A preliminary version of this paper appears in *Advances in Cryptology – EUROCRYPT* 2004, volume 3027 of Lecture Notes in Computer Science, C. Cachin and J. Camenisch ed., Springer-Verlag, 2004. This is the full version.

# Security Proofs for Identity-Based Identification and Signature Schemes

Mihir Bellare [*]  Chanathip Namprempre [†]  Gregory Neven [‡]

March 2007

**Abstract**

This paper provides either security proofs or attacks for a large number of identity-based identification and signature schemes defined either explicitly or implicitly in existing literature. Underlying these is a framework that on the one hand helps explain how these schemes are derived, and on the other hand enables modular security analyses, thereby helping to understand, simplify and unify previous work. We also analyze a generic folklore construction that in particular yields identity-based identification and signature schemes without random oracles.

[*] Dept. of Computer Science & Engineering, University of California at San Diego, 9500 Gilman Drive, La Jolla, California 92093, USA. E-Mail: `mihir@cs.ucsd.edu`. URL: `http://www-cse.ucsd.edu/users/mihir`. Supported in part by NSF grants CCR-0098123, ANR-0129617, CCR-0208842, CNS-0524765, and a gift from Intel Corporation.

[†] Electrical Engineering Dept., Thammasat University, Klong Luang, Patumtani 12121, Thailand. E-Mail: `cnamprem@engr.tu.ac.th`. URL: `http://www.engr.tu.ac.th/~nchanath`. Supported in part by the above-mentioned grants of first author and the Thailand Research Fund. Part of this work was done while the author was at University of California, San Diego.

[‡] Dept. of Computer Science, Katholieke Universiteit Leuven, Celestijnenlaan 200A, 3001 Heverlee-Leuven, Belgium. E-Mail: `Gregory.Neven@cs.kuleuven.ac.be`. URL: `http://www.cs.kuleuven.ac.be/~gregory`. Supported by Research Assistantship and travel credit from the Fund for Scientific Research, Flanders (Belgium).

# Contents

# 1 Introduction

IBI and IBS. In an identity-based identification (IBI) scheme, there is an authority having a master public key and a master secret key. This authority can provide a user with a secret key based on its identity. The user, playing the role of a prover, can then identify itself to a verifier in a protocol in which the verifier begins by knowing only the claimed identity of the prover and the master key of the authority. An identity-based signature (IBS) scheme is similar except that the user signs messages, rather than identifying itself, and verification of a signature requires knowledge only of the identity of the signer and the master public key.

Current state of the area. The late eighties and early nineties saw the proposal of many IBI and IBS schemes. These include the Fiat-Shamir IBI and IBS schemes [FS86], the Guillou-Quisquater IBI and IBS schemes [GQ89], the IBS scheme in Shamir's paper [Sha84] introducing identity-based cryptography, and others [Bet88, Oka93, Gir90]. Now, new pairing-based IBS schemes are being proposed [SOK00, Pat02, Hes03, CC03, Yi03].

There is a lot of work on proving security in the identification domain, but it pertains to standard rather than identity-based schemes. (For example, security proofs have been provided for standard identification schemes underlying the Fiat-Shamir and Guillou-Quisquater IBI schemes [FS86, FFS88, GQ89, BP02], but not for the IBI schemes themselves.) In fact, a provable-security treatment of IBI schemes is entirely lacking: there are no security definitions, and none of the existing schemes is proven secure.

Cha and Cheon provide a definition of security for IBS schemes and prove their scheme secure [CC03]. Dodis, Katz, Xu, and Yung [DKXY03] define a class of standard signature (SS) schemes that they call trapdoor, and then present a random-oracle-using transform (let us call it tSS-2-IBS) that turns any secure trapdoor SS (tSS) scheme into a secure IBS scheme. Security proofs for several existing IBS schemes, including those of [FS86, GQ89], are obtained by observing that these are the result of applying tSS-2-IBS to underlying tSS schemes already proven secure in the literature [OO98, PS00, AABN02]. However, as we will see, there are several IBS schemes not yet proven secure (one example is Shamir's IBS scheme [Sha84]), either because they are not the result of applying tSS-2-IBS to a tSS scheme, or because, although they are, the tSS scheme in question has not yet been analyzed.

In summary, there are important gaps with regard to proven security in this area. Namely, it is absent for IBI and absent in some cases for IBS.

Contributions in brief. We fill the above-mentioned gaps, providing security proofs for a large number of existing and new IBI and IBS schemes via a common framework that unifies and explains the area. We also show that it is easy to implement IBI and IBS without random oracles and in fact from any one-way function.

Context and motivation. There are several motivations to provide firm foundations in this area. One comes from existing usage. IBI schemes such as GQ are in use for smartcard-based identification. (For example, a fast implementation of GQ was put on a chip as a part of a payment protocol in as early as 1996 [DVQ96].) Our work provides the first provable-security support for this usage in the identity-based setting. Another motivation comes from the recent implementation of IBE [BF01]: A full-fledged identity-based system would require identity-based authentication, i.e. IBI or IBS, in addition to IBE, so the emergence of the latter renews interest in the former. The final motivation is to bring some clarity to the area. Our work highlights the fact that IBI and IBS are easier to achieve than IBE. In particular, they do not require pairings and are easy to achieve without random oracles. Also, unlike IBE, the first schemes for which are recent, schemes for IBI and IBS have been proposed since the eighties (albeit without security proofs in some cases). We now discuss our contributions in more depth.

## 1.1 Definitions

We extend to the IBI setting the three notions of security for standard identification (SI) schemes, namely security against impersonation under passive attacks (imp-pa), active attacks (imp-aa) [FFS88], and concurrent attacks (imp-ca) [BP02]. Our model allows the adversary to expose user (prover) keys, and to mount either passive, active, or concurrent attacks on the provers, winning if it succeeds in impersonating a prover of its choice. We remark that although existing security definitions for other identity-based primitives [BF01, CC03, DKXY03] give us some guidance as to what adversary capabilities to consider, there are some issues in the definition for IBI that need thought, for example with regard to capabilities the adversary gets in what stage of its two-stage attack. See Section 2.

The security notion for SS schemes is the standard notion of unforgeability under chosen-message attack (uf-cma) [GMR88]. An appropriate extension of it for IBS schemes exists [CC03, DKXY03] and we refer to it also as uf-cma. These definitions are recalled in Section 2.

## 1.2 Certificate-based IBI and IBS: Schemes without Random Oracles

Before executing the main task of analyzing dedicated IBI and IBS schemes, we pause to note a very simple and natural design of an IBI scheme, based on any given SI scheme. The design is based on the *certification paradigm*. The authority picks a public and secret key pair $(pk, sk)$ for a SI scheme, and provides these to prover with identity $I$ along with a certificate $cert$ consisting of the authority's signature on $I, pk$. The prover can now send $pk, cert$ to the verifier and then identify itself via the SI scheme under $pk$. The verifier needs to know only the prover's identity $I$ and the public key of the authority in order to authenticate the prover. Theorem 3.2 says that this yields a secure IBI scheme. Analogously, there is a certificate-based construction of an IBS scheme from any standard signature (SS) scheme.

Although simple, we believe this is worth noting. It highlights the fact that, unlike IBE [BF01], IBI and IBS are trivial to achieve. In particular, it shows that IBI and IBS can be achieved without random oracles and in fact based on any one-way function (because there are non-random-oracle and one-way function based implementations of SI and SS schemes). It also enables us to better understand what the dedicated schemes are trying to do, namely to beat the trivial certificate-based schemes in performance.

## 1.3 Analysis of Dedicated Schemes

We provide security proofs for a large number of dedicated IBI and IBS schemes, including not only the ones mentioned above, but many more that we surface as having been, with hindsight, implicit in the literature. We do this in two steps. In the first step, we provide a framework that (in most cases) reduces proving security of IBI or IBS schemes to proving security of an underlying SI scheme. In a few cases, we found that the SI schemes in question were already analyzed in the literature, but in many cases they were not. The second step, wherein lies the main technical work of the paper, is to provide security proofs for those SI schemes not already proven secure, and then provide direct security proofs for the few exceptional IBI or IBS schemes that escape being captured by our framework.

The framework is of value beyond its ability to reduce proving security of IBI and IBS schemes to proving security of SI schemes. It helps understand how schemes are being derived, and in the process surfaces the implicit schemes we mentioned above. Overall, the framework contributes to simplifying and unifying our picture of the area. We now explain the framework, which is based on a set of transforms, and then summarize the results for particular dedicated schemes.

$$\begin{tikzcd} \mathcal{N}ame\text{-}\mathcal{SI} \arrow[r, "\mathsf{cSI\text{-}2\text{-}IBI}"] \arrow[d, "\mathsf{fs\text{-}I\text{-}2\text{-}S}"'] & \mathcal{N}ame\text{-}\mathcal{IBI} \arrow[d, "\mathsf{fs\text{-}I\text{-}2\text{-}S}"] \\ \mathcal{N}ame\text{-}\mathcal{SS} \arrow[r, "\mathsf{cSS\text{-}2\text{-}IBS}"] & \mathcal{N}ame\text{-}\mathcal{IBS} \end{tikzcd}$$

Figure 1: Family of schemes associated with a cSI scheme $\mathcal{N}ame\text{-}\mathcal{SI}$. If $\mathcal{N}ame\text{-}\mathcal{SI}$ is imp-atk secure then $\mathcal{N}ame\text{-}\mathcal{IBI}$ is also imp-atk secure, for all atk $\in \{\mathrm{pa}, \mathrm{aa}, \mathrm{ca}\}$. If $\mathcal{N}ame\text{-}\mathcal{SI}$ is imp-pa secure then $\mathcal{N}ame\text{-}\mathcal{IBS}$ is uf-cma secure. Implicit in drawing the diagram this way is that $\mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathsf{cSI\text{-}2\text{-}IBI}(\mathcal{N}ame\text{-}\mathcal{SI})) = \mathsf{cSS\text{-}2\text{-}IBS}(\mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathcal{N}ame\text{-}\mathcal{SI}))$.

The transforms. We introduce (cf. Definition 4.2) a class of SI schemes that we call convertible. The idea is that the public key contains the description of a (trapdoor samplable, as we define in Definition 4.1) relation $\mathbf{R}$ and an element $y$, while the secret key contains an element $x$ such that $(x, y) \in \mathbf{R}$. We then present a random-oracle-using transform $\mathsf{cSI\text{-}2\text{-}IBI}$ that transforms a convertible SI (cSI) scheme into an IBI scheme (cf. Construction 4.3) by defining $y$ as the hash of the user's identity, and by letting the authority compute the corresponding $x$ using a piece of trapdoor information associated to $\mathbf{R}$. Theorem 4.4 shows that $\mathsf{cSI\text{-}2\text{-}IBI}$ is security-preserving, meaning that if the starting cSI scheme is imp-atk secure then so is the resulting IBI scheme (in the random oracle model), for each atk $\in \{\mathrm{pa}, \mathrm{aa}, \mathrm{ca}\}$. This will be our main tool for proving security of IBI schemes.

It is useful to analogously define convertible standard signature (cSS) schemes and a transform $\mathsf{cSS\text{-}2\text{-}IBS}$ that turns a uf-cma secure cSS scheme into a uf-cma secure IBS scheme. Special cases of this transform considering trapdoor permutations (rather than trapdoor samplable relations) and pairing-based schemes were previously presented in [DKXY03]. Our generalization is slight, but important, as some existing schemes can only be captured under our more general definition.

Now let $\mathsf{fs\text{-}I\text{-}2\text{-}S}$ denote the (random-oracle-using) Fiat-Shamir transform [FS86] which turns a SI scheme into a SS scheme. We know that if the former is imp-pa secure then the latter is uf-cma secure in the random oracle model[AABN02]. (Application of the transform and this last result requires that the starting SI scheme be a three-move public-coin protocol satisfying a certain technical condition, but all these will always be true for the applications we consider.)

Putting the above together yields Corollary 4.10, which says that, as long as a cSI scheme $X$ is imp-pa secure, the IBS scheme $\mathsf{cSS\text{-}2\text{-}IBS}(\mathsf{fs\text{-}I\text{-}2\text{-}S}(X))$ is uf-cma secure. This will be our main tool for proving security of IBS schemes. We note that $\mathsf{fs\text{-}I\text{-}2\text{-}S}$ also transforms a given IBI scheme into an IBS scheme. Furthermore, $\mathsf{cSS\text{-}2\text{-}IBS}(\mathsf{fs\text{-}I\text{-}2\text{-}S}(X)) = \mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathsf{cSI\text{-}2\text{-}IBI}(X))$ for any cSI scheme $X$. In other words, the diagram of Figure 1 "commutes."

As an aside, we remark that the analogue of the result of [AABN02] does *not* hold for $\mathsf{fs\text{-}I\text{-}2\text{-}S}$ as a transform of IBI schemes to IBS schemes: Proposition 4.11 shows that there exists an imp-pa secure IBI scheme $Y$ which under $\mathsf{fs\text{-}I\text{-}2\text{-}S}$ yields an insecure IBS scheme. This does not contradict the above since this $Y$ is not the result of $\mathsf{cSI\text{-}2\text{-}IBI}$ applied to a cSI scheme, but it makes things more difficult in a few exception cases (that we will see later) in which we need to consider an IBS scheme $Z = \mathsf{fs\text{-}I\text{-}2\text{-}S}(Y)$ where $Y$ is an IBI scheme that is not equal to $\mathsf{cSI\text{-}2\text{-}IBI}(X)$ for any cSI scheme $X$. See the end of Section 4 for more information.

Scheme families. We seek to explain any IBI scheme $Y$ in the literature by surfacing a cSI scheme $X$ such that $\mathsf{cSI\text{-}2\text{-}IBI}(X) = Y$. We seek to explain any IBS scheme $Z$ in the literature by surfacing a cSI scheme $X$ such that $\mathsf{cSS\text{-}2\text{-}IBS}(\mathsf{fs\text{-}I\text{-}2\text{-}S}(X)) = Z$. We are able to do this for the schemes in [Sha84, FS86, Bet88, GQ89, Gir90, Hes03, CC03, Yi03] and for the RSA-based IBI scheme in [Oka93].

| *Name* | Origin | *Name-SI* | | | *Name-IBI* | | | *Name-SS* | *Name-IBS* |
|---|---|---|---|---|---|---|---|---|---|
| | | imp-pa | imp-aa | imp-ca | imp-pa | imp-aa | imp-ca | uf-cma | uf-cma |
| *FFS* | IBI,IBS [FS86, FFS88] | [FFS88] | [FFS88] | **I** | **I** | **I** | **I** | [PS00] | [DKXY03] |
| *ItR* | SI, SS [OO90, OS90] | [Sch96] | [Sch96] | **U** | **I** | **I** | **U** | [PS00] | [DKXY03] |
| *FF* | SI,SS [FF02] | [FF02] | [FF02] | [FF02] | **I** | **I** | **I** | [FF02] | **I** |
| *GQ* | IBI, IBS [GQ89] | [GQ89] | [BP02] | [BP02] | **I** | **I** | **I** | [PS00] | [DKXY03] |
| *Sh* | IBS [Sha84] | **P** | **A** | **A** | **I** | **A** | **A** | **I** | **I** |
| *Sh*$^*$ | SI | **P** | **P** | **P** | **I** | **I** | **I** | **I** | **I** |
| *OkRSA* | SI, IBI, SS [Oka93] | [Oka93] | [Oka93] | **I** | **I** | **I** | **I** | [PS00] | **I** |
| *Gir* | SI, IBI [Gir90, SSN98] | **A** | **A** | **A** | **A** | **A** | **A** | **A** | **A** |
| *SOK* | IBS [SOK00] | **P** | **A** | **A** | **I** | **A** | **A** | **I** | **I** |
| *Hs* | IBS [Hes03] | **P** | **P** | **P** | **I** | **I** | **I** | [Hes03] | [DKXY03] |
| *ChCh* | IBS [CC03, Yi03] | **P** | **P** | **P** | **I** | **I** | **I** | [CC03] | [CC03] |
| *Beth* | IBI [Bet88] | **P** | **U** | **U** | **I** | **U** | **U** | **I** | **I** |
| *OkDL* | IBI [Oka93] | **I** | **I** | **I** | **P** | **P** | **P** | **I** | **I** |
| *BNN* | SI,IBI | **I** | **I** | **I** | **P** | **P** | **P** | **I** | **I** |

Figure 2: Summary of security results for dedicated IBI and IBS schemes. Column 1 is the family name of a family of schemes. Column 2 indicates which of the four member-schemes of the family existed in the literature. (The others we surface.) In the security columns, a known result is indicated via a reference to the paper establishing it. The marks **I**, **P**, and **A** all indicate new results obtained in this paper. An **I** indicates a proof of security obtained by implication. (If under *Name-IBI* it means we obtain it via Theorem 4.4, if under *Name-IBS* it means we obtain it either via Corollary 4.10 or via our modified fs-I-2-S transform, if elsewhere it means it follows easily from, or is an easy extension of, existing work.) A **P** indicates a new security proof, such as a from-scratch analysis of some SI or IBI scheme. An **A** indicates an attack that we have found. A **U** indicates that the security status is unknown. In all but the last two rows, the SI scheme is convertible. The first set of schemes are factoring based, the next RSA based, the next pairing based, and the last DL based. For each of the schemes above except for the last two, *Name-IBS* is obtained through the fs-I-2-S transform. The schemes *OkDL-IBS* and *BNN-IBS* are obtained through a modified version of the fs-I-2-S transform.

By Theorem 4.4 and Corollary 4.10, this reduces the task of showing that $Y, Z$ are secure to showing that $X$ is secure in these cases.

We remark that the above gives rise to numerous schemes that are "new" in the sense that they were not provided explicitly in the literature. For example, Shamir [Sha84] defined an IBS scheme but no IBI scheme. (He even says providing an IBI scheme is an open question.) Denoting Shamir's IBS scheme by *Sh-IBS*, we surface the cSI scheme *Sh-SI* such that cSS-2-IBS(fs-I-2-S(*Sh-SI*)) = fs-I-2-S(cSI-2-IBI(*Sh-SI*)) = *Sh-IBS*. As a consequence, we surface the IBI scheme *Sh-IBI* = cSI-2-IBI(*Sh-SI*) that is related in a natural way to *Sh-IBS*, namely by the fact that fs-I-2-S(*Sh-IBI*) = *Sh-IBS*. In an analogous way we surface IBI schemes *Hs-IBI* and *ChCh-IBI* underlying the IBS schemes of [Hes03] and [CC03, Yi03], respectively.

Beside explaining existing IBI or IBS schemes, we are able to derive some new ones. We found papers in the literature [OO90, OS90, FF02] not defining IBI or IBS schemes, but defining SI schemes that we can show are convertible. Our transforms then yield new IBI and IBS schemes that we analyze.

We feel that this systematic surfacing of implicit schemes helps to homogenize, unify, and simplify the area. Figure 1 summarizes the perspective that emerges. We view schemes as occurring in families. Each family has a family name *Name*. At the core of the family is a cSI scheme *Name-SI*. The other schemes are related to it via *Name-IBI* = cSI-2-IBI(*Name-SI*), *Name-SS* = fs-I-2-S(*Name-SI*), and *Name-IBS* = cSS-2-IBS(*Name-SS*). If *Name-SI* is secure, so are all other schemes in the family.

Results for particular dedicated schemes. In order to complete the task of obtaining security proofs for the existing and new IBI and IBS schemes we have discussed, it remains to analyze the cSI schemes underlying the families in question. This turns out to be a large task, for although in a few cases the cSI scheme is one already analyzed in the literature, we found (perhaps surprisingly) that in many cases it is not. Additionally, we need to directly analyze two IBI schemes not underlain by cSI schemes, namely the DL-based scheme in [Oka93], and a somewhat more efficient Schnorr-based [Sch90] variant that we introduce.

A summary of our results is in Figure 2. Section 5 and Section 6 provide scheme descriptions and more precise result statements. Note all security proofs for SS, IBI, and IBS schemes are in the random oracle (RO) model of [BR93]. Here, we highlight some of the important elements of these results.

Cases captured by our framework. Section 5 begins by surfacing SI schemes underlying the first 12 (i.e. all but the last two) families of Figure 2 and shows that they are convertible, so that the picture of Figure 1 holds in all these cases and we need only consider security of the cSI schemes. The analysis of these schemes follows.

Easy cases are *FFS*, *ItR* (the iterated-root, also called $2^t$-th root, family), *FF*, *GQ*, and *OkRSA* (an RSA-based family from [Oka93]) where the SI schemes are already present and analyzed in the literature [FFS88, Oka93, Sch96, FF02, BP02].

The *Sh-SI* scheme turns out to be a mirror-image of *GQ-SI*, and is interesting technically because we show that it is honest-verifier zero-knowledge (HVZK) even though it might not at first appear to be so. Based on this, we prove that it is imp-pa (cf. Theorem 5.2), but simple attacks show that imp-aa and imp-ca do not hold. A slight modification *Sh*$^*$*-SI* of this scheme however is not only imp-pa but also proven imp-aa and imp-ca secure under the one-more RSA assumption of [BNPS03] (cf. Theorem 5.3), so that its security is like that of *GQ-SI* [BP02].

An attack and a fix for Girault's IBI scheme [Gir90] were proposed in [SSN98], but we find attacks on the fixed scheme as well, breaking all schemes in the family.

We prove imp-pa security of the pairing-based *SOK-SI*, *Hs-SI* and *ChCh-SI* schemes under a computational DH assumption and imp-aa, imp-ca security under a one-more computational DH assumption (cf. Theorems 5.5 and 5.6). We remark that the *SOK-IBS* scheme defined via our transforms is not the one of [SOK00], but is slightly different. This suggests the value of our framework, for it is unclear whether the IBS scheme of [SOK00] can be proved uf-cma secure, whereas Corollary 4.10 implies that *SOK-IBS* is uf-cma secure.

Since the discrete-log function has no known trapdoor it is not an obvious starting point for IBI schemes, but some do exist. Beth's (unproven) IBI scheme [Bet88] is based on ElGamal signatures. The proof of convertibility of the *Beth-SI* scheme we surface is interesting in that it exploits the existential forgeability of ElGamal signatures. Theorem 5.7 says that *Beth-SI* is imp-pa secure if the hashed-message ElGamal signature scheme is universally unforgeable under no-message attack in the random oracle model.

Exceptions. The last two rows of Figure 2 represent cases where our framework does not apply and direct analyses are needed. The first such case is an unproven DL-based IBI scheme *OkDL-IBI* due to Okamoto [Oka93], who here introduces an interesting SS-based method for constructing IBI schemes and instantiates it with his own DL-based SS scheme. We were unable to surface any cSI scheme which under cSI-2-IBI maps to *OkDL-IBI*. (*OkDL-IBI* can be "dropped" in a natural way to a SI scheme *OkDL-SI*, but the latter does not appear to be convertible.) However, Theorem 6.2 shows that *OkDL-IBI* is nevertheless imp-pa, imp-aa, and imp-ca secure assuming hardness of the DL problem. This direct proof is probably the most technical in the paper and uses the security of Okamoto's DL-based SS scheme under a weakened notion of non-malleability [SPMLS02], which is established via an extension of the result of [AABN02] combined with results from [Oka93]. We also present a new IBI scheme *BNN-IBI* that is based on the paradigm underlying *OkDL-IBI* but uses Schnorr signatures [Sch90] instead of Okamoto signatures. It is slightly more efficient than *OkDL-IBI*. Security results are analogous to those above (cf. Theorems 6.3, 6.4).

Proposition 4.11 precludes proving security of the IBS schemes fs-I-2-S(*OkDL-IBI*) and fs-I-2-S( *BNN-IBI*) based merely on the security properties of the IBI schemes. However, we show that by including the user's identity in the argument of the hash function in the classical fs-I-2-S transform, we obtain a transform (we call it efs-IBI-2-IBS) that yields a secure uf-cma IBS scheme when applied to any imp-pa IBI scheme. We can then apply this transform to *OkDL-IBI* or *BNN-IBI* to obtain uf-cma IBS schemes.

### 1.4 Discussion and Related Work

Above, we have concentrated on the security of the schemes. A choice as to which schemes to use must also consider efficiency. In order to facilitate comparisons, we provide in Section 7 a table showing signature sizes, signing time, and verification time for all the IBS schemes we have considered here.

A preliminary version of this paper appeared in Eurocrypt 2004 [BNN04].

Independent of our work, Kurosawa and Heng [KH04] presented a transform from a certain class of "zero-knowledge" SS schemes to IBI schemes. However, the IBI scheme resulting from their transform is only shown to be secure against impersonation under *passive* attacks.

Consider the IBS scheme efs-IBI-2-IBS(*SOK-IBI*), obtained by applying our extended Fiat-Shamir transform to our modified version *SOK-IBI* of the IBI scheme of [SOK00]. This IBS scheme is different from the *SOK-IBS* scheme that we noted we prove secure above, but a proof of security of this scheme (in the RO model under the CDH assumption) too follows by combining Theorems 5.5 and 4.13. Interestingly, following our work, Libert and Quisquater [LQ04] show that this scheme has a *tight* security reduction from the CDH problem, which seems to be a rather unique feature for IBS schemes.

It might be worth clarifying that there are many SI schemes in the literature that are not cSI and have no IBI or IBS counterparts. These include the Schnorr scheme [Sch90].

## 2 Notation and Standard Definitions

NOTATION. We let $\mathbb{N} = \{1, 2, 3, \ldots\}$ denote the set of positive integers. If $k \in \mathbb{N}$, then $1^k$ is the string of $k$ ones. The empty string is denoted $\varepsilon$. If $x_1, \ldots, x_n$ are strings, then we denote by $x_1 \| \cdot \| x_n$ a string encoding of $x_1, \ldots, x_n$ from which the constituent objects are uniquely recoverable. (If the lengths of the string encodings of $x_1, \ldots, x_n$ are known, then concatenation will serve the purpose.) If $x$ is a string, then $|x|$ denotes its length, and if $S$ is a set, then $|S|$ is its cardinality. If $\mathsf{A}$ is a randomized algorithm, then

$$\mathsf{A}(x_1, x_2, \ldots : \mathrm{O}_1, \mathrm{O}_2, \ldots)$$

means that $\mathsf{A}$ has inputs $x_1, x_2, \ldots$ and access to oracles $\mathrm{O}_1, \mathrm{O}_2, \ldots$. Also

$$y \overset{\$}{\leftarrow} \mathsf{A}(x_1, x_2, \ldots : \mathrm{O}_1, \mathrm{O}_2, \ldots)$$

means that we run the randomized algorithm $\mathsf{A}$ on inputs $x_1, x_2, \ldots$ and with access to oracles $\mathrm{O}_1, \mathrm{O}_2, \ldots$, and let $y$ denote the output obtained. The set of all possible outputs is denoted by

$$[\mathsf{A}(x_1, x_2, \ldots : \mathrm{O}_1, \mathrm{O}_2, \ldots)] \ .$$

PROVERS AND VERIFIERS. An *interactive algorithm* (modeling a party such as prover or verifier in a protocol) is a stateful algorithm that on input an incoming message $M_{\text{in}}$ (this is $\varepsilon$ if the party is initiating the protocol) and state information $St$ outputs an outgoing message $M_{\text{out}}$ and updated state

```
Oracle $\textsc{Conv}()$
    $(C, d) \xleftarrow{\$} \mathbf{Run}[P(sk) \leftrightarrow V(pk)]$
    Return $C$
```

```
Oracle $\textsc{Prov}(s, M_{\text{in}})$
    If $s \notin PS$ then
        If atk = aa then $PS \leftarrow \{s\}$
        If atk = ca then $PS \leftarrow PS \cup \{s\}$
        Pick random coins $\rho$ for $\mathsf{P}$ ; $St_{\mathsf{P}}[s] \leftarrow (sk, \rho)$
    $(M_{\text{out}}, St_{\mathsf{P}}[s]) \leftarrow \mathsf{P}(M_{\text{in}}, St_{\mathsf{P}}[s])$
    Return $M_{\text{out}}$
```

```
$\mathbf{Exp}^{\text{imp-atk}}_{\mathcal{SI},\mathsf{A}}(k)$                // atk $\in \{\text{pa}, \text{aa}, \text{ca}\}$, $\mathsf{A} = (\mathsf{CV}, \mathsf{CP})$
    $(pk, sk) \xleftarrow{\$} \mathsf{Kg}(1^k)$ ; $PS \leftarrow \emptyset$   // $PS$ is set of active prover sessions
    If atk = pa then let $\textsc{Or}$ denote $\textsc{Conv}$ else let $\textsc{Or}$ denote $\textsc{Prov}$
    $St_{\mathsf{CP}} \leftarrow \mathsf{CV}(1^k, pk : \textsc{Or})$ ; $(C, d) \xleftarrow{\$} \mathbf{Run}[\mathsf{CP}(St_{\mathsf{CP}}) \leftrightarrow V(pk)]$
    Return $d$
```

Figure 3: Oracles given to adversary attacking SI scheme $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$, and experiment used to define imp-atk security of the scheme.

$St'$. The initial state of $\mathsf{A}$ contains its *initial inputs* and optionally a random tape $\rho$; if no random tape is explicitly given in the initial state, $\mathsf{A}$ is assumed to toss its own coins. We say that $\mathsf{A}$ accepts if $St = \texttt{acc}$ and rejects if $St = \texttt{rej}$. An interaction between a prover $\mathsf{P}$ and verifier $\mathsf{V}$ (both modeled as interactive algorithms) ends when $\mathsf{V}$ either accepts or rejects. We write

$$(C, d) \xleftarrow{\$} \mathbf{Run}[\mathsf{P}(p_1, \ldots : \mathrm{OP}_1, \ldots) \leftrightarrow \mathsf{V}(v_1, \ldots : \mathrm{OV}_1, \ldots)]$$

to indicate that we let $\mathsf{P}$ (with initial inputs $p_1, \ldots$ and indicated oracles) interact with $\mathsf{V}$ (with initial inputs $v_1, \ldots$ and indicated oracles), having provided both $\mathsf{P}$ and $\mathsf{V}$ with fresh random coins, to get a conversation transcript $C$ and a boolean decision $d$ with 1 meaning that $\mathsf{V}$ accepted and 0 meaning it rejected.

Resource usage conventions. We may want to talk of the resources of an adversary, such as its running time and the number of its oracle queries. The context will be an overlying experiment, depending on a security parameter $k$, in which the adversary is executed. We measure its resource usage as a function of $k$. We say that the running time of adversary $\mathsf{A}$ is at most $\boldsymbol{T}_{\mathsf{A}}$ if for every $k$, the running time of $\mathsf{A}$ in the experiment does not exceed $\boldsymbol{T}_{\mathsf{A}}(k)$ steps. Similarly, we say that $\mathsf{A}$ makes at most $\boldsymbol{Q}^{\mathrm{O}}_{\mathsf{A}}$ queries to oracle O if for every $k$, the number of queries made by $\mathsf{A}$ to O in the experiment does not exceed $\boldsymbol{Q}^{\mathrm{O}}_{\mathsf{A}}(k)$. These bounds must hold not only for all executions of the experiment, meaning all coin tosses used, but also across all possible answers to oracle queries. We always assume that functions such as $\boldsymbol{T}_{\mathsf{A}}$, $\boldsymbol{Q}^{\mathrm{O}}_{\mathsf{A}}$ are poly($k$) bounded and poly($k$) time computable. For a pair of algorithms $\mathsf{A} = (\mathsf{P}, \mathsf{V})$, we use the shorthand notations $\boldsymbol{T}_{\mathsf{A}} = \boldsymbol{T}_{\mathsf{P}} + \boldsymbol{T}_{\mathsf{V}}$ and $\boldsymbol{Q}^{\mathrm{O}}_{\mathsf{A}} = \boldsymbol{Q}^{\mathrm{O}}_{\mathsf{P}} + \boldsymbol{Q}^{\mathrm{O}}_{\mathsf{V}}$.

Standard identification schemes. A *standard identification (SI) scheme* is a tuple $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$ where $\mathsf{Kg}$ is the randomized polynomial-time key generation algorithm, and $\mathsf{P}$ and $\mathsf{V}$ are polynomial-time interactive algorithms called the prover and verifier algorithms, respectively. In an initialization step, the prover runs $\mathsf{Kg}(1^k)$, where $k$ is a security parameter, to obtain a key pair $(pk, sk)$, and publishes the public key $pk$ while keeping the secret key $sk$ private. In the interactive identification protocol, the prover runs $\mathsf{P}$ with initial state $sk$, and the verifier runs $\mathsf{V}$ with initial state $pk$. We require that for all $k \in \mathbb{N}$ and all $(pk, sk) \in [\mathsf{Kg}(1^k)]$, the decision taken by the $\mathsf{V}$ in the interaction between $\mathsf{P}$ (initialized with $sk$) and $\mathsf{V}$ (initialized with $pk$) is $\texttt{acc}$ with probability one.

Security of SI schemes. We recall the notions of impersonation under passive (imp-pa), active (imp-aa) [FFS88], and concurrent (imp-ca) attack [BP02]. Let $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$ be a SI scheme, $\mathsf{A} = (\mathsf{CV}, \mathsf{CP})$ an adversary (consisting of a cheating verifier $\mathsf{CV}$ and a cheating prover $\mathsf{CP}$) and $k \in \mathbb{N}$ a security parameter. Consider the experiment of Figure 3. The type of attack atk $\in \{\text{pa}, \text{aa}, \text{ca}\}$ is a parameter, and the adversary has access to the oracles shown in the same Figure. The imp-atk *advantage* of $\mathsf{A}$ in attacking $\mathcal{SI}$ is

$$\mathbf{Adv}^{\text{imp-atk}}_{\mathcal{SI},\mathsf{A}}(k) \;=\; \Pr\left[\mathbf{Exp}^{\text{imp-atk}}_{\mathcal{SI},\mathsf{A}}(k) = 1\right] .$$

We say that $\mathcal{SI}$ is an imp-atk-*secure SI scheme* if $\mathbf{Adv}^{\text{imp-atk}}_{\mathcal{SI},\mathsf{A}}(\cdot)$ is negligible for every polynomial-time $\mathsf{A}$. We now explain the experiment.

The cheating verifier $\mathsf{CV}$ gets initial inputs $1^k, pk$. In the case of a passive (pa) attack, $\mathsf{CV}$ gets a conversation oracle, which, upon a query, returns a transcript of the conversation between $\mathsf{P}$ (with initial state $sk$) and $\mathsf{V}$ (with initial state $pk$), each time generated under fresh coins for both parties. For an active attack (aa) or concurrent attack (ca), $\mathsf{CV}$ gets a prover oracle Prov. Upon a query $(s, M)$ where $s$ is a session number and $M$ is a message, the Prov oracle runs the prover algorithm using $M$ as an incoming message and returns the prover's outgoing message while maintaining the prover's state associated with the session $s$ across the invocations. (For each new session, Prov uses fresh random coins to start the prover, initializing it with $sk$.) The difference between active and concurrent attacks is that the former allows only a single session to be active at a time, while the latter allows for a polynomial number of arbitrarily interleaved sessions. Eventually, $\mathsf{CV}$ halts with some output that is given to $\mathsf{CP}$, and $\mathsf{A}$ wins if the interaction between $\mathsf{CP}$ and $\mathsf{V}$ (initialized with $pk$) leads the latter to accept. The advantage of the adversary is the probability that it wins.

