]_\lambda, \left[x_2 y_2 a^{\left\lfloor \frac{x_1 + y_1}{\lambda} \right\rfloor}\right]_N\right),$$

as we prove in Theorem 8.4. Furthermore, for $s \in \mathcal{S}$, $x \in \mathcal{D}$ and $Z \in \mathcal{R}$, we define $sx := s \cdot x = x^s$, $sZ = Z^s$, i.e., as the s-fold application of $\circ$ to x with itself when applied to $x \in \mathcal{D}$ and the s-fold application of multiplication modulo N to Z with itself when applied to $Z \in \mathcal{R}$. We write sx, sZ rather than $s \cdot x, s \cdot Z$ to distinguish the way that elements from $\mathcal{S}$ are applied to elements of $\mathcal{D}$ and $\mathcal{R}$ from the way that elements from $\mathcal{D}$ and $\mathcal{R}$ are composed via $\circ$ and the multiplication modulo N, respectively. Given that $\mathcal{D}$ forms a group with the operation $\circ$, it is easy to see that $\mathcal{S}$ and $\mathcal{D}$ and $\mathcal{S}$ and $\mathcal{R}$ form pseudo modules.

Lemma 8.4 *$\langle \mathcal{D}, \circ \rangle$ is an abelian group.*

Proof. We show that the group axioms are satisfied for $\langle \mathcal{D}, \circ \rangle$. It follows by inspection that $\mathcal{D}$ is closed under $\circ$ and that $\circ$ is commutative. Furthermore:

- Neutral Element: We see that the element $(0, 1) \in \mathcal{D}$ satisfies $(0,1)\circ(x_1, x_2) = \left([x_1 + 0]_\lambda, [x_2 \cdot 1 \cdot a^{\lfloor \frac{x_1+0}{\lambda} \rfloor}]_N\right) = \left(x_1, [x_2 \cdot 1 \cdot a^0]_N\right) = (x_1, x_2)$ for all $(x_1, x_2) \in \mathcal{D}$. The second to last equality follows from the fact that since $x_1 \in \mathbb{Z}_\lambda$, we have that $\lfloor \frac{x_1}{\lambda} \rfloor = 0$.
- Inverses: Let $(x_1, x_2) \in \mathcal{D}$. Then the element $(-x_1, x_2^{-1}a^{-1}) \in \mathcal{D}$ satisfies $(x_1, x_2)\circ(-x_1, x_2^{-1}a^{-1}) = (x_1, x_2)\circ(\lambda - x_1, x_2^{-1}a^{-1}) = \left([x_1 - x_1]_\lambda, [x_2 x_2^{-1} \cdot a^{\lfloor \frac{x_1+\lambda-x_1}{\lambda} \rfloor} a^{-1}]_N\right) = \left([x_1 - x_1]_\lambda, [x_2 x_2^{-1} \cdot a^1 a^{-1}]_N\right) = \left(0, [a^0]_N\right) = (0, 1)$.

We now show that $\circ$ is associative.

Claim 8.5 $\circ$ is an associative operation on $\mathcal{D}$.

Proof. Let $(x_1, y_1), (x_2, y_2), (x_3, y_3) \in \mathcal{D}$. We have to show that

$$(x_1, y_1) \circ \left((x_2, y_2) \circ (x_3, y_3)\right) = \left((x_1, y_1) \circ (x_2, y_2)\right) \circ (x_3, y_3). \tag{22}$$

Equation (22) leads to

$$\left([x_1 + [x_2 + x_3]_\lambda]_\lambda, \left[y_1 y_2 y_3 a^{\left\lfloor \frac{x_1 + [x_2+x_3]_\lambda}{\lambda} \right\rfloor} a^{\left\lfloor \frac{x_2+x_3}{\lambda} \right\rfloor}\right]_N\right) \tag{23}$$

$$= \left([[x_1 + x_2]_\lambda + x_3]_\lambda, \left[y_1 y_2 y_3 a^{\left\lfloor \frac{[x_1+x_2]_\lambda + x_3}{\lambda} \right\rfloor} a^{\left\lfloor \frac{x_1+x_2}{\lambda} \right\rfloor}\right]_N\right). \tag{24}$$

Let

$$A_0 := \left\lfloor \frac{x_1 + [x_2 + x_3]_\lambda}{\lambda} \right\rfloor, \quad B_0 := \left\lfloor \frac{x_2 + x_3}{\lambda} \right\rfloor,$$
$$A_1 := \left\lfloor \frac{[x_1 + x_2]_\lambda + x_3}{\lambda} \right\rfloor, \quad B_1 := \left\lfloor \frac{x_1 + x_2}{\lambda} \right\rfloor.$$

The two sides of Equation (23) are equal if either

$$(A_0 = A_1) \wedge (B_0 = B_1) \tag{25}$$

or

$$(A_0 = B_1) \wedge (B_0 = A_1). \tag{26}$$

Observe that $A_0, A_1, B_0, B_1 \in \{0, 1\}$. We now carry out a case distinction over the possible combinations of values that can be taken by A_0 and B_0 which rules out all cases for which both Equation (25) and Equation (26) are violated.