Standard signature schemes. A *standard signature (SS) scheme* $\mathcal{SS}$ is a triple of algorithms $(\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$. On input $1^k$, where $k$ is the security parameter, the randomized key generation algorithm $\mathsf{Kg}$ returns a fresh key pair $(pk, sk)$. On input $sk$ and a message $M$, the possibly randomized signing algorithm $\mathsf{Sign}$ returns a signature $\sigma$. On input $pk, M$, and a signature $\sigma$, the deterministic verification algorithm $\mathsf{Vf}$ returns a decision (0 or 1) on whether $\sigma$ is a valid signature for $M$ relative to $pk$. In the random oracle model, the last two algorithms have oracle access to a function $H$ drawn at random from an appropriate space, with a range that might depend on $pk$. We require that, for all $k \in \mathbb{N}$, all $(pk, sk) \in [\mathsf{Kg}(1^k)]$ and all messages $M$, it is the case that $\mathsf{Vf}(pk, M, \mathsf{Sign}(sk, M)) = 1$.

Security of SS schemes. We use the standard notion of unforgeability under chosen-message attack (uf-cma) [GMR88]. Associated with a SS scheme $\mathcal{SS} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$, adversary $\mathsf{F}$ and value $k$ of the security parameter is an experiment in which we begin by running $\mathsf{Kg}$ on input $1^k$ to get keys $(pk, sk)$. Then we run $\mathsf{F}$ on input $1^k, pk$, providing it oracle access to $\mathsf{Sign}(sk, \cdot)$, until it halts with output a pair $(M, \sigma)$. We say that $\mathsf{F}$ wins if $\mathsf{Vf}(pk, M, \sigma) = 1$ but $M$ was not queried to $\mathsf{Sign}(sk, \cdot)$. The uf-cma *advantage* of $\mathsf{F}$ in breaking $\mathcal{SS}$, denoted $\mathbf{Adv}^{\text{uf-cma}}_{\mathsf{F},\mathcal{SS}}(k)$, is the probability that $\mathcal{SS}$ wins. We say that $\mathcal{SS}$ is uf-cma secure if $\mathbf{Adv}^{\text{uf-cma}}_{\mathsf{F},\mathcal{SS}}(\cdot)$ is negligible for every polynomial-time adversary $\mathsf{F}$.

Identity-based identification schemes. An *identity-based identification (IBI) scheme* is a four-tuple $\mathcal{IBI} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{P}}, \overline{\mathsf{V}})$ of polynomial-time algorithms. The trusted, key-issuing authority runs the *master-key generation* algorithm $\mathsf{MKg}$ on input $1^k$, where $k$ is a security parameter, to obtain a master public and secret key pair $(mpk, msk)$. It can then run the *user-key generation* algorithm $\mathsf{UKg}$ on $msk$ and the identity $I \in \{0,1\}^*$ of a user to generate for this user a secret key $usk$ which is then assumed to be securely communicated to the user in question. In the interactive identification protocol, the prover with identity $I$ runs interactive algorithm $\overline{\mathsf{P}}$ with initial state $usk$, and the verifier runs $\overline{\mathsf{V}}$ with initial state $mpk, I$.

In the random oracle model, $\mathsf{UKg}, \overline{\mathsf{P}}, \overline{\mathsf{V}}$ additionally have oracle access to a function H whose range

```
Oracle $\textsc{Init}(I)$
    If $I \in CU \cup HU \cup AU$ then return $\bot$
    $usk[I] \xleftarrow{\$} \mathsf{UKg}(msk, I)$ ; $HU \leftarrow HU \cup \{I\}$
    Return 1
```

```
Oracle $\textsc{Conv}(I)$
    If $I \notin HU$ then return $\bot$
    $(C, d) \xleftarrow{\$} \mathbf{Run}[\overline{\mathsf{P}}(usk[I]) \leftrightarrow \overline{\mathsf{V}}(mpk, I)]$
    Return $C$
```

```
Oracle $\textsc{Corr}(I)$
    If $I \notin HU \setminus AU$ then return $\bot$
    $CU \leftarrow CU \cup \{I\}$ ; $HU \leftarrow HU \setminus \{I\}$
    Return $usk[I]$
```

```
Oracle $\textsc{Prov}(I, s, M_{\mathrm{in}})$
    If $I \notin HU \setminus AU$ then return $\bot$
    If $(I, s) \notin PS$ then
        If $\mathrm{atk} = \mathrm{aa}$ then $PS \leftarrow \{(I, s)\}$
        If $\mathrm{atk} = \mathrm{ca}$ then $PS \leftarrow PS \cup \{(I, s)\}$
        Pick random coins $\rho$ for $\overline{\mathsf{P}}$
        $St_{\overline{\mathsf{P}}}[I, s] \leftarrow (usk[I], \rho)$
    $(M_{\mathrm{out}}, St_{\overline{\mathsf{P}}}[I, s]) \leftarrow \overline{\mathsf{P}}(M_{\mathrm{in}}, St_{\overline{\mathsf{P}}}[I, s])$
    Return $M_{\mathrm{out}}$
```

```
Experiment $\mathbf{Exp}^{\mathrm{imp\text{-}atk}}_{\mathcal{IBI},\overline{\mathsf{A}}}(k)$          // $\mathrm{atk} \in \{\mathrm{pa}, \mathrm{aa}, \mathrm{ca}\}$, $\overline{\mathsf{A}} = (\overline{\mathsf{CV}}, \overline{\mathsf{CP}})$
    $(mpk, msk) \xleftarrow{\$} \mathsf{MKg}(1^k)$
    $HU \leftarrow \emptyset$ ; $CU \leftarrow \emptyset$ ; $AU \leftarrow \emptyset$   // sets of honest, corrupt and attacked users
    $PS \leftarrow \emptyset$                                     // set of active prover sessions
    If $\mathrm{atk} = \mathrm{pa}$ then let $\textsc{Or}$ denote $\textsc{Conv}$ else let $\textsc{Or}$ denote $\textsc{Prov}$
    $(I_{\mathrm{b}}, St_{\overline{\mathsf{CP}}}) \leftarrow \overline{\mathsf{CV}}(1^k, mpk : \textsc{Init}, \textsc{Corr}, \textsc{Or})$
    $AU \leftarrow \{I_{\mathrm{b}}\}$ ; If $I_{\mathrm{b}} \notin HU$ then return 0
    $(C, d) \xleftarrow{\$} \mathbf{Run}[\overline{\mathsf{CP}}(St_{\overline{\mathsf{CP}}} : \textsc{Init}, \textsc{Corr}, \textsc{Or}) \leftrightarrow \overline{\mathsf{V}}(mpk, I_{\mathrm{b}})]$
    Return $d$
```

Figure 4: Oracles given to adversary attacking IBI scheme $\mathcal{IBI} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{P}}, \overline{\mathsf{V}})$, and experiment used to define imp-atk security of the scheme.

may depend on *mpk*. We require that for all $k \in \mathbb{N}$, $I \in \{0,1\}^*$, $(mpk, msk) \in [\mathsf{MKg}(1^k)]$, functions H with appropriate domain and range, and $usk \in [\mathsf{UKg}(msk, I : \mathrm{H})]$, the interaction between $\overline{\mathsf{P}}$ (initialized with *usk*) and $\overline{\mathsf{V}}$ (initialized with $mpk, I$) is $\mathtt{acc}$ with probability one.

Security of IBI schemes. We first provide the formal definitions and then the explanations. Let $\mathcal{IBI} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{P}}, \overline{\mathsf{V}})$ be an IBI scheme, $\overline{\mathsf{A}} = (\overline{\mathsf{CV}}, \overline{\mathsf{CP}})$ an adversary (consisting of a cheating verifier $\overline{\mathsf{CV}}$ and a cheating prover $\overline{\mathsf{CP}}$) and $k \in \mathbb{N}$ a security parameter. Consider the experiment of Figure 4. The type of attack $\mathrm{atk} \in \{\mathrm{pa}, \mathrm{aa}, \mathrm{ca}\}$ is a parameter, and the adversary has access to the oracles shown in the same Figure. The imp-atk *advantage* of $\overline{\mathsf{A}}$ in attacking $\mathcal{IBI}$ is

$$\mathbf{Adv}^{\mathrm{imp\text{-}atk}}_{\mathcal{IBI},\overline{\mathsf{A}}}(k) \;=\; \Pr\left[\mathbf{Exp}^{\mathrm{imp\text{-}atk}}_{\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1\right] .$$

We say that $\mathcal{IBI}$ is an imp-atk *secure IBI scheme* if $\mathbf{Adv}^{\mathrm{imp\text{-}atk}}_{\mathcal{IBI},\overline{\mathsf{A}}}(\cdot)$ is negligible for every polynomial-time $\overline{\mathsf{A}}$.

The main difference from the SI experiment is that $\overline{\mathsf{A}}$ can initialize or corrupt identities of its choice through the $\textsc{Init}$ and $\textsc{Corr}$ oracles. When an identity is initialized, it is issued a secret key by the authority. When an (honest) identity is corrupted, its secret key is returned to the adversary. *HU* is the set of honest users, and *CU* is the set of corrupted users. In the case of a passive attack the adversary gets a conversation oracle $\textsc{Conv}$ that, when queried with the identity $I$ of an honest

```
Oracle $\textsc{Init}(I)$
    If $I \in CU \cup HU$ then return $\perp$
    $usk[I] \xleftarrow{\$} \mathsf{UKg}(msk, I)$
    $MSG[I] \leftarrow \emptyset$ ; $HU \leftarrow HU \cup \{I\}$
    Return 1
```

```
Oracle $\textsc{Sign}(I, M)$
    If $I \notin HU$ then return $\perp$
    $\sigma \xleftarrow{\$} \overline{\mathsf{Sign}}(usk[I], M)$ ; $MSG[I] \leftarrow MSG[I] \cup \{M\}$
    Return $\sigma$
```

```
Experiment $\mathbf{Exp}^{\text{uf-cma}}_{\mathcal{IBS},\overline{\mathsf{F}}}(k)$
    $(mpk, msk) \xleftarrow{\$} \mathsf{MKg}(1^k)$
    $HU \leftarrow \emptyset$ ; $CU \leftarrow \emptyset$                // sets of honest and corrupt users
    $(I, M, \sigma) \xleftarrow{\$} \overline{\mathsf{F}}(1^k, mpk : \textsc{Init}(\cdot), \textsc{Sign}(\cdot,\cdot), \textsc{Corr}(\cdot))$
    If ($I \in HU$ and $\mathsf{Vf}(mpk, I, M, \sigma) = 1$ and $M \notin MSG[I]$) then return 1 else return 0
```

Figure 5: Oracles given to adversary attacking IBS scheme $\mathcal{IBS} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{Sign}}, \overline{\mathsf{Vf}})$, and experiment used to define uf-cma security of the scheme. The oracle Corr is the same as that in Figure 4 and thus is not shown here.

and initialized user, returns a transcript of a conversation between $I$ (playing the role of prover and using its issued secret key) and the verifier, each time using fresh coins. In the case of an active or concurrent attack, the adversary gets access to the prover oracle Prov. Its arguments are an identity, a session number, and a message that the adversary, playing the role of verifier, sends to $I$ in its role as a prover. The oracle maintains state for the prover for each active session, but allows only one session to be active at any point if the attack is an active one rather than a concurrent one. At the end of its execution, $\overline{\mathsf{CV}}$ transfers its state to $\overline{\mathsf{CP}}$ and outputs an uncorrupted identity $I_{\mathrm{b}}$. In the second stage, $\overline{\mathsf{CP}}$ will try to impersonate $I_{\mathrm{b}}$. An element of this definition worth drawing attention to is that we have allowed $\overline{\mathsf{CP}}$ to query the same oracles as $\overline{\mathsf{CV}}$. This allows $\overline{\mathsf{CP}}$ to initialize, corrupt, interact with, or see conversations involving certain identities depending on the challenge it gets from the verifier. The only restriction is that $\overline{\mathsf{CP}}$ *cannot* submit queries involving $I_{\mathrm{b}}$ because otherwise impersonating $I_{\mathrm{b}}$ would become trivial. The restrictions are all enforced by the oracles themselves. (At the end of the first stage, $I_{\mathrm{b}}$ is added to the set of users under attack $AU$ and, in the case of active or concurrent attacks, removed from the set of honest users $HU$.)

Identity-based signature schemes. An *identity-based signature (IBS) scheme* is a tuple $\mathcal{IBS} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{Sign}}, \overline{\mathsf{Vf}})$ of polynomial-time algorithms. The first three may be randomized but the last is not. The trusted, key-issuing authority runs the master-key generation algorithm $\mathsf{MKg}$ on input $1^k$, where $k$ is a security parameter, to obtain a master public and secret key pair $(mpk, msk)$. It can then run the user-key generation algorithm $\mathsf{UKg}$ on $msk$ and the identity $I \in \{0,1\}^*$ thus generating for the user $I$ a secret key $usk$ which is then assumed to be securely communicated to the user in question. On input $usk$ and a message $M$, the signing algorithm $\overline{\mathsf{Sign}}$ returns a signature of $M$. On input $mpk, I, M$, and a signature $\sigma$, the verification algorithm $\overline{\mathsf{Vf}}$ returns a decision on whether $\sigma$ is valid for $I$ and $M$. We require that, for all $k \in \mathbb{N}$, $M \in \{0,1\}^*$, and $I \in \{0,1\}^*$,

$$\Pr\left[\,(mpk, msk) \xleftarrow{\$} \mathsf{MKg}(1^k)\;;\; usk \xleftarrow{\$} \mathsf{UKg}(msk, I)\;;\; \sigma \xleftarrow{\$} \overline{\mathsf{Sign}}(usk, M) \;:\; \overline{\mathsf{Vf}}(mpk, I, M, \sigma) = 1\,\right] \;=\; 1\;.$$

Security of IBS schemes. We first provide the formal definition following [CC03, DKXY03] and then the explanations. Let $\mathcal{IBS} = (\mathsf{MKg}, \mathsf{UKg}, \mathsf{Sign}, \mathsf{Vf})$ be an IBS scheme, $\overline{\mathsf{F}}$ an adversary, and $k \in \mathbb{N}$ a security parameter. Consider the experiment of Figure 5. The adversary has access to the oracles shown in the same Figure. The uf-cma *advantage* of $\overline{\mathsf{F}}$ in attacking $\mathcal{IBS}$ is

$$\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{IBS},\overline{\mathsf{F}}}(k) \;=\; \Pr\left[\mathbf{Exp}^{\text{uf-cma}}_{\mathcal{IBS},\overline{\mathsf{F}}}(k) = 1\right].$$

We say that $\mathcal{IBS}$ is a uf-cma *secure IBS scheme* if $\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{IBS},\overline{\mathsf{F}}}(\cdot)$ is negligible for every polynomial-time adversary $\overline{\mathsf{F}}$.

Via $\textsc{Init}(I)$, the adversary $\overline{\mathsf{F}}$ can create a user $I$ and give it a secret key denoted $\mathit{usk}[I]$. Via $\textsc{Sign}(I, M)$, it can obtain $I$'s signature on a message $M$ of its choice. Via $\textsc{Corr}(I)$, it can obtain $I$'s secret key $\mathit{usk}[I]$. To win, $\overline{\mathsf{F}}$ must output the identity $I$ of an uncorrupted user, a message $M$, and a signature $\sigma$ such that $I$ did not previously sign $M$ but $\overline{\mathsf{Vf}}(\mathit{mpk}, I, M, \sigma) = 1$. Here, $\mathit{HU}$ is the set of honest users, $\mathit{CU}$ is the set of corrupted users, and $\mathit{MSG}[I]$ is the set of messages that $I$ has signed. The uf-cma advantage of $\overline{\mathsf{F}}$ is its success probability.

## 3 Certificate-based IBI and IBS: Schemes without Random Oracles

There is a natural way to construct IBI and IBS schemes using certificates. This may sound paradoxical since the purpose of identity-based cryptography is to avoid certificates, but certification here refers to a technique, not a PKI. The idea is simply that the authority can issue a certificate, consisting of a signature of a user's identity and "public key," the latter being a value it chooses and provides to the user along with a matching secret key. Now, to accomplish IBI, a prover can send this public key and certificate to the verifier, and then the parties can run a SI protocol based on the public key. Since the verifier needs to know only the authority public key and identity of the prover, this is identity-based. Similarly, by adding to a standard signature under $\mathit{pk}$ the value $\mathit{pk}$ itself and its certificate, verification of this signature becomes possible given only the authority public key and identity of the user, and hence is identity-based. (Note that no such simple trick works for identity-based encryption, which is a much harder problem.)

We believe these facts are folklore, but are worth detailing and proving them for several reasons. One is that they show that IBI and IBS can be achieved without random oracles (all the dedicated schemes we consider use random oracles) and thereby enable us to answer the foundational question of finding the minimal assumptions for the existence of IBI and IBS schemes (cf. Corollaries 3.3 and 3.6). Another reason is that these simple schemes are benchmarks relative to which dedicated schemes measure their efficiency. We now provide some details.

### 3.1 Certificate-based IBI

We show the design of an IBI scheme based on any SI scheme and any SS scheme. Let $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$ be a SI scheme, and let $\mathcal{SS} = (\mathsf{SKg}, \mathsf{Sign}, \mathsf{Vf})$ be a SS scheme. We associate to them an IBI scheme $\mathcal{C}\mathit{ert}\text{-}\mathcal{IBI} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{P}}, \overline{\mathsf{V}})$ whose constituent algorithms are as follows. The master key generation algorithm $\mathsf{MKg}$ is simply $\mathsf{SKg}$, so that the master secret key $\mathit{msk}$ can be used to produce signatures verifiable under $\mathit{mpk}$. To issue a secret key $\mathit{usk}$ to a user with identity $I$, the authority first runs $\mathsf{Kg}(1^k)$ to obtain a public and secret key pair $(\mathit{pk}, \mathit{sk})$ for the SI scheme. It then creates the certificate $\mathit{cert} \leftarrow (\mathit{pk}, \mathsf{Sign}(\mathit{msk}, \mathit{pk}\|I))$. It sets $\mathit{usk} \leftarrow (\mathit{sk}, \mathit{cert})$ and sends the latter to $I$. The interactive algorithm $\overline{\mathsf{P}}$, run by $I$ to identify itself, runs $\mathsf{P}$, initializing the latter with $\mathit{sk}$, and includes $\mathit{cert}$ in its first message to the verifier. The interactive algorithm $\overline{\mathsf{V}}$, run by the verifier, has initial input $(\mathit{mpk}, I)$. It retrieves $\mathit{cert}$ from the first message sent by the prover. It then verifies the signature on the certificate $\mathit{cert}$ by parsing $\mathit{cert}$ as $(\mathit{pk}, \sigma)$ and running $\mathsf{Vf}(\mathit{mpk}, \mathit{pk}\|I, \sigma)$. If the certificate is invalid, $\overline{\mathsf{V}}$ halts and rejects. Otherwise, it runs $\mathsf{V}$, initializing the latter with $\mathit{pk}$. It accepts if $\mathsf{V}$ accepts.

**Construction 3.1 (Certificate-based IBI)** Given a standard identification scheme $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$ and a (standard) signature scheme $\mathcal{SS} = (\mathsf{SKg}, \mathsf{Sign}, \mathsf{Vf})$, we associate to them an IBI scheme $\mathcal{C}ert\text{-}\mathcal{IBI} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{P}}, \overline{\mathsf{V}})$ as described above. ▮

The proof of the following is based on standard ideas and is detailed in Appendix A.

**Theorem 3.2 (Security of Certificate-based IBI)** Let $\mathcal{SI}$ be a SI scheme, and $\mathcal{SS}$ a uf-cma secure SS scheme. Let $\mathcal{C}ert\text{-}\mathcal{IBI}$ be the corresponding certificate-based IBI scheme as per Construction 3.1. If $\mathcal{SI}$ is imp-atk secure then $\mathcal{C}ert\text{-}\mathcal{IBI}$ is imp-atk secure, for any atk $\in \{\mathrm{pa}, \mathrm{aa}, \mathrm{ca}\}$. ▮

Given a digital signature scheme one can easily construct an imp-ca secure SI scheme [BFGM01]. (The scheme consists simply of the verifier sending a random challenge which the prover signs.) Since there are numerous constructions of signature schemes without random oracles, we obtain from Construction 3.1 and Theorem 3.2 numerous constructions of IBI schemes without random oracles. In particular, we have the following corollary.

**Corollary 3.3** There exists a secure (imp-pa, imp-aa, or imp-ca) IBI scheme if and only if there exists a one-way function. ▮

**Proof:** The existence of one-way functions implies the existence of uf-cma digital signature schemes [Rom90], and we noted above that any uf-cma digital signature scheme yields a imp-ca secure SI scheme. So Construction 3.1 and Theorem 3.2 give us an imp-ca secure IBI scheme. Since imp-ca security implies imp-pa and imp-aa security, this completes one direction of the proof.

For the other direction, note that both imp-aa and imp-ca security imply imp-pa security, and it is easy to see that even the existence of an imp-pa secure IBI scheme implies the existence of a one-way function. This follows from [IL89] or by noting that if $\mathcal{C}ert\text{-}\mathcal{IBI} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{P}}, \overline{\mathsf{V}})$ is an imp-pa secure IBI scheme and $\rho$ is the length of the random tape for $\mathsf{MKg}$, then the function that maps $x \in \{0,1\}^{\rho}$ to the master public key obtained by running $\mathsf{MKg}$ on random tape $x$ is one-way. ▮

### 3.2 Certificate-based IBS

Similar ideas and results hold for IBS schemes, and we outline them briefly. The construction is folklore, and essentially the same as the generic key-insulated signature scheme of [DKXY03]. To any standard digital signature scheme $\mathcal{SS} = (\mathsf{SKg}, \mathsf{Sign}, \mathsf{Vf})$, we associate the following IBS scheme $\mathcal{C}ert\text{-}\mathcal{IBS} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{Sign}}, \overline{\mathsf{Vf}})$. Just like in the IBI scheme of Construction 3.1, the master key generation algorithm $\mathsf{MKg}$ is simply $\mathsf{SKg}$, and the secret key $usk$ of a user with identity $I$ is a pair $(sk, cert)$, where $sk$ is a secret key generated by $\mathsf{SKg}$, and $cert = (pk, \mathsf{Sign}(msk, pk \| I))$ is a certificate for the corresponding public key $pk$. The signature on a message $M$ by user $I$ consists of a signature $\sigma \leftarrow \mathsf{Sign}(sk, M)$ and the certificate $cert$. The verification algorithm parses the signature as $(\sigma, (pk, \sigma'))$ and returns 1 if and only if $\mathsf{Vf}(pk, M, \sigma) = \mathsf{Vf}(mpk, pk \| I, \sigma') = 1$.

**Construction 3.4 (Certificate-based IBS)** Given a standard digital signature scheme $\mathcal{SS} = (\mathsf{SKg}, \mathsf{Sign}, \mathsf{Vf})$, we associate to it an IBS scheme $\mathcal{C}ert\text{-}\mathcal{IBS} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{Sign}}, \overline{\mathsf{Vf}})$ as described above. ▮

**Theorem 3.5 (Security of Certificate-based IBS)** Let $\mathcal{SS}$ be a uf-cma secure SS scheme. Let $\mathcal{IBS}$ be the corresponding certificate-based IBS scheme as per Construction 3.4. Then $\mathcal{IBS}$ is a uf-cma secure IBS scheme. ▮

We omit the proof since it is similar to the proof of Theorem 3.2 and that of the generic construction of [DKXY03]. Since there are numerous constructions of uf-cma secure SS schemes without random oracles, we obtain from the above IBS schemes without random oracles. In particular, we have the following corollary.

**Corollary 3.6** There exists a uf-cma secure IBS scheme if and only if there exists a one-way function. ▌

**Proof:** Given that the existence of uf-cma secure digital signature schemes is equivalent to the existence of one-way functions [Rom90], this follows from Theorem 3.5 and the fact that a SS scheme can be constructed from an IBS scheme by including an arbitrary identity $I$ in the public key of the SS scheme and including the user secret key corresponding to $I$ in the secret key. ▌

### 3.3 Discussion of Certificate-based Constructs

One can obtain fairly efficient constructions of IBI and IBS schemes through the above. The prover of the $\mathcal{C}ert\text{-}\mathcal{IBI}$ scheme of Construction 3.1 is as efficient as the prover of the underlying SI scheme. (However, the verification cost grows by the cost of verifying one signature, and the communication increases due to the transmission of the certificate.) Signing in the $\mathcal{C}ert\text{-}\mathcal{IBS}$ scheme of Construction 3.4 costs the same as for the underlying SS scheme. Verification arguably costs the same too if one takes into account that in a SS scheme one must also verify a CA-issued certificate of the public key. The size of the signature increases due to inclusion of the certificate, but again one can argue that with a SS scheme one will in practice transmit a CA-issued certificate with the signature, making the two comparable again.

We remark that implementing the schemes of Constructions 3.1 and 3.4 with signature schemes permitting aggregation [BGLS03] will reduce the communication costs. The dedicated IBI and IBS schemes that follow attempt to reduce costs below that of even the best instantiations of Constructions 3.1 and 3.4.

Finally, we remark that, while none of the IBI schemes that follow are secure against reset attack [BFGM01] (in which the adversary is allowed to rewind the prover and run it again on the same random coins), one can be obtained from Construction 3.1. To do this, use as SI scheme one of the reset-attack secure SI schemes from [BFGM01].

## 4 Transformations

We begin by defining trapdoor samplable relations. Then we define convertible SI (cSI) schemes and related transforms.

### 4.1 Trapdoor Samplable Relations

A *relation* is a finite set of ordered pairs. We define the *range* of a relation $\mathbf{R}$, the set of images of $x$, and the set of inverses of $y$, respectively, as

$$\begin{aligned} \mathrm{Rng}(\mathbf{R}) &= \{\, y \,:\, \exists\, x \text{ such that } (x,y) \in \mathbf{R}\,\} \\ \mathbf{R}(x) &= \{\, y \,:\, (x,y) \in \mathbf{R}\,\} \\ \mathbf{R}^{-1}(y) &= \{\, x \,:\, (x,y) \in \mathbf{R}\,\}\;. \end{aligned}$$

**Definition 4.1 (Trapdoor Samplable Relations)** A family of *trapdoor samplable relations* $\mathcal{F}$ is a triplet of polynomial-time algorithms $(\mathsf{TDG}, \mathsf{Smp}, \mathsf{Inv})$ such that the following properties hold:

- *Efficient generation:* On input $1^k$, where $k \in \mathbb{N}$ is the security parameter, $\mathsf{TDG}$ outputs the description $\langle \mathbf{R} \rangle$ of a relation $\mathbf{R}$ together with its trapdoor information $t$;
- *Samplability:* The output of the algorithm $\mathsf{Smp}$ on an input $\langle \mathbf{R} \rangle$ is uniformly distributed over $\mathbf{R}$;
- *Inversion:* On input a relation description $\langle \mathbf{R} \rangle$, the corresponding trapdoor $t$, and an element $y \in \mathrm{Rng}(\mathbf{R})$, the randomized algorithm $\mathsf{Inv}$ outputs a random element of $\mathbf{R}^{-1}(y)$;
- *Regularity:* For every relation $\mathbf{R}$ in the family, there is an integer $d$ such that $|\mathbf{R}^{-1}(y)| = d$ for all $y \in \mathrm{Rng}(\mathbf{R})$. ▮

When we refer to the family of relations defined by $\mathcal{F}$ we simply mean

$$\{\ \mathbf{R}\ :\ \exists k, t \text{ such that } (\langle \mathbf{R} \rangle, t) \in [\mathsf{TDG}(1^k)]\ \}\ .$$

A family of trapdoor one-way permutations gives rise to a family of trapdoor samplable relations in a natural way. Namely, to every member $\mathbf{f}$ of the former family corresponds the relation $\mathbf{R}$ consisting of the set of pairs $(x, \mathbf{f}(x))$ for $x$ in the domain of the function $\mathbf{f}$. However, trapdoor samplable relations are a more general concept, and we will see examples where this greater generality is needed.

## 4.2 Convertible Schemes and the cSI-2-IBI Transform

In analogy with the definition of trapdoor signature schemes [DKXY03], we define the concept of *convertible identification schemes* and show how to transform these into IBI schemes. A SI scheme is called convertible if its key-generation process is underlain by a family of trapdoor samplable relations in the manner specified below.

**Definition 4.2 (Convertible SI Schemes)** A SI scheme $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$ is said to be *convertible* if there exists a family of trapdoor samplable relations $\mathcal{F} = (\mathsf{TDG}, \mathsf{Smp}, \mathsf{Inv})$ such that for all $k \in \mathbb{N}$ the output of the following is distributed identically to the output of $\mathsf{Kg}(1^k)$:

$$(\langle \mathbf{R} \rangle, t) \stackrel{\$}{\leftarrow} \mathsf{TDG}(1^k)\ ;\ (x, y) \stackrel{\$}{\leftarrow} \mathsf{Smp}(\langle \mathbf{R} \rangle)\ ;\ pk \leftarrow (\langle \mathbf{R} \rangle, y)\ ;\ sk \leftarrow (\langle \mathbf{R} \rangle, x)\ ;\ \text{Return } (pk, sk)$$ ▮

The following describes the $\mathsf{cSI\text{-}2\text{-}IBI}$ transform of a convertible SI (cSI) scheme into an IBI scheme. The idea is that to each identity $I$ we can associate a "pseudo-public-key" that is derivable from the master public key and $I$ and plays the role of a public key for the underlying cSI scheme. This "pseudo-public-key" is $(\langle \mathbf{R} \rangle, \mathrm{H}(I))$, where H is a random oracle.

**Construction 4.3 (The cSI-2-IBI Transform)** Let $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$ be a cSI scheme, and let $\mathcal{F} = (\mathsf{TDG}, \mathsf{Smp}, \mathsf{Inv})$ be the family of trapdoor samplable relations that underlies it as per Definition 4.2. The $\mathsf{cSI\text{-}2\text{-}IBI}$ transform associates to $\mathcal{SI}$ the random oracle model IBI scheme $\mathcal{IBI} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{P}}, \overline{\mathsf{V}})$ whose components we now describe. The master and user key generation algorithms are defined as

```
Algorithm $\mathsf{MKg}(1^k)$
    $(\langle \mathbf{R} \rangle, t) \stackrel{\$}{\leftarrow} \mathsf{TDG}(1^k)$
    $mpk \leftarrow \langle \mathbf{R} \rangle$ ; $msk \leftarrow (\langle \mathbf{R} \rangle, t)$
    Return $(mpk, msk)$
```

```
Algorithm $\mathsf{UKg}(msk, I : \mathrm{H})$
    Parse $msk$ as $(\langle \mathbf{R} \rangle, t)$
    $x \stackrel{\$}{\leftarrow} \mathsf{Inv}(\langle \mathbf{R} \rangle, t, \mathrm{H}(I))$ ; $usk \leftarrow (\langle \mathbf{R} \rangle, x)$
    Return $usk$
```

where $\mathrm{H} : \{0,1\}^* \to \mathrm{Rng}(\mathbf{R})$ is a random oracle. The prover algorithm $\overline{\mathsf{P}}$ is identical to $\mathsf{P}$. The verifier algorithm $\overline{\mathsf{V}}$, which takes initial input $\langle \mathbf{R} \rangle, I$ and oracle H, runs $\mathsf{V}$ on initial input $(\langle \mathbf{R} \rangle, \mathrm{H}(I))$. ▮

The following theorem says that $\mathsf{cSI\text{-}2\text{-}IBI}$ is security-preserving.

**Theorem 4.4 (Security of cSI-2-IBI)** Let $\mathcal{SI}$ be a cSI scheme and let $\mathcal{IBI} = \mathsf{cSI\text{-}2\text{-}IBI}(\mathcal{SI})$ be the associated IBI scheme as per Construction 4.3. For any atk $\in \{\mathrm{pa}, \mathrm{aa}, \mathrm{ca}\}$, if $\mathcal{SI}$ is imp-atk secure then $\mathcal{IBI}$ is imp-atk secure. ▌

**Proof:** Let $\overline{\mathsf{A}} = (\overline{\mathsf{CV}}, \overline{\mathsf{CP}})$ be a polynomial time adversary mounting an imp-atk attack on $\mathcal{IBI}$. Say $\overline{\mathsf{CV}}$ makes at most $\boldsymbol{Q}^{\mathrm{H}}_{\overline{\mathsf{CV}}}(\cdot)$ queries to its H oracle and at most $\boldsymbol{Q}^{\textsc{Init}}_{\overline{\mathsf{CV}}}(\cdot)$ to its Init oracle. We construct a polynomial-time adversary $\mathsf{A} = (\mathsf{CV}, \mathsf{CP})$ mounting an imp-atk attack on $\mathcal{SI}$ such that for all $k \in \mathbb{N}$

$$\mathbf{Adv}^{\text{imp-atk}}_{\mathcal{IBI},\overline{\mathsf{A}}}(k) \le \left[\boldsymbol{Q}^{\mathrm{H}}_{\overline{\mathsf{CV}}}(k) + \boldsymbol{Q}^{\textsc{Init}}_{\overline{\mathsf{CV}}}(k)\right] \cdot \mathbf{Adv}^{\text{imp-atk}}_{\mathcal{SI},\mathsf{A}}(k) \ .$$

The theorem follows.

Algorithms $\mathsf{CV}$ and $\mathsf{CP}$ are described in Figure 6. These algorithms run $\overline{\mathsf{CV}}$ and $\overline{\mathsf{CP}}$, replacing their oracles with subroutines that they themselves define. The subroutines are also shown in Figure 6.

Algorithm $\mathsf{CV}$ takes input $1^k, pk = (\langle \mathbf{R} \rangle, y)$ and has access to either a Conv oracle (in the case of a passive attack) or a Prov oracle (in the case of an active or concurrent attack). It will run $\overline{\mathsf{CV}}$ on input $1^k$, $mpk = \langle \mathbf{R} \rangle$. Its strategy is to guess in advance the identity $I_{\mathrm{b}}$ that $\overline{\mathsf{CV}}$ will try to attack, and ensure that the hash of this identity equals $y$. This means that in the second phase, the pseudo-public-key of $I_{\mathrm{b}}$, the identity that $\overline{\mathsf{CP}}$ is attacking, is $pk$, so $\overline{\mathsf{CP}}$ can be used by $\mathsf{CP}$ to attack $pk$. To ensure that the correspondence between the pseudo-public-key of $I_{\mathrm{b}}$ and $pk$ is accurate, $\mathsf{CV}$ will simulate the conversation or prover oracles for $I_{\mathrm{b}}$ via its own conversation or prover oracles. It will arrange to know the secret keys corresponding to identities other than $I_{\mathrm{b}}$ and thus simulate the conversation or prover oracles for these directly.

Guessing an identity from the infinite set $\{0,1\}^*$ is of course infeasible. Instead, $\mathsf{CV}$ picks at random a value $q_{\mathrm{g}}$ in the range $1, \ldots, \boldsymbol{Q}^{\mathrm{H}}_{\overline{\mathsf{CV}}}(k) + \boldsymbol{Q}^{\textsc{Init}}_{\overline{\mathsf{CV}}}(k)$, and then views itself as guessing the identity $I_{\mathrm{g}}$ corresponding to the $q_{\mathrm{g}}$-th hash oracle query made by $\mathsf{CV}$.

$\mathsf{CP}$ simply forwards $\overline{\mathsf{CP}}$'s reply to the same message, answering $\overline{\mathsf{CP}}$'s oracle queries in the same way as $\mathsf{CV}$ did before.

For the analysis, we begin by noting that the input provided by $\mathsf{CV}$ to $\overline{\mathsf{CV}}$ is correctly distributed because the relation description included in the public key of a convertible SI scheme and the one included in the master public key $mpk$ of its $\mathsf{cSI\text{-}2\text{-}IBI}$ transform are both generated by the $\mathsf{TDG}(1^k)$ algorithm. Let **Good** be the event that $\overline{\mathsf{A}}$ does not corrupt identity $I_{\mathrm{g}}$ during the attack. We now explain why $\mathsf{A}$ provides a perfect simulation of $\overline{\mathsf{A}}$'s environment as long as event **Good** is true.

We first prove that, in the event **Good**, $\overline{\mathsf{A}}$'s view follows the same distribution as in a real attack against $\mathcal{IBI}$. (Since $\mathsf{A}$ treats oracle queries made by $\overline{\mathsf{CV}}$ or $\overline{\mathsf{CP}}$ in the same way, we don't distinguish between $\overline{\mathsf{CV}}$ and $\overline{\mathsf{CP}}$ in this analysis, but rather view $\overline{\mathsf{A}}$ as a single algorithm.) We already argued that $\overline{\mathsf{CV}}$'s input is correctly distributed. The initial state of $\overline{\mathsf{CP}}$ is generated by $\overline{\mathsf{CV}}$ as in the real game, and $\overline{\mathsf{CP}}$'s incoming protocol messages are correctly distributed because by Construction 4.3 $\overline{\mathsf{V}}$ runs $\mathsf{V}$ as a subroutine. The responses to $\overline{\mathsf{A}}$'s oracle queries can be seen to be correctly distributed as follows:

- H-sim($I$): The regularity of $\mathbf{R}$ and the uniform distribution of the output of the $\mathsf{Smp}$ algorithm over $\mathbf{R}$ imply that the response $\mathrm{HT}[I]$ is uniformly distributed over $\mathrm{Rng}(\mathbf{R})$ for $I \neq I_{\mathrm{g}}$. By Definition 4.2, the value $y$ included in the public key of $\mathcal{SI}$ is also generated via the $\mathsf{Smp}$ algorithm, and hence $y$ is uniformly distributed over $\mathrm{Rng}(\mathbf{R})$ for the same reasons.

- Init-sim($I$): The output of the Init-sim oracle is only determined by the sets $HU$ and $CU$, which the Init-sim and Corr-sim subroutines maintain in the exact same way as the Init and Corr oracles in the real game.

```
Subroutine $\textsc{Init-sim}(I)$
  If $I \in CU \cup HU$ then return $\perp$
  $temp \leftarrow \text{H-}\textsc{sim}(I)$ ; $HU \leftarrow HU \cup \{I\}$
  Return 1
```

```
Subroutine $\textsc{Corr-sim}(I)$
  If $I \notin HU$ then return $\perp$
  $CU \leftarrow CU \cup \{I\}$ ; $HU \leftarrow HU \setminus \{I\}$
  If $I = I_{\rm g}$ then abort
  Return $(\langle \mathbf{R} \rangle, \mathrm{USK}[I])$
```

```
Subroutine $\text{H-}\textsc{sim}(I)$
  If $I \notin QU$ then
    $QU \leftarrow QU \cup \{I\}$
    If $|QU| = q_{\rm g}$ then
      $I_{\rm g} \leftarrow I$ ; $\mathrm{HT}[I] \leftarrow y$
    Else $(\mathrm{USK}[I], \mathrm{HT}[I]) \xleftarrow{\$} \mathsf{Smp}(\langle \mathbf{R} \rangle)$
  Return $\mathrm{HT}[I]$
```

```
Subroutine $\textsc{Conv-sim}(I)$
  If $I \notin HU$ then return $\perp$
  If $I = I_{\rm g}$ then $C \leftarrow \textsc{Or}(\varepsilon)$
  Else
    $pk[I] \leftarrow (\langle \mathbf{R} \rangle, \mathrm{HT}[I])$ ; $sk[I] \leftarrow (\langle \mathbf{R} \rangle, \mathrm{USK}[I])$
    $(C, d) \xleftarrow{\$} \mathbf{Run}[\mathsf{P}(sk[I]) \leftrightarrow \mathsf{V}(pk[I])]$
  Return $C$
```

```
Subroutine $\textsc{Prov-sim}(I, s, M_{\rm in})$
  If $I \notin HU$ then return $\perp$
  If $(I, s) \notin PS$ then
    If $\mathrm{atk} = \mathrm{aa}$ then $PS \leftarrow \{(I, s)\}$
    If $\mathrm{atk} = \mathrm{ca}$ then $PS \leftarrow PS \cup \{(I, s)\}$
    Pick random coins $\rho_{\overline{\mathsf{P}}}$ for $\overline{\mathsf{P}}$
    $St_{\overline{\mathsf{P}}}[I, s] \leftarrow ((\langle \mathbf{R} \rangle, \mathrm{USK}[I]), \rho_{\overline{\mathsf{P}}})$
  If $I = I_{\rm g}$ then $M_{\rm out} \leftarrow \textsc{Or}(s, M_{\rm in})$
  Else $(M_{\rm out}, St_{\overline{\mathsf{P}}}[I, s]) \leftarrow \overline{\mathsf{P}}(M_{\rm in}, St_{\overline{\mathsf{P}}}[I, s])$
  Return $M_{\rm out}$
```

```
Algorithm $\mathsf{CV}(1^k, pk : \textsc{Or})$
    Parse $pk$ as $(\langle \mathbf{R} \rangle, y)$ ; $HU \leftarrow \emptyset$ ; $CU \leftarrow \emptyset$ ; $QU \leftarrow \emptyset$
    $q_{\rm g} \xleftarrow{\$} \{1, \ldots, \boldsymbol{Q}^{\rm H}_{\overline{\mathsf{CV}}}(k) + \boldsymbol{Q}^{\textsc{Init}}_{\overline{\mathsf{CV}}}(k)\}$ ; $mpk \leftarrow \langle \mathbf{R} \rangle$
    If $\mathrm{atk} = \mathrm{pa}$ then $(I_{\rm b}, St_{\overline{\mathsf{CP}}}) \xleftarrow{\$} \overline{\mathsf{CV}}(1^k, mpk : \textsc{Init-sim}, \textsc{Corr-sim}, \textsc{Conv-sim}, \text{H-}\textsc{sim})$
    Else $(I_{\rm b}, St_{\overline{\mathsf{CP}}}) \xleftarrow{\$} \overline{\mathsf{CV}}(1^k, mpk : \textsc{Init-sim}, \textsc{Corr-sim}, \textsc{Prov-sim}, \text{H-}\textsc{sim})$
    If $|QU| < q_{\rm g}$ or $I_{\rm b} \neq I_{\rm g}$ then abort
    $HU \leftarrow HU \setminus \{I_{\rm b}\}$ ; $CU \leftarrow CU \cup \{I_{\rm b}\}$
    $St_{\mathsf{CP}} \leftarrow (St_{\overline{\mathsf{CP}}}, \langle \mathbf{R} \rangle, HU, CU, QU, \mathrm{HT}, \mathrm{USK}, I_{\rm g}, q_{\rm g})$
    Return $St_{\mathsf{CP}}$
```

```
Algorithm $\mathsf{CP}(M_{\rm in}, St_{\mathsf{CP}})$
    Parse $St_{\mathsf{CP}}$ as $(St_{\overline{\mathsf{CP}}}, \langle \mathbf{R} \rangle, HU, CU, QU, \mathrm{HT}, \mathrm{USK}, I_{\rm g}, q_{\rm g})$
    If $\mathrm{atk} = \mathrm{pa}$ then $(M_{\rm out}, St_{\overline{\mathsf{CP}}}) \xleftarrow{\$} \overline{\mathsf{CP}}(M_{\rm in}, St_{\overline{\mathsf{CP}}} : \textsc{Init-sim}, \textsc{Corr-sim}, \textsc{Conv-sim}, \text{H-}\textsc{sim})$
    Else $(M_{\rm out}, St_{\overline{\mathsf{CP}}}) \xleftarrow{\$} \overline{\mathsf{CP}}(M_{\rm in}, St_{\overline{\mathsf{CP}}} : \textsc{Init-sim}, \textsc{Corr-sim}, \textsc{Prov-sim}, \text{H-}\textsc{sim})$
    $St_{\mathsf{CP}} \leftarrow (St_{\overline{\mathsf{CP}}}, \langle \mathbf{R} \rangle, HU, CU, QU, \mathrm{HT}, \mathrm{USK}, I_{\rm g}, q_{\rm g})$
    Return $(M_{\rm out}, St_{\mathsf{CP}})$
```

Figure 6: Algorithms $\mathsf{CV}$ and $\mathsf{CP}$ constituting adversary $\mathsf{A}$ of the proof of Theorem 4.4, and their subroutines. Above, $\textsc{Or}$ is a conversation oracle if $\mathrm{atk} = \mathrm{pa}$ and a prover oracle if $\mathrm{atk} \in \{\mathrm{aa}, \mathrm{ca}\}$.

- $\textsc{Conv-sim}(I)$: Simulated conversations for $I = I_{\rm g}$ are easily seen to be correctly distributed from Construction 4.12. Due to the regularity of $\mathbf{R}$, the user secret key used to generate conversations for identity $I \neq I_{\rm g}$ in a real attack against $\mathcal{IBI}$ is uniformly distributed over $\mathbf{R}^{-1}(\mathrm{H}(I))$. Since in the simulation the pair $(\mathrm{USK}[I], \mathrm{HT}[I])$ was generated by the $\mathsf{Smp}$ algorithm, the user secret key used for the simulated conversations are distributed identically to the one used in a real attack, and hence also the output of the $\overline{\mathsf{P}}$ and $\overline{\mathsf{V}}$ algorithms which make up the conversation are identically distributed.

- PROV-SIM$(I, s, M_{\rm in})$: Just as for the output of the CONV oracle, the perfectness of the simulation can be seen from Construction 4.3 for $I = I_{\rm g}$, and from the identical distribution of user secret keys for $I \neq I_{\rm g}$.
- CORR-SIM$(I)$: Since we assume that $\overline{\mathsf{A}}$ does not corrupt $I_{\rm g}$, we only need to consider the case $I \neq I_{\rm g}$. The CORR-SIM oracle returns USK$[I]$ as the user secret key of identity $I$, which is correctly distributed for the same reasons as explained for the CONV-SIM and PROV-SIM oracles.

So conditioned on the event **Good**, the simulation of $\overline{\mathsf{A}}$'s environment is perfect. This means that $\mathsf{A}$'s impersonation is successful whenever (1) $\overline{\mathsf{A}}$ succeeds (i.e. $\mathbf{Exp}^{\text{imp-atk}}_{\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1$), (2) $\mathsf{A}$ correctly guesses the identity that $\overline{\mathsf{A}}$ attacks (i.e. $|HU| \geq q_{\rm g} \wedge I_{\rm b} = I_{\rm g}$), and (3) $\overline{\mathsf{A}}$ doesn't corrupt identity $I_{\rm g}$ (i.e. event **Good** occurs). The advantage of $\mathsf{A}$ can therefore be bounded from below by

$$\begin{aligned}
\mathbf{Adv}^{\text{imp-atk}}_{\mathcal{SI},\mathsf{A}}(k) &\geq \Pr\left[\mathbf{Exp}^{\text{imp-atk}}_{\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1 \wedge I_{\rm b} = I_{\rm g} \wedge \mathbf{Good}\right] \\
&= \Pr\left[\mathbf{Exp}^{\text{imp-atk}}_{\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1 \wedge I_{\rm b} = I_{\rm g}\right] \\
&= \Pr\left[\mathbf{Exp}^{\text{imp-atk}}_{\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1\right] \cdot \Pr\left[\, I_{\rm b} = I_{\rm g}\,\right] \\
&\geq \frac{1}{\boldsymbol{Q}^{\rm H}_{\overline{\mathsf{CV}}}(k) + \boldsymbol{Q}^{\rm INIT}_{\overline{\mathsf{CV}}}(k)} \cdot \mathbf{Adv}^{\text{imp-atk}}_{\mathcal{IBI},\overline{\mathsf{A}}}(k)\ .
\end{aligned}$$

The first equality above needs some clarification. In order to be successful, the identity $I_{\rm b}$ attacked by $\overline{\mathsf{A}}$ cannot have been previously corrupted by $\overline{\mathsf{A}}$. So if $\overline{\mathsf{A}}$ is successful and $I_{\rm b} = I_{\rm g}$, this means that $I_{\rm g}$ cannot have been corrupted by $\overline{\mathsf{A}}$, which is exactly the definition of the event **Good**. Therefore, **Good** is implied by the other two conditions and can be removed from the expression without affecting the probability. The second equality holds because $\mathsf{A}$'s simulation of $\overline{\mathsf{A}}$'s environment is perfect, and hence $\overline{\mathsf{A}}$'s view is independent of $\mathsf{A}$'s choice of $I_{\rm g}$. ▌

## 4.3 The cSS-2-IBS Transform

Convertibility of a standard signature (SS) scheme $\mathcal{SS} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$ is defined by analogy to Definition 4.2 as shown below.

**Definition 4.5 (Convertible SS Schemes)** A SS scheme $\mathcal{SS} = (\mathsf{SKg}, \mathsf{Sign}, \mathsf{Vf})$ is said to be *convertible* if there exists a family of trapdoor samplable relations $\mathcal{F} = (\mathsf{TDG}, \mathsf{Smp}, \mathsf{Inv})$ such that for all $k \in \mathbb{N}$ the output of the following is distributed identically to the output of $\mathsf{SKg}(1^k)$:

$$(\langle \mathbf{R} \rangle, t) \xleftarrow{\$} \mathsf{TDG}(1^k)\ ;\ (x, y) \xleftarrow{\$} \mathsf{Smp}(\langle \mathbf{R} \rangle)\ ;\ pk \leftarrow (\langle \mathbf{R} \rangle, y)\ ;\ sk \leftarrow (\langle \mathbf{R} \rangle, x)\ ;\ \text{Return } (pk, sk) \ ▌$$

The cSS-2-IBS transform is defined analogously to the cSI-2-IBI transform:

**Construction 4.6 (The cSS-2-IBS Transform)** To any convertible SS (cSS) scheme $\mathcal{SS} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$, the cSS-2-IBS transform assocciates an IBS scheme $\mathcal{IBS} = \text{cSS-2-IBS}(\mathcal{SS}) = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{Sign}}, \overline{\mathsf{Vf}})$ where the master and the user key generators are exactly as in Construction 4.3, and $\overline{\mathsf{Sign}}(usk, \cdot)$ and where $\overline{\mathsf{Vf}}(mpk, I, \cdot, \cdot : \mathrm{H})$ are identical to $\mathsf{Sign}(usk, \cdot)$ and $\mathsf{Vf}((mpk, \mathrm{H}(I)), \cdot, \cdot)$, respectively. ▌

The proof of the following analogue of Theorem 4.4 is similar to the proof of Theorem 4.4 and is thus omitted.

**Theorem 4.7 (Security of cSS-2-IBS)** Let $\mathcal{SS}$ be a cSS scheme and let $\mathcal{IBS}$ = cSS-2-IBS($\mathcal{SS}$) be the associated IBS scheme as defined in Construction 4.6. If $\mathcal{SS}$ is uf-cma secure then $\mathcal{IBS}$ is also uf-cma secure. ▮

One can check that the class of trapdoor SS (tSS) schemes as defined in [DKXY03] contains all cSS schemes where $\mathcal{F}$ is a family of trapdoor permutations, and that their tSS-2-IBS transform coincides with cSS-2-IBS in case the starting cSS scheme is trapdoor. Thus, Theorem 4.7 represents a (slight) extension of their result. However, the extension is important, for we will see cases of cSS schemes that are not trapdoor and where the extension is needed.

### 4.4 The fs-I-2-S Transform

So-called *canonical* SI schemes can be transformed into signature schemes using the Fiat-Shamir transform [FS86], referred to as the fs-I-2-S transform here. A standard identification scheme $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$ is said to be *canonical* if it follows a three-move structure where the prover initiates the communication with a "commitment" *Cmt* distributed uniformly over a set CmtSet(*sk*) possibly depending on the secret key; the verifier sends back a "challenge" *Ch* chosen uniformly from a set ChSet(*pk*) that possibly depends on the public key; and the prover replies with a "response" *Rsp*. The verifier's decision to accept or reject is a deterministic function $\mathrm{Dec}(pk, \mathit{Cmt} \| \mathit{Ch} \| \mathit{Rsp}) \in \{0,1\}$ of the public key and the communication transcript. We say that $\mathcal{SI}$ has *commitment length* $\beta(\cdot)$ if $|\mathrm{CmtSet}(sk)| \geq 2^{\beta(k)}$ for every $k \in \mathbb{N}$ and every $(pk, sk) \in [\mathsf{Kg}(1^k)]$. We say that $\mathcal{SI}$ is *non-trivial* if the function $2^{-\beta(k)}$ is negligible in $k$.[1] All SI schemes considered in this paper are canonical.

**Construction 4.8 (The fs-I-2-S Transform [FS86])** Let $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$ be a non-trivial canonical SI scheme with challenge set function ChSet and decision function Dec. The *Fiat-Shamir transform* fs-I-2-S associates to it the SS scheme $\mathcal{SS}$ = fs-I-2-S($\mathcal{SI}$) = (Kg, Sign, Vf) whose signing and verification algorithms are defined as follows:

```
Algorithm $\mathsf{Sign}(sk, M : \mathrm{H})$
    $(\mathit{Cmt}, \mathit{St}_\mathsf{P}) \xleftarrow{\$} \mathsf{P}(\varepsilon, sk)$
    $\mathit{Ch} \leftarrow \mathrm{H}(\mathit{Cmt} \| M)$
    $(\mathit{Rsp}, \mathit{St}_\mathsf{P}) \xleftarrow{\$} \mathsf{P}(\mathit{Ch}, \mathit{St}_\mathsf{P})$
    Return $\mathit{Cmt} \| \mathit{Rsp}$
```

```
Algorithm $\mathsf{Vf}(pk, M, \sigma : \mathrm{H})$
    Parse $\sigma$ as $\mathit{Cmt} \| \mathit{Rsp}$
    $\mathit{Ch} \leftarrow \mathrm{H}(\mathit{Cmt} \| M)$
    Return $\mathrm{Dec}(pk, \mathit{Cmt} \| \mathit{Ch} \| \mathit{Rsp})$
```

Above, H: $\{0,1\}^* \to \mathrm{ChSet}(pk)$ is a random oracle. ▮

The following theorem is a special case of [AABN02, Lemma 3.5]. It relates the security of $\mathcal{SS}$ to that of the underlying identification scheme.

**Theorem 4.9** Let $\mathcal{SI}$ be a non-trivial canonical SI scheme, and let $\mathcal{SS}$ = fs-I-2-S($\mathcal{SI}$) be the associated signature scheme as per Construction 4.8. If $\mathcal{SI}$ is imp-pa secure, then $\mathcal{SS}$ is uf-cma secure in the random oracle model. ▮

It is also easy to see that the fs-I-2-S transform of a cSI scheme is a cSS scheme. Combining Theorems 4.9 and 4.7 yields the following, which will be our main tool to prove security of IBS schemes.

**Corollary 4.10** Let $\mathcal{SI}$ be a non-trivial canonical cSI scheme, and let $\mathcal{IBS}$ = cSS-2-IBS(fs-I-2-S($\mathcal{SI}$)). If $\mathcal{SI}$ is imp-pa secure then $\mathcal{IBS}$ is uf-cma secure. ▮

[1] The canonicity definition used here is more restrictive than the one used in [AABN02], which allows *Cmt* to be chosen according to *any* distribution over CmtSet(*sk*), instead of only the uniform one. This however complicates the non-triviality condition, requiring $\beta(\cdot)$ to be defined as the *min-entropy* of the distribution. Since all schemes treated in this work have uniformly distributed commitments, we restrict ourselves to the simpler definition here.

## 4.5 The efs-IBI-2-IBS Transform

The canonicity definition for SI schemes is easily extended to IBI schemes, the only modification being that the set from which challenges are drawn may depend on both the master public key *mpk* and the user's identity $I$, and that the verifier's decision is a deterministic function $\mathrm{Dec}((mpk, I), Cmt \| Ch \| Rsp)$ of the master public key, the user's identity and the communication transcript. It is easily seen that the cSI-2-IBI transform of a canonical cSI scheme is also canonical. One can apply the fs-I-2-S transform to a canonical IBI scheme to obtain an IBS scheme, and one can check that $\mathsf{cSS\text{-}2\text{-}IBS}($ $\mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathcal{SI})) = \mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathsf{cSI\text{-}2\text{-}IBI}(\mathcal{SI}))$ for any canonical cSI scheme $\mathcal{SI}$. It follows that fs-I-2-S yields a uf-cma secure IBS scheme if it is applied to a *converted* IBI scheme, meaning one that is obtained as the result of applying cSI-2-IBI to some (canonical) cSI scheme. However, one can also apply fs-I-2-S to a canonical IBI scheme that is not converted and get an IBS scheme, and there will be instances later where we would like to do this. Unfortunately, the IBS scheme so obtained need not be secure, in the sense that the analogue of the result of Theorem 4.9 does not hold, as stated below.

**Proposition 4.11** Assume there exists an imp-pa secure canonical IBI scheme. Then, there exists an imp-pa secure canonical IBI scheme $\mathcal{IBI}$ such that the IBS scheme given by $\mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathcal{IBI})$ is not uf-cma secure. ▌

**Proof:** Let $\mathrm{Dec}'$ be the decision function of the given IBI scheme. The new scheme $\mathcal{IBI}$ is identical to the given one, except that the decision function is given by

$$\mathrm{Dec}((mpk, I), Cmt \| Ch \| Rsp) \quad = \quad \begin{cases} 1 & \text{if } \mathrm{Dec}'((mpk, I), Cmt \| Ch \| Rsp) = 1 \text{ or } Ch = I \\ 0 & \text{otherwise.} \end{cases}$$

In other words, the decision function is relaxed so that the verifier also accepts when the challenge is equal to the identity being verified.

We first claim that the new IBI scheme $\mathcal{IBI}$ inherits the imp-pa security of the given IBI scheme. We provide the intuition, from which a formal proof by reduction is easy constructed. Namely, an imp-pa adversary attacking $\mathcal{IBI}$ has to commit to an identity $I_{\mathrm{b}}$ in the first phase, before seeing the challenge issued by the verifier in the second phase. But since the challenge is drawn from a set of super-polynomial size (this follows from the assumed imp-pa security of the original IBI scheme), the probability that it equals $I_{\mathrm{b}}$ is negligible. So the adversary is effectively left attacking the original scheme, but the latter is assumed secure.

Next, we note that the IBS scheme $\mathcal{IBS} = \mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathcal{IBI})$ is insecure. To forge a signature of a message $M$, pick any values $Cmt, Rsp$, then compute $I = \mathrm{H}(Cmt \| M)$. Then $(Cmt, Rsp)$ is a valid signature of $M$ under identity $I$. ▌

We now provide a remedy for the above. We consider the *extended Fiat-Shamir transform* efs-IBI-2-IBS, a modified version of the fs-I-2-S transform that hashes the identity of the signer (prover) along with the commitment and message, rather than merely hashing the commitment and message as in fs-I-2-S. We show (by an extension of the proof of [AABN02]) that, if this transform is applied to a canonical imp-pa secure IBI scheme, then the outcome is a uf-cma secure IBS scheme. We apply this in Section 6 to obtain uf-cma secure IBS schemes from the two unconverted IBI schemes we consider, namely $\mathcal{OKDL}\text{-}\mathcal{IBI}$ and $\mathcal{BNN}\text{-}\mathcal{IBI}$.

**Construction 4.12 (The efs-IBI-2-IBS Transform)** Let $\mathcal{IBI} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{P}}, \overline{\mathsf{V}})$ be a non-trivial canonical IBI scheme with challenge set function ChSet and decision function Dec. The *extended Fiat-Shamir transform* efs-IBI-2-IBS associates to it the IBS scheme $\mathcal{IBS} = \mathsf{efs\text{-}IBI\text{-}2\text{-}IBS}(\mathcal{IBI}) =$

$(\mathsf{MKg}, \mathsf{UKg}', \overline{\mathsf{Sign}}, \overline{\mathsf{Vf}})$ as follows. Let H: $\{0,1\}^* \to \mathrm{ChSet}(pk)$ be a random oracle. The secret key *usk* of a signer with identity $I$ is the pair $(\mathsf{UKg}(msk, I), I)$. To sign a message $M$, the $\overline{\mathsf{Sign}}$ algorithm parses the secret key as $(usk, I)$, computes

$$(Cmt, St_{\overline{\mathsf{P}}}) \xleftarrow{\$} \overline{\mathsf{P}}(\varepsilon, usk)\ ;\ Ch \leftarrow \mathrm{H}(I \| Cmt \| M)\ ;\ (Rsp, St_{\overline{\mathsf{P}}}) \xleftarrow{\$} \overline{\mathsf{P}}(Ch, St_{\overline{\mathsf{P}}})$$

and outputs $Cmt \| Rsp$ as the signature. The verification algorithm $\overline{\mathsf{Vf}}$ parses the signature as $Cmt \| Rsp$, computes $Ch \leftarrow \mathrm{H}(I \| Cmt \| M)$, and returns $\mathrm{Dec}((mpk, I), Cmt \| Ch \| Rsp)$. ▌

**Theorem 4.13** Let $\mathcal{IBI}$ be a non-trivial canonical IBI scheme, and let $\mathcal{IBS} = \textsf{efs-IBI-2-IBS}(\mathcal{IBI})$ be the associated IBS scheme as per Construction 4.12. If $\mathcal{IBI}$ is imp-pa secure, then $\mathcal{IBS}$ is uf-cma secure in the random oracle model. ▌

**Proof Sketch:** The proof of Theorem 4.13 follows the approach of [AABN02]. Given a polynomial-time adversary $\overline{\mathsf{F}}$ attacking $\mathcal{IBS}$ using at most $Q_{\overline{\mathsf{F}}}^{\mathrm{H}}(\cdot)$ random oracle queries and $Q_{\overline{\mathsf{F}}}^{\textsc{Sign}}(\cdot)$ signature queries, we construct a polynomial-time adversary $\overline{\mathsf{A}} = (\overline{\mathsf{CV}}, \overline{\mathsf{CP}})$ attacking $\mathcal{IBI}$ such that for every $k \in \mathbb{N}$

$$\mathbf{Adv}_{\mathcal{IBS},\overline{\mathsf{F}}}^{\text{uf-cma}}(k) \ \leq\ [1 + Q_{\overline{\mathsf{F}}}^{\mathrm{H}}(k)] \cdot \mathbf{Adv}_{\mathcal{IBI},\overline{\mathsf{A}}}^{\text{imp-pa}}(k) + \frac{[1 + Q_{\overline{\mathsf{F}}}^{\mathrm{H}}(k) + Q_{\overline{\mathsf{F}}}^{\textsc{Sign}}(k)] \cdot Q_{\overline{\mathsf{F}}}^{\textsc{Sign}}(k)}{2^{\beta(k)}}\ , \qquad (1)$$

where $\beta(\cdot)$ is the commitment length of $\mathcal{IBI}$.

On input a security parameter $1^k$ and master public key *mpk*, the cheating verifier $\overline{\mathsf{CV}}$ first guesses the index $q_{\mathrm{g}}$ of the random oracle query that will be involved in $\overline{\mathsf{F}}$'s forgery. Algorithm $\overline{\mathsf{CV}}$ then runs $\overline{\mathsf{F}}$ on input $1^k, mpk$, answering $\overline{\mathsf{F}}$'s Init oracle queries using its own Init oracle, and answering $\overline{\mathsf{F}}$'s Corr oracle using its own Corr oracle. If $\overline{\mathsf{F}}$ queries the Sign oracle for a signature of message $M$ under identity $I$ then $\overline{\mathsf{CV}}$ queries its Conv oracle to retrieve a valid conversation $Cmt \| Ch \| Rsp$ for identity $I$. It returns $Cmt \| Rsp$ to $\overline{\mathsf{F}}$ as the signature, and saves $Ch$ as the random oracle value corresponding to $I \| Cmt \| M$. (If a value has already been assigned to $\mathrm{H}(I \| Cmt \| M)$ during previous random oracle or signature queries, then $\overline{\mathsf{CV}}$ gives up. This will only happen with negligible probability though, since *Cmt* is uniformly distributed over a set of size $\geq 2^{\beta(k)}$ and $\mathcal{IBI}$ is non-trivial.) If $\overline{\mathsf{F}}$ queries H on a string $x$ that has no value assigned to it yet, and if this is not the $q_{\mathrm{g}}$-th query to the random oracle, then $\overline{\mathsf{CV}}$ simply picks a random element from $\mathrm{ChSet}(mpk, I)$, assigns this as the value of $\mathrm{H}(x)$, and also returns it to $\overline{\mathsf{F}}$ as the oracle response.

At the $q_{\mathrm{g}}$-th query $I_{\mathrm{g}} \| Cmt_{\mathrm{g}} \| M_{\mathrm{g}}$ made by $\overline{\mathsf{F}}$ to H, however, $\overline{\mathsf{CV}}$ indicates that it wants to impersonate identity $I_{\mathrm{g}}$ in the second phase of the game by outputting $I_{\mathrm{g}}$, and all the state information it gathered so far, for use by its accomplice $\overline{\mathsf{CP}}$. (Actually, it might have to initialize identity $I_{\mathrm{g}}$ first if $\overline{\mathsf{F}}$ didn't do so before. Also, $\overline{\mathsf{CV}}$ has to query a batch of $Q_{\overline{\mathsf{F}}}^{\textsc{Sign}}(k)$ conversations for identity $I_{\mathrm{b}}$ before it halts and pass these to $\overline{\mathsf{CP}}$ as well, as $\mathsf{CP}$ is not given Conv.) The cheating prover $\overline{\mathsf{CP}}$ immediately sends $Cmt_{\mathrm{g}}$ as the first message of its impersonation attempt, receives the challenge $Ch$ from the honest verifier, and returns $Ch$ to $\overline{\mathsf{F}}$ as the response to its random oracle query. $\overline{\mathsf{CP}}$ continues the execution of $\overline{\mathsf{F}}$, answering its oracle queries in the same way as $\overline{\mathsf{CV}}$ did before, with the following exceptions. For Sign queries, not having access to Conv, it uses conversations from the batch to generate signatures for $I_{\mathrm{g}}$. Also if $\overline{\mathsf{F}}$ corrupts $I_{\mathrm{g}}$ then $\overline{\mathsf{CP}}$ gives up. At the end of its execution, $\overline{\mathsf{F}}$ outputs its forgery $(I_{\mathrm{b}}, M, Cmt \| Rsp)$. If $I_{\mathrm{b}} \| M \| Cmt \neq I_{\mathrm{g}} \| M_{\mathrm{g}} \| Cmt_{\mathrm{g}}$, then $\overline{\mathsf{CP}}$ gives up, but otherwise it sends *Rsp* as its response to the honest verifier. It is clear from Construction 4.12 that this is a valid response if $\overline{\mathsf{F}}$'s forgery is valid, and hence $\overline{\mathsf{A}}$ succeeds in impersonating identity $I_{\mathrm{b}}$.

The analysis establishing Equation (1) closely resembles the analysis in the proof of Theorem 4.9 as given in [AABN02], and we refer to the latter for more details. ▌

# 5 Applying the Framework

We now apply the above transform-based framework to prove security of numerous existing and new IBI and IBS schemes. To do this, we consider numerous (existing and new) SI schemes. We show that they are convertible, and then analyze their security if this has not already been done. The implications for the corresponding IBI and IBS schemes, obtained via the transforms discussed above, follow from Theorem 4.4 and Corollary 4.10.

When proving the imp-atk security of identification schemes ($\mathrm{atk} \in \{\mathrm{pa}, \mathrm{aa}, \mathrm{ca}\}$), we will use Bellare and Palacio's [BP02] Reset Lemma that upper bounds the success probability of a cheating prover $\mathsf{CP}$ in any canonical identification scheme as a function of the probability of obtaining two accepting conversations in a resetting experiment. By using the abstract notation $St_{\mathsf{V}}$ for the verifier's initial state, the Reset Lemma is applicable to both SI and IBI schemes: for SI schemes, $St_{\mathsf{V}}$ is simply the public key $pk$, while for IBI schemes, it is a tuple $(mpk, I)$ containing the master public key and the identity.

We say that a canonical SI scheme $\mathcal{SI}$, with challenge set function ChSet and decision function Dec, has *challenge length* $\ell(\cdot)$ if $|\mathrm{ChSet}(pk)| \geq 2^{\ell(k)}$ for all $k \in \mathbb{N}$ and all $(pk, sk) \in [\mathsf{Kg}(1^k)]$. Analogously, we say that a canonical IBI scheme $\mathcal{IBI}$, with challenge set function ChSet and decision function Dec, has challenge length $\ell(\cdot)$ if $|\mathrm{ChSet}((mpk, I))| \geq 2^{\ell(k)}$ for all $k \in \mathbb{N}$, all $(mpk, msk) \in [\mathsf{MKg}(1^k)]$ and all $I \in \{0,1\}^*$.