- **Case** $A_0 = B_0 = 0$. In this case, we have that

  $$(x_1 + [x_2 + x_3]_\lambda < \lambda) \wedge (x_2 + x_3 < \lambda), \tag{27}$$

  which leads to

  $$x_1 + x_2 + x_3 < \lambda. \tag{28}$$

  We analyse two subcases:

- **Case** $A_1 = 1$. In this case, $x_1 + x_2 + x_3 \geq [x_1 + x_2]_\lambda + x_3 \geq \lambda$, which is in contradiction to Equation (28).
- **Case** $B_1 = 1$. In this case, $x_1 + x_2 + x_3 \geq x_1 + x_2 \geq \lambda$, which is in contradiction to Equation (28).

- **Case** $A_0 = 1, B_0 = 0$. In this case, we have that

$$(x_1 + [x_2 + x_3]_\lambda \geq \lambda) \wedge (x_2 + x_3 < \lambda), \tag{29}$$

which leads to

$$\lambda \leq x_1 + x_2 + x_3. \tag{30}$$

We analyse two subcases:

- **Case** $B_1 = 0, A_1 = 0$. In this case, $x_1 + x_2 < \lambda$ and $[x_1 + x_2]_\lambda + x_3 < \lambda$, which simplifies to $x_1 + x_2 + x_3 < \lambda$ and is in contradiction to Equation (30).
- **Case** $B_1 = 1, A_1 = 1$. In this case, $x_1 + x_2 \geq \lambda$ and $[x_1 + x_2]_\lambda + x_3 \geq \lambda$, which simplifies to $x_1 + x_2 + x_3 - \lambda \geq \lambda$. From Equation (29) it now follows that $\lambda \leq x_1 + x_2 + x_3 - \lambda < x_1 + \lambda - \lambda = x_1$, which contradicts $x_1 < \lambda$.

- **Case** $A_0 = 0, B_0 = 1$. In this case, we have that

$$(x_1 + [x_2 + x_3]_\lambda < \lambda) \wedge (x_2 + x_3 \geq \lambda), \tag{31}$$

which leads to

$$\lambda > x_1 + x_2 + x_3 - \lambda. \tag{32}$$

We analyse two subcases:

- **Case** $B_1 = 0, A_1 = 0$. In this case, $x_1 + x_2 < \lambda$ and $[x_1 + x_2]_\lambda + x_3 < \lambda$, which simplifies to $x_1 + x_2 + x_3 < \lambda$. Now, Equation (31) leads to a contradiction since $\lambda \leq x_2 + x_3 \leq x_1 + x_2 + x_3 < \lambda$.
- **Case** $B_1 = 1, A_1 = 1$. In this case, $x_1 + x_2 \geq \lambda$ and $[x_1 + x_2]_\lambda + x_3 \geq \lambda$, which simplifies to $x_1 + x_2 + x_3 - \lambda \geq \lambda$. This is in direct contradiction to Equation (32).

- **Case** $A_0 = 1, B_0 = 1$. In this case, we have that

$$(x_1 + [x_2 + x_3]_\lambda \geq \lambda) \wedge (x_2 + x_3 \geq \lambda), \tag{33}$$

which leads to

$$\lambda \leq x_1 + x_2 + x_3 - \lambda. \tag{34}$$

We analyse two subcases:

- **Case** $B_1 = 0$. In this case, $x_1 + x_2 < \lambda$. Equation (34) now leads to $\lambda \leq x_1 + x_2 + x_3 - \lambda < x_3 + \lambda - \lambda = x_3$, contradicting that $\lambda > x_3$.
- **Case** $A_1 = 0, B_1 = 1$. In this case, $[x_1 + x_2]_\lambda + x_3 < \lambda$ and $x_1 + x_2 \geq \lambda$, leading to $x_1 + x_2 - \lambda + x_3 < \lambda$ which again contradicts Equation (34).

■

This concludes the proof. ■

For the following proofs, we will use the identity $[x]_\lambda = x - \lambda \left\lfloor \frac{x}{\lambda} \right\rfloor$, which holds for all $x \in \mathbb{Z}, \lambda \in \mathbb{N}$.[4] The evaluation function F is defined as

$$\mathsf{F}\colon \mathbb{Z}_\lambda \times \mathbb{Z}_N^* \to \mathbb{Z}_N^*, \mathsf{F}(x_1, x_2) := \left[a^{x_1} x_2^\lambda\right]_N.$$

[4]More precisely, we will implicitly use the identity $[x]_\lambda \equiv_{\mathsf{ord}(a)} x - \lambda \left\lfloor \frac{x}{\lambda} \right\rfloor$.