**Lemma 5.1 (Reset Lemma [BP02])** Let $\mathsf{CP}$ be a prover in a canonical SI or IBI scheme with challenge set ChSet, challenge length $\ell$ and decision function Dec. Let $St_{\mathsf{V}}$ and $St_{\mathsf{CP}}$ be initial states for the verifier and $\mathsf{CP}$, respectively. Let $\mathrm{acc}(St_{\mathsf{CP}}, St_{\mathsf{V}})$ be the probability that the verifier accepts on initial state $St_{\mathsf{V}}$ after interacting with $\mathsf{CP}$ initiated with $St_{\mathsf{CP}}$, and let $\mathrm{res}(St_{\mathsf{CP}}, St_{\mathsf{V}})$ be the probability that the following *reset experiment* returns 1:

```
Choose random tape $\rho$ for $\mathsf{CP}$ ; $(Cmt, St_{\mathsf{CP}}) \leftarrow \mathsf{CP}(\varepsilon, St_{\mathsf{CP}}, \rho)$
$Ch_1 \stackrel{\$}{\leftarrow} \mathrm{ChSet}(St_{\mathsf{V}})$ ; $(Rsp_1, St'_{\mathsf{CP}}) \leftarrow \mathsf{CP}(Ch_1, St_{\mathsf{CP}})$ ; $d_1 \leftarrow \mathrm{Dec}(St_{\mathsf{V}}, Cmt \| Ch_1 \| Rsp_1)$
$Ch_2 \stackrel{\$}{\leftarrow} \mathrm{ChSet}(St_{\mathsf{V}})$ ; $(Rsp_2, St'_{\mathsf{CP}}) \leftarrow \mathsf{CP}(Ch_2, St_{\mathsf{CP}})$ ; $d_2 \leftarrow \mathrm{Dec}(St_{\mathsf{V}}, Cmt \| Ch_2 \| Rsp_2)$
If ($d_1 = 1$ and $d_2 = 1$ and $Ch_1 \neq Ch_2$) then return 1 else return 0
```

Then,

$$\mathrm{acc}(St_{\mathsf{CP}}, St_{\mathsf{V}}) \leq 2^{-\ell(k)} + \sqrt{\mathrm{res}(St_{\mathsf{CP}}, St_{\mathsf{V}})} \quad \blacksquare$$

When presenting schemes, we always explicitly include membership tests on all messages sent by the prover to prevent the type of attacks described by Burmester and Desmedt [BD89] sending e.g. zero as the commitment.

## 5.1 Schemes based on Factoring

Definitions and assumptions. The key generation algorithms of all factoring-based schemes are underlain by a *modulus generator* $\mathsf{K}_{\mathrm{fact}}$. This is a polynomial-time algorithm that on input $1^k$ outputs a *modulus* $N$ and two distinct, odd primes $p, q$ such that $N = pq$ and $2^{k-1} \leq N < 2^k$. We assume that the factoring problem associated to $\mathsf{K}_{\mathrm{fact}}$ is hard. This means that the function

$$\mathbf{Adv}^{\mathrm{fact}}_{\mathsf{A},\mathsf{K}_{\mathrm{fact}}}(k) \;=\; \Pr\Big[\, \mathsf{A}(1^k, N) \in \{p, q\} \;\;:\;\; (N, p, q) \stackrel{\$}{\leftarrow} \mathsf{K}_{\mathrm{fact}}(1^k) \,\Big]$$

is negligible for any polynomial-time algorithm $\mathsf{A}$. A *Blum-Williams generator* is a modulus generator that returns Blum-Williams (BW) moduli $N$ [Wil80, Blu82], meaning that $N = pq$ with $p \equiv q \equiv 3 \bmod 4$.

```
Algorithm $\mathsf{Kg}(1^k)$
  $(N,p,q) \xleftarrow{\$} \mathsf{K}_{\text{fact}}(1^k)$
  For $i = 1 \ldots t(k)$ do
    $x_i \xleftarrow{\$} \mathbb{Z}_N^*$
    $X_i \xleftarrow{\$} \pm {x_i}^{-2^{m(k)}} \bmod N$
  $pk \leftarrow ((1^k,N),(X_1,\ldots,X_{t(k)}))$
  $sk \leftarrow ((1^k,N),(x_1,\ldots,x_{t(k)}))$
  Return $(pk, sk)$
```

| <u>Prover</u> $\mathsf{P}$ | | <u>Verifier</u> $\mathsf{V}$ |
|---|---|---|
| $y \xleftarrow{\$} \mathbb{Z}_N^*$ | | |
| $Y \leftarrow y^{2^{m(k)}} \bmod N$ | $\xrightarrow{Y}$ | |
| | $\xleftarrow{c}$ | $c = (c_1,\ldots,c_{t(k)}) \xleftarrow{\$} \mathbb{Z}^{t(k)}_{2^{m(k)}}$ |
| $z \leftarrow y \prod_{i=1}^{t(k)} x_i^{c_i} \bmod N$ | $\xrightarrow{z}$ | |
| | | If $Y \equiv \pm z^{2^{m(k)}} \prod_{i=1}^{t(k)} X_i^{c_i} \bmod N$<br>and $Y, z \in \mathbb{Z}_N^*$<br>then $\mathtt{acc}$ else $\mathtt{rej}$ |

Figure 7: **The *ItR-SI* and *FFS-SI* schemes.** The scheme is parameterized with a Blum-Williams modulus generator $\mathsf{K}_{\text{fact}}$, key multiplicity $t : \mathbb{N} \to \mathbb{N}$ and iteration depth $m : \mathbb{N} \to \mathbb{N}$ such that $t(k) \cdot m(k)$ is super-logarithmic in $k$. The prover $\mathsf{P}$ and verifier $\mathsf{V}$ are run on initial states $sk = ((1^k, N), (x_1, \ldots, x_t))$ and $pk = ((1^k, N), (X_1, \ldots, X_t))$, respectively. The *FFS-SI* scheme is the special case where $m(k) = 1$.

Let $\mathrm{QR}_N = \{x^2 \bmod N \mid x \in \mathbb{Z}_N^*\}$ be the set of all quadratic residues modulo $N$, let $\mathrm{Jac}_N(x)$ be the Jacobi symbol of $x$ with respect to $N$, and let $\mathbb{Z}_N^*[+1] = \{x \in \mathbb{Z}_N^* \mid \mathrm{Jac}_N(x) = +1\}$ be the set of elements of $\mathbb{Z}_N^*$ with Jacobi symbol $+1$. It is known that if $N$ is a BW modulus, then squaring is a permutation on $\mathrm{QR}_N$, and $-1 \in \mathbb{Z}_N^*[+1] \setminus \mathrm{QR}_N$, so that, for any $x \in \mathbb{Z}_N^*[+1]$, we have either $x \in \mathrm{QR}_N$ or $-x \in \mathrm{QR}_N$.

Scheme modifications. Some of the schemes presented hereafter are slightly altered versions of the corresponding schemes proposed in the literature. The changes are necessary because of an issue regarding instantiation of the random oracle related to the $\mathsf{cSI\text{-}2\text{-}IBI}$ and $\mathsf{cSS\text{-}2\text{-}IBS}$ transforms. Using the original schemes, the random oracle would have to map arbitrary strings to random elements of $\mathrm{QR}_N$. While theoretically one can assume the availability of such an oracle, it is not clear how it can be implemented in practice without revealing a square root of the hash value during the computation (because deciding whether an element $x \in \mathbb{Z}_N^*$ is a quadratic residue modulo $N$ is assumed to be hard when the factorization of $N$ is unknown). The depicted *FFS*, *ItR* and *FF* schemes overcome this problem by having the random oracle map to random elements of $\mathbb{Z}_N^*[+1]$, membership of which can be efficiently tested without knowledge of the factorization of $N$.

### 5.1.1 The *FFS* and *ItR* Families

The scheme. The *iterated-root* scheme *ItR-SI* depicted in Figure 7 is parameterized with a Blum-Williams generator $\mathsf{K}_{\text{fact}}$, key multiplicity $t : \mathbb{N} \to \mathbb{N}$ and iteration depth $m : \mathbb{N} \to \mathbb{N}$ such that $t(k) \cdot m(k)$ is a super-logarithmic function in $k$, i.e. $t(k) \cdot m(k) = \omega(\log k)$. The Feige-Fiat-Shamir scheme *FFS-IBI* is the special case of *ItR-SI* for $m(k) = 1$. We use the shorthand notation $a \xleftarrow{\$} \pm b$ for $a \xleftarrow{\$} \{+b, -b\}$.

The *FFS-SI* scheme coincides perfectly with the scheme by Feige et al. [FFS88]. The identification scheme by Fiat and Shamir [FS86] was originally presented as an IBI scheme that almost coincides with the *FFS-IBI* = $\mathsf{cSI\text{-}2\text{-}IBI}$(*FFS-SI*) scheme, the difference being that the latter is not restricted to BW moduli and doesn't have the ± signs. The *ItR-SI* differs from the Ong-Schnorr scheme [OS90] in exactly the same way. Another variant by Ohta and Okamoto [OO90] is based on the difficulty of taking $L$-th roots with $\gcd(L, \varphi(n)) > 1$.

Convertibility. Since the *FFS-SI* scheme is a special case of the *ItR-SI* scheme, it suffices to show that the latter is convertible. To any Blum-Williams modulus generator $\mathsf{K}_{\text{fact}}$ and to any function $m : \mathbb{N} \to \mathbb{N}$, we associate a family of trapdoor samplable relations $\mathcal{F} = (\mathsf{TDG}, \mathsf{Smp}, \mathsf{Inv})$ described as follows:

```
Algorithm $\mathsf{TDG}(1^k)$:
    $(N,p,q) \xleftarrow{\$} \mathsf{K}_{\mathrm{fact}}(1^k)$
    Return $((1^k,N),(p,q))$
```

```
Algorithm $\mathsf{Smp}((1^k,N))$:
    For $i = 1 \ldots t(k)$ do $x_i \xleftarrow{\$} \mathbb{Z}_N^*$ ; $X_i \xleftarrow{\$} \pm x_i^{-2^{m(k)}} \bmod N$
    Return $((x_1,\ldots,x_{t(k)}),(X_1,\ldots,X_{t(k)}))$
```

```
Algorithm $\mathsf{Inv}((1^k,N),(p,q),(X_1,\ldots,X_{t(k)}))$:
    For $i = 1 \ldots t(k)$ do $x_i \xleftarrow{\$} (\pm X_i)^{-1/2^{m(k)}} \bmod N$
    Return $(x_1,\ldots,x_{t(k)})$
```

where the notation $x \xleftarrow{\$} (\pm X)^{1/2^{m(k)}} \bmod N$ is used to indicate that an element $x$ is chosen uniformly over all $2^{m(k)}$-th roots of $X$ or $-X$ (whichever is a square) modulo $N$, which can be computed using the factors $p, q$ of $N$. The relation described by $(1^k, N)$ is

$$\begin{aligned} \mathbf{R} \;=\; & \{\, ((x_1,\ldots,x_{t(k)}),(X_1,\ldots,X_{t(k)})) \in (\mathbb{Z}_N^*)^{t(k)} \times (\mathbb{Z}_N^*[+1])^{t(k)} \;: \\ & x_i^{-2^{m(k)}} \equiv \pm X_i \bmod N \text{ for all } i = 1 \ldots t(k) \} \;. \end{aligned}$$

Obviously, for all $x \in \mathbb{Z}_N^*$, there exist exactly two values of $X$ such that $x^{-2^{m(k)}} \equiv \pm X \bmod N$, namely $x^{-2^{m(k)}}$ and $-x^{-2^{m(k)}} \bmod N$. Since squaring is a permutation over $\mathrm{QR}_N$, all $X \in \mathrm{QR}_N$ have exactly four $2^{m(k)}$-th roots in $\mathbb{Z}_N^*$. Likewise, for all $X \in \mathbb{Z}_N^*[+1] \setminus \mathrm{QR}_N$ there exist exactly four $x \in \mathbb{Z}_N^*$ such that $x^{-2^{m(k)}} \equiv -X \bmod N$, and hence the relation is regular. The output of the $\mathsf{Smp}$ algorithm is uniformly distributed over $\mathbf{R}$ due to the random choice of $x_i$ and the sign of $X_i$. The output of the $\mathsf{Inv}$ algorithm is uniformly distributed over $\mathbf{R}^{-1}((X_1,\ldots,X_{t(k)}))$ since for each $X_i$ all $-2^{m(k)}$-th roots are computed and $x_i$ is chosen at random from these.

Security. We note that $\mathcal{FFS}\text{-}\mathcal{SI}$ is exactly the scheme in [FFS88], which is known to be imp-pa and imp-aa secure for super-logarithmic $t(k)$ assuming that the factoring problem related to $\mathsf{K}_{\mathrm{fact}}$ is hard, and this easily extends to imp-ca. By Theorem 4.4, the imp-pa, imp-aa and imp-ca security of the $\mathcal{FFS}\text{-}\mathcal{IBI} = \mathsf{cSI\text{-}2\text{-}IBI}(\mathcal{FFS}\text{-}\mathcal{SI})$ scheme follows. Theorem 4.9 and Corollary 4.10 imply that the $\mathcal{FFS}\text{-}\mathcal{SS} = \mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathcal{FFS}\text{-}\mathcal{SI})$ and $\mathcal{FFS}\text{-}\mathcal{IBS} = \mathsf{cSS\text{-}2\text{-}IBS}(\mathcal{FFS}\text{-}\mathcal{SS})$ are uf-cma secure, but this was known [PS00, DKXY03].

The $\mathit{It}\mathcal{R}\text{-}\mathcal{SI}$ scheme depicted in Figure 7 is a close variant of the Ong-Schnorr SI scheme [OS90], the only difference being the absence of ± signs in the latter. The Ong-Schnorr scheme is known to be imp-pa and imp-aa secure for super-logarithmic $t(k) \cdot m(k)$ if the factoring problem related to $\mathsf{K}_{\mathrm{fact}}$ is hard [Sho99, Sch96]. These results extend to the $\mathit{It}\mathcal{R}\text{-}\mathcal{SI}$ scheme. Theorem 4.4 implies that $\mathit{It}\mathcal{R}\text{-}\mathcal{IBI} = \mathsf{cSI\text{-}2\text{-}IBI}(\mathit{It}\mathcal{R}\text{-}\mathcal{SI})$ is imp-pa and imp-aa secure assuming factoring moduli generated by $\mathsf{K}_{\mathrm{fact}}$ is hard. Theorem 4.7 and Corollary 4.10 imply that $\mathit{It}\mathcal{R}\text{-}\mathcal{SS} = \mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathit{It}\mathcal{R}\text{-}\mathcal{SI})$ and $\mathit{It}\mathcal{R}\text{-}\mathcal{IBS} = \mathsf{cSS\text{-}2\text{-}IBS}(\mathit{It}\mathcal{R}\text{-}\mathcal{SS})$ are uf-cma secure assuming factoring is hard, but this was known [PS00, DKXY03]. Whether $\mathit{It}\mathcal{R}\text{-}\mathcal{SI}$ and $\mathit{It}\mathcal{R}\text{-}\mathcal{IBI}$ are imp-ca secure remains open.

### 5.1.2 The $\mathcal{FF}$ Family

The scheme. To any Blum-Williams modulus generator $\mathsf{K}_{\mathrm{fact}}$ and any super-logarithmic iteration depth $m : \mathbb{N} \to \mathbb{N}$, we associate the $\mathcal{FF}\text{-}\mathcal{SI}$ scheme as depicted in Figure 8. The scheme is closely related to a SI scheme introduced by Fischlin and Fischlin [FF02] as a fix to an attack they found on a scheme in [Oka93]. They did not introduce IBI or IBS schemes. Due to the restriction of the $\mathcal{FF}\text{-}\mathcal{SI}$ scheme to BW moduli, it is actually a simplified variant of the scheme of [FF02], which also does not include the ± signs in the key generation and verification algorithms.

```
Algorithm $\mathsf{Kg}(1^k)$
    $(N, p, q) \xleftarrow{\$} \mathsf{K}_{\mathrm{fact}}(1^k)$
    $g \xleftarrow{\$} \mathrm{QR}_N$
    $x_1 \xleftarrow{\$} \mathbb{Z}_{2^{m(k)}}$ ; $x_2 \xleftarrow{\$} \mathbb{Z}_N^*$
    $X \xleftarrow{\$} \pm g^{x_1} x_2^{2^{m(k)}} \bmod N$
    $pk \leftarrow ((1^k, N, g), X)$
    $sk \leftarrow ((1^k, N, g), (x_1, x_2))$
    Return $(pk, sk)$
```

```
Prover P                                                                      Verifier V
$y_1 \xleftarrow{\$} \mathbb{Z}_{2^{m(k)}}$ ; $y_2 \xleftarrow{\$} \mathbb{Z}_N^*$
$Y \leftarrow g^{y_1} y_2^{2^{m(k)}} \bmod N$                  $\xrightarrow{Y}$
                                                 $\xleftarrow{c}$             $c \xleftarrow{\$} \mathbb{Z}_{2^{m(k)}}$
$z_1 \leftarrow y_1 + cx_1 \bmod 2^{m(k)}$
$z_2 \leftarrow g^{\lfloor (y_1+cx_1)/2^{m(k)} \rfloor} y_2 x_2^c \bmod N$   $\xrightarrow{z_1, z_2}$
                                                                              If $g^{z_1} z_2^{2^{m(k)}} \equiv \pm Y X^c \bmod N$
                                                                              and $Y, z_2 \in \mathbb{Z}_N^*$ and $z_1 \in \mathbb{Z}_{2^{m(k)}}$
                                                                              then acc else rej
```

Figure 8: **The $\mathcal{FF}$-$\mathcal{SI}$ scheme.** The scheme is parameterized with Blum-Williams modulus generator $\mathsf{K}_{\mathrm{fact}}$ and super-logarithmic iteration depth $m : \mathbb{N} \to \mathbb{N}$. The prover $\mathsf{P}$ and verifier $\mathsf{V}$ are initialized with states $sk = ((1^k, N, g), (x_1, x_2))$ and $pk = ((1^k, N, g), X)$, respectively.

Convertibility. Consider the following family of trapdoor samplable relations $\mathcal{F} = (\mathsf{TDG}, \mathsf{Smp}, \mathsf{Inv})$ associated to Blum-Williams generator $\mathsf{K}_{\mathrm{fact}}$ and iteration depth $m(k)$:

```
Algorithm $\mathsf{TDG}(1^k)$:
    $(N, p, q) \xleftarrow{\$} \mathsf{K}_{\mathrm{fact}}(1^k)$
    $g \xleftarrow{\$} \mathrm{QR}_N$
    Return $((1^k, N, g), (p, q))$
```

```
Algorithm $\mathsf{Smp}((1^k, N, g))$:
    $x_1 \xleftarrow{\$} \mathbb{Z}_{2^{m(k)}}$ ; $x_2 \xleftarrow{\$} \mathbb{Z}_N^*$
    $X \xleftarrow{\$} \pm g^{x_1} x_2^{2^{m(k)}} \bmod N$
    Return $((x_1, x_2), X)$
```

```
Algorithm $\mathsf{Inv}((1^k, N, g), (p, q), X)$:
    $x_1 \xleftarrow{\$} \mathbb{Z}_{2^{m(k)}}$
    $x_2 \xleftarrow{\$} (\pm X g^{-x_1})^{1/2^{m(k)}} \bmod pq$
    Return $(x_1, x_2)$.
```

A tuple $(1^k, N, g)$ describes the relation

$$\mathbf{R} = \{\, ((x_1, x_2), X) \in (\mathbb{Z}_{2^{m(k)}} \times \mathbb{Z}_N^*) \times \mathbb{Z}_N^*[+1] \;:\; g^{x_1} x_2^{2^{m(k)}} \equiv \pm X \bmod N \,\} \;.$$

For each pair $(x_1, x_2) \in \mathbb{Z}_{2^{m(k)}} \times \mathbb{Z}_N^*$, there exist exactly two values $X \in \mathbb{Z}_N^*[+1]$ such that $((x_1, x_2), X) \in \mathbf{R}$. Since squaring is a permutation over $\mathrm{QR}_N$ and $\mathrm{Jac}_N(-1) = +1$, there exist exactly four values $x_2 \in \mathbb{Z}_N^*$ for each $X \in \mathbb{Z}_N^*[+1]$ and $x_1 \in \mathbb{Z}_{2^{m(k)}}$ such that $((x_1, x_2), X)$. So $\mathbf{R}$ is regular with $|\mathbf{R}^{-1}(X)| = 4 \cdot 2^{m(k)}$ for all $X \in \mathbb{Z}_N^*[+1]$. The output of the $\mathsf{Smp}$ algorithm above is uniformly distributed over $\mathbf{R}$ by the random choice of $x_1, x_2$ and the sign of $X$. The uniform distribution of the output of the $\mathsf{Inv}$ algorithm follows from the random choice of $x_1$ and of the $2^{m(k)}$-th root.

Security. The $\mathcal{FF}$-$\mathcal{SI}$ scheme is a slight variation on (a special case of) the scheme of [FF02], the only difference being the absence of ± signs in the latter. The latter is proven to be imp-pa, imp-aa and imp-ca secure for super-logarithmic exponent $m(\cdot)$ assuming that the factoring problem associated to $\mathsf{K}_{\mathrm{fact}}$ is hard [FF02]. This result easily extends to the $\mathcal{FF}$-$\mathcal{SI}$ scheme. Likewise, the $\mathcal{FF}$-$\mathcal{SS}$ = $\mathsf{sfs\text{-}I\text{-}2\text{-}S}(\mathcal{FF}$-$\mathcal{SI})$ scheme is closely related to the SS scheme presented in [FF02], and it inherits its uf-cma security. The imp-pa, imp-aa and imp-ca security of the new $\mathcal{FF}$-$\mathcal{IBI}$ = $\mathsf{cSI\text{-}2\text{-}IBI}(\mathcal{FF}$-$\mathcal{SI})$ scheme and the uf-cma security of the new $\mathcal{FF}$-$\mathcal{IBS}$ = $\mathsf{cSS\text{-}2\text{-}IBS}(\mathcal{FF}$-$\mathcal{SS})$ scheme under the factoring assumption related to $\mathsf{K}_{\mathrm{fact}}$ follow from Theorem 4.4 and Corollary 4.10, respectively.

## 5.2 Schemes based on RSA

Definitions and assumptions. Similarly to the modulus generators used in factoring-based schemes, we describe all schemes based on RSA in terms of an *RSA key generator* $\mathsf{K}_{\mathrm{rsa}}$ that on input $1^k$ outputs a modulus $N$ that is the product of two distinct odd primes, and exponents $e, d$ such that $ed \equiv 1 \bmod \varphi(N)$ where $\varphi(N) = (p-1)(q-1)$ is Euler's totient function. A *prime-exponent* RSA key

```
Algorithm $\mathsf{Kg}(1^k)$
    $(N, e, d) \xleftarrow{\$} \mathsf{K}_{\mathrm{rsa}}(1^k)$
    $x \xleftarrow{\$} \mathbb{Z}_N^*$
    $X \leftarrow x^e \bmod N$
    $pk \leftarrow ((1^k, N, e), X)$
    $sk \leftarrow ((1^k, N, e), x)$
    Return $(pk, sk)$
```

| Prover $\mathsf{P}$ | | Verifier $\mathsf{V}$ |
|---|---|---|
| $y \xleftarrow{\$} \mathbb{Z}_N^*$ | | |
| $Y \leftarrow y^e \bmod N$ | $\xrightarrow{Y}$ | |
| | $\xleftarrow{c}$ | $c \xleftarrow{\$} \mathbb{Z}_{2^{l(k)}}$ |
| $z \leftarrow x^c y \bmod N$ | $\xrightarrow{z}$ | |
| | | If $z^e \equiv X^c Y \bmod N$ and $Y, z \in \mathbb{Z}_N^*$<br>then `acc` else `rej` |

Figure 9: **The $\mathcal{GQ}$-$\mathcal{SI}$ scheme.** The scheme is parameterized with a prime-exponent RSA key generator $\mathsf{K}_{\mathrm{rsa}}$ and a superlogarithmic challenge length $l : \mathbb{N} \to \mathbb{N}$ such that $2^{l(k)} < e$ for all $(N, e, d) \in [\mathsf{K}_{\mathrm{rsa}}(1^k)]$. The prover $\mathsf{P}$ and verifier $\mathsf{V}$ are initialized with states $sk = ((1^k, N, e), x)$ and $pk = ((1^k, N, e), X)$, respectively.

generator only outputs keys with $e$ prime. Security of schemes is based on either the one-wayness of the RSA function associated with the key generator, or on the hardness of the so-called one-more RSA inversion problem [BNPS03]. In particular, for the former, we say that the RSA function associated with the key generator $\mathsf{K}_{\mathrm{rsa}}$ is one-way if

$$\mathbf{Adv}^{\mathrm{rsa}}_{\mathsf{K}_{\mathrm{rsa}},\mathsf{A}}(k) = \Pr\left[\, x^e \equiv y \bmod N \;\;:\;\; (N, e, d) \xleftarrow{\$} \mathsf{K}_{\mathrm{rsa}}(1^k) \;;\; y \xleftarrow{\$} \mathbb{Z}_N^* \;;\; x \leftarrow \mathsf{A}(1^k, N, e, y) \,\right]$$

is negligible in $k$ for all polynomial-time algorithms $\mathsf{A}$. For the latter, the hardness is defined with respect to the following game:

```
Experiment $\mathbf{Exp}^{\mathrm{1m\text{-}rsa}}_{\mathsf{K}_{\mathrm{rsa}},\mathsf{A}}(k)$ :
    $(N, e, d) \xleftarrow{\$} \mathsf{K}_{\mathrm{rsa}}(1^k)$
    $i \leftarrow 0 \,;\, n \leftarrow 0$
    $(x_1, \ldots, x_m) \xleftarrow{\$} \mathsf{A}(1^k, N, e : \textsc{Chall}, \textsc{Inv})$
    If $m = i$ and $n < m$ and $x_i^e \equiv y_i \bmod N$ for all $i \in \{1, \ldots, m\}$
    Then return 1 else return 0.

Oracle $\textsc{Chall}$:
    $i \leftarrow i + 1 \,;\, y_i \xleftarrow{\$} \mathbb{Z}_N^*$
    Return $y_i$

Oracle $\textsc{Inv}(y)$:
    $n \leftarrow n + 1 \,;\, x \leftarrow y^d \bmod N$
    Return $x$
```

The adversary $\mathsf{A}$ is given $1^k, N, e$ as input and access to two oracles: a challenge oracle $\textsc{Chall}$ that on any input returns a new random target point $y_i \in \mathbb{Z}_N^*$ and an inversion oracle $\textsc{Inv}(\cdot) = (\cdot)^d \bmod N$. The adversary's goal is to invert all target points output by the challenge oracle using strictly fewer queries to the inversion oracle. We say that the one-more RSA inversion problem associated with $\mathsf{K}_{\mathrm{rsa}}$ is hard if the advantage

$$\mathbf{Adv}^{\mathrm{1m\text{-}rsa}}_{\mathsf{K}_{\mathrm{rsa}},\mathsf{A}}(k) = \Pr\left[\, \mathbf{Exp}^{\mathrm{1m\text{-}rsa}}_{\mathsf{K}_{\mathrm{rsa}},\mathsf{A}}(k) = 1 \,\right]$$

is negligible in $k$ for any polynomial-time adversary $\mathsf{A}$.

### 5.2.1 The $\mathcal{GQ}$ Family

The scheme. The $\mathcal{GQ}$-$\mathcal{SI}$ scheme associated to RSA key generator $\mathsf{K}_{\mathrm{rsa}}$ and challenge length $l(\cdot)$ is defined via Figure 9. The schemes originally presented by Guillou and Quisquater [GQ89] are actually $\mathcal{GQ}$-$\mathcal{IBI}$ = cSI-2-IBI($\mathcal{GQ}$-$\mathcal{SI}$) and $\mathcal{GQ}$-$\mathcal{IBS}$ = cSS-2-IBS(fs-I-2-S($\mathcal{GQ}$-$\mathcal{SI}$)).

Convertibility. To any RSA key generator $\mathsf{K}_{\mathrm{rsa}}$, we associate the following family of trapdoor samplable permutations:

```
Algorithm $\mathsf{TDG}(1^k)$:
    $(N, e, d) \xleftarrow{\$} \mathsf{K}_{\mathrm{rsa}}(1^k)$
    Return $((1^k, N, e), d)$

Algorithm $\mathsf{Smp}((1^k, N, e))$:
    $x \xleftarrow{\$} \mathbb{Z}_N^* \,;\, X \leftarrow x^e \bmod N$
    Return $(x, X)$

Algorithm $\mathsf{Inv}((1^k, N, e), d, X)$:
    $x \leftarrow X^d \bmod N$
    Return $x$.
```

```
Algorithm $\mathsf{Kg}(1^k)$
    $(N, e, d) \xleftarrow{\$} \mathsf{K}_{\rm rsa}(1^k)$
    $x \xleftarrow{\$} \mathbb{Z}_N^*$
    $X \leftarrow x^e \bmod N$
    $pk \leftarrow ((1^k, N, e), X)$
    $sk \leftarrow ((1^k, N, e), x)$
    Return $(pk, sk)$
```

| Prover P | | Verifier V |
|---|---|---|
| $y \xleftarrow{\$} \mathbb{Z}_N^*$ | | |
| $Y \leftarrow y^e \bmod N$ | $\xrightarrow{Y}$ | |
| | $\xleftarrow{c}$ | $c \xleftarrow{\$} \mathrm{ChSet}(pk)$ |
| If $c \notin \mathrm{ChSet}(pk)$ then abort | | |
| else $z \leftarrow xy^c \bmod N$ | $\xrightarrow{z}$ | |
| | | If $z^e \equiv XY^c \bmod N$ and $Y, z \in \mathbb{Z}_N^*$<br>then `acc` else `rej` |

Figure 10: **The *Sh-SI* and *Sh*$^*$-*SI* schemes.** Both schemes are specified in terms of a prime-exponent RSA generator $\mathsf{K}_{\rm rsa}$ and super-logarithmic challenge length $l(\cdot)$ such that $2^{l(k)} < e$ for all $(N, e, d) \in [\mathsf{K}_{\rm rsa}(1^k)]$. The prover P and verifier V are initialized with states $sk = ((1^k, N, e), x)$ and $pk = ((1^k, N, e), X)$, respectively. The *Sh-SI* and *Sh*$^*$-*SI* differ from each other in that the former uses challenge set $\mathrm{ChSet}(pk) = \{0, \ldots, 2^{l(k)} - 1\}$, while the latter uses $\mathrm{ChSet}(pk) = \{1, \ldots, 2^{l(k)}\}$.

The relation described by $(1^k, N, e)$ is $\mathbf{R} = \{(x, X) \in \mathbb{Z}_N^* \times \mathbb{Z}_N^* \,|\, x^e \equiv X \bmod N\}$. It is regular because raising to the $e$-th power is a permutation on the elements of $\mathbb{Z}_N^*$. The correct distribution of the output of the $\mathsf{Smp}$ algorithm follows from the random choice of $x$ from $\mathbb{Z}_N^*$, and the $\mathsf{Inv}$ algorithm returns the unique element $x \in \mathbb{Z}_N^*$ such that $x^e \equiv X \bmod N$.

Security. The *GQ-SI* scheme associated with a prime-exponent RSA key generator $\mathsf{K}_{\rm cg}$ and with super-logarithmic challenge length $l(\cdot)$ such that $2^{l(k)} < e$ for all $(N, e, d) \leftarrow [\mathsf{K}_{\rm rsa}(1^k)]$ is known to be imp-pa secure assuming the one-wayness of RSA [GQ89], and imp-aa and imp-ca secure assuming the hardness of the one-more RSA inversion problem [BP02]. According to Theorem 4.4, these results extend to *GQ-IBI*. Also, Corollary 4.10 says that *GQ-IBS* is uf-cma assuming RSA is one-way, but this was known [DKXY03].

### 5.2.2 The *Sh* and *Sh*$^*$ Families

The *Sh* family. In the same work in which he introduced the concept of identity-based cryptography, Shamir [Sha84] also proposed the first IBS scheme, but no SI or IBI schemes. He did not give a security proof for his IBS scheme, and none has been provided until now. We surface the SI scheme *Sh-SI* defined via Figure 10. One can check that *Sh-IBS* = cSS-2-IBS(fs-I-2-S(*Sh-SI*)) is exactly the IBS scheme in [Sha84]. The *Sh-SI* scheme is interesting both historically and technically. It turns out to be a "mirror-image" of *GQ-SI* that closely resembles the latter.

Convertibility. Convertibility of *Sh-SI* follows from the convertibility of *GQ-SI* since the two schemes have the same key generation algorithm.

Security. The first question to ask is whether *Sh-SI* is honest-verifier zero-knowledge (HVZK). While this was obvious for *GQ-SI* (and in fact, if true for an SI scheme, is usually obvious), it is in fact not apparent at first glance for *Sh-SI*, and one might suspect that the scheme is not HVZK. However, using a trick involving greatest common divisors, we show that *Sh-SI* is statistical (not perfect) HVZK. We also show that it is a proof of knowledge and thereby obtain the following:

**Theorem 5.2** The *Sh-SI* scheme associated with prime-exponent RSA generator $\mathsf{K}_{\rm rsa}$ and with super-logarithmic challenge length $l(\cdot)$ such that $2^{l(k)} < e$ for all $(N, e, d) \in [(\mathsf{K}_{\rm rsa}(1^k)]$ is imp-pa secure assuming that the RSA function associated to $\mathsf{K}_{\rm rsa}$ is one-way. ▌

**Proof:** The *Sh-SI* scheme is statistical honest-verifier zero-knowledge since the following algorithm simulates communication transcripts using only the public key:

```
Algorithm $\mathsf{Conv\text{-}sim}(1^k, N, e, X)$
    $c \xleftarrow{\$} \mathbb{Z}_{2^{l(k)}}$
    Compute $a, b \in \mathbb{Z}$ such that $ac + be = 1$ (using extended Euclidean algorithm)
    $y \xleftarrow{\$} Z_N^*$ ; $Y \leftarrow X^{-a} \cdot y^e \bmod N$ ; $z \leftarrow X^b \cdot y^c \bmod N$
    Return $(Y, c, z)$
```

The transcripts generated by $\mathsf{Conv\text{-}sim}$ are correctly distributed since $Y$ is uniformly distributed over $\mathbb{Z}_N^*$, $c$ is uniformly distributed over $\mathbb{Z}_{2^{l(k)}}$ and $z$ is the unique element of $\mathbb{Z}_N^*$ such that $z^e \equiv XY^c \bmod N$ because $z^e \equiv X^{be}y^{ec} \equiv X^{ac+be}Y^c \bmod N$. The second line of the algorithm may fail if $\gcd(c, e) \neq 1$. However, since $e$ is prime with $2^{l(k)} < e$, the only problematic value is $c = 0$, which occurs only with negligible probability $2^{-l(k)}$ when the challenge length $l$ is super-logarithmic in the security parameter.

The protocol is also a proof of knowledge of $x$, since from two valid challenge-response pairs $(c_1, z_1)$, $(c_2, z_2)$ for the same commitment $Y$, one can extract the secret key $x$ as follows: use the extended Euclidean algorithm to compute $a, b \in Z$ such that $a(c_1 - c_2) + be = 1$. Because $(z_1/z_2)^e \equiv Y^{c_1 - c_2} \bmod N$, it holds that $Y \equiv Y^{a(c_1-c_2)+be} \equiv \left((z_1/z_2)^a Y^b\right)^e \bmod N$, so that we can let $y \leftarrow (z_1/z_2)^a Y^b \bmod N$ and compute $x$ as $z_1 y^{-c_1} \bmod N$. The extraction does not work if $\gcd(c_1 - c_2, e) > 1$, but since $e$ is prime, this only occurs when $c_1 = c_2$.