Lemma 8.6 F *is a pseudo module homomorphism.*

Proof. Let $(x_1, x_2), (y_1, y_2) \in \mathcal{D}$ and $s \in \mathcal{S}$. Then

$$\begin{aligned}
\mathsf{F}((x_1, x_2) \circ (y_1, y_2)) &= \mathsf{F}\left([x_1 + y_1]_\lambda, \left[x_2 y_2 a^{\lfloor \frac{x_1+y_1}{\lambda} \rfloor}\right]_N\right) \\
&= \left[a^{[x_1+y_1]_\lambda}\left(x_2 y_2 a^{\lfloor \frac{x_1+y_1}{\lambda} \rfloor}\right)^\lambda\right]_N \\
&\equiv_N a^{[(x_1+y_1)]_\lambda + \lambda\lfloor \frac{x_1+y_1}{\lambda} \rfloor}(x_2 y_2)^\lambda \\
&\equiv_N a^{x_1+y_1}(x_2 y_2)^\lambda \equiv_N \left(a^{x_1} x_2^\lambda\right)\left(a^{y_1} y_2^\lambda\right) \\
&\equiv_N \left[a^{x_1} x_2^\lambda\right]_N \left[a^{y_1} y_2^\lambda\right]_N = \mathsf{F}(x_1, x_2)\mathsf{F}(y_1, y_2).
\end{aligned}$$

Since for all $s \in \mathcal{S}$ and $(x_1, x_2) \in \mathcal{D}$, we have defined $s \cdot (x_1, x_2) = (x_1, x_2)^s = (x_1, x_2) \circ \cdots \circ (x_1, x_2)$, it immediately follows that

$$\mathsf{F}(s(x_1, x_2) \circ (y_1, y_2) = \mathsf{F}((x_1, x_2)^s \circ (y_1, y_2)) = \mathsf{F}(x_1, x_2)^s \cdot \mathsf{F}(y_1, y_2) = s\mathsf{F}(x_1, x_2) \cdot \mathsf{F}(y_1, y_2).$$

■

Lemma 8.7 *If* $\mathbf{RSA}_{\mathsf{PGen}}$ *is* (ε, t)*-hard then* LF *is* (ε, t)*-collision resistant.*

Proof. Let A be an adversary that breaks (ε, t)-collision resistance of LF. We show an adversary B that (ε, t)-breaks $\mathbf{RSA}_{\mathsf{PGen}}$. B obtains the problem instance $(N, \lambda, y = u^\lambda,)$. B now runs A on input $(N, \lambda, a := y)$. When A returns pairs $(x_1, x_2), (x_1', x_2')$ s.t. $\mathsf{F}(x_1, x_2) = \mathsf{F}(x_1', x_2')$, we have that

$$u^{\lambda \cdot (x_1 - x_1')} \equiv_N a^{(x_1 - x_1')} \equiv_N (x_2'/x_2)^\lambda \tag{35}$$

and thus

$$u^{x_1 - x_1'} \equiv_N x_2'/x_2,$$

which follows from $\gcd(\lambda, \varphi(N)) = 1$.

Claim 8.8 If $(x_1, x_2) \neq (x_1', x_2')$ then $x_1 \neq x_1'$.

Proof. Suppose that $(x_1, x_2) \neq (x_1', x_2')$ and $x_1 = x_1'$. Therefore, $x_2 \neq x_2'$. Here, $x_1, x_1' \in \mathbb{Z}_\lambda, x_2, x_2' \in \mathbb{Z}_N^*$. Since $\gcd(\lambda, \varphi(N)) = 1$, we know that 1 has the unique λ-th root 1 modulo N, i.e., $z^\lambda \equiv_N 1$ if and only if $z \equiv_N 1$. Setting $z := x_2'/x_2 \not\equiv_N 1$, Equality 35 leads to the contradiction

$$1 \equiv_N a^0 \equiv_N a^{x_1 - x_1'} \equiv_N (x_2'/x_2)^\lambda = z^\lambda,$$

i.e., $z \not\equiv_N 1$ is a λ-th root of 1. ■

By the claim, we may assume in the following that $x_1 \neq x_1'$. Furthermore, since $x_1, x_1' \in \mathbb{Z}_\lambda$ and λ is a prime number, we also have $\gcd(\lambda, x_1 - x_1') = 1$. Thus, B can use the extended Euclidean Algorithm to efficiently compute values d, e such that $d \cdot (x_1 - x_1') + e \cdot \lambda = 1$. In turn, it can compute

$$(x_2'/x_2)^d a^e \equiv_N u^{d \cdot (x_1 - x_1')} u^{e \cdot \lambda} \equiv_N u^{d \cdot (x_1 - x_1') + e \cdot \lambda} \equiv_N u.$$

B returns u and terminates. Clearly, B wins $\mathbf{RSA}_{\mathsf{PGen}}$ whenever A wins $\mathbf{CR}_{\mathsf{LF}}$, which completes the proof. ■