Given an imp-pa adversary, one can use the fact that the protocol is honest-verifier zero-knowledge and a proof of knowledge to build an RSA inverter. ▌

Theorem 4.9 and Corollary 4.10 now imply that the $\mathcal{S}h\text{-}\mathcal{SS} = \mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathcal{S}h\text{-}\mathcal{SI})$ and $\mathcal{S}h\text{-}\mathcal{IBS}$ schemes are uf-cma secure under the same assumptions. Also, the imp-pa security of the $\mathcal{S}h\text{-}\mathcal{IBI} = \mathsf{cSI\text{-}2\text{-}IBI}(\mathcal{S}h\text{-}\mathcal{SI})$ scheme now follows from Theorem 4.4.

The $\mathcal{S}h\text{-}\mathcal{SI}$ scheme is trivially insecure under active attacks however, since the cheating verifier can learn the secret key by sending a zero challenge. This minor weakness is easily fixed by "removing" the zero challenge, leading to the $\mathcal{S}h^*\text{-}\mathcal{SI}$ scheme.

The $\mathcal{S}h^*$ family. In Figure 10, we define a modified scheme that we denote $\mathcal{S}h^*\text{-}\mathcal{SI}$. This scheme turns out to have security attributes analogous to those of $\mathcal{GQ}\text{-}\mathcal{SI}$ in that we can show the following:

**Theorem 5.3** The $\mathcal{S}h^*\text{-}\mathcal{SI}$ scheme associated with prime-exponent RSA generator $\mathsf{K}_{\mathrm{rsa}}$ and with super-logarithmic challenge length $l(\cdot)$ such that $2^{l(k)} < e$ for all $(N, e, d) \in [(\mathsf{K}_{\mathrm{rsa}}(1^k)]$ is imp-pa secure assuming that the RSA function associated to $\mathsf{K}_{\mathrm{rsa}}$ is one-way, and is imp-aa and imp-ca secure assuming that the one-more RSA inversion problem associated to $\mathsf{K}_{\mathrm{rsa}}$ is hard. ▌

**Proof:** The imp-pa security of the $\mathcal{S}h^*\text{-}\mathcal{SI}$ scheme follows from the fact that it is perfect honest-verifier zero-knowledge and a proof of knowledge of $x$. Conversations can be simulated by an algorithm similar to the $\mathsf{Conv\text{-}sim}$ algorithm in the proof of Theorem 5.2, but drawing $c$ from $\{1, \ldots, 2^{l(k)}\}$. Extracting $x$ is done exactly as in the proof of Theorem 5.2.

As one might expect, the proof of imp-aa and imp-ca security are very similar to the corresponding proofs for the GQ identification scheme [BP02]. Given imp-ca adversary $\mathsf{A} = (\mathsf{CV}, \mathsf{CP})$ for the $\mathcal{S}h^*\text{-}\mathcal{SI}$ scheme, we construct a one-more RSA adversary $\mathsf{B}$ as follows. On input $(1^k, N, e)$, $\mathsf{B}$ queries its challenge oracle the first time and stores the output as $X$. It then runs $\mathsf{CV}$ on input $1^k, pk = ((1^k, N, e), X)$. When $\mathsf{CV}$ requests to interact with a new prover session $s$, $\mathsf{B}$ queries the challenge oracle for a fresh target point $Y_s$ and returns $Y_s$ to $\mathsf{CV}$. When confronted with challenge $c_s \neq 0$, $\mathsf{B}$ uses the inversion oracle to compute $z_s \leftarrow \textsc{Inv}(XY_s^{c_s} \bmod N)$ and returns it to $\mathsf{CV}$. At the end of its execution, $\mathsf{CV}$ outputs initial state $St_{\mathsf{CP}}$ for the cheating prover $\mathsf{CP}$.

```
Algorithm $\mathsf{Kg}(1^k)$
    $(N, e, d) \xleftarrow{\$} \mathsf{K}_{\text{rsa}}(1^k)$
    $g \xleftarrow{\$} \mathbb{Z}_N^*$
    $x_1 \xleftarrow{\$} \mathbb{Z}_e$ ; $x_2 \xleftarrow{\$} \mathbb{Z}_N^*$
    $X \leftarrow g^{-x_1} x_2^{-e} \bmod N$
    $pk \leftarrow ((1^k, N, e, g), X)$
    $sk \leftarrow ((1^k, N, e, g), (x_1, x_2))$
    Return $(pk, sk)$
```

```
Prover P                                                          Verifier V
$y_1 \xleftarrow{\$} \mathbb{Z}_e$ ; $y_2 \xleftarrow{\$} \mathbb{Z}_N^*$
$Y \leftarrow g^{y_1} y_2^e \bmod N$                  $\xrightarrow{Y}$
                                                      $\xleftarrow{c}$   $c \xleftarrow{\$} \mathbb{Z}_{2^{l(k)}}$
$z_1 \leftarrow y_1 + cx_1 \bmod e$
$z_2 \leftarrow g^{\lfloor (y_1+cx_1)/e \rfloor} y_2 x_2^c \bmod N$   $\xrightarrow{z_1, z_2}$
                                                                  If $Y \equiv g^{z_1} z_2^e X^c \bmod N$
                                                                  and $Y, z_2 \in \mathbb{Z}_N^*$ and $z_1 \in \mathbb{Z}_e$
                                                                  then acc else rej
```

Figure 11: **The *OkRSA-SI* scheme.** The scheme is parameterized with a prime-exponent RSA generator $\mathsf{K}_{\text{rsa}}$ and a challenge length $l : \mathbb{N} \rightarrow \mathbb{N}$ such that $2^{l(k)} < e$ for any $e$ output by $\mathsf{K}_{\text{rsa}}(1^k)$. The prover P and verifier V are initialized with states $sk = ((1^k, N, e, g), (x_1, x_2))$ and $pk = ((1^k, N, e, g), X)$, respectively.

Algorithm B then runs CP in a reset experiment as in Lemma 5.1 to generate two communication transcripts $(Y, \tilde{c}_1, \tilde{z}_1)$ and $(Y, \tilde{c}_2, \tilde{z}_2)$ where challenges $\tilde{c}_1, \tilde{c}_2$ are uniformly distributed over $S_1$. With probability $\Pr[\text{res}(St_{\mathsf{CP}}, pk) = 1]$ these will both be accepting transcripts and $\tilde{c}_1 \neq \tilde{c}_2$. Since $e$ is prime and $2^{l(k)} < e$, we can compute $a, b \in \mathbb{Z}$ such that $a(\tilde{c}_1 - \tilde{c}_2) + be = 1$ and compute $x \in \mathbb{Z}_N^*$ such that $x^e \equiv X \bmod N$ as in the proof of Theorem 5.2. Inversions of all other target points $Y_s$ are either computed using the inversion oracle for unfinished sessions $s$, or are computed by applying the gcd trick again to get $a, b$ such that $ac_s + be = 1$ and using the fact that $y_s \equiv y_s^{ac_s+be} \equiv (z_s/x)^a Y^b \bmod N$.

In summary, B needed one target point and one inversion query for each prover session, but succeeded in inverting $X$ without the help of the inversion oracle, so it wins the game whenever the rewinding experiment succeeded. We have

$$\begin{aligned}
\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{Sh}^*\text{-}\mathcal{SI},\mathsf{A}}(k) &= \text{acc}(St_{\mathsf{CP}}, pk) \\
&\leq 2^{-l(k)} + \sqrt{\Pr[\text{res}(St_{\mathsf{CP}}, pk) = 1]} \\
&\leq 2^{-l(k)} + \sqrt{\mathbf{Adv}^{\text{1m-rsa}}_{\mathsf{K}_{\text{rsa}},\mathsf{B}}(k)}
\end{aligned}$$

by the Reset Lemma. ▌

By Theorem 4.4, the $\mathcal{Sh}^*$-$\mathcal{IBI}$ = cSI-2-IBI($\mathcal{Sh}^*$-$\mathcal{SI}$) scheme is imp-pa secure under the one-wayness of the RSA function related to $\mathsf{K}_{\text{rsa}}$, and is imp-aa and imp-ca secure under the hardness of the one-more RSA inversion problem associated to $\mathsf{K}_{\text{rsa}}$. The uf-cma security of the $\mathcal{Sh}^*$-$\mathcal{SS}$ = fs-I-2-S($\mathcal{Sh}^*$-$\mathcal{SI}$) and $\mathcal{Sh}^*$-$\mathcal{IBS}$ = cSS-2-IBS($\mathcal{Sh}^*$-$\mathcal{SS}$) under the one-wayness of the RSA function related to $\mathsf{K}_{\text{rsa}}$ follows from Theorem 4.9 and Corollary 4.10.

### 5.2.3 The *OkRSA* Family

The scheme. Okamoto [Oka93] presented an RSA-based SI scheme and a related RSA-based IBI scheme. The former is the *OkRSA-SI* scheme associated to a prime-exponent RSA key generator $\mathsf{K}_{\text{rsa}}$ and challenge length $l(\cdot)$ defined in Figure 11, and the latter is exactly *OkRSA-IBI* = cSI-2-IBI(*OkRSA-SI*).

Convertibility. Associated to any RSA key generator $\mathsf{K}_{\text{rsa}}$, consider the family of trapdoor samplable relations $\mathcal{F}$ = (TDG, Smp, Inv) given by:

```
Algorithm $\mathsf{Kg}(1^k)$
    $(N, e, d, f) \xleftarrow{\$} \mathsf{K}_{\mathrm{grsa}}(1^k)$
    Choose $g \in \mathbb{Z}_N^*$ of order $f$
    $h \leftarrow g^e \bmod N$
    $s \xleftarrow{\$} \mathbb{Z}_f$ ; $S \leftarrow g^{-s} \bmod N$
    $X \xleftarrow{\$} \mathbb{Z}_N^*$ ; $P \leftarrow X^{-d} S \bmod N$
    $pk \leftarrow ((1^k, N, e, f, g, h), X)$
    $sk \leftarrow ((1^k, N, e, f, g, h), (P, s))$
    Return $(pk, sk)$
```

```
Prover P                                          Verifier V
$y \xleftarrow{\$} \mathbb{Z}_f$
$Y \leftarrow h^y \bmod N$        $\xrightarrow{P,Y}$
                                  $\xleftarrow{c}$      $c \xleftarrow{\$} \mathbb{Z}_{2^{l(k)}}$
$z \leftarrow y + sc \bmod f$     $\xrightarrow{z}$
                                                  If $h^z (P^e X)^c \equiv Y \bmod N$
                                                  and $P, Y \in \mathbb{Z}_N^*$ and $z \in \mathbb{Z}_f$
                                                  then acc else rej
```

Figure 12: **The $\mathcal{Gir}$-$\mathcal{SI}$ scheme.** The scheme is parameterized with a challenge length $l(k)$ and a Girault-RSA key generator $\mathsf{K}_{\mathrm{grsa}}$. The prover $\mathsf{P}$ and verifier $\mathsf{V}$ are initialized with states $sk = ((1^k, N, e, f, g, h), (P, s))$ and $pk = ((1^k, N, e, f, g, h), X)$, respectively.

```
Algorithm $\mathsf{TDG}(1^k)$:
    $(N, e, d) \xleftarrow{\$} \mathsf{K}_{\mathrm{rsa}}(1^k)$
    $g \xleftarrow{\$} \mathbb{Z}_N^*$
    Return $((1^k, N, e, g), d)$

Algorithm $\mathsf{Smp}((1^k, N, e, g))$:
    $x_1 \xleftarrow{\$} \mathbb{Z}_e$ ; $x_2 \xleftarrow{\$} \mathbb{Z}_N^*$
    $X \leftarrow g^{-x_1} x_2^{-e} \bmod N$
    Return $((x_1, x_2), X)$

Algorithm $\mathsf{Inv}((1^k, N, e, g), d, X)$:
    $x_1 \xleftarrow{\$} \mathbb{Z}_e$
    $x_2 \leftarrow (g^{x_1} X)^{-d} \bmod N$
    Return $(x_1, x_2)$.
```

The relation described by $(1^k, N, e, g)$ is $\mathbf{R} = \{((x_1, x_2), X) \in (\mathbb{Z}_e \times \mathbb{Z}_N^*) \times \mathbb{Z}_N^* \mid g^{-x_1} x_2^{-e} \equiv X \bmod N\}$. Since raising to the $e$-th exponent induces a permutation on $\mathbb{Z}_N^*$, there exists a unique $x_2 \in \mathbb{Z}_N^*$ for each $X \in \mathbb{Z}_N^*$ and $x_1 \in \mathbb{Z}_e$ such that $((x_1, x_2), X) \in \mathbf{R}$. Obviously, each pair $(x_1, x_2) \in \mathbb{Z}_e \times \mathbb{Z}_N^*$ uniquely determines $X \in \mathbb{Z}_N^*$ such that $((x_1, x_2), X) \in \mathbf{R}$. Hence, $|\mathbf{R}^{-1}(X)| = e$ for all $X \in \mathbb{Z}_N^*$, and $\mathbf{R}$ is regular. As a consequence, the output of the $\mathsf{Smp}$ algorithm is uniformly distributed over $\mathbf{R}$ due to the random choice of $x_1$ and $x_2$, and the output of the $\mathsf{Inv}$ algorithm is uniformly distributed over $\mathbf{R}^{-1}(X)$ due to the random choice of $x_1$.

Security. Okamoto [Oka93] proved that the $\mathcal{OkRSA}$-$\mathcal{SI}$ scheme is imp-pa and imp-aa secure under the one-wayness of the RSA function associated to $\mathsf{K}_{\mathrm{rsa}}$ when the scheme is instantiated with super-logarithmic challenge length $l(\cdot)$ and a prime-exponent RSA key generator $\mathsf{K}_{\mathrm{rsa}}$ such that $2^{l(k)} < e$ for each $(N, e, d) \in [\mathsf{K}_{\mathrm{rsa}}(1^k)]$. The proof easily extends to imp-ca security as well. The imp-pa, imp-aa and imp-ca security of the $\mathcal{OkRSA}$-$\mathcal{IBI}$ = cSI-2-IBI($\mathcal{OkRSA}$-$\mathcal{SI}$) scheme follows from the convertibility of $\mathcal{OkRSA}$-$\mathcal{SI}$ and Theorem 4.4, and the uf-cma security of $\mathcal{OkRSA}$-$\mathcal{IBS}$ follows from Theorem 4.7.

### 5.2.4 The $\mathcal{Gir}$ Family

The scheme. In [Gir90], Girault proposed an SI scheme that we have defined in Figure 12 and named $\mathcal{Gir}$-$\mathcal{SI}$. The scheme is inspired by the Schnorr identification scheme [Sch90] and is parameterized with challenge length $l(\cdot)$ and a *Girault-RSA key generator* $\mathsf{K}_{\mathrm{grsa}}$, which is an algorithm that on input $1^k$ outputs $(N, e, d, f)$ such that $N = pq$ with $2^{k-1} \le N < 2^k$ and with $p, q$ of the special form $p = 2fp' + 1$ and $q = 2fq' + 1$, where $f, p', q', p, q$ are all primes. He also proposed a related IBI scheme. This IBI scheme did not use hash functions, which lead to an attack and later a fix [SSN98]. The fixed IBI scheme turns out to be exactly $\mathcal{Gir}$-$\mathcal{IBI}$ = cSI-2-IBI($\mathcal{Gir}$-$\mathcal{SI}$).

Convertibility. Consider the following family of trapdoor samplable relations $\mathcal{F}$ = ($\mathsf{TDG}$, $\mathsf{Smp}$, $\mathsf{Inv}$) associated to $\mathsf{K}_{\mathrm{grsa}}$:

```
Algorithm $\mathsf{TDG}(1^k)$:
    $(N, e, d, f) \xleftarrow{\$} \mathsf{K}_{\mathrm{grsa}}(1^k)$
    Choose $g \in \mathbb{Z}_N^*$ of order $f$
    $h \leftarrow g^e \bmod N$
    Return $((1^k, N, e, f, g, h), d)$
```

```
Algorithm $\mathsf{Smp}((1^k, N, e, f, g, h))$:
    $s \xleftarrow{\$} \mathbb{Z}_f$ ; $P \xleftarrow{\$} \mathbb{Z}_N^*$ ; $X \leftarrow P^{-e}h^{-s} \bmod N$
    Return $((P, s), X)$
```

```
Algorithm $\mathsf{Inv}((1^k, N, e, f, g, h), d, X)$:
    $s \xleftarrow{\$} \mathbb{Z}_f$ ; $S \leftarrow g^{-s} \bmod N$ ; $P \leftarrow X^{-d}S \bmod N$
    Return $(P, s)$.
```

The relation described by $(1^k, N, e, f, g, h)$ is $\mathbf{R} = \{((P, s), X) \in (\mathbb{Z}_N^* \times \mathbb{Z}_f) \times \mathbb{Z}_N^* \mid P^e \equiv X^{-1}h^{-s} \bmod N\}$. Each $X \in \mathbb{Z}_N^*$ has exactly $|\mathbf{R}^{-1}(X)| = f$ inverses, namely one for each $s \in \mathbb{Z}_f$. On the other hand, each pair $(P, s) \in \mathbb{Z}_N^* \times \mathbb{Z}_f$ uniquely determines $X \in \mathbb{Z}_N^*$ such that $((P, s), X) \in \mathbf{R}$. From this, the regularity of $\mathbf{R}$ and the correct output distribution of the Smp and Inv algorithms above follow. The $\mathcal{G}ir$-$\mathcal{SI}$ scheme is convertible with respect to $\mathcal{F}$.

Security. The convertibility does not help here, however, because we found that all schemes in the family are insecure. In particular, $\mathcal{G}ir$-$\mathcal{SI}$ is not even imp-pa secure, and neither is the fixed IBI scheme $\mathcal{G}ir$-$\mathcal{IBI}$. The signature scheme $\mathcal{G}ir$-$\mathcal{IBS}$ = cSS-2-IBS(fs-I-2-S($\mathcal{G}ir$-$\mathcal{IBI}$)) is not uf-cma secure either.

**Theorem 5.4 (Insecurity of the $\mathcal{G}ir$ Family)** The $\mathcal{G}ir$-$\mathcal{SI}$ scheme depicted in Figure 12 and the $\mathcal{G}ir$-$\mathcal{IBI}$ = cSI-2-IBI($\mathcal{G}ir$-$\mathcal{SI}$) scheme as presented in [Gir90, SSN98] are insecure against impersonation under passive, active and concurrent attack. The $\mathcal{G}ir$-$\mathcal{SS}$ = fs-I-2-S($\mathcal{G}ir$-$\mathcal{SI}$) and the $\mathcal{G}ir$-$\mathcal{IBS}$ = cSS-2-IBS($\mathcal{G}ir$-$\mathcal{SS}$) schemes are universally forgeable under known-message attack. ▌

**Proof:** We attack only the $\mathcal{G}ir$-$\mathcal{IBS}$ scheme, since the insecurity of the SI, IBI, and SS schemes then follows as a consequence. The attack we present is a universal forgery under known-message attack, which is an even stronger attack than the standard existential forgery under chosen-message attack. In the $\mathcal{G}ir$-$\mathcal{IBS}$ scheme, a signature of a user $I$ on a message $M$ under the master public key $mpk = (1^k, N, e, f, g, h)$ is a tuple $(P, Y, z)$ such that $Y \equiv h^z(P^e \cdot \mathrm{H}_1(I))^{\mathrm{H}_2(P\|Y\|M)} \bmod N$, where $\mathrm{H}_1$ is the random oracle associated to the cSI-2-IBI transform and $\mathrm{H}_2$ is the random oracle associated to the fs-I-2-S transform. The flaw at the heart of the attack is that in the subgroup generated by $g$, computing RSA inverses is easy because the order $f$ of the subgroup is known. Given a valid signature $(P_1, Y_1, z_1)$ for message $M_1$ and identity $I$, an adversary can forge $I$'s signature for any message $M_2$ as follows. It first computes $d' \leftarrow e^{-1} \bmod f$, and then computes $S' \leftarrow (P_1^e \cdot \mathrm{H}_1(I))^{d'} \bmod N$ so that

$$\begin{aligned} S' &\equiv \left(\mathrm{H}_1(I)^{-1}S^e \cdot \mathrm{H}_1(I)\right)^{d'} \bmod N \\ &\equiv S \bmod N. \end{aligned}$$

Then, it chooses $s_2$ from $\mathbb{Z}_f$ and computes $P_2 \leftarrow P_1 S'^{-1} g^{-s_2} \bmod N$. Since $P_2 \equiv \mathrm{H}_1(I)^{-d} g^{-s_2} \bmod N$, the pair $(P_2, s_2)$ might have been output by the UKg algorithm as part of the user secret key corresponding to identity $I$. Therefore, any signature the adversary generates using this pair will be considered valid for identity $I$. The adversary now follows the normal signing algorithm to compute the forgery: it chooses $y_2$ from $\mathbb{Z}_q$, sets $Y_2 \leftarrow h^{y_2} \bmod N$, computes $z_2 \leftarrow y_2 + s_2\mathrm{H}_2(P_2\|Y_2\|M_2) \bmod f$. The forgery is $(P_2, Y_2, z_2)$. ▌

It is natural to consider counteracting the above attack by including $f$ only in the secret key, and not in the public key. The resulting scheme however is no longer a cSI scheme, preventing it from being automatically transformed into an IBI or IBS scheme. It is also not clear how to design a secure IBI or IBS scheme directly (i.e. without using the transforms), as $f$ will probably have to be included in the secret key of every user. An adversary can then easily extract $f$ by corrupting a single user.

### 5.3 Schemes based on Pairings

Many recent papers propose pairing-based IBS schemes [SOK00, Pat02, Hes03, CC03, Yi03] (the schemes independently published by [CC03] and [Yi03] are actually equivalent). Barring [CC03], none of these papers prove their scheme secure. (Some proofs in weak models were however provided in [Hes03, Yi03].) However, the first scheme presented in [Hes03] was proven secure in [DKXY03]. A second scheme of [Hes03] was later found to be insecure [Che02].

None of these papers define SI or IBI schemes. We surface new SI schemes that, through our transformations, yield exactly the proposed IBS schemes (for the schemes of [Hes03, CC03, Yi03]), or a close variant thereof (for the scheme of [SOK00]). By analyzing the security of the SI scheme, we obtain security results for all schemes in the families. The scheme of [Pat02] does not seem to be related to any convertible SI scheme, leaving its security an open problem.

Definitions and Assumptions A *pairing generator* is a polynomial-time randomized algorithm $\mathsf{K}_{\text{pair}}$ that on input $1^k$ outputs $(\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P)$, where $\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle$ are the descriptions of an additive group $\mathbb{G}_1$ and a multiplicative group $\mathbb{G}_2$ of the same prime order $q$ such that $2^{k-1} \leq q < 2^k$, $P$ is a generator of $\mathbb{G}_1$, and $\langle\hat{e}\rangle$ is the description of a non-degenerate computable bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ associated to $\mathbb{G}_1$ and $\mathbb{G}_2$. This means that (1) $\hat{e}$ does not map all pairs of elements in $\mathbb{G}_1$ to the identity element of $\mathbb{G}_2$; (2) the pairing $\hat{e}(Q, R)$ is computable in polynomial time given descriptions $\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle$ for all $Q, R \in \mathbb{G}_1$; and (3) for all $Q, R \in \mathbb{G}_1$ and for all $a, b \in \mathbb{Z}_q$, $\hat{e}(aQ, bR) = \hat{e}(Q, R)^{ab}$. Pairings can be constructed from the Weil and Tate pairings over supersingular elliptic curves [BF01]. The computational Diffie-Hellman (CDH) problem in $\mathbb{G}_1$ associated to $\mathsf{K}_{\text{pair}}$ is said to be hard if

$$\begin{aligned}\mathbf{Adv}^{\text{cdh}}_{\mathsf{K}_{\text{pair}},\mathsf{A}}(k) \quad = \quad \Pr\Big[\ & \mathsf{A}(1^k, \langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, aP, bP) = abP \ : \\ & (\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P) \xleftarrow{\$} \mathsf{K}_{\text{pair}}(1^k)\ ;\ a, b \xleftarrow{\$} \mathbb{Z}_q\ \Big]\end{aligned}$$

is negligible in $k$ for any polynomial-time algorithm $\mathsf{A}$. The assumption that CDH is hard is a weaker assumption than the bilinear CDH assumption used by Boneh and Franklin [BF01] which states that, given $(aP, bP, cP)$, computing $\hat{e}(P, P)^{abc}$ is hard.

The *one-more computational Diffie-Hellman problem* [Bol03] in $\mathbb{G}_1$ associated to $\mathsf{K}_{\text{pair}}$ is defined through the following experiment:

```
Experiment $\mathbf{Exp}^{\text{1m-cdh}}_{\mathsf{K}_{\text{pair}},\mathsf{A}}(k)$ :
    $(\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P) \xleftarrow{\$} \mathsf{K}_{\text{pair}}(1^k)$
    $a \xleftarrow{\$} \mathbb{Z}_q\ ;\ i \leftarrow 0\ ;\ n \leftarrow 0$
    $(Q_1, \ldots, Q_m) \xleftarrow{\$} \mathsf{A}(1^k, \langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, aP : \textsc{Chall}, \textsc{Cdh})$
    If $m = i$ and $n < m$ and $Q_i \equiv aP_i$ for all $i \in \{1, \ldots, m\}$
    Then return 1 else return 0.

Oracle $\textsc{Chall}$:
    $i \leftarrow i + 1\ ;\ P_i \xleftarrow{\$} \mathbb{G}_1$
    Return $P_i$

Oracle $\textsc{Cdh}(P)$:
    $n \leftarrow n + 1\ ;\ Q \leftarrow aP$
    Return $Q$
```

The adversary $\mathsf{A}$ is given $1^k, \langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, aP$ as input and access to two oracles: a challenge oracle $\textsc{Chall}$ that on any input returns a new random target point $P_i \in \mathbb{G}_1$ and a CDH oracle $\textsc{Cdh}(\cdot) = a(\cdot)$. The adversary's goal is to compute CDH solutions for all target points output by the challenge oracle using strictly fewer queries to the CDH oracle. We say that the one-more RSA inversion problem in $\mathbb{G}_1$ associated to $\mathsf{K}_{\text{pair}}$ is hard if the advantage

$$\mathbf{Adv}^{\text{1m-cdh}}_{\mathsf{K}_{\text{pair}},\mathsf{A}}(k) = \Pr\Big[\mathbf{Exp}^{\text{1m-cdh}}_{\mathsf{K}_{\text{pair}},\mathsf{A}}(k) = 1\Big]$$

is negligible in $k$ for any polynomial-time adversary $\mathsf{A}$. This assumption was used before in the proofs of a group signature scheme [Bol03] and a transitive signature scheme [BN05].

```
Algorithm $\mathsf{Kg}(1^k)$
    $(\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P) \xleftarrow{\$} \mathsf{K}_{\mathrm{pair}}(1^k)$
    $s \xleftarrow{\$} \mathbb{Z}_q$ ; $S \leftarrow sP$
    $U \xleftarrow{\$} \mathbb{G}_1$ ; $V \leftarrow sU$
    $pk \leftarrow (((\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, S), U)$
    $sk \leftarrow (((\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, S), V)$
    Return $(pk, sk)$
```

| Prover $\mathsf{P}$ | | Verifier $\mathsf{V}$ |
|---|---|---|
| $y \xleftarrow{\$} \mathbb{Z}_q$ | | |
| $Y \leftarrow yP$ | $\xrightarrow{Y}$ | |
| | $\xleftarrow{C}$ | $C \xleftarrow{\$} \mathbb{G}_1$ |
| $Z \leftarrow yC + V$ | $\xrightarrow{Z}$ | |
| | | If $\hat{e}(Z, P) = \hat{e}(U, S)\hat{e}(C, Y)$<br>and $Y, Z \in \mathbb{G}_1$<br>then `acc` else `rej` |

| Prover $\mathsf{P}$ | | Verifier $\mathsf{V}$ |
|---|---|---|
| $y \xleftarrow{\$} \mathbb{Z}_q$ | | |
| $\alpha \leftarrow \hat{e}(P, P)^y$ | $\xrightarrow{\alpha}$ | |
| | $\xleftarrow{c}$ | $c \xleftarrow{\$} \mathbb{Z}_q$ |
| $Z \leftarrow yP + cV$ | $\xrightarrow{Z}$ | |
| | | If $\hat{e}(Z, P) = \alpha \cdot \hat{e}(U, S)^c$<br>and $\alpha \in \mathbb{G}_2$ and $Z \in \mathbb{G}_1$<br>then `acc` else `rej` |

| Prover $\mathsf{P}$ | | Verifier $\mathsf{V}$ |
|---|---|---|
| $y \xleftarrow{\$} \mathbb{Z}_q$ | | |
| $Y \leftarrow yU$ | $\xrightarrow{Y}$ | |
| | $\xleftarrow{c}$ | $c \xleftarrow{\$} \mathbb{Z}_q$ |
| $Z \leftarrow (y + c)V$ | $\xrightarrow{Z}$ | |
| | | If $\hat{e}(Z, P) = \hat{e}(Y + cU, S)$<br>and $Y, Z \in \mathbb{G}_1$<br>then `acc` else `rej` |

Figure 13: **The pairing-based IBS schemes as SI schemes.** All schemes use the same key generation algorithm $\mathsf{Kg}$. Presented here are the $\mathcal{SOK}$-$\mathcal{SI}$ (upper right), $\mathcal{H}s$-$\mathcal{SI}$ (lower left) and $\mathcal{C}h\mathcal{C}h$-$\mathcal{SI}$ (lower right) schemes. The provers $\mathsf{P}$ and verifiers $\mathsf{V}$ are initialized with states $sk = (((\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, S), V)$ and $pk = (((\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, S), U)$, respectively.

### 5.3.1 The $\mathcal{SOK}$, $\mathcal{H}s$ and $\mathcal{C}h\mathcal{C}h$ Families

The schemes. In Figure 13, we associate to a pairing generator $\mathsf{K}_{\mathrm{pair}}$ the $\mathcal{SOK}$-$\mathcal{SI}$ scheme that we surface from [SOK00], the $\mathcal{H}s$-$\mathcal{SI}$ scheme that we surface from [Hes03] and the $\mathcal{C}h\mathcal{C}h$-$\mathcal{SI}$ scheme that we surface from [CC03, Yi03]. The $\mathcal{H}s$-$\mathcal{IBS}$ = cSS-2-IBS(fs-I-2-S($\mathcal{H}s$-$\mathcal{SI}$)) and $\mathcal{C}h\mathcal{C}h$-$\mathcal{IBS}$ = cSS-2-IBS( fs-I-2-S($\mathcal{C}h\mathcal{C}h$-$\mathcal{SI}$)) schemes are exactly the original IBS schemes, while $\mathcal{SOK}$-$\mathcal{IBS}$ = cSS-2-IBS(fs-I-2-S( $\mathcal{SOK}$-$\mathcal{SI}$)) slightly differs from the scheme of [SOK00] in the sense that the latter uses $\mathrm{H}(M)$ to generate the challenge when computing a signature, rather than $\mathrm{H}(Y \| M)$.

Convertibility. We now show that all these pairing-based SI schemes are convertible. They all have the same key-generation algorithm, so a common argument applies. For any pairing generator $\mathsf{K}_{\mathrm{pair}}$, consider the family of samplable trapdoor relations $\mathcal{F} = (\mathsf{TDG}, \mathsf{Smp}, \mathsf{Inv})$:

```
Algorithm $\mathsf{TDG}(1^k)$:
    $(\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, q, P) \xleftarrow{\$} \mathsf{K}_{\mathrm{grsa}}(1^k)$
    $s \xleftarrow{\$} \mathbb{Z}_q$ ; $S \leftarrow sP$
    Return $(((\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, S), s)$

Algorithm $\mathsf{Smp}(((\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, S))$:
    $u \xleftarrow{\$} \mathbb{Z}_q$ ; $U \leftarrow uP$ ; $V \leftarrow uS$ ; Return $(V, U)$

Algorithm $\mathsf{Inv}(((\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, S), s, U)$:
    $V \leftarrow sU$ ; Return $V$.
```

The relation $\mathbf{R}$ described by $(\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, S)$ is $\mathbf{R} = \{\, (V, U) \in \mathbb{G}_1 \times \mathbb{G}_1 \,:\, \hat{e}(V, P) = \hat{e}(U, S) \,\}$. Since both $\mathbb{G}_1$ and $\mathbb{G}_2$ are of prime order and since $\hat{e}$ is a bilinear map, $\hat{e}(P, P)$ is a generator of $\mathbb{G}_2$. Let $s, u, v$ be the unique elements of $\mathbb{Z}_q$ such that $S \equiv sP$, $U \equiv uP$ and $V \equiv vP$. Then the equation $\hat{e}(V, P) \equiv \hat{e}(U, S)$ implies that $v \equiv us \bmod q$. Since $q$ is prime, for each $v \in \mathbb{Z}_Q$ there exists exactly one $u \in \mathbb{Z}_Q$ that satisfies this equation. Hence, $\mathbf{R}$ is actually a permutation on $\mathbb{G}_1$, from which the regularity of $\mathbf{R}$ follows. The output of the $\mathsf{Smp}$ algorithm is uniformly distributed of $\mathbf{R}$ because by the random choice of $u$, $U$ is uniformly distributed over $\mathbb{G}_1$ and $V$ is the only element of $\mathbb{G}_1$ such that $(V, U) \in \mathbf{R}$. Since $|\mathbf{R}^{-1}(V)| = 1$ for all $V \in \mathbb{G}_1$, $\mathsf{Inv}$'s output is trivially correctly distributed.

```
Simulator for $\mathcal{SOK}$-$\mathcal{SI}$:
    $y \xleftarrow{\$} \mathbb{Z}_q$ ; $Y \leftarrow yS$
    $z \xleftarrow{\$} \mathbb{Z}_q$ ; $Z \leftarrow zS$
    $C \leftarrow y^{-1}(zP - U)$
    Return $(Y, C, Z)$
```

```
Simulator for $\mathcal{H}s$-$\mathcal{SI}$:
    $Z \xleftarrow{\$} \mathbb{G}_1$
    $c \xleftarrow{\$} \mathbb{Z}_q$
    $\alpha \leftarrow \hat{e}(Z, P)\hat{e}(U, S)^{-c}$
    Return $(\alpha, c, Z)$
```

```
Simulator for $\mathcal{C}h\mathcal{C}h$-$\mathcal{SI}$:
    $z \xleftarrow{\$} \mathbb{Z}_q$ ; $Z \leftarrow zS$
    $c \xleftarrow{\$} \mathbb{Z}_q$
    $Y \leftarrow zP - cU$
    Return $(Y, c, Z)$
```

Figure 14: Conversation simulator algorithms for the pairing-based schemes.