Lemma 8.9 F *has a pseudo torsion-free element from the kernel.*

Proof. Let $par = (N = PQ, \lambda, a) \in \mathsf{PGen}(1^\kappa)$. We show that the element $\boldsymbol{z}^* = (-1, a^{1/\lambda} a^{-1}) = (\lambda - 1, a^{1/\lambda} a^{-1})$ satisfies the required properties.[5]

- We have $\mathsf{F}(\boldsymbol{z}^*) = a^{\lambda - 1}(a^{1/\lambda})^\lambda a^{-\lambda} \equiv_N a^{-1} a \equiv_N 1$, where $1 = 0_\mathcal{R}$ is the neutral element in $\mathcal{R}$.

[5] Here, $1/\lambda$ denotes the inverse of λ modulo $\varphi(N)$.

- For all $s, s' \in \mathbb{Z}_\lambda, s \neq s'$, we have that $s\boldsymbol{z}^* \neq s'\boldsymbol{z}^*$, since $[-s]_\lambda \neq [-s']_\lambda$.

This proves the claim. ■

For $Z \in \mathcal{R}, s', s \in \mathcal{S}$ the distributor function Ψ is defined as

$$\Psi : \mathbb{Z}_N^* \times \mathbb{Z}_\lambda \times \mathbb{Z}_\lambda \longrightarrow \mathcal{D}, (Z, s, s') \mapsto \left(0, Z^{-\left\lfloor \frac{s+s'}{\lambda} \right\rfloor}\right).$$

Lemma 8.10 *For all $x = (x_1, x_2) \in \mathcal{D}, s', s \in \mathcal{S}$, $\mathsf{F}((s+s')x) = \mathsf{F}(sx + s'x + \Psi(\mathsf{F}(x), s, s'))$.*

Proof. Let $\boldsymbol{x} = (x_1, x_2) \in \mathcal{D}, s', s \in \mathcal{S}$. We have

$$\mathsf{F}((s+s')\boldsymbol{x}) \equiv_N [(s+s')]_\lambda \mathsf{F}(\boldsymbol{x}) \equiv_N a^{[(s+s')]_\lambda x_1} x_2^{\lambda [(s+s')]_\lambda}$$

and

$$\begin{aligned}
&\mathsf{F}(s\boldsymbol{x} + s'\boldsymbol{x} + \Psi(\mathsf{F}(\boldsymbol{x}), s, s')) \\
&\equiv_N s\mathsf{F}(\boldsymbol{x}) + s'\mathsf{F}(\boldsymbol{x}) + \mathsf{F}(\Psi(\mathsf{F}(\boldsymbol{x}), s, s')) \\
&\equiv_N a^{sx_1} x_2^{\lambda s} a^{s' x_1} x_2^{\lambda s'} \mathsf{F}\left(\left(0, (a^{x_1} x_2^\lambda)^{-\left\lfloor \frac{s+s'}{\lambda} \right\rfloor}\right)\right) \\
&\equiv_N a^{sx_1} x_2^{\lambda s} a^{s' x_1} x_2^{\lambda s'} (a^{x_1} x_2^\lambda)^{-\lambda \left\lfloor \frac{s+s'}{\lambda} \right\rfloor} \\
&\equiv_N a^{(s+s')x_1 - x_1 \lambda \left\lfloor \frac{s+s'}{\lambda} \right\rfloor} x_2^{\lambda (s+s') - \lambda^2 \left\lfloor \frac{s+s'}{\lambda} \right\rfloor}.
\end{aligned}$$

We want to show that

$$\begin{aligned}
&a^{[(s+s')]_\lambda x_1} x_2^{\lambda [(s+s')]_\lambda} \equiv_N [(s+s')]_\lambda \mathsf{F}(\boldsymbol{x}) \equiv_N s\mathsf{F}(\boldsymbol{x}) + s'\mathsf{F}(\boldsymbol{x}) + \mathsf{F}(\Psi(\mathsf{F}(\boldsymbol{x}), s, s')) \\
&\equiv_N a^{(s+s')x_1 - x_1 \lambda \left\lfloor \frac{s+s'}{\lambda} \right\rfloor} x_2^{\lambda (s+s') - \lambda^2 \left\lfloor \frac{s+s'}{\lambda} \right\rfloor}.
\end{aligned}$$

It suffices to notice that

$$[(s+s')]_\lambda x_1 = (s+s')x_1 - x_1 \lambda \left\lfloor \frac{s+s'}{\lambda} \right\rfloor$$

and

$$\lambda [(s+s')]_\lambda = \lambda \left((s+s') - \lambda \left\lfloor \frac{s+s'}{\lambda} \right\rfloor \right) = \lambda (s+s') - \lambda^2 \left\lfloor \frac{s+s'}{\lambda} \right\rfloor.$$

■

We remark that our detailed description of the Okamoto-Guillou-Quisquater linear function family lets us present the associated blind signature scheme in a more clear and even simplified manner as compared to the description in [28, 29]: There, it was required that λ also satisfy $\mathsf{ord}(a) > \lambda$.

FIAT-SHAMIR. PGen returns parameters $par := (N = P \cdot Q, k)$, where P, Q are prime and $k > 2\kappa$ is an integer. Parameters par define

$$\begin{aligned}
&\mathcal{S} := \mathbb{Z}_2^k; \quad \mathcal{D} := (\mathbb{Z}_N^*)^k, \mathcal{R} := (\mathbb{Z}_N^*)^k; \\
&\mathsf{F} : (\mathbb{Z}_N^*)^k \to (\mathbb{Z}_N^*)^k, \mathsf{F}(x_1, \ldots, x_k) \mapsto (x_1^2, \ldots, x_k^2).
\end{aligned}$$

For $\boldsymbol{s} \in \mathcal{S}, \boldsymbol{z} \in \mathcal{R} = \mathcal{D}$, we define $\boldsymbol{s} \cdot \boldsymbol{z} := (z_1^{\boldsymbol{s}_1}, \ldots, z_k^{\boldsymbol{s}_k}) = ([z_1^{\boldsymbol{s}_1}]_N, \ldots, [z_k^{\boldsymbol{s}_k}]_N)$. Furthermore, we define the group operation on $\mathcal{S}$ as the component wise addition modulo 2 and the group operation on $\mathcal{D}$ and $\mathcal{R}$ as the component wise multiplication modulo N.[6] It is easy to see that $\mathcal{S}$ forms a pseudo module with $\mathcal{D} = \mathcal{R}$ and that F is a pseudo module homomorphism. Moreover, it is straight-forward to verify that LF is (ε, t)-collision resistant if $\mathbf{FAC}_{\mathsf{PG}}$ is (ε, t)-secure (for PG defined as in Section 8).

[6] Inversion is also defined as inverting the elements component wise modulo the respective modulus.

Lemma 8.11 F *has a pseudo torsion-free element from the kernel.*

Proof. For all parameters $par = (N = P{\cdot}Q, k) \in \mathsf{PGen}(1^\kappa)$. We show that the element $\boldsymbol{z}^* = (z_1^*, \ldots, z_k^*) := (-1, \ldots, -1)$ satisfies the required properties.

- We have $\mathsf{F}(\boldsymbol{z}^*) = \left(-1^2, \ldots, -1^2\right) = (1, \ldots, 1)$, where $(1, \ldots, 1) = 0_{\mathcal{R}}$ is the neutral element in $\mathcal{R}$.
- For all $\boldsymbol{s}, \boldsymbol{s}' \in \mathbb{Z}_2^k, \boldsymbol{s} \neq \boldsymbol{s}'$, we have $\boldsymbol{s} \cdot \boldsymbol{z}^* = (-1^{\boldsymbol{s}_1}, \ldots, -1^{\boldsymbol{s}_k}) \neq (-1^{\boldsymbol{s}'_1}, \ldots, -1^{\boldsymbol{s}'_k}) = \boldsymbol{s'} \cdot \boldsymbol{z}^*$.

This completes the proof. ∎

For $\boldsymbol{z} \in \mathcal{R}, \boldsymbol{s}', \boldsymbol{s} \in \mathcal{S}$ we define the distributor function $\Psi : (\mathbb{Z}_N^*)^k \times \mathbb{Z}_2^k \times \mathbb{Z}_2^k \longrightarrow (\mathbb{Z}_N^*)^k$ component wise as $\Psi(\boldsymbol{z}, \boldsymbol{s}, \boldsymbol{s}') = (\Psi_1(\boldsymbol{z}_1, \boldsymbol{s}_1, \boldsymbol{s}'_1), \ldots, \Psi_k(\boldsymbol{z}_k, \boldsymbol{s}_k, \boldsymbol{s}'_k))$, where for $i \in [k]$,

$$\Psi_i(\boldsymbol{z}_i, \boldsymbol{s}_i, \boldsymbol{s}'_i) := \boldsymbol{z}_i^{-(\boldsymbol{s}_i > [\boldsymbol{s}'_i + \boldsymbol{s}_i]_2)}.$$

Here, the boolean operation $b > b'$ on the binary inputs b, b' returns 1 iff $b = 1$ and $b' = 0$, and returns 0 otherwise. In other words, $\Psi_i(\boldsymbol{z}_i, \boldsymbol{s}_i, \boldsymbol{s}'_i) = \boldsymbol{z}_i^{-1}$ if $\boldsymbol{s}_i > [\boldsymbol{s}'_i + \boldsymbol{s}_i]_2$, and $\Psi_i(\boldsymbol{z}_i, \boldsymbol{s}_i, \boldsymbol{s}'_i) = 1$ otherwise. Equivalently, $\Psi_i(\boldsymbol{z}_i, \boldsymbol{s}_i, \boldsymbol{s}'_i) = \boldsymbol{z}_i^{-1}$ if $\boldsymbol{s}_i = \boldsymbol{s}'_i = 1$ and $\Psi_i(\boldsymbol{z}_i, \boldsymbol{s}_i, \boldsymbol{s}'_i) = 1$ otherwise. Note that this also implies that $\Psi_i(\boldsymbol{z}_i, \boldsymbol{s}_i, \boldsymbol{s}'_i) = \Psi_i(\boldsymbol{z}_i, \boldsymbol{s}'_i, \boldsymbol{s}_i)$.