SECURITY. In the following, we prove that the $\mathcal{SOK}$-$\mathcal{SI}$, $\mathcal{H}s$-$\mathcal{SI}$ and $\mathcal{C}h\mathcal{C}h$-$\mathcal{SI}$ schemes are imp-pa secure under the CDH assumption associated to $\mathsf{K}_{\text{pair}}$, and that the $\mathcal{H}s$-$\mathcal{SI}$ and $\mathcal{C}h\mathcal{C}h$-$\mathcal{SI}$ schemes are imp-ca secure under the one-more CDH assumption associated to $\mathsf{K}_{\text{pair}}$. Security results for the IBI, SS and IBS schemes follow through the transforms.

**Theorem 5.5** The $\mathcal{SOK}$-$\mathcal{SI}$, $\mathcal{H}s$-$\mathcal{SI}$ and $\mathcal{C}h\mathcal{C}h$-$\mathcal{SI}$ schemes are imp-pa secure assuming that the computational Diffie-Hellman problem associated with the underlying generator $\mathsf{K}_{\text{pair}}$ is hard. ▌

**Proof:** We prove imp-pa security by showing that all three schemes are honest-verifier zero-knowledge and proofs of knowledge for $V$. The former can be seen from the conversation simulators given in Figure 14. It is easily verified that their outputs are correctly distributed. We demonstrate the proof of knowledge property by showing how any cheating prover $\mathsf{CP}$ can be used to extract the prover's secret $V$. For the $\mathcal{SOK}$-$\mathcal{SI}$ scheme, the extractor chooses $c \xleftarrow{\$} \mathbb{Z}_q$ upon receiving $Y$ from $\mathsf{CP}$, and sends $C \leftarrow cP$ as the challenge. From $\mathsf{CP}$'s response $Z$, the extractor computes $V$ as $Z - cY$. The extractor of the two other schemes runs the cheating prover in a reset experiment to obtain two responses $Z_1, Z_2$ to randomly chosen challenges $c_1, c_2$ for the same commitment $Y$ (or $\alpha$). If both transcripts are valid, $V$ can be computed as $(c_1 - c_2)^{-1}(Z_1 - Z_2)$. Using the Reset Lemma, we obtain

$$\begin{aligned}\mathbf{Adv}^{\text{imp-pa}}_{\mathcal{SOK}\text{-}\mathcal{SI},\mathsf{A}}(k) &\leq \mathbf{Adv}^{\text{cdh}}_{\mathsf{K}_{\text{pair}},\mathsf{B}}(k) \\ \mathbf{Adv}^{\text{imp-pa}}_{\mathcal{SI},\mathsf{A}}(k) &\leq 2^{-k+1} + \sqrt{\mathbf{Adv}^{\text{cdh}}_{\mathsf{K}_{\text{pair}},\mathsf{B}}(k)} \quad \text{for } \mathcal{SI} \in \{\mathcal{H}s\text{-}\mathcal{SI}, \mathcal{C}h\mathcal{C}h\text{-}\mathcal{SI}\}\end{aligned}$$

as the bounds on the advantage of any imp-pa adversary $\mathsf{A}$. ▌

Theorem 4.4 implies that $\mathcal{C}h\mathcal{C}h$-$\mathcal{IBI}$, $\mathcal{SOK}$-$\mathcal{IBI}$ and $\mathcal{H}s$-$\mathcal{IBI}$ are imp-pa secure, and Theorem 4.9 implies that $\mathcal{C}h\mathcal{C}h$-$\mathcal{SS}$, $\mathcal{SOK}$-$\mathcal{SS}$ and $\mathcal{H}s$-$\mathcal{SS}$ are uf-cma secure. Corollary 4.10 implies that $\mathcal{C}h\mathcal{C}h$-$\mathcal{IBS}$, $\mathcal{SOK}$-$\mathcal{IBS}$ and $\mathcal{H}s$-$\mathcal{IBS}$ are uf-cma secure IBS schemes, but of these only the result about $\mathcal{SOK}$-$\mathcal{IBS}$ is new.

**Theorem 5.6** The $\mathcal{C}h\mathcal{C}h$-$\mathcal{SI}$ and $\mathcal{H}s$-$\mathcal{SI}$ schemes are imp-aa and imp-ca secure assuming that the one-more computational Diffie-Hellman problem associated with the underlying pairing generator $\mathsf{K}_{\text{pair}}$ is hard. ▌

**Proof:** The way to construct a one-more CDH algorithm $\mathsf{B}$ out of an imp-aa/ca adversary $\mathsf{A} = (\mathsf{CV}, \mathsf{CP})$ is actually very similar for the $\mathcal{C}h\mathcal{C}h$-$\mathcal{SI}$ and $\mathcal{H}s$-$\mathcal{SI}$ schemes. We present a single construction here and emphasize the differences. When run on input $(1^k, \langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, aP)$, algorithm $\mathsf{B}$ lets $S \leftarrow aP$, queries the challenge oracle a first time to get $U \leftarrow \textsc{Chall}$, and runs $\mathsf{CV}$ on input security parameter $1^k$ and public key $pk = ((\langle\mathbb{G}_1\rangle, \langle\mathbb{G}_2\rangle, \langle\hat{e}\rangle, q, P, S), U)$. Each time $\mathsf{CV}$ asks for interaction with a new prover session $i$, it queries the the challenge oracle to get $Y_i \leftarrow \textsc{Chall}(\varepsilon)$. This value is returned to the cheating verifier for the $\mathcal{C}h\mathcal{C}h$-$\mathcal{SI}$ scheme, while $\alpha_i \leftarrow \hat{e}(Y_i, S)$ is returned for the $\mathcal{H}s$-$\mathcal{SI}$

```
Algorithm $\mathsf{Kg}(1^k)$
    $(\langle\mathbb{G}\rangle, q, g) \xleftarrow{\$} \mathsf{K}_{\mathrm{cg}}(1^k)$
    $r \xleftarrow{\$} \mathbb{Z}_q$ ; $R \leftarrow g^r$
    $x \xleftarrow{\$} \mathbb{Z}_q$ ; $X \leftarrow g^x$
    $h \xleftarrow{\$} Z_q$ ; $s \leftarrow r^{-1}(h - Rx) \bmod q$
    $pk \leftarrow ((1^k, \langle\mathbb{G}\rangle, q, g, X), h)$
    $sk \leftarrow ((1^k, \langle\mathbb{G}\rangle, q, g, X), (R, s))$
    Return $(pk, sk)$
```

```
Prover P                                      Verifier V
$y \xleftarrow{\$} \mathbb{Z}_q$
$Y \leftarrow R^{-y}$            $\xrightarrow{R,Y}$
                                 $\xleftarrow{c}$           $c \xleftarrow{\$} \mathbb{Z}_{2^{l(k)}}$
$z \leftarrow y + cs \bmod q$    $\xrightarrow{z}$
                                                            If $g^{ch} \equiv R^z Y X^{cR}$
                                                            and $R, Y \in \mathbb{G}$ and $z \in \mathbb{Z}_q$
                                                            then acc else rej
```

Figure 15: **The *Beth-SI* scheme.** The scheme is parameterized with a prime-order cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ and super-logarithmic challenge length $l : \mathbb{N} \to \mathbb{N}$ such that $2^{l(k)} < q$ for all $(\langle\mathbb{G}\rangle, q, g) \in [\mathsf{K}_{\mathrm{cg}}(1^k)]$. The prover P and verifier V are run on initial states $sk = ((1^k, \langle\mathbb{G}\rangle, q, g, X), (R, s))$ and $pk = ((1^k, \langle\mathbb{G}\rangle, q, g, X), h)$, respectively.

scheme. Upon receiving challenge $c_i$ from CV, the one-more CDH adversary B uses its CDH oracle to compute $Z_i \leftarrow \textsc{Cdh}(Y_i + c_iU)$ and returns it to CV. The validity of this response can be verified by observing that for the *ChCh-SI* scheme it holds that $\hat{e}(Z_i, P) = \hat{e}(a(Y_i + c_iU), P) = \hat{e}(Y_i + c_iU, S)$, and for the *Hs-SI* scheme that $\hat{e}(Z_i, P) = \hat{e}(a(Y_i + c_iU), P) = \hat{e}(Y_i, S)\hat{e}(c_iU, S) = \alpha_i \cdot \hat{e}(U, S)^{c_i}$. When CV outputs initial state $St_{\mathsf{CP}}$ for the cheating prover, B extracts a value $V$ from CP such that $V = aU$ by running CP in a reset experiment as in the proof of Theorem 5.5. This is the solution to B's first challenge, and it can compute solutions to all other challenges as $Q_i \leftarrow Z_i - c_iV$. (The solution for $Y_i$ in unfinished prover sessions can be queried directly from the Cdh oracle.) In summary, if CV interacted with $n$ different prover sessions, then B succeeded in solving $n + 1$ challenges using only $n$ queries to the Cdh oracle, and hence wins the game. Therefore, the advantage of an imp-aa/ca adversary A for $\mathcal{SI} \in \{\mathit{ChCh\text{-}SI}, \mathit{Hs\text{-}SI}\}$ is bounded by

$$\mathbf{Adv}^{\text{imp-aa/ca}}_{\mathcal{SI},\mathsf{A}}(k) \leq 2^{-k+1} + \sqrt{\mathbf{Adv}^{\text{1m-cdh}}_{\mathsf{K}_{\text{pair}},\mathsf{B}}(k)}\ .$$

due to the Reset Lemma. ▌

Theorem 4.4 implies that the *ChCh-IBI* and *Hs-IBI* schemes are imp-aa and imp-ca secure assuming that the one-more computational Diffie-Hellman problem in the group $\mathbb{G}_1$ associated to $\mathsf{K}_{\text{pair}}$ is hard. Thus, we obtain new, pairing-based IBI schemes with proofs of security.

*SOK-SI* and *SOK-IBI* are insecure under active or concurrent attacks: upon receiving a commitment $Y$, an adversary can choose $c \xleftarrow{\$} \mathbb{Z}_q$, submit $C \leftarrow cP$ as the challenge, and compute the prover's secret key from the response $Z$ as $V \leftarrow Z - cY$.

As indicated above, *SOK-IBS*, that we prove uf-cma secure, is slightly different from the IBS scheme in [SOK00]. It is unclear whether the latter can be proved secure, so *SOK-IBS* might be preferable to the original one. This highlights a benefit of our framework, namely that we can obtain provable schemes in a systematic way.

### 5.4 A Scheme based on Discrete Logarithms

The scheme. A *cyclic group generator* $\mathsf{K}_{\mathrm{cg}}$ is a randomized polynomial-time algorithm that on input $1^k$ returns a tuple $(\langle\mathbb{G}\rangle, q, g)$, where $\langle\mathbb{G}\rangle$ is the description of a cyclic multiplicative group $\mathbb{G}$ of order $q$ and where $g$ is a generator of $\mathbb{G}$. A *prime-order* cyclic group generator is a group generator that $q$ is prime for all $(\langle\mathbb{G}\rangle, q, g) \in [\mathsf{K}_{\mathrm{cg}}(1^k)]$ for all $k \in \mathbb{N}$. We surface the *Beth-SI* scheme defined in Figure 15 from [Bet88]. It is parameterized with a cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ and challenge length $l(\cdot)$. The

$\mathcal{B}eth\text{-}\mathcal{IBI} = \textsf{cSI-2-IBI}(\mathcal{B}eth\text{-}\mathcal{SI})$ scheme is a more efficient variant of the IBI scheme actually presented in [Bet88]. The scheme of [Bet88] is actually more general, allowing for higher "key multiplicities", just as the $\mathcal{I}t\mathcal{R}$ family. We don't have any results for the more general scheme though, and limit our presentation to the special case above.

Convertibility. Consider the following family of trapdoor samplable relations $\mathcal{F} = (\mathsf{TDG}, \mathsf{Smp}, \mathsf{Inv})$ associated to $\mathsf{K}_{\mathrm{cg}}$:

```
Algorithm $\mathsf{TDG}(1^k)$:
    $(\langle\mathbb{G}\rangle, q, g) \xleftarrow{\$} \mathsf{K}_{\mathrm{cg}}(1^k)$
    $x \xleftarrow{\$} \mathbb{Z}_q$ ; $X \leftarrow g^x$
    Return $((1^k, \langle\mathbb{G}\rangle, q, g, X), x)$
```

```
Algorithm $\mathsf{Smp}((1^k, \langle\mathbb{G}\rangle, q, g, X))$:
    $a, b \xleftarrow{\$} \mathbb{Z}_q$ ; $R \leftarrow X^a g^b$
    $s \leftarrow a^{-1}R \bmod q$ ; $h \leftarrow bs \bmod q$
    Return $((R, s), h)$
```

```
Algorithm $\mathsf{Inv}((1^k, \langle\mathbb{G}\rangle, q, g, X), x, h)$:
    $r, h \xleftarrow{\$} \mathbb{Z}_q$ ; $R \leftarrow g^r$ ; $s \leftarrow r^{-1}(h - xR) \bmod q$
    Return $(R, s)$.
```

The relation described by $(1^k, \langle\mathbb{G}\rangle, q, g, X)$ is $\mathbf{R} = \{((R,s),h) \in (\mathbb{G} \times \mathbb{Z}_q) \times \mathbb{Z}_q \,|\, g^h \equiv R^s X^R\}$. For each pair $(R, s) \in (\mathbb{G} \times \mathbb{Z}_q)$, there exists exactly one $h \in \mathbb{Z}_q$ such that $((R,s),h) \in \mathbf{R}$, namely the unique discrete logarithm of $R^s X^R$. On the other hand, for each $h \in \mathbb{Z}_q$, there exist $q$ pairs $(R, s)$ such that $((R,s),h) \in \mathbf{R}$, namely one for each $s \in \mathbb{Z}_q$. The output of the $\mathsf{Smp}$ algorithm is uniformly distributed over $\mathbf{R}$ because $s$ and $h$ are uniformly and independently distributed over $\mathbb{Z}_q$ due to the random choice of $a$ and $b$, respectively, and $R$ is the unique element of $\mathbb{G}$ such that $((R,s),h) \in \mathbf{R}$. The output of the $\mathsf{Inv}$ algorithm is also correctly distributed, since we can see $s$ as uniformly distributed over $\mathbb{Z}_q$ due to the choice of $r$, and $R$ as the unique element of $\mathbb{G}$ such that $((R,s),h) \in \mathbf{R}$. It can be seen from the construction of the scheme that the $\mathcal{B}eth\text{-}\mathcal{SI}$ scheme is convertible with respect to this family $\mathcal{F}$.

Assumptions. The way the above sampling algorithm works is closely related to the well-known two-parameter attack on textbook ElGamal signatures (see e.g. [MvOV96, p. 455]). Our security result is based on the security of what we call the *hashed-message* ElGamal signature scheme $\mathcal{E}l\mathcal{G}\text{-}\mathcal{SS}$ that is described by the following algorithms:

```
Algorithm $\mathsf{Kg}(1^k)$:
    $(\langle\mathbb{G}\rangle, q, g) \xleftarrow{\$} \mathsf{K}_{\mathrm{cg}}(1^k)$
    $x \xleftarrow{\$} \mathbb{Z}_q$ ; $X \leftarrow g^x$
    $pk \leftarrow (\mathbb{G}, q, g, X)$ ; $sk \leftarrow (\mathbb{G}, q, g, x)$
    Return $(pk, sk)$
```

```
Algorithm $\mathsf{Sign}(sk, M : \mathrm{H})$:
    $r \xleftarrow{\$} \mathbb{Z}_q$ ; $R \leftarrow g^r$
    $s \leftarrow r^{-1}(\mathrm{H}(M) - xR) \bmod q$
    Return $(R, s)$
```

```
Algorithm $\mathsf{Vf}(pk, M, \sigma : \mathrm{H})$:
    If $X^R R^s \equiv g^{\mathrm{H}(M)}$
    then return 1
    else return 0.
```

The only difference with the provably secure Modified ElGamal scheme [PS00] is that the latter includes $R$ in the argument of the hash function.

Universal unforgeability under no-message attack is a (weak) security notion for signature schemes in which the forger $\mathsf{F}$, on input $1^k$, the public key $pk$ and a message $M$, has to come up with a valid signature for $M$, without the help of any signing oracle. We say that a signature scheme $\mathcal{SS} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf})$ associated with $\mathsf{K}_{\mathrm{cg}}$ is universally unforgeable under no-message attacks if, for any polynomial-time forger $\mathsf{F}$ and for any message $M \in \{0,1\}^*$, the advantage of $\mathsf{F}$

$$\mathbf{Adv}^{\text{uuf-nma}_{\mathcal{SS},\mathsf{F}}}(k) = \Pr\left[\ (M, \sigma) \xleftarrow{\$} \mathsf{F}(1^k, pk, M) \ : \ (pk, sk) \xleftarrow{\$} \mathsf{Kg}(1^k) \ ; \ \mathsf{Vf}(pk, M, \sigma) = 1\ \right]$$

is a negligible function in $k$.

Security. The following theorem proves the imp-pa security of the *Beth-SI* scheme based on the universal unforgeability of the *ElG-SS* scheme under no-message attack in the random oracle model. While the *ElG-SS* scheme has never been formally proven secure, we note that no attacks have been found against it either, and that *universal* forgery under *no-message* attack is a very weak security notion for signature schemes.

**Theorem 5.7** The *Beth-SI* scheme associated with prime-order cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ and challenge length $l(\cdot)$ such that $2^{l(k)} < q$ for all $(\langle\mathbb{G}\rangle, q, g) \in [\mathsf{K}_{\mathrm{cg}}(1^k)]$ is imp-pa secure assuming that the hashed-message ElGamal signature scheme associated with $\mathsf{K}_{\mathrm{cg}}$ is universally unforgeable under no-message attacks in the random oracle model. ▌

**Proof:** Given an imp-pa adversary $\mathsf{A} = (\mathsf{CV}, \mathsf{CP})$, we construct a universal forger $\mathsf{F}$ as follows. On input $1^k, pk = (\langle\mathbb{G}\rangle, q, g, X)$, the forger first chooses $r \xleftarrow{\$} \mathbb{Z}_q$, lets $R \leftarrow g^r$ and runs $\mathsf{CV}$ on input $1^k, ((1^k, \langle\mathbb{G}\rangle, q, g, X), \mathrm{H}(M))$. Note that since H is a random oracle, this public key is correctly distributed. It answers $\mathsf{CV}$'s conversation queries by each time choosing $c$ and $z$ at random from $\{0,1\}^{l(k)}$ and $\mathbb{Z}_q$, respectively, computing $Y \leftarrow g^{cH(M)} R^{-z} X^{-cR}$ and returning $(Y, c, z)$ as the transcript. When $\mathsf{CV}$ outputs $St_{\mathsf{CP}}$, the forger runs the cheating prover in a reset experiment as in Lemma 5.1 to get commitment $\tilde{R}, Y$ and responses $z_1, z_2$ to challenges $c_1, c_2$ chosen at random from $\{0,1\}^{l(k)}$. Note that $\tilde{R}$ does not have to be equal to $R$. If the reset experiment is successful (meaning that both responses are valid and $c_1 \neq c_2$), the forger computes $s \leftarrow (c_1 - c_2)^{-1}(z_1 - z_2) \bmod q$ and outputs $(\tilde{R}, s)$ as the signature for $M$. By dividing the two verification equations of the reset experiment, it is easily seen that this is a valid signature for $M$. Due to the Reset Lemma, the imp-pa advantage of $\mathsf{A}$ is bounded by

$$\mathbf{Adv}^{\text{imp-pa}}_{\mathit{Beth\text{-}SI},\mathsf{A}}(k) \leq 2^{-l(k)} + \sqrt{\mathbf{Adv}^{\text{uuf-nma}_{\mathit{ElG\text{-}SS},\mathsf{F}}}(k)}$$

which is negligible for any super-logarithmic function $l(k)$, thereby concluding the proof. ▌

Theorem 4.4 implies that *Beth-IBI* inherits the imp-pa security of *Beth-SI*, and Theorem 4.9 and Corollary 4.10 imply that *Beth-SS* = fs-I-2-S(*Beth-SI*) and *Beth-IBS* = cSS-2-IBS(*Beth-SS*) are uf-cma secure under the same assumptions. The imp-aa and imp-ca security of *Beth-SI* remains an open question.

# 6 Exceptions: Schemes needing Direct Proofs

In this section, we discuss two schemes that escape being captured by our framework, in the sense that they do not seem to originate from a cSI scheme. The first is the *OkDL-IBI* scheme, which was known [Oka93] but never proven secure, the second is the *BNN-IBI* scheme which is new. We prove the security of both schemes as IBI schemes directly, rather than by making use of our framework of transforms.

## 6.1 Definitions and Lemmas

Assumptions. If $\mathbb{G}$ is a cyclic group with generator $g$ and $Y \in \mathbb{G}$ then $\mathrm{dlog}_{\mathbb{G},g}(Y)$ denotes the discrete logarithm of $Y$ to base $g$, namely the unique value $y \in \mathbb{Z}_q$ such that $g^y \equiv Y$, where $q$ is the order of $\mathbb{G}$. Let $\mathsf{K}_{\mathrm{cg}}$ be a cyclic group generator. We say that the discrete logarithm problem associated with $\mathsf{K}_{\mathrm{cg}}$ is hard if, for any polynomial-time algorithm $\mathsf{A}$, the function

$$\mathbf{Adv}^{\mathrm{dlog}}_{\mathsf{K}_{\mathrm{cg}},\mathsf{A}}(k) = \Pr\left[\, \mathsf{A}(1^k, \langle\mathbb{G}\rangle, q, g, Y) = \mathrm{dlog}_{\mathbb{G},g}(Y) \;:\; (\langle\mathbb{G}\rangle, q, g) \xleftarrow{\$} \mathsf{K}_{\mathrm{cg}}(1^k)\,;\, Y \xleftarrow{\$} \mathbb{G} \,\right]$$

is negligible. The one-more discrete logarithm problem associated with $\mathsf{K}_{\mathrm{cg}}$ is defined through the following game:

```
Experiment $\mathbf{Exp}^{\text{1m-dlog}}_{\mathsf{K}_{\mathrm{cg}},\mathsf{A}}(k)$ :
    $(\langle\mathbb{G}\rangle, q, g) \xleftarrow{\$} \mathsf{K}_{\mathrm{cg}}(1^k)$
    $i \leftarrow 0$ ; $n \leftarrow 0$
    $(y_1, \ldots, y_m) \xleftarrow{\$} \mathsf{A}(1^k, \langle\mathbb{G}\rangle, q, g : \textsc{Chall}, \textsc{DLog})$
    If $m = i$ and $n < m$ and $g^{y_i} \equiv Y_i$ for all $i \in \{1, \ldots, m\}$
    Then return 1 else return 0.

Oracle $\textsc{Chall}$:
    $i \leftarrow i + 1$ ; $Y_i \xleftarrow{\$} \mathbb{G}$
    Return $Y_i$

Oracle $\textsc{DLog}(Y)$:
    $n \leftarrow n + 1$ ; $y \leftarrow \mathrm{dlog}_{\mathbb{G},g}(Y)$
    Return $y$
```

The adversary $\mathsf{A}$ is given $1^k, \langle\mathbb{G}\rangle, q, g$ as input and access to two oracles: a challenge oracle $\textsc{Chall}$ that on any input returns a new random target point $Y_i \in \mathbb{Z}_N^*$ and a discrete logarithm oracle $\textsc{DLog}(\cdot) = \mathrm{dlog}_{\mathbb{G},g}(\cdot)$. The adversary's goal is to compute the discrete logarithms of all target points output by the challenge oracle using strictly fewer queries to the inversion oracle. We say that the one-more discrete logarithm problem associated with $\mathsf{K}_{\mathrm{cg}}$ is hard if, for any polynomial-time adversary $\mathsf{A}$, the advantage

$$\mathbf{Adv}^{\text{1m-dlog}}_{\mathsf{K}_{\mathrm{cg}},\mathsf{A}}(k) = \Pr\left[\mathbf{Exp}^{\text{1m-dlog}}_{\mathsf{K}_{\mathrm{cg}},\mathsf{A}}(k) = 1\right]$$

is a negligible function in $k$.

Semi-strong unforgeability. Both the $\mathcal{OKDL}$-$\mathcal{IBI}$ and $\mathcal{BNN}$-$\mathcal{IBI}$ schemes are essentially a zero-knowledge proof of knowledge of a standard signature on the user's identity. However, standard uf-cma security of the underlying SS scheme does not seem to be sufficient to prove the security of the IBI scheme. The notion of *strong unforgeability* [BN00, ADR02, SPMLS02] (referred to as *non-malleability* in [SPMLS02]) would be sufficient for our purposes, but unfortunately the SS schemes in question do not satisfy it. Therefore, we introduce a new notion that we call *semi-strong unforgeability* (ss-cma), which is related to strong unforgeability, but is tailored to SS schemes that are obtained as the $\mathsf{fs\text{-}I\text{-}2\text{-}S}$ transform of a canonical SI scheme. Essentially, the security notion requires that signatures be strongly unforgeable in their first component (the commitment): after seeing a signature $\mathit{Cmt}\|\mathit{Rsp}$ on message $M$, it should be hard to find a second valid signature $\mathit{Cmt}'\|\mathit{Rsp}$ on $M$, where $\mathit{Cmt}' \neq \mathit{Cmt}$.

Let $\mathcal{SI}$ be a canonical SI scheme, and let $\mathcal{SS} = (\mathsf{Kg}, \mathsf{Sign}, \mathsf{Vf}) = \mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathcal{SI})$ be the corresponding SS scheme as per Construction 4.8. Consider the following experiment:

```
Experiment $\mathbf{Exp}^{\text{ss-cma}}_{\mathcal{SS},\mathsf{F}}(k)$ :
    $(pk, sk) \xleftarrow{\$} \mathsf{Kg}(1^k)$ ; $n \leftarrow 0$
    $(M, \mathit{Cmt}\|\mathit{Rsp}) \xleftarrow{\$} \mathsf{F}(pk : \textsc{Sign})$
    If $\mathsf{Vf}(pk, M, \mathit{Cmt}\|\mathit{Rsp}) = 1$
    and $\nexists\, i \in \{1, \ldots n\}$ : $(M, \mathit{Cmt}) = (M_i, \mathit{Cmt}_i)$
    then return 1 else return 0.

Oracle $\textsc{Sign}(M)$:
    $n \leftarrow n + 1$ ; $M_n \leftarrow M$
    $\mathit{Cmt}_n\|\mathit{Rsp}_n \xleftarrow{\$} \mathsf{Sign}(sk, M)$
    Return $\mathit{Cmt}_n\|\mathit{Rsp}_n$
```

The scheme $\mathcal{SS}$ is said to be ss-cma secure if for any polynomial-time adversary $\mathsf{F}$ the advantage

$$\mathbf{Adv}^{\text{ss-cma}}_{\mathcal{SS},\mathsf{F}}(k) = \Pr\left[\mathbf{Exp}^{\text{ss-cma}}_{\mathcal{SS},\mathsf{F}}(k) = 1\right]$$

is a negligible function in $k$.

**Lemma 6.1** Let $\mathcal{SI}$ be a non-trivial canonical SI scheme, and let $\mathcal{SS} = \mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathcal{SI})$ as defined by Construction 4.8. If $\mathcal{SI}$ is imp-pa secure, then $\mathcal{SS}$ is ss-cma secure in the random oracle model. ▌

**Proof Sketch:** The description of algorithm $\mathsf{A}$ is identical to the impersonator described in the proof of Lemma 3.5 of [AABN02] (of which our Theorem 4.9 is a special case). In a nutshell, $\mathsf{A}$ uses the

```
Algorithm $\mathsf{MKg}(1^k)$
    $(\langle\mathbb{G}\rangle, q, g_1) \xleftarrow{\$} \mathsf{K}_{\mathrm{cg}}(1^k)$
    $\alpha \xleftarrow{\$} \mathbb{Z}_q$ ; $g_2 \leftarrow g_1^{\alpha}$
    $x_1, x_2 \xleftarrow{\$} \mathbb{Z}_q$ ; $X \leftarrow g_1^{-x_1} g_2^{-x_2}$
    $mpk \leftarrow (1^k, \langle\mathbb{G}\rangle, q, g_1, g_2, X)$
    $msk \leftarrow (\langle\mathbb{G}\rangle, q, g_1, g_2, x_1, x_2)$
    Return $(mpk, msk)$
```

```
Algorithm $\mathsf{UKg}(msk, I : \mathrm{H})$
    $(\langle\mathbb{G}\rangle, q, g_1, g_2, x_1, x_2) \leftarrow msk$
    $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_q$ ; $R \leftarrow g_1^{r_1} g_2^{r_2}$
    $s_1 \leftarrow -r_1 - \mathrm{H}(R\|I) \cdot x_1 \bmod q$
    $s_2 \leftarrow -r_2 - \mathrm{H}(R\|I) \cdot x_2 \bmod q$
    $usk \leftarrow (\langle\mathbb{G}\rangle, q, g_1, g_2, R, s_1, s_2)$
    Return $usk$
```

```
Prover $\overline{\mathsf{P}}$                                             Verifier $\overline{\mathsf{V}}$
$S \leftarrow g_1^{-s_1} g_2^{-s_2}$
$y_1, y_2 \xleftarrow{\$} \mathbb{Z}_q$ ; $Y \leftarrow g_1^{y_1} g_2^{y_2}$    $\xrightarrow{R, S, Y}$
                                            $\xleftarrow{c}$       $c \xleftarrow{\$} \mathbb{Z}_{2^{l(k)}}$
$z_1 \leftarrow y_1 + cs_1 \bmod q$
$z_2 \leftarrow y_2 + cs_2 \bmod q$                  $\xrightarrow{z_1, z_2}$     If $Y \equiv g_1^{z_1} g_2^{z_2} S^c$ and $R \equiv SX^{\mathrm{H}(R\|I)}$
                                                         and $h \in \{0,1\}^{l(k)}$and $S, Y \in \mathbb{G}$
                                                         and $z_1, z_2 \in \mathbb{Z}_q$ then acc else rej
```

Figure 16: **The *OkDL-IBI* scheme.** The scheme is parameterized by super-logarithmic challenge length $l : \mathbb{N} \to \mathbb{N}$, a random oracle $\mathrm{H} : \{0,1\}^* \to 0, \underline{1}^{\ell}(k)$ and a prime-order cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ such that $2^{l(k)} < q$ for all $(\langle\mathbb{G}\rangle, q, g) \in [\mathsf{K}_{\mathrm{cg}}(1^k)]$. The prover $\overline{\mathsf{P}}$ and verifier $\overline{\mathsf{V}}$ are run on initial states $usk = (\langle\mathbb{G}\rangle, q, g_1, g_2, R, s_1, s_2)$ and $(mpk, I)$ where $mpk = (1^k, \langle\mathbb{G}\rangle, q, g_1, g_2, X)$, respectively.

forger $\mathsf{F}$ as a subroutine to impersonate itself as a prover to an honest verifier $\mathsf{V}$ as follows. Algorithm $\mathsf{A}$ uses its conversation oracle to reply to $\mathsf{F}$'s signing and hash queries, except for one hash query $\mathrm{H}(Cmt\|M)$ that it guesses to be the "crucial" query that $\mathsf{F}$ will use later in its forgery. When this query occurs, $\mathsf{A}$ sends $Cmt$ as the first move of its identification to $\mathsf{V}$, and returns the challenge it received from $\mathsf{V}$ as the response to $\mathsf{F}$'s hash query. If at the end $\mathsf{F}$ indeed outputs a valid forgery $Cmt\|Rsp$ for message $M$, then $\mathsf{A}$ successfully completes the identification protocol by sending $Rsp$ as the response to $\mathsf{V}$.

It is important that when the crucial hash query occurs, $\mathsf{A}$ is still free to program the hash value that will be returned to $\mathsf{F}$ for $Cmt\|M$. We can assume without loss of generality that $\mathsf{F}$ never queries the hash oracle on the same argument twice, but the hash value might also have been fixed by a previous signature query for message $M$. At this point in the proof, [AABN02] exploits the fact that $\mathsf{F}$ is not allowed to make such query if it later wants to forge a signature on $M$. Here, we observe here that even if $\mathsf{F}$ retrieved a signature $Cmt_i\|Rsp_i$ for message $M$ from the signing oracle before, then the value of $\mathrm{H}(Cmt_i\|M)$ is still undecided as long as $Cmt_i \neq Cmt$, and this is exactly what is enforced by our definition of semi-strong unforgeability. The rest of the analysis is the same as in [AABN02], resulting in an almost identical reduction equation. ▌

### 6.2 The *OkDL-IBI* and *OkDL-IBS* Schemes

The scheme. Figure 16 depicts the *OkDL-IBI* scheme associated to cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ and challenge length $l(\cdot)$. The security of the scheme is equivalent to that of an IBI scheme presented in [Oka93]. (The scheme in [Oka93] actually uses shorter user secret keys by including $h = \mathrm{H}(R\|I)$ instead of $R$ in *usk*.) This is the only IBI scheme we found in the literature that is *not* based on a convertible SI scheme. At first sight one may think it is convertible based on a relation containing tuples $((x_1, x_2), X = g_1^{x_1} g_2^{x_2})$, but there does not seem to exist any trapdoor information allowing to

```
Algorithm $\mathsf{Kg}(1^k)$
    $(\langle\mathbb{G}\rangle, q, g_1) \xleftarrow{\$} \mathsf{K}_{\mathrm{cg}}(1^k)$
    $\alpha \xleftarrow{\$} \mathbb{Z}_q$ ; $g_2 \leftarrow g_1^{\alpha}$
    $x_1, x_2 \xleftarrow{\$} \mathbb{Z}_q$ ; $X \leftarrow g_1^{-x_1} g_2^{-x_2}$
    $pk \leftarrow (1^k, \langle\mathbb{G}\rangle, q, g_1, g_2, X)$
    $sk \leftarrow (\langle\mathbb{G}\rangle, q, g_1, g_2, x_1, x_2)$
    Return $(pk, sk)$
```

```
Prover P                                        Verifier V
$y_1, y_2 \xleftarrow{\$} \mathbb{Z}_q$
$Y \leftarrow g_1^{y_1} g_2^{y_2}$        $\xrightarrow{Y}$
                                 $\xleftarrow{c}$          $c \xleftarrow{\$} \mathbb{Z}_{2^{l(k)}}$
$z_1 \leftarrow y_1 + cx_1 \bmod q$
$z_2 \leftarrow y_2 + cx_2 \bmod q$     $\xrightarrow{z_1, z_2}$
                                                If $Y \equiv g_1^{z_1} g_2^{z_2} X^c$
                                                and $Y \in \mathbb{G}$ and $z_1, z_2 \in \mathbb{Z}_q$
                                                then acc else rej
```

Figure 17: **The *OkCL-SI* scheme.** The scheme is parameterized with cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ and super-logarithmic challenge length $l : \mathbb{N} \to \mathbb{N}$ such that $2^{l(k)} < q$ for all $q$ output by $\mathsf{K}_{\mathrm{cg}}(1^k)$. The prover $\mathsf{P}$ and verifier $\mathsf{V}$ are run on initial states $sk = (\langle\mathbb{G}\rangle, q, g_1, g_2, x_1, x_2)$ and $pk = (1^k, \langle\mathbb{G}\rangle, q, g_1, g_2, X)$, respectively.

invert this relation.