Lemma 8.12 *For all* $x \in \mathcal{D}, \boldsymbol{s}', \boldsymbol{s} \in \mathcal{S}$, $\mathsf{F}\left((\boldsymbol{s} + \boldsymbol{s}')x\right) = \mathsf{F}(\boldsymbol{s}x + \boldsymbol{s}'x + \Psi(\mathsf{F}(x), \boldsymbol{s}, \boldsymbol{s}'))$.

Proof. Let $\boldsymbol{s}', \boldsymbol{s} \in \mathcal{S}, x \in \mathcal{D}$. We have

$$\mathsf{F}\left((\boldsymbol{s} + \boldsymbol{s}')x\right) = \left(x_1^{2[\boldsymbol{s}_1 + \boldsymbol{s}'_1]_2}, \ldots, x_k^{2[\boldsymbol{s}_k + \boldsymbol{s}'_k]_2}\right)$$

and

$$\begin{aligned}
&\mathsf{F}(\boldsymbol{s}\boldsymbol{x} + \boldsymbol{s}'\boldsymbol{x} + \Psi(\mathsf{F}(x), \boldsymbol{s}, \boldsymbol{s}')) \\
&= \left(x_1^{2(\boldsymbol{s}_1 + \boldsymbol{s}'_1)}, \ldots, x_k^{2(\boldsymbol{s}_k + \boldsymbol{s}'_k)}\right) + \mathsf{F}\left(\left(x_1^{-2(\boldsymbol{s}_1 > [\boldsymbol{s}'_1 + \boldsymbol{s}_1]_2)}, \ldots, x_k^{-2(\boldsymbol{s}_k > [\boldsymbol{s}'_k + \boldsymbol{s}_k]_2)}\right)\right) \\
&= \left(x_1^{2(\boldsymbol{s}_1 + \boldsymbol{s}'_1)}, \ldots, x_k^{2(\boldsymbol{s}_k + \boldsymbol{s}'_k)}\right) + \left(x_1^{-4(\boldsymbol{s}_1 > [\boldsymbol{s}'_1 + \boldsymbol{s}_1]_2)}, \ldots, x_k^{-4(\boldsymbol{s}_k > [\boldsymbol{s}'_k + \boldsymbol{s}_k]_2)}\right) \\
&= \left(x_1^{2(\boldsymbol{s}_1 + \boldsymbol{s}'_1) - 4(\boldsymbol{s}_1 > [\boldsymbol{s}'_1 + \boldsymbol{s}_1]_2)}, \ldots, x_k^{2(\boldsymbol{s}_k + \boldsymbol{s}'_k) - 4(\boldsymbol{s}_k > [\boldsymbol{s}'_k + \boldsymbol{s}_k]_2)}\right).
\end{aligned}$$

In the following, we show that

$$\begin{aligned}
&\left(x_1^{2[\boldsymbol{s}_1 + \boldsymbol{s}'_1]_2}, \ldots, x_k^{2[\boldsymbol{s}_k + \boldsymbol{s}'_k]_2}\right) = \mathsf{F}\left((\boldsymbol{s} + \boldsymbol{s}')x\right) = \mathsf{F}(\boldsymbol{s}\boldsymbol{x} + \boldsymbol{s}'\boldsymbol{x} + \Psi(\mathsf{F}(x), \boldsymbol{s}, \boldsymbol{s}')) \\
&= \left(x_1^{2(\boldsymbol{s}_1 + \boldsymbol{s}'_1) - 4(\boldsymbol{s}_1 > [\boldsymbol{s}'_1 + \boldsymbol{s}_1]_2)}, \ldots, x_k^{2(\boldsymbol{s}_k + \boldsymbol{s}'_k) - 4(\boldsymbol{s}_k > [\boldsymbol{s}'_k + \boldsymbol{s}_k]_2)}\right),
\end{aligned}$$

by performing a case distinction (for component i) over the possible values that $\boldsymbol{s}_i, \boldsymbol{s}'_i$ can take.

- $\boldsymbol{s}_i = \boldsymbol{s}'_i = 0$: In this case, we have

$$\begin{aligned}
x_i^{2[\boldsymbol{s}_i + \boldsymbol{s}'_i]_2} &= x_i^{2[0+0]_2} = 1 = x_i^{2\cdot 0 - 4\cdot 0} \\
x_i^{2(0+0) - 4\cdot(0 > [0+0]_2)} &= x_i^{2(\boldsymbol{s}_i + \boldsymbol{s}'_i) - 4\cdot(\boldsymbol{s}_i > [\boldsymbol{s}'_i + \boldsymbol{s}_i]_2)}.
\end{aligned}$$

- $\boldsymbol{s}_i = 0, \boldsymbol{s}'_i = 1$: In this case, we have

$$\begin{aligned}
x_i^{2[\boldsymbol{s}_i + \boldsymbol{s}'_i]_2} &= x_i^{2[0+1]_2} = x_i^2 = x_i^{2\cdot 1 - 4\cdot 0} \\
x_i^{2(0+1) - 4\cdot(0 > [1+0]_2)} &= x_i^{2(\boldsymbol{s}_i + \boldsymbol{s}'_i) - 4\cdot(\boldsymbol{s}_i > [\boldsymbol{s}'_i + \boldsymbol{s}_i]_2)}.
\end{aligned}$$

- $\boldsymbol{s}_i = 1, \boldsymbol{s}'_i = 0$: In this case, we have

$$\begin{aligned}
&x_i^{2[\boldsymbol{s}_i + \boldsymbol{s}'_i]_2} = x_i^{2[1+0]_2} = x_i^2 = x_i^{2\cdot 1 - 4\cdot 0} \\
&= x_i^{2(1+0) - 4\cdot(1 > [0+1]_2)} = x_i^{2(\boldsymbol{s}_i + \boldsymbol{s}'_i) - 4\cdot(\boldsymbol{s}_i > [\boldsymbol{s}'_i + \boldsymbol{s}_i]_2)}.
\end{aligned}$$

- $\boldsymbol{s}_i = 1, \boldsymbol{s}'_i = 1$: In this case, we have

$$x_i^{2[\boldsymbol{s}_i+\boldsymbol{s}'_i]_2} = x_i^{2[1+1]_2} = 1 = x_i^{2\cdot 2-4\cdot 1}$$
$$= x_i^{2(1+1)-4\cdot(1>[1+1]_2)} = x_i^{2(\boldsymbol{s}_i+\boldsymbol{s}'_i)-4\cdot(\boldsymbol{s}_i>[\boldsymbol{s}'_i+\boldsymbol{s}_i]_2)}.$$

This concludes the proof. ■

Acknowledgments

We would like to thank David Pointcheval for helpful discussions and for answering many of our questions. We also thank Jesse Selover for pointing out to us that a previous version of our framework was incorrect and did not capture the Okamoto-Guillou-Quisquater and Fiat-Shamir instantiations. This work was supported by the Deutsche Forschungsgemeinschaft (DFG) under Germany's Excellence Strategy (EXC 2092 CASA) and the ERC Project ERCC (FP7/615074).

References

[1] M. Abdalla, J. H. An, M. Bellare, and C. Namprempre. From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In L. R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 418–433. Springer, Heidelberg, Apr. / May 2002. (Cited on page 8.)

[2] M. Abdalla and L. Reyzin. A new forward-secure digital signature scheme. In T. Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 116–129. Springer, Heidelberg, Dec. 2000. (Cited on page 5.)

[3] M. Abe and T. Okamoto. Provably secure partially blind signatures. In M. Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 271–286. Springer, Heidelberg, Aug. 2000. (Cited on page 5.)

[4] M. Backendal, M. Bellare, J. Sorrell, and J. Sun. The fiat-shamir zoo: Relating the security of different signature variants. Cryptology ePrint Archive, Report 2018/775, 2018. `https://eprint.iacr.org/2018/775`. (Cited on page 1, 2, 6, 9.)

[5] F. Baldimtsi and A. Lysyanskaya. Anonymous credentials light. In A.-R. Sadeghi, V. D. Gligor, and M. Yung, editors, *ACM CCS 2013*, pages 1087–1098. ACM Press, Nov. 2013. (Cited on page 1.)

[6] F. Baldimtsi and A. Lysyanskaya. On the security of one-witness blind signature schemes. In K. Sako and P. Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 82–99. Springer, Heidelberg, Dec. 2013. (Cited on page 1.)

[7] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, and H. Shacham. Randomizable proofs and delegatable anonymous credentials. In S. Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 108–125. Springer, Heidelberg, Aug. 2009. (Cited on page 1.)

[8] M. Bellare and G. Neven. Multi-signatures in the plain public-key model and a general forking lemma. In A. Juels, R. N. Wright, and S. De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 390–399. ACM Press, Oct. / Nov. 2006. (Cited on page 2, 3, 17, 18.)

[9] M. Bellare and A. Palacio. GQ and Schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 162–177. Springer, Heidelberg, Aug. 2002. (Cited on page 2, 8.)