Security. No security proof for this scheme was provided in [Oka93]. However, here we prove it imp-ca secure in the random oracle model under the assumption that the discrete logarithm problem associated to $\mathsf{K}_{\mathrm{cg}}$ is hard.

**Theorem 6.2** The *OkDL-IBI* scheme associated to prime-order cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ and super-logarithmic challenge length $l : \mathbb{N} \to \mathbb{N}$ such that $2^{l(k)} < q$ for all $(\langle\mathbb{G}\rangle, q, g) \in [\mathsf{K}_{\mathrm{cg}}(1^k)]$ is imp-ca secure in the random oracle model if the discrete logarithm problem associated with the underlying generator $\mathsf{K}_{\mathrm{cg}}$ is hard. ▌

**Proof:** A user's secret key in the *OkDL-IBI* scheme is essentially a signature of his identity under a signature scheme that is commonly known as the (classical) Okamoto signature scheme [Oka93], referred to as the *OkCL-SS* scheme here. This scheme is the fs-I-2-S transform of the *OkCL-SI* scheme associated to cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ and challenge length $l(\cdot)$ as depicted in Figure 17. (Note that the *OkCL-SI* and *OkCL-SS* schemes are not known to be convertible, so the corresponding IBI and IBS schemes are not defined.)

The *OkCL-SI* scheme with super-logarithmic challenge length $l(\cdot)$ is known to be imp-pa secure if the discrete logarithm problem associated to the underlying generator $\mathsf{K}_{\mathrm{cg}}$ is hard [Oka93]. Since it is also a non-trivial canonical SI scheme, Lemma 6.1 implies that the *OkCL-SS* = fs-I-2-S(*OkCL-SI*) scheme is semi-strongly unforgeable under the same assumption.

The idea of the proof is to distinguish between two types of attacks, based on the values for $R, S$ that the adversary sends in the first move of the impersonation. The first type considers those attacks where $R, S$ are equal to the values that the experiment used in previous transcripts of (or interactions with) the user under attack $I_{\mathrm{b}}$. Using the Reset Lemma, we show that this type of attacks can be transformed into an algorithm for computing $\mathrm{dlog}_{\mathbb{G},g_1}(g_2)$. The second type of attacks are those where the $R, S$ used in the impersonation are different from the values previously used for $I_{\mathrm{b}}$. Again using the Reset Lemma, we show how this type of attacks can be transformed into an algorithm computing semi-strong forgeries for the *OkCL-SS* scheme, which we know is infeasible under the discrete logarithm assumption.

Given a polynomial-time impersonator $\overline{\mathsf{A}}$ breaking $\mathcal{OkDL}\text{-}\mathcal{IBI}$ in a concurrent attack, we construct a discrete logarithm algorithm $\mathsf{B}$ and a forger $\mathsf{F}$ such that

$$\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{OkDL}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) \leq \sqrt{\mathbf{Adv}^{\text{dlog}}_{\mathsf{K}_{\text{cg}},\mathsf{B}}(k)} + \sqrt{\mathbf{Adv}^{\text{ss-cma}}_{\mathcal{OkCL}\text{-}\mathcal{SS},\mathsf{F}}(k)} + 3 \cdot 2^{-l(k)/2} \ . \tag{2}$$

Since $l(\cdot)$ is super-logarithmic and the discrete logarithm problem associated to $\mathsf{K}_{\text{cg}}$ is hard, all terms on the right-hand side are negligible, and the theorem follows.

The discrete logarithm algorithm $\mathsf{B}$ operates as follows. Given an imp-ca adversary $\overline{\mathsf{A}} = (\overline{\mathsf{CV}}, \overline{\mathsf{CP}})$ and input $(1^k, \langle\mathbb{G}\rangle, q, g_1, g_2)$, the algorithm $\mathsf{B}$ chooses $x_1, x_2$ at random from $\mathbb{Z}_q$, computes $X \leftarrow g_1^{-x_1} g_2^{-x_2}$ and runs $\overline{\mathsf{CV}}$ on input $1^k, mpk = (1^k, \langle\mathbb{G}\rangle, q, g_1, g_2, X)$. It answers all $\overline{\mathsf{CV}}$'s oracle queries by running the real algorithms of the $\mathcal{OkDL}\text{-}\mathcal{IBI}$ scheme. (It is able to do so since it knows the master secret key $msk = (1^k, \langle\mathbb{G}\rangle, q, g_1, g_2, x_1, x_2)$, and storing the last three components of the user secret key it generates for each identity $I$ as $(R_I, s_{1,I}, s_{2,I})$.) At the end of its execution, $\overline{\mathsf{CV}}$ outputs the identity $I_{\text{b}}$ that will be attacked, together with state information $St_{\overline{\mathsf{CP}}}$ for the cheating prover. For ease of notation, we let $(\tilde{R}, \tilde{s}_1, \tilde{s}_2)$ denote the components $(R_{I_{\text{b}}}, s_{1,I_{\text{b}}}, s_{2,I_{\text{b}}})$ of the user secret key that $\mathsf{B}$ stored for identity $I_{\text{b}}$, and we let $\tilde{S} \leftarrow g_1^{-\tilde{s}_1} g_2^{-\tilde{s}_2}$.

Define a verifier algorithm $\overline{\mathsf{V}}'$ that, on initial state $(mpk, I)$, only accepts a conversation $R\|S\|Y\|c\|z_1\| z_2$ if $\overline{\mathsf{V}}$ accepts the conversation on the same initial state and moreover $R = \tilde{R}$. Let $\text{acc}'(St_{\overline{\mathsf{CP}}}, (mpk, I_{\text{b}}))$ be the probability that $\overline{\mathsf{V}}'$ accepts on initial state $(mpk, I_{\text{b}})$ after interacting with $\overline{\mathsf{CP}}$ initialized with $St_{\overline{\mathsf{CP}}}$. Then, by the Reset Lemma (Lemma 5.1), $\mathsf{B}$ can extract two such accepting conversations $R\|S\|Y\|c_1\|z_{11}\|z_{21}$ and $R\|S\|Y\|c_2\|z_{12}\|z_{22}$ with $c_1 \neq c_2$ with probability

$$\text{res}'(St_{\overline{\mathsf{CP}}}, (mpk, I_{\text{b}})) \geq \left(\text{acc}'(St_{\overline{\mathsf{CP}}}, (mpk, I_{\text{b}})) - 2^{-l(k)}\right)^2 \ .$$

Since $R = \tilde{R}$ and the two conversation transcripts are valid, we also have that $S = \tilde{S} \equiv RX^{-\text{H}(R\|I_{\text{b}})}$. Algorithm $\mathsf{B}$ extracts $(s_1, s_2)$ such that $S \equiv g_1^{-s_1} g_2^{-s_2}$ as

$$\begin{aligned} s_1 &\leftarrow (z_{11} - z_{12})/(c_1 - c_2) \bmod q \\ s_2 &\leftarrow (z_{21} - z_{22})/(c_1 - c_2) \bmod q \ . \end{aligned}$$

Since $\overline{\mathsf{A}}$'s view is independent of $\mathsf{B}$'s choice of $\tilde{s}_1, \tilde{s}_2$, with probability $1 - 1/q \geq 1 - 2^{-l(k)}$ we have that $(s_1, s_2) \neq (\tilde{s}_1, \tilde{s}_2)$. From these, $\mathsf{B}$ computes the discrete logarithm of $g_2$ relative to $g_1$ as $-(s_1 - \tilde{s}_1)/(s_2 - \tilde{s}_2) \bmod q$. It is easy to see that the simulation of $\overline{\mathsf{CV}}$'s and $\overline{\mathsf{CP}}$'s environment is perfect, since the same algorithms were used as in a real attack against $\mathcal{OkDL}\text{-}\mathcal{IBI}$. The advantage of $\mathsf{B}$ can be lower bounded by:

$$\begin{aligned} \mathbf{Adv}^{\text{dlog}}_{\mathsf{K}_{\text{cg}},\mathsf{B}}(k) &\geq (1 - 2^{-l(k)}) \cdot \text{res}'(St_{\overline{\mathsf{CP}}}, (mpk, I_{\text{b}})) \\ &\geq \left(\text{acc}'(St_{\overline{\mathsf{CP}}}, (mpk, I_{\text{b}})) - 2^{-l(k)}\right)^2 - 2^{-l(k)} \end{aligned} \tag{3}$$

Now we define the forger $\mathsf{F}$ breaking the semi-strong unforgeability of $\mathcal{OkCL}\text{-}\mathcal{SS}$. Given an imp-ca adversary $\overline{\mathsf{A}} = (\overline{\mathsf{CV}}, \overline{\mathsf{CP}})$, input $1^k, pk = (1^k, \langle\mathbb{G}\rangle, q, g_1, g_2, X)$ and oracle access to a signing oracle $\textsc{Sign}(\cdot)$ and random oracle $\text{H}(\cdot)$, $\mathsf{F}$ first initiates sets $HU$, $CU$ and $PS$ to $\emptyset$. It then runs $\overline{\mathsf{CV}}(1^k, mpk = (1^k, \langle\mathbb{G}\rangle, q, g_1, g_2, X) : \textsc{Init-sim}, \textsc{Corr-sim}, \textsc{Prov-sim}, \text{H})$, simulating $\overline{\mathsf{CV}}$'s oracles as indicated in Figure 18, until it outputs $(St_{\overline{\mathsf{CP}}}, I_{\text{b}})$. For ease of notation, we again denote $(R_{I_{\text{b}}}, s_{1,I_{\text{b}}}, s_{2,I_{\text{b}}})$ as $(\tilde{R}, \tilde{s}_1, \tilde{s}_2)$, and we let $\tilde{S} \leftarrow g_1^{-\tilde{s}_1} g_2^{-\tilde{s}_2}$. Algorithm $\mathsf{F}$ also updates $HU \leftarrow HU \setminus \{I_{\text{b}}\}$ and $CU \leftarrow CU \cup \{I_{\text{b}}\}$.

```
Oracle $\textsc{Init-sim}(I)$
    If $I \in CU \cup HU$ then return $\perp$
    $R_I \| z_{1,I} \| z_{2,I} \leftarrow \textsc{Sign}(I)$
    $s_{1,I} \leftarrow -z_{1,I} \bmod q$ ; $s_{2,I} \leftarrow -z_{2,I} \bmod q$
    $HU \leftarrow HU \cup \{I\}$
    Return 1
```

```
Oracle $\textsc{Corr-sim}(I)$
    If $I \notin HU$ then return $\perp$
    $CU \leftarrow CU \cup \{I\}$ ; $HU \leftarrow HU \setminus \{I\}$
    Return $(\langle \mathbb{G} \rangle, q, g_1, g_2, R_I, s_{1,I}, s_{2,I})$
```

```
Oracle $\textsc{Prov-sim}(I, s, M_{\rm in})$
    If $I \notin HU$ then return $\perp$
    If $(I, s) \notin PS$ then
        $PS \leftarrow PS \cup \{(I, s)\}$
        Pick random coins $\rho$ for $\overline{\mathsf{P}}$
        $St_{\overline{\mathsf{P}}}[I, s] \leftarrow (((\langle \mathbb{G} \rangle, q, g_1, g_2, R_I, s_{1,I}, s_{2,I}), \rho)$
    $(M_{\rm out}, St_{\overline{\mathsf{P}}}[I, s]) \leftarrow \overline{\mathsf{P}}(M_{\rm in}, St_{\overline{\mathsf{P}}}[I, s])$
    Return $M_{\rm out}$
```

Figure 18: Subroutines used by forger $\mathsf{F}$ to simulate oracle queries of $\overline{\mathsf{CV}}$ and $\overline{\mathsf{CP}}$ in the proof of Theorem 6.2.

Define a verifier algorithm $\overline{\mathsf{V}}''$ that, on initial state $(mpk, I)$, only accepts a conversation $R\|S\|Y\|c\|z_1\|z_2$ if $\overline{\mathsf{V}}$ accepts the conversation on the same initial state and moreover $R \neq \tilde{R}$. Let $\mathrm{acc}''(St_{\overline{\mathsf{CP}}}, (mpk, I_{\rm b}))$ be the probability that $\overline{\mathsf{V}}''$ accepts on input $(mpk, I_{\rm b})$ after interacting with $\overline{\mathsf{CP}}$ initialized with $St_{\overline{\mathsf{CP}}}$ (granting $\overline{\mathsf{CP}}$ access to the oracles depicted in Figure 18). Then by the Reset Lemma, $\mathsf{F}$ can extract $\mathsf{B}$ can extract two such accepting conversations $R\|S\|Y\|c_1\|z_{11}\|z_{21}$ and $R\|S\|Y\|c_2\|z_{12}\|z_{22}$ with $c_1 \neq c_2$ with probability

$$\mathrm{res}''(St_{\overline{\mathsf{CP}}}, (mpk, I_{\rm b})) \geq \left(\mathrm{acc}''(St_{\overline{\mathsf{CP}}}, (mpk, I_{\rm b})) - 2^{-l(k)}\right)^2 .$$

From these conversations, $\mathsf{F}$ extracts $(s_1, s_2)$ such that $S \equiv g_1^{-s_1} g_2^{-s_2}$ as

$$\begin{aligned} s_1 &\leftarrow (z_{11} - z_{12})/(c_1 - c_2) \bmod q \\ s_2 &\leftarrow (z_{21} - z_{22})/(c_1 - c_2) \bmod q . \end{aligned}$$

Since $R \equiv g_1^{-s_1} g_2^{-s_2} X^{\mathrm{H}(R\|I_{\rm b})}$, the string $\sigma = R\| -s_1 \bmod q \| -s_2 \bmod q$ is a valid $\mathcal{OKCL}\text{-}\mathcal{SS}$ signature for message $I_{\rm b}$. The only signature for message $I_{\rm b}$ output by $\mathsf{F}$'s $\textsc{Sign}$ oracle is $\tilde{R}\|, -\tilde{s}_1 \bmod q \| -\tilde{s}_2 \bmod q$, so since $R \neq \tilde{R}$, signature $\sigma$ is a valid semi-strong forgery. Algorithm $\mathsf{F}$ halts and outputs $(I_{\rm b}, \sigma)$.

The simulation of $\overline{\mathsf{CV}}$'s and $\overline{\mathsf{CP}}$'s environment is perfect, since the same algorithms were used as in a real attack against $\mathcal{OKDL}\text{-}\mathcal{IBI}$. The ss-cma advantage of $\mathsf{F}$ is lower bounded by

$$\begin{aligned} \mathbf{Adv}^{\text{ss-cma}}_{\mathcal{OKCL}\text{-}\mathcal{SS},\mathsf{F}}(k) &\geq \mathrm{res}''(St_{\overline{\mathsf{CP}}}, (mpk, I_{\rm b})) \\ &\geq \left(\mathrm{acc}''(St_{\overline{\mathsf{CP}}}, (mpk, I_{\rm b})) - 2^{-l(k)}\right)^2 . \end{aligned} \tag{4}$$

Now let $\mathbf{E}$ be the event that $\overline{\mathsf{CP}}$ sends $R \neq \tilde{R}$ as part of the first move of its impersonation attempt, where $\tilde{R}$ is the value that was returned to it as part of the first message in $\overline{\mathsf{CV}}$'s previous interactions with identity $I_{\rm b}$ through the $\textsc{Prov}$ oracle. (If $\overline{\mathsf{CV}}$ didn't interact with $I_{\rm b}$, we can make it do a dummy interaction.) Using the notation

$$\overline{\mathsf{V}}(\cdot, (mpk, I_{\rm b})) \text{ accepts } \overline{\mathsf{CP}}(\varepsilon, St_{\overline{\mathsf{CP}}})$$

```
Algorithm $\mathsf{MKg}(1^k)$
    $(\langle\mathbb{G}\rangle, q, g) \xleftarrow{\$} \mathsf{K}_{\mathrm{cg}}(1^k)$
    $x \xleftarrow{\$} \mathbb{Z}_q$ ; $X \leftarrow g^x$
    $mpk \leftarrow (1^k, \langle\mathbb{G}\rangle, q, g, X)$
    $msk \leftarrow (\langle\mathbb{G}\rangle, q, g, x)$
    Return $(mpk, msk)$
```

```
Algorithm $\mathsf{UKg}(msk, I : \mathrm{H})$
    $(\langle\mathbb{G}\rangle, q, g, x) \leftarrow msk$
    $r \xleftarrow{\$} \mathbb{Z}_q$ ; $R \leftarrow g^r$
    $s \leftarrow r + \mathrm{H}(R\|I) \cdot x \bmod q$
    $usk \leftarrow (\mathbb{G}, q, g, R, s)$
    Return $usk$
```

| Prover $\overline{\mathsf{P}}$ | | Verifier $\overline{\mathsf{V}}$ |
|---|---|---|
| $S \leftarrow g^s$ ; $y \xleftarrow{\$} \mathbb{Z}_q$ ; $Y \leftarrow g^y$ | $\xrightarrow{R,S,Y}$ | |
| | $\xleftarrow{c}$ | $c \xleftarrow{\$} \mathbb{Z}_{2^{l(k)}}$ |
| $z \leftarrow y + cs \bmod q$ | $\xrightarrow{z}$ | |
| | | If $g^z \equiv YS^c$ and $S \equiv RX^{\mathrm{H}(R\|I)}$<br>and $S, Y \in \mathbb{G}$ and $z_1, z_2 \in \mathbb{Z}_q$<br>then `acc` else `rej` |

Figure 19: **The $\mathcal{BNN}\text{-}\mathcal{IBI}$ scheme.** The scheme is parameterized by super-logarithmic challenge length $l : \mathbb{N} \to \mathbb{N}$, a random oracle $\mathrm{H} : \{0,1\}^* \to \{0,1\}^{\ell(k)}$ and a prime-order cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ such that $2^{l(k)} < q$ for all $(\langle\mathbb{G}\rangle, q, g) \in [\mathsf{K}_{\mathrm{cg}}(1^k)]$. The prover $\overline{\mathsf{P}}$ and verifier $\overline{\mathsf{V}}$ are run on initial states $usk = (\langle\mathbb{G}\rangle, q, g, R, s)$ and $(mpk, I)$ where $mpk = (1^k, \langle\mathbb{G}\rangle, q, g, X)$, respectively.

as shorthand for the event that algorithm $\overline{\mathsf{V}}$, when initialized with state $(mpk, I_{\mathrm{b}})$, accepts after interacting with $\overline{\mathsf{CP}}$ initialized with $St_{\overline{\mathsf{CP}}}$, we can upper bound the advantage of $\overline{\mathsf{A}}$ as

$$\begin{aligned}
\mathbf{Adv}^{\text{imp-ca}}_{\mathcal{OKDL}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) &= \Pr\left[\overline{\mathsf{V}}(\cdot,(mpk,I_{\mathrm{b}})) \text{ accepts } \overline{\mathsf{CP}}(\varepsilon, St_{\overline{\mathsf{CP}}})\right] \\
&= \Pr\left[\overline{\mathsf{V}}(\cdot,(mpk,I_{\mathrm{b}})) \text{ accepts } \overline{\mathsf{CP}}(\varepsilon, St_{\overline{\mathsf{CP}}}) \wedge \mathbf{E}\right] + \Pr\left[\overline{\mathsf{V}}(\cdot,(mpk,I_{\mathrm{b}})) \text{ accepts } \overline{\mathsf{CP}}(\varepsilon, St_{\overline{\mathsf{CP}}}) \wedge \neg\mathbf{E}\right] \\
&\leq \mathrm{acc}'(St_{\overline{\mathsf{CP}}}, (mpk, I_{\mathrm{b}})) + \mathrm{acc}''(St_{\overline{\mathsf{CP}}}, (mpk, I_{\mathrm{b}})) \\
&\leq \sqrt{\mathbf{Adv}^{\mathrm{dlog}}_{\mathsf{K}_{\mathrm{cg}},\mathsf{B}}(k) + 2^{-l(k)}} + 2^{-l(k)} + \sqrt{\mathbf{Adv}^{\text{ss-cma}}_{\mathcal{OKCL}\text{-}\mathcal{SS},\mathsf{F}}(k)} + 2^{-l(k)} \\
&\leq \sqrt{\mathbf{Adv}^{\mathrm{dlog}}_{\mathsf{K}_{\mathrm{cg}},\mathsf{B}}(k)} + \sqrt{\mathbf{Adv}^{\text{ss-cma}}_{\mathcal{OKCL}\text{-}\mathcal{SS},\mathsf{F}}} + 3 \cdot 2^{-l(k)/2}
\end{aligned}$$

which is exactly Equation (2), thereby proving the theorem. In the one but last step above, we used Equations 3 and 4. In the last step, we used the fact that $\sqrt{x+y} \leq \sqrt{x} + \sqrt{y}$ for all $x, y \geq 0$, and the fact that $2^{-l(k)} \leq 2^{-l(k)/2}$ for $l(k) \geq 0$. ▌

As already noted in Section 4.5, the uf-cma security of the IBS scheme obtained as $\mathsf{fs\text{-}I\text{-}2\text{-}S}(\mathcal{OKDL}\text{-}\mathcal{IBI})$ scheme is *not* implied by Corollary 4.10 since it does not originate from a convertible SI scheme. However, since $\mathcal{OKDL}\text{-}\mathcal{IBI}$ is easily seen to be canonical and non-trivial, Theorem 4.13 implies that $\mathcal{OKDL}\text{-}\mathcal{IBS} = \mathsf{efs\text{-}IBI\text{-}2\text{-}IBS}(\mathcal{OKDL}\text{-}\mathcal{IBI})$ scheme is uf-cma secure in the random oracle model under the discrete logarithm assumption associated to $\mathsf{K}_{\mathrm{cg}}$.

### 6.3 The $\mathcal{BNN}\text{-}\mathcal{IBI}$ and $\mathcal{BNN}\text{-}\mathcal{IBS}$ Schemes

In Figure 19, we introduce a new IBI scheme associated to any prime-order cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ and challenge length $l(\cdot)$ called $\mathcal{BNN}\text{-}\mathcal{IBI}$. The scheme can be viewed as the single-generator variant of the $\mathcal{OKDL}\text{-}\mathcal{IBI}$ scheme.

Security. Just like the $\mathcal{OKDL}\text{-}\mathcal{IBI}$ scheme, the $\mathcal{BNN}\text{-}\mathcal{IBI}$ scheme does not seem to originate from a cSI scheme by lack of an appropriate trapdoor, so we have to prove its security directly as an IBI

```
Algorithm $\mathsf{Kg}(1^k)$
    $(\langle\mathbb{G}\rangle, q, g) \xleftarrow{\$} \mathsf{K}_{\mathrm{cg}}(1^k)$
    $x \xleftarrow{\$} \mathbb{Z}_q$ ; $X \leftarrow g^x$
    $pk \leftarrow (\langle\mathbb{G}\rangle, q, g, X)$
    $sk \leftarrow (\langle\mathbb{G}\rangle, q, g, x)$
    Return $(pk, sk)$
```

$$\begin{array}{lcl} \underline{\text{Prover}}\ \mathsf{P} & & \underline{\text{Verifier}}\ \mathsf{V} \\ y \xleftarrow{\$} \mathbb{Z}_q & & \\ Y \leftarrow g^y & \xrightarrow{\quad Y \quad} & \\ & \xleftarrow{\quad c \quad} & c \xleftarrow{\$} \mathbb{Z}_{2^{l(k)}} \\ z \leftarrow y + cx \bmod q & \xrightarrow{\quad z \quad} & \\ & & \text{If } g^z \equiv YX^c \\ & & \text{and } Y \in \mathbb{G} \text{ and } z \in \mathbb{Z}_q \\ & & \text{then } \mathtt{acc} \text{ else } \mathtt{rej} \end{array}$$

Figure 20: **The *Schnorr-SI* scheme.** The scheme is parameterized by prime-order cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ and super-logarithmic challenge length $l : \mathbb{N} \to \mathbb{N}$ such that $2^{l(k)} < q$ for all $(\langle\mathbb{G}\rangle, q, g) \in [\mathsf{K}_{\mathrm{cg}}(1^k)]$. The prover $\mathsf{P}$ and verifier $\mathsf{V}$ are run on initial states $sk = (\langle\mathbb{G}\rangle, q, g, x)$ and $pk = (\langle\mathbb{G}\rangle, q, g, X)$, respectively.

scheme.

**Theorem 6.3** The *BNN-IBI* scheme associated with super-logarithmic challenge length $l(\cdot)$ and prime-order cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ such that $2^{l(k)} < q$ for all $(\langle\mathbb{G}\rangle, q, g) \in [\mathsf{K}_{\mathrm{cg}}(1^k)]$ is imp-pa secure in the random oracle model if the discrete logarithm problem associated with $\mathsf{K}_{\mathrm{cg}}$ is hard. ▮

**Proof:** The user secret key of the *BNN-IBI* scheme is actually a Schnorr signature [Sch90] on the user's identity. The *Schnorr-SI* scheme associated with prime-order cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ and challenge length $l(\cdot)$ is depicted in Figure 20. The Schnorr signature scheme is referred to as *Schnorr-SS* = fs-I-2-S(*Schnorr-SI*) here.

The *Schnorr-SI* scheme is a non-trivial canonical SI scheme, and is imp-pa secure under the discrete logarithm assumption associated with $\mathsf{K}_{\mathrm{cg}}$ when instantiated with super-logarithmic challenge length $l(\cdot)$ such that $2^{l(k)} < q$ for all $(\langle\mathbb{G}\rangle, q, g) \in [\mathsf{K}_{\mathrm{cg}}(1^k)]$ [Sch90]. By Lemma 6.1, the *Schnorr-SS* scheme is semi-strongly unforgeable under chosen-message attack under the same assumptions.

We prove the theorem by showing that if there exists a polynomial-time imp-pa impersonator $\overline{\mathsf{A}} = (\overline{\mathsf{CP}}, \overline{\mathsf{CV}})$ attacking *BNN-IBI*, then there exist a discrete logarithm algorithm $\mathsf{B}$ and a forger algorithm $\mathsf{F}$ such that

$$\mathbf{Adv}^{\text{imp-pa}}_{\mathit{BNN\text{-}IBI},\overline{\mathsf{A}}}(k) \;\le\; \sqrt{\boldsymbol{Q}^{\textsc{Init}}_{\overline{\mathsf{CV}}}(k) \cdot \mathbf{Adv}^{\mathrm{dlog}}_{\mathsf{K}_{\mathrm{cg}},\mathsf{B}}(k)} + \sqrt{\mathbf{Adv}^{\text{ss-cma}}_{\mathit{Schnorr\text{-}SS},\mathsf{F}}(k)} + 2 \cdot 2^{-l(k)} \;. \tag{5}$$

The latter is a straightforward adaptation of algorithm $\mathsf{F}$ in the proof of Theorem 6.2, the former requires a bit more explanation.

On input $(1^k, \langle\mathbb{G}\rangle, q, g, \tilde{S})$, algorithm $\mathsf{B}$ computes $\tilde{s} = \mathrm{dlog}_{\mathbb{G},g}(\tilde{S})$ as follows. It chooses $x \xleftarrow{\$} \mathbb{Z}_q$, computes $X \leftarrow g^x$ and runs $\overline{\mathsf{CV}}$ on input $1^k$, $mpk = (1^k, \mathbb{G}, q, g, X)$. It also chooses $q_{\mathrm{g}} \xleftarrow{\$} \{1, \ldots, \boldsymbol{Q}^{\textsc{Init}}_{\overline{\mathsf{CV}}}(k)\}$, hoping that the identity $I_{\mathrm{g}}$ initialized in $\overline{\mathsf{CV}}$'s $q_{\mathrm{g}}$-th Init query will be the one under attack in the second phase of the game. All $\overline{\mathsf{CV}}$'s Init$(\cdot)$, Conv$(\cdot)$ and Corr$(\cdot)$ oracle queries are simulated using the real protocol algorithms, except for queries involving identity $I_{\mathrm{g}}$. At the initialization of $I_{\mathrm{g}}$, $\mathsf{B}$ chooses $\tilde{h} \xleftarrow{\$} \{0,1\}^{l(k)}$ and computes $\tilde{R} \leftarrow \tilde{S}X^{-\tilde{h}}$. It also programs the random oracle so that $\mathrm{H}(\tilde{R} \| I_{\mathrm{g}}) = \tilde{h}$. (If $\overline{\mathsf{CV}}$ queried $\mathrm{H}(\tilde{R} \| I_{\mathrm{g}})$ before, then $\mathsf{B}$ gives up. Because up to that point $\overline{\mathsf{CV}}$'s view is independent of $\tilde{R}$, this happens only with probability $2^{-l(k)}$.) Conversations for $I_{\mathrm{g}}$ are generated by choosing $c \xleftarrow{\$} \{0,1\}^{l(k)}$, $z \xleftarrow{\$} \mathbb{Z}_q$ and by computing $Y \leftarrow g^z \tilde{S}^{-c}$. The returned conversation is $\tilde{R} \| \tilde{S} \| Y \| c \| z$. If $\overline{\mathsf{CV}}$ decides to corrupt identity $I_{\mathrm{g}}$, then $\mathsf{B}$ gives up.

With probability $1/\boldsymbol{Q}^{\textsc{Init}}_{\overline{\mathsf{CV}}}(k)$, algorithm $\overline{\mathsf{CV}}$ outputs $(I_{\mathrm{b}}, St_{\overline{\mathsf{CP}}})$ with identity $I_{\mathrm{b}} = I_{\mathrm{g}}$. Let $\overline{\mathsf{V}}'$ be a verifier algorithm that only accepts a conversation $R\|S\|Y\|c\|z$ if $\overline{\mathsf{V}}$ accepts and if moreover $R = \tilde{R}$, and let $\mathrm{acc}'(St_{\overline{\mathsf{CP}}}, (mpk, I_{\mathrm{b}}))$ be $\overline{\mathsf{CP}}$'s probability of making $\overline{\mathsf{V}}'$ accept. Then by the Reset Lemma, with probability $\mathrm{res}'(St_{\overline{\mathsf{CP}}}, (mpk, I_{\mathrm{b}}))$ algorithm $\mathsf{B}$ can generate two accepting conversations $R\|S\|Y\|c_1\|z_1$ and $R\|S\|Y\|c_2\|z_2$ with $c_1 \neq c_2$. Since $R = \tilde{R}$, we also have that $S = \tilde{S} \equiv RX^{\mathrm{H}(R\|I_{\mathrm{b}})}$. Finally, algorithm computes $\tilde{s} \leftarrow (z_1 - z_2)/(c_1 - c_2) \bmod q$ and outputs $\tilde{s}$ as the discrete logarithm of $\tilde{S}$. Overall, $\mathsf{B}$'s advantage is at least

$$\begin{aligned}\mathbf{Adv}^{\mathrm{dlog}}_{\mathsf{K}_{\mathrm{cg}},\mathsf{B}}(k) \;&\geq\; \frac{1}{\boldsymbol{Q}^{\textsc{Init}}_{\overline{\mathsf{CV}}}(k)} \cdot \mathrm{res}'(St_{\overline{\mathsf{CP}}}, (mpk, I_{\mathrm{b}})) \\ &\geq\; \frac{1}{\boldsymbol{Q}^{\textsc{Init}}_{\overline{\mathsf{CV}}}(k)} \cdot \left(\mathrm{acc}'(St_{\overline{\mathsf{CP}}}, (mpk, I_{\mathrm{b}})) - 2^{-l(k)}\right)^2 \,. \qquad (6)\end{aligned}$$

Combining Equation (6) with $\mathsf{F}$'s advantage through an analysis similar to that in the proof of Theorem 6.2, we obtain Equation (5). Since all terms on the righthand side of Equation (5) are negligible, the theorem follows. ▌

The uf-cma security of the $\mathcal{BNN}$-$\mathcal{IBS}$ = efs-IBI-2-IBS($\mathcal{BNN}$-$\mathcal{IBI}$) scheme under the hardness of the discrete logarithm problem associated with $\mathsf{K}_{\mathrm{cg}}$ follows from Theorem 6.3 and Theorem 4.13.

It is unknown if the $\mathcal{BNN}$-$\mathcal{IBI}$ is also secure against impersonation under active and concurrent attacks under the plain discrete logarithm assumption. A proof does exist however under the stronger one-more discrete logarithm assumption.

**Theorem 6.4** The $\mathcal{BNN}$-$\mathcal{IBI}$ scheme associated with super-logarithmic challenge length $l(\cdot)$ and prime-order cyclic group generator $\mathsf{K}_{\mathrm{cg}}$ such that $2^{l(k)} < q$ for all $(\langle\mathbb{G}\rangle, q, g) \xleftarrow{\$} \mathsf{K}_{\mathrm{cg}}(1^k)$ is imp-ca secure in the random oracle model if the one-more discrete logarithm problem associated with $\mathsf{K}_{\mathrm{cg}}$ is hard. ▌

**Proof:** Given a polynomial-time impersonator $\overline{\mathsf{A}}$ breaking $\mathcal{BNN}$-$\mathcal{IBI}$ under concurrent attack, we show how to build a one-more discrete logarithm algorithm $\mathsf{B}$ and a forger $\mathsf{F}$ such that

$$\mathbf{Adv}^{\mathrm{imp\text{-}ca}}_{\mathcal{BNN}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) \;\leq\; \sqrt{\mathbf{Adv}^{\mathrm{1m\text{-}dlog}}_{\mathsf{K}_{\mathrm{cg}},\mathsf{B}}(k)} + \sqrt{\mathbf{Adv}^{\mathrm{ss\text{-}cma}}_{\mathcal{S}\mathit{chnorr}\text{-}\mathcal{SS},\mathsf{F}}(k)} + 2 \cdot 2^{-l(k)} \,. \qquad (7)$$

The description of algorithm $\mathsf{F}$ is identical to that in the proof of Theorem 6.3, but using the user secret keys to simulate interactive prover protocols, rather than generating conversations. Since the one-more discrete logarithm assumption implies the discrete logarithm assumption, the second term on the righthand side of Equation (7) is negligible.

Algorithm $\mathsf{B}$, on input $\langle\mathbb{G}\rangle, q, g$, chooses $x \xleftarrow{\$} \mathbb{Z}_q$, computes $X \leftarrow g^x$ and runs $\overline{\mathsf{CV}}$ on input $1^k, mpk = (1^k, \langle\mathbb{G}\rangle, q, g, X)$. When $\overline{\mathsf{CV}}$ initializes identity $I$, it uses the challenge oracle to produce $S_I \leftarrow \textsc{Chall}$, it chooses $h_I \xleftarrow{\$} \mathbb{Z}_q$ and computes $R_I \leftarrow SX^{-h}$. It programs the random oracle so that $\mathrm{H}(R_I\|I) = h_I$, or gives up if $\mathrm{H}(R_I\|h_I)$ was queried before. Simulation of an interactive prover session $s$ for identity $I$ is done by querying $Y_{I,s} \leftarrow \textsc{Chall}$ and sending $R_I, S_I, Y_I$as the first message. The response for challenge $c_{I,s}$ is computed as $z_{I,s} \leftarrow \textsc{DLog}(Y_{I,s}S_I^{c_{I,s}})$. When $\overline{\mathsf{CV}}$ asks to corrupt identity $I$, algorithm $\mathsf{B}$ calls its discrete logarithm oracle for $s_I \leftarrow \textsc{DLog}(S_I)$ and returns $(R_I, s_I)$.