[10] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In D. E. Denning, R. Pyle, R. Ganesan, R. S. Sandhu, and V. Ashby, editors, *ACM CCS 93*, pages 62–73. ACM Press, Nov. 1993. (Cited on page 1, 4.)

[11] M. Bellare and P. Rogaway. Code-based game-playing proofs and the security of triple encryption. Cryptology ePrint Archive, Report 2004/331, 2004. http://eprint.iacr.org/2004/331. (Cited on page 4.)

[12] S. Brands. Untraceable off-line cash in wallets with observers (extended abstract). In D. R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 302–318. Springer, Heidelberg, Aug. 1994. (Cited on page 1.)

[13] J. Camenisch, S. Hohenberger, and A. Lysyanskaya. Compact e-cash. In R. Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 302–321. Springer, Heidelberg, May 2005. (Cited on page 1.)

[14] J. Camenisch and A. Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In B. Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, Heidelberg, May 2001. (Cited on page 1.)

[15] D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. L. Rivest, and A. T. Sherman, editors, *CRYPTO'82*, pages 199–203. Plenum Press, New York, USA, 1982. (Cited on page 1.)

[16] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In S. Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 319–327. Springer, Heidelberg, Aug. 1990. (Cited on page 1.)

[17] U. Feige, A. Fiat, and A. Shamir. Zero-knowledge proofs of identity. *Journal of Cryptology*, 1(2):77–94, June 1988. (Cited on page 2.)

[18] M. Fischlin. Round-optimal composable blind signatures in the common reference string model. In C. Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 60–77. Springer, Heidelberg, Aug. 2006. (Cited on page 12.)

[19] R. Gennaro. Multi-trapdoor commitments and their applications to proofs of knowledge secure under concurrent man-in-the-middle attacks. In M. Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 220–236. Springer, Heidelberg, Aug. 2004. (Cited on page 2.)

[20] E. Hauck, E. Kiltz, and J. Loss. A modular treatment of blind signatures from identification schemes. In Y. Ishai and V. Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 345–375. Springer, Heidelberg, May 2019. (Cited on page 3, 36.)

[21] J. L. W. V. Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, 30(1):175–193. (Cited on page 5.)

[22] A. Juels, M. Luby, and R. Ostrovsky. Security of blind digital signatures (extended abstract). In B. S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 150–164. Springer, Heidelberg, Aug. 1997. (Cited on page 12.)

[23] V. Lyubashevsky. Lattice-based identification schemes secure under active attacks. In R. Cramer, editor, *PKC 2008*, volume 4939 of *LNCS*, pages 162–179. Springer, Heidelberg, Mar. 2008. (Cited on page 3.)

[24] L. Minder and A. Sinclair. The extended k-tree algorithm. In C. Mathieu, editor, *20th SODA*, pages 586–595. ACM-SIAM, Jan. 2009. (Cited on page 7.)

[25] T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In E. F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 31–53. Springer, Heidelberg, Aug. 1993. (Cited on page 1, 2.)

[26] T. Okamoto and K. Ohta. Universal electronic cash. In J. Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 324–337. Springer, Heidelberg, Aug. 1992. (Cited on page 1.)

[27] D. Pointcheval. Strengthened security for blind signatures. In K. Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 391–405. Springer, Heidelberg, May / June 1998. (Cited on page 19.)

[28] D. Pointcheval and J. Stern. New blind signatures equivalent to factorization (extended abstract). In R. Graveman, P. A. Janson, C. Neuman, and L. Gong, editors, *ACM CCS 97*, pages 92–99. ACM Press, Apr. 1997. (Cited on page 1, 2, 32.)

[29] D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, June 2000. (Cited on page 1, 2, 3, 5, 17, 19, 32.)

[30] J. M. Pollard. Monte Carlo methods for index computation mod p. *Mathematics of Computation*, 32:918–924, 1978. (Cited on page 27.)

[31] C. Pomerance and R. Crandall. *Prime Numbers: A Computational Perspective*. Springer, 2001. (Cited on page 27.)

[32] F. Rodriuguez-Henriquez, D. Ortiz-Arroyo, and C. Garcia-Zamora. Yet another improvement over the mu-varadharajan e-voting protocol. *Comput. Stand. Interfaces*, 29(4):471–480, 2007. (Cited on page 1.)

[33] M. Rückert. Lattice-based blind signatures. In M. Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 413–430. Springer, Heidelberg, Dec. 2010. (Cited on page 1, 3.)

[34] C.-P. Schnorr. Security of blind discrete log signatures against interactive attacks. In S. Qing, T. Okamoto, and J. Zhou, editors, *ICICS 01*, volume 2229 of *LNCS*, pages 1–12. Springer, Heidelberg, Nov. 2001. (Cited on page 2, 7.)

[35] D. Wagner. A generalized birthday problem. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 288–303. Springer, Heidelberg, Aug. 2002. (Cited on page 2, 7.)