In the second phase of the game, $\overline{\mathsf{CP}}$ will impersonate an uncorrupted identity $I_{\mathrm{b}}$. With a probability given by the Reset Lemma, algorithm $\mathsf{B}$ extracts two accepting conversations $R\|S\|Y\|c_1\|z_1$ and $R\|S\|Y\|c_2\|z_2$ with $(R, S) = (R_I, S_I)$ and $c_1 \neq c_2$. From these, $\mathsf{B}$ computes $s_{I_{\mathrm{b}}}$ as $(z_1 - z_2)/(c_1 -$

$c_2) \bmod q$ and uses it to compute discrete logarithms of all values $Y_{I_\mathrm{b},s}$ as $y_{I_\mathrm{b},i} \leftarrow z_{I_\mathrm{b},i} - s_{I_\mathrm{b}} c_{I_\mathrm{b},s} \bmod q$. For all other initialized identities $I \neq I_\mathrm{b}$, algorithm $\mathsf{B}$ simply queries $s_I \leftarrow \mathrm{DL{\small OG}}(S_I)$ itself and computes the discrete logarithms $y_{I,s} \leftarrow z_{I,s} - s_I c_{I,s} \bmod q$.

Let $n$ be the number of identities initialized by $\overline{\mathsf{A}}$, and let $n_I$ be the number of prover sessions initiated for identity $I$. Then for each identity $I$, $\mathsf{B}$ calculated the discrete logarithm of $n_I + 1$ target points (all $Y_{I,s}$ and $S_I$) using $n_I + 1$ queries to the DLog oracle (one for each prover session, and an additional one at the end of the game), *except* for identity $I_\mathrm{b}$ where the discrete logarithms of $n_{I_\mathrm{b}} + 1$ target points were computed using only $n_{I_\mathrm{b}}$ queries to the DLog oracle. So in total, $\mathsf{B}$ saved one DLog query and wins the game.

Again, an analysis similar to that in the proof of Theorem 6.2 yields Equation (7), thereby concluding the proof. ▌

## 7 Efficiency Comparison

We compare the signature sizes and the efficiency in signing and verification of all the schemes studied in this paper in Table 1. Let us explain some of the notation used in the table. We denote by $\mathrm{mexp}(n)$ an $n$-fold multi-exponentiation in the underlying group, meaning the operation of computing $a_1^{b_1} \cdot \ldots \cdot a_n^{b_n}$ given group elements $a_1, \ldots, a_n$ and exponents $b_1, \ldots, b_n$. The cost is that of 1 exponentiation plus $2^n$ multiplications, which for small $n$ (e.g. $n = 2$ or 3) is essentially the same as 1 exponentiation. We let

$$\begin{aligned} s_{m,t}(k) &= \min\left[m(k) \cdot (2t(k)+1) + t(k)\ ,\ \mathrm{mexp}(t(k))\right] \text{ and} \\ v_{m,t}(k) &= \min\left[m(k) \cdot (2t(k)+1) + t(k)\ ,\ \mathrm{mexp}(t(k)+1)\right]\ . \end{aligned}$$

These terms appear for *ItR-IBS* in the table.

Notice that the pairing-based schemes yield shorter signatures than other schemes but we pay in verification time which now involves pairing operations. *ChCh-IBS* is more efficient than *Hs-IBS*. *BNN-IBS* is more efficient than *OkDL-IBS*.

## Acknowledgments

We thank Marc Fischlin for pointing out that the *Sh-SI* scheme is zero-knowledge. We also would like to thank Bart De Decker, Ann Haegemans, Kenneth G. Paterson, Frank Piessens, Bart Preneel and the anonymous referees for their valuable comments on earlier versions of this work. The second author was supported by the Thailand Research Fund.

## References

[AABN02] Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the Fiat-Shamir transform: Minimizing assumptions for security and forward-security. In L. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 418–433. Springer-Verlag, April 2002. (Cited on page 3, 5, 7, 20, 21, 22, 39, 40.)

[ADR02] Jee Hea An, Yevgeniy Dodis, and Tal Rabin. On the security of joint signature and encryption. In L. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 83–107, 2002. (Cited on page 39.)

| Scheme | Signature size | Signing time | Verification time | Security assumption |
|---|---|---|---|---|
| $\mathcal{Cert}$-$\mathcal{IBS}$ | 2 sig. of $\mathcal{SS}$<br>1 public key of $\mathcal{SS}$ | 1 $\mathsf{Sign}$ of $\mathcal{SS}$ | 2 $\mathsf{Vf}$ of $\mathcal{SS}$ | $\mathcal{SS}$ is uf-cma |
| $\mathcal{FFS}$-$\mathcal{IBS}$ | 2 el. of $\mathbb{Z}_N^*$ | $t(k)+1$ mult. in $\mathbb{Z}_N^*$ | $t(k)+1$ mult. in $\mathbb{Z}_N^*$ | factoring |
| $\mathcal{ItR}$-$\mathcal{IBS}$ | 2 el. of $\mathbb{Z}_N^*$ | $s_{m,t}(k)$ mult. in $\mathbb{Z}_N^*$ | $v_{m,t}(k)$ mult. in $\mathbb{Z}_N^*$ | factoring |
| $\mathcal{FF}$-$\mathcal{IBS}$ | 2 el. of $\mathbb{Z}_N^*$<br>1 el. of $\mathbb{Z}_{2^{m(k)}}$ | 2 mexp(2) in $\mathbb{Z}_N^*$ | 1 mexp(3) in $\mathbb{Z}_N^*$ | factoring |
| $\mathcal{GQ}$-$\mathcal{IBS}$ | 2 el. of $\mathbb{Z}_N^*$ | 2 exp. in $\mathbb{Z}_N^*$ | 1 mexp(2) in $\mathbb{Z}_N^*$ | one-wayness of RSA |
| $\mathcal{Sh}$-$\mathcal{IBS}$ | 2 el. of $\mathbb{Z}_N^*$ | 2 exp. in $\mathbb{Z}_N^*$ | 1 mexp(2) in $\mathbb{Z}_N^*$ | one-wayness of RSA |
| $\mathcal{Sh}^*$-$\mathcal{IBS}$ | 2 el. of $\mathbb{Z}_N^*$ | 2 exp. in $\mathbb{Z}_N^*$ | 1 mexp(2) in $\mathbb{Z}_N^*$ | one-more RSA inversion |
| $\mathcal{OkRSA}$-$\mathcal{IBS}$ | 2 el. of $\mathbb{Z}_N^*$<br>1 el. of $\mathbb{Z}_e$ | 2 mexp(2) in $\mathbb{Z}_N^*$ | 1 mexp(3) in $\mathbb{Z}_N^*$ | one-wayness of RSA |
| $\mathcal{SOK}$-$\mathcal{IBS}$ | 2 el. of $\mathbb{G}_1$ | 2 exp. in $\mathbb{G}_1$ | 3 pairings | CDH in $\mathbb{G}_1$ |
| $\mathcal{Hs}$-$\mathcal{IBS}$ | 1 el. of $\mathbb{G}_1$<br>1 el. of $\mathbb{G}_2$ | 1 exp. in $\mathbb{G}_2$<br>1 mexp(2) in $\mathbb{G}_1$ | 2 pairings<br>1 exp. in $\mathbb{G}_2$ | CDH in $\mathbb{G}_1$ |
| $\mathcal{ChCh}$-$\mathcal{IBS}$ | 2 el. of $\mathbb{G}_1$ | 2 exp. in $\mathbb{G}_1$ | 2 pairings | CDH in $\mathbb{G}_1$ |
| $\mathcal{Beth}$-$\mathcal{IBS}$ | 2 el. of $\mathbb{G}$<br>1 el. of $\mathbb{Z}_q$ | 1 exp. in $\mathbb{G}$ | 1 mexp(3) in $\mathbb{G}$ | $\mathcal{ElG}$-$\mathcal{SS}$ is uuf-nma |
| $\mathcal{OkDL}$-$\mathcal{IBS}$ | 3 el. of $\mathbb{G}$<br>2 el. of $\mathbb{Z}_q$ | 1 mexp(2) in $\mathbb{G}$ | 1 mexp(3) in $\mathbb{G}$<br>1 exp. in $\mathbb{G}$ | discrete log |
| $\mathcal{BNN}$-$\mathcal{IBS}$ | 3 el. of $\mathbb{G}$<br>1 el. of $\mathbb{Z}_q$ | 1 exp. in $\mathbb{G}$ | 1 mexp(2) in $\mathbb{G}$<br>1 exp. in $\mathbb{G}$ | discrete log |

Table 1: Efficiency comparison of all treated IBS schemes. Column 1 is the name of the scheme. Column 2 gives the size of a signature under each scheme. Columns 3 and 4 give (the dominating term in) the computation cost associated to creating and verifying a signature, respectively. The last column gives the assumption under which the scheme is secure. In the table, we use the abbreviations "sig." for signature, "el." for element, "sq." for squaring, "exp." for exponentiation, and "mult." for multiplication. Also, $N$ is an RSA modulus, $e$ is an RSA encryption exponent, $\mathbb{G}_1$ and $\mathbb{G}_2$ are prime-order groups such that a pairing $\hat{e}: \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ exists, and $\mathbb{G}$ is a prime-order group.

[BD89] Mike Burmester and Yvo Desmedt. Remarks on soundness of proofs. *Electronics Letters*, 25(22):1509–1511, 1989. (Cited on page 23.)

[Bet88] Thomas Beth. Efficient zero-knowledged identification scheme for smart cards. In C. Gunther, editor, *EUROCRYPT 1988*, volume 330 of *LNCS*, pages 77–86. Springer-Verlag, May 1988. (Cited on page 3, 5, 6, 7, 36, 37.)

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer-Verlag, August 2001. (Cited on page 3, 4, 33.)

[BFGM01] Mihir Bellare, Marc Fischlin, Shafi Goldwasser, and Silvio Micali. Identification protocols secure against reset attacks. In B. Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 268–286. Springer-Verlag, May 2001. (Cited on page 14, 15.)

[BGLS03] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In E. Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 416–432. Springer-Verlag, 2003. (Cited on page 15.)

[Blu82] Manuel Blum. Coin flipping by telephone. In A. Gersho, editor, *Advances in Cryptology: A Report on CRYPTO 81*, University of California, Santa Barbara, Department of ECE Report No 82-04, pages 11–15, 1982. (Cited on page 23.)

[BN00] Mihir Bellare and Chanathip Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In T. Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 531–545. Springer-Verlag, December 2000. (Cited on page 39.)

[BN05] Mihir Bellare and Gregory Neven. Transitive signatures: New schemes and proofs. *IEEE Trans. Inf. Theory*, 51(6):2133–2151, 2005. (Cited on page 33.)

[BNN04] Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Security proofs for identity-based identification and signature schemes. In C. Cachin and J. Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 268–286. Springer-Verlag, May 2004. (Cited on page 8.)

[BNPS03] Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme. *J. Cryptology*, 16(3):185–215, 2003. (Cited on page 7, 27.)

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In Y. Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 31–46. Springer-Verlag, January 2003. (Cited on page 33.)

[BP02] Mihir Bellare and Adriana Palacio. GQ and Schnorr identification schemes: Proofs of security against impersonation under active and concurrent attack. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 162–177. Springer-Verlag, August 2002. (Cited on page 3, 4, 6, 7, 9, 23, 28, 29.)

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proc. of the 1st CCS*, pages 62–73. ACM Press, November 1993. (Cited on page 7.)

[CC03] Jae Choon Cha and Jung Hee Cheon. An identity-based signature from gap diffie-hellman groups. In Y. Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 18–30. Springer-Verlag, January 2003. (Cited on page 3, 4, 5, 6, 12, 33, 34.)

[Che02] Jung Hee Cheon. A universal forgery of Hess's second ID-based signature against the known-message attack. Cryptology ePrint Archive, Report 2002/028, 2002. `http://eprint.iacr.org/2002/028`. (Cited on page 33.)

[DKXY03] Yevgeniy Dodis, Jonathan Katz, Shouhuai Xu, and Moti Yung. Strong key-insulated signature schemes. In Y. Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 130–144. Springer-Verlag, January 2003. (Cited on page 3, 4, 5, 6, 12, 14, 16, 20, 25, 28, 33.)

[DVQ96] Jean-François Dhem, Daniel Veithen, and Jean-Jacques Quisquater. SCALPS: Smart card for limited payment systems. *IEEE Micro*, 16(3):42–51, 1996. (Cited on page 3.)

[FF02] Marc Fischlin and Roger Fischlin. The representation problem based on factoring. In B. Preneel, editor, *CT-RSA 2002*, volume 2271 of *LNCS*, pages 96–113. Springer-Verlag, February 2002. (Cited on page 6, 7, 25, 26.)

[FFS88] Uriel Feige, Amos Fiat, and Adi Shamir. Zero knowledge proofs of identity. *J. Cryptology*, 1(2):77–94, 1988. (Cited on page 3, 4, 6, 7, 9, 24, 25.)

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In A. Odlyzko, editor, *CRYPTO 1986*, volume 263 of *LNCS*, pages 186–194. Springer-Verlag, August 1986. (Cited on page 3, 5, 6, 20, 24.)

[Gir90] Marc Girault. An identity-based identification scheme based on discrete logarithms modulo a composite number. In I. Damgård, editor, *EUROCRYPT 1990*, volume 473 of *LNCS*, pages 481–486. Springer-Verlag, May 1990. (Cited on page 3, 5, 6, 7, 31, 32.)

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Computing*, 17(2):281–308, April 1988. (Cited on page 4, 10.)

[GQ89] Louis C. Guillou and Jean-Jacques Quisquater. A "paradoxical" identity-based signature scheme resulting from zero-knowledge. In S. Goldwasser, editor, *CRYPTO 1988*, volume 403 of *LNCS*, pages 216–231. Springer-Verlag, August 1989. (Cited on page 3, 5, 6, 27, 28.)

[Hes03] Florian Hess. Efficient identity based signature schemes based on pairings. In K. Nyberg and H. Heys, editors, *Selected Areas in Cryptography, SAC 2002*, pages 310–324. Springer-Verlag, 2003. (Cited on page 3, 5, 6, 33, 34.)

[IL89] Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography. In *Proc. of the 30th FOCS*, pages 230–235, Research Triangle Park, North Carolina, October 1989. IEEE Computer Society Press. (Cited on page 14.)

[KH04] Kaoru Kurosawa and Swee-Huay Heng. From digital signature to ID-based identification/signature. In F. Bao, R. Deng, and J. Zhou, editors, *PKC 2004*, volume 2947 of *LNCS*, pages 248–261. Springer-Verlag, 2004. (Cited on page 8.)

[LQ04] Benoît Libert and Jean-Jacques Quisquater. The exact security of an identity based signature and its applications. Cryptology ePrint Archive, Report 2004/102, 2004. `http://eprint.iacr.org/2004/102`. (Cited on page 8.)

[MvOV96] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, 1996. (Cited on page 37.)

[Oka93] Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In E. Brickell, editor, *CRYPTO 1992*, volume 740 of *LNCS*, pages 31–53. Springer-Verlag, August 1993. (Cited on page 3, 5, 6, 7, 25, 30, 31, 38, 40, 41.)

[OO90] Kazuo Ohta and Tatsuaki Okamoto. A modification of the Fiat-Shamir scheme. In S. Goldwasser, editor, *CRYPTO 1988*, volume 403 of *LNCS*, pages 232–243. Springer-Verlag, August 1990. (Cited on page 6, 24.)

[OO98] Kazuo Ohta and Tatsuaki Okamoto. On concrete security treatment of signatures derived from identification. In H. Krawczyk, editor, *CRYPTO 1998*, volume 1462 of *LNCS*, pages 354–370. Springer-Verlag, August 1998. (Cited on page 3.)

[OS90] H. Ong and Claus-Peter Schnorr. Fast signature generation with a Fiat-Shamir–like scheme. In I. Damgård, editor, *EUROCRYPT 1990*, volume 473 of *LNCS*, pages 432–440. Springer-Verlag, May 1990. (Cited on page 6, 24, 25.)

[Pat02] Kenneth G. Paterson. ID-based signatures from pairings on elliptic curves. Cryptology ePrint Archive, Report 2002/004, 2002. `http://eprint.iacr.org/`. (Cited on page 3, 33.)

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *J. Cryptology*, 13(3):361–396, 2000. (Cited on page 3, 6, 25, 37.)

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proc. of the 22nd ACM STOC*, pages 387–394, Baltimore, Maryland, May 14–16 1990. ACM Press. (Cited on page 14, 15.)

[Sch90] Claus-Peter Schnorr. Efficient identification and signatures for smartcards. In G. Brassard, editor, *CRYPTO 1989*, volume 435 of *LNCS*, pages 239–252. Springer-Verlag, August 1990. (Cited on page 7, 8, 31, 45.)

[Sch96] Claus-Peter Schnorr. Security of $2^t$-root identification and signatures. In N. Koblitz, editor, *CRYPTO 1996*, volume 1109 of *LNCS*, pages 143–156. Springer-Verlag, August 1996. (Cited on page 6, 7, 25.)

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In G.R. Blakely and D. Chaum, editors, *CRYPTO 1984*, volume 196 of *LNCS*, pages 47–53. Springer-Verlag, 1984. (Cited on page 3, 5, 6, 28.)

[Sho99] Victor Shoup. On the security of a practical identification scheme. *J. Cryptology*, 12(4):247–260, 1999. (Cited on page 25.)

[SOK00] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *SCIS 2000*, Okinawa, Japan, January 2000. (Cited on page 3, 6, 7, 8, 33, 34, 36.)

[SPMLS02] Jacques Stern, David Pointcheval, John Malone-Lee, and Nigel P. Smart. Flaws in applying proof methodologies to signature schemes. In M. Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 93–110. Springer-Verlag, 2002. (Cited on page 7, 39.)

[SSN98] Shahrokh Saeednia and Reihaneh Safavi-Naini. On the security of Girault's identification scheme. In H. Imai and Y. Zheng, editors, *PKC 1998*, volume 1431 of *LNCS*, pages 149–153. Springer-Verlag, 1998. (Cited on page 6, 7, 31, 32.)

[Wil80] Hugh C. Williams. A modification of the RSA public-key encryption procedure. *IEEE Trans. Inf. Theory*, 26(6):726–729, 1980. (Cited on page 23.)

[Yi03] Xun Yi. An identity-based signature scheme from the Weil pairing. *IEEE Communications Letters*, 7(2):76–78, 2003. (Cited on page 3, 5, 6, 33, 34.)

# A Proof of Theorem 3.2

Let $\overline{\mathsf{A}} = (\overline{\mathsf{CV}}, \overline{\mathsf{CP}})$ be a polynomial-time imp-atk adversary against $\mathcal{C}ert\text{-}\mathcal{IBI} = (\mathsf{MKg}, \mathsf{UKg}, \overline{\mathsf{P}}, \overline{\mathsf{V}})$. The intuition is as follows. Let $pk_{I_{\rm b}}$ denote the public key (of the SI scheme) assigned by Init to $I_{\rm b}$, the identity that $\overline{\mathsf{CV}}$ outputs as the one $\overline{\mathsf{CP}}$ will impersonate. Let $cert = (pk, \sigma)$ be the certificate sent by $\overline{\mathsf{CP}}$ to $\overline{\mathsf{V}}$ as part of its first flow in the identification protocol. If $\overline{\mathsf{A}}$ wins, there are two possibilities. Either $pk \neq pk_{I_{\rm b}}$, in which case $\sigma$ is a forgery under the master public key $mpk$ of message $pk \| I_{\rm b}$, or $pk = pk_{I_{\rm b}}$, in which case $\overline{\mathsf{CP}}$ succeeded in identifying itself under $pk_{I_{\rm b}}$ in the underlying SI protocol. The first possibility is ruled out by the security of $\mathcal{SS}$ and the second by the security of $\mathcal{SI}$.

```
Subroutine $\text{I}{\scriptstyle\text{NIT-SIM}}(I)$
    If $I \in CU \cup HU$ then return $\perp$
    $(pk_I, sk_I) \leftarrow \mathsf{Kg}(1^k)$ ; $cert_I \leftarrow (pk_I, \text{S}{\scriptstyle\text{IGN}}(pk_I \| I))$ ; $usk[I] \leftarrow (sk_I, cert_I)$ ; $HU \leftarrow HU \cup \{I\}$
    Return 1
```

```
Algorithm $\mathsf{F}(1^k, mpk : \text{S}{\scriptstyle\text{IGN}}(\cdot))$
    $HU \leftarrow \emptyset$ ; $CU \leftarrow \emptyset$ ; $PS \leftarrow \emptyset$
    If atk = pa then $(I_{\mathrm{b}}, St_{\overline{\mathsf{CP}}}) \stackrel{\$}{\leftarrow} \overline{\mathsf{CV}}(1^k, mpk : \text{I}{\scriptstyle\text{NIT-SIM}}, \text{C}{\scriptstyle\text{ORR}}, \text{C}{\scriptstyle\text{ONV}})$
    Else $(I_{\mathrm{b}}, St_{\overline{\mathsf{CP}}}) \stackrel{\$}{\leftarrow} \overline{\mathsf{CV}}(1^k, mpk : \text{I}{\scriptstyle\text{NIT-SIM}}, \text{C}{\scriptstyle\text{ORR}}, \text{P}{\scriptstyle\text{ROV}})$
    $HU \leftarrow HU \setminus \{I_{\mathrm{b}}\}$ ; $CU \leftarrow CU \cup \{I_{\mathrm{b}}\}$
    If atk = pa then $(M_{\mathrm{out}}, St_{\overline{\mathsf{CP}}}) \stackrel{\$}{\leftarrow} \overline{\mathsf{CP}}(\varepsilon, St_{\overline{\mathsf{CP}}} : \text{I}{\scriptstyle\text{NIT-SIM}}, \text{C}{\scriptstyle\text{ORR}}, \text{C}{\scriptstyle\text{ONV}})$
    Else $(M_{\mathrm{out}}, St_{\overline{\mathsf{CP}}}) \stackrel{\$}{\leftarrow} \overline{\mathsf{CP}}(\varepsilon, St_{\overline{\mathsf{CP}}} : \text{I}{\scriptstyle\text{NIT-SIM}}, \text{C}{\scriptstyle\text{ORR}}, \text{P}{\scriptstyle\text{ROV}})$
    Parse $M_{\mathrm{out}}$ as $cert \| M'$ ; Parse $cert$ as $(pk, \sigma)$
    Return $(pk \| I_{\mathrm{b}}, \sigma)$
```

Figure 21: The adversary $\mathsf{F}$ against the standard signature scheme $\mathcal{SS} = (\mathsf{SKg}, \mathsf{Sign}, \mathsf{Vf})$ with access to a signing oracle $\text{S}{\scriptstyle\text{IGN}}(\cdot) = \mathsf{Sign}(msk, \cdot)$, for the proof of Theorem 3.2.

We now proceed to the actual proof. Assume the number of queries $\overline{\mathsf{CV}}$ makes to $\text{I}{\scriptstyle\text{NIT}}$ is at most $\boldsymbol{Q}^{\text{I}{\scriptscriptstyle\text{NIT}}}_{\overline{\mathsf{CV}}}(\cdot)$. We construct polynomial-time adversaries $\mathsf{F}$ attacking $\mathcal{SS} = (\mathsf{SKg}, \mathsf{Sign}, \mathsf{Vf})$ and $\mathsf{A} = (\mathsf{CV}, \mathsf{CP})$ attacking $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$, such that for every $k \in \mathbb{N}$

$$\mathbf{Adv}^{\text{imp-atk}}_{\mathit{Cert}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) \;\leq\; \mathbf{Adv}^{\text{uf-cma}}_{\mathcal{SS},\mathsf{F}}(k) + \boldsymbol{Q}^{\text{I}{\scriptscriptstyle\text{NIT}}}_{\overline{\mathsf{CV}}}(k) \cdot \mathbf{Adv}^{\text{imp-atk}}_{\mathcal{SI},\mathsf{A}}(k)\ . \tag{8}$$

The theorem follows. We now describe $\mathsf{F}, \mathsf{A}$ in turn.

Adversary $\mathsf{F}$ (attacking $\mathcal{SS}$) is depicted in Figure 21. It takes input the security parameter $1^k$ and a public key $mpk$ of the $\mathcal{SS}$ scheme, and has access to the signing oracle $\text{S}{\scriptstyle\text{IGN}}(\cdot) = \mathsf{Sign}(msk, \cdot)$ where $msk$ is the secret key corresponding to $mpk$. It will run $\overline{\mathsf{CV}}, \overline{\mathsf{CP}}$ as subroutines, itself providing answers to their oracle queries. It answers a query to $\text{I}{\scriptstyle\text{NIT}}$ by running the subroutine also shown in the same Figure. It can do this even though it does not have $msk$ via its access to the $\text{S}{\scriptstyle\text{IGN}}$ oracle. In this way, $\mathsf{F}$ knows the secret keys of all initiated users, and can thus easily execute the code for all the other oracles, by simply following Figure 4. (Thus, these other oracles are simply shown as provided to $\overline{\mathsf{CV}}$ in the figure.) Eventually $\overline{\mathsf{CV}}$ outputs some identity $I_{\mathrm{b}}$ and state information $St_{\overline{\mathsf{CV}}}$. Now $\mathsf{F}$ obtains from $\overline{\mathsf{CP}}$ the first message $M_{\mathrm{out}}$ of its interaction with $\overline{\mathsf{V}}$. (It does this by running $\overline{\mathsf{CP}}$ with inputs $\varepsilon, St_{\overline{\mathsf{CV}}}$, simulating its oracles just as above, to get the message $M_{\mathrm{out}}$ and updated state information $St_{\overline{\mathsf{CP}}}$.) It parses $M_{\mathrm{out}}$ as $cert \| M_1$, and parses $cert$ as $(pk, \sigma)$. $\mathsf{F}$ is betting that $pk$ is not $pk_{I_{\mathrm{b}}}$, the "real" public key of $I_{\mathrm{b}}$, but a value chosen by $\overline{\mathsf{A}}$, and thus the certificate is forged. Accordingly it will output $(pk, \sigma)$ as its forgery.

Adversary $\mathsf{A} = (\mathsf{CV}, \mathsf{CP})$ (attacking $\mathcal{SI}$) is depicted in Figure 22. The cheating verifier component $\mathsf{CV}$ gets input security parameter $1^k$ and a public key $pk$, and has access to a conversation oracle (if atk = pa) or a prover oracle (if atk = aa or atk = ca). It will run $\overline{\mathsf{CV}}$ as a subroutine, itself providing answers to $\overline{\mathsf{CV}}$'s oracle queries. It begins by running the key-generation algorithm $\mathsf{SKg}$ of the signature scheme $\mathcal{SS}$ on input $1^k$ to get back a master public key $mpk$ and matching secret key $msk$. (This gives it the ability to create certificates and thus simulate the $\text{I}{\scriptstyle\text{NIT}}$ oracle.) It also guesses an identity $I_{\mathrm{g}}$ that it hopes equals the identity $I_{\mathrm{b}}$ of the prover that $\overline{\mathsf{A}}$ will eventually impersonate. (Since this is a string in $\{0,1\}^*$ and the number of possible values for it is a priori infinite, $\mathsf{CV}$ cannot guess it directly. Instead, it picks at random the index $q_{\mathrm{g}}$ of $I_{\mathrm{b}}$ in the sequence of queries made to $\text{I}{\scriptstyle\text{NIT}}$, eventually

```
Subroutine INIT-SIM($I$)
    If $I \in CU \cup HU$ then return $\perp$
    $HU \leftarrow HU \cup \{I\}$
    If $|HU| = q_{\mathrm{g}}$
        then $I_{\mathrm{g}} \leftarrow I$ ; $upk[I] \leftarrow pk$ ; $sk[I] \leftarrow \perp$
        else $(upk[I], sk[I]) \xleftarrow{\$} \mathsf{Kg}(1^k)$
    $cert[I] \leftarrow (upk[I], \mathsf{Sign}(msk, upk[I] \| I))$
    $usk[I] \leftarrow (sk[I], cert[I])$
    Return 1
```

```
Subroutine CORR-SIM($I$)
    If $I \notin HU$ then return $\perp$
    $CU \leftarrow CU \cup \{I\}$ ; $HU \leftarrow HU \setminus \{I\}$
    If $I = I_{\mathrm{g}}$ then abort
    Return $usk[I]$
```

```
Subroutine CONV-SIM($I$)
    If $I \notin HU$ then return $\perp$
    If $I = I_{\mathrm{g}}$ then $C \leftarrow cert[I] \| \mathrm{OR}(\varepsilon)$
    Else $(C, d) \xleftarrow{\$} \mathbf{Run}[\overline{\mathsf{P}}(usk[I]) \leftrightarrow \overline{\mathsf{V}}(mpk, I)]$
    Return $C$
```

```
Subroutine PROV-SIM($I, s, M_{\mathrm{in}}$)
    If $I \notin HU$ then return $\perp$
    If $(I, s) \notin PS$ then
        If atk = aa then $PS \leftarrow \{(I, s)\}$
        If atk = ca then $PS \leftarrow PS \cup \{(I, s)\}$
        If $I = I_{\mathrm{g}}$ then
            $M_{\mathrm{out}} \leftarrow cert[I] \| \mathrm{OR}(s, M_{\mathrm{in}})$
            Return $M_{\mathrm{out}}$
        Pick random coins $\rho_{\overline{\mathsf{P}}}$ for $\overline{\mathsf{P}}$
        $St_{\overline{\mathsf{P}}}[I, s] \leftarrow (usk[I], \rho_{\overline{\mathsf{P}}})$
    If $I = I_{\mathrm{g}}$
        then $M_{\mathrm{out}} \leftarrow \mathrm{OR}(s, M_{\mathrm{in}})$
        else $(M_{\mathrm{out}}, St_{\overline{\mathsf{P}}}[I, s]) \leftarrow \overline{\mathsf{P}}(M_{\mathrm{in}}, St_{\overline{\mathsf{P}}}[I, s])$
    Return $M_{\mathrm{out}}$
```

```
Algorithm $\mathsf{CV}(1^k, pk : \mathrm{OR})$
    $(mpk, msk) \xleftarrow{\$} \mathsf{SKg}(1^k)$ ; $HU \leftarrow \emptyset$ ; $CU \leftarrow \emptyset$ ; $PS \leftarrow \emptyset$ ; $q_{\mathrm{g}} \xleftarrow{\$} \{1, \ldots, \boldsymbol{Q}^{\mathrm{INIT}}_{\overline{\mathsf{CV}}}(k)\}$
    If atk = pa then $(I_{\mathrm{b}}, St_{\overline{\mathsf{CP}}}) \xleftarrow{\$} \overline{\mathsf{CV}}(1^k, mpk$ : INIT-SIM, CORR-SIM, CONV-SIM$)$
    Else $(I_{\mathrm{b}}, St_{\overline{\mathsf{CP}}}) \xleftarrow{\$} \overline{\mathsf{CV}}(1^k, mpk$ : INIT-SIM, CORR-SIM, PROV-SIM$)$
    If $|HU| < q_{\mathrm{g}}$ or $I_{\mathrm{b}} \neq I_{\mathrm{g}}$ then abort
    $HU \leftarrow HU \setminus \{I_{\mathrm{b}}\}$ ; $CU \leftarrow CU \cup \{I_{\mathrm{b}}\}$ ; $St_{\mathsf{CP}} \leftarrow (St_{\overline{\mathsf{CP}}}, HU, CU, usk[\cdot], I_{\mathrm{b}})$
    Return $(I_{\mathrm{b}}, St_{\mathsf{CP}})$
```

```
Algorithm $\mathsf{CP}(M_{\mathrm{in}}, St_{\mathsf{CP}})$
    Parse $St_{\mathsf{CP}}$ as $(St_{\overline{\mathsf{CP}}}, HU, CU, usk[\cdot], I_{\mathrm{b}})$
    If atk = pa then $(M_{\mathrm{out}}, St_{\overline{\mathsf{CP}}}) \xleftarrow{\$} \overline{\mathsf{CP}}(M_{\mathrm{in}}, St_{\overline{\mathsf{CP}}}$ : INIT-SIM, CORR-SIM, CONV-SIM$)$
    Else $(M_{\mathrm{out}}, St_{\overline{\mathsf{CP}}}) \xleftarrow{\$} \overline{\mathsf{CP}}(M_{\mathrm{in}}, St_{\overline{\mathsf{CP}}}$ : INIT-SIM, CORR-SIM, PROV-SIM$)$
    $St_{\mathsf{CP}} \leftarrow (St_{\overline{\mathsf{CP}}}, HU, CU, usk[\cdot], I_{\mathrm{b}})$
    Return $(M_{\mathrm{out}}, St_{\mathsf{CP}})$
```

Figure 22: Adversary $\mathsf{A} = (\mathsf{CP}, \mathsf{CV})$ attacking SI scheme $\mathcal{SI} = (\mathsf{Kg}, \mathsf{P}, \mathsf{V})$, and its subroutines, for the proof of Theorem 3.2. Above, OR is a conversation oracle if atk = pa and a prover oracle if atk $\in \{\mathrm{aa}, \mathrm{ca}\}$.

assigning $I_{\mathrm{g}}$ a value while simulating this oracle, as shown in the Figure). Now $\mathsf{CV}$ runs $\overline{\mathsf{CV}}$ on input $1^k, mpk$, simulating its oracles in such a way that the public key corresponding to $I_{\mathrm{g}}$ is $pk$. To do this, it invokes its own oracle (conversation if atk = pa or prover if atk = aa or atk = ca) to answer queries to the corresponding oracles of $\overline{\mathsf{CV}}$ when the identity queried is $I_{\mathrm{g}}$, appropriately inserting a certificate for $I_{\mathrm{g}}$ in the flows. For identities other than $I_{\mathrm{g}}$, it follows the scheme $\mathcal{C}ert\text{-}\mathcal{IBI}$, generating secret keys via its knowledge of *msk* and then using them. It aborts if asked to answer a query of $I_{\rm g}$ to the CORR oracle, since in this case it does not know the corresponding secret key. If $\mathsf{CV}$ correctly guesses $I_{\rm b}$, then $\mathsf{CP}$ uses $\overline{\mathsf{CP}}$, the latter playing the role of $I_{\rm b}$, to interact with $\mathsf{V}$ and try to convince $\mathsf{V}$ to accept.

For the analysis, consider $\mathbf{Exp}^{\text{imp-atk}}_{\mathit{Cert}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k)$. Let *pk* denote the public key assigned by INIT to $I_{\rm b}$, and let $\mathit{cert} = (\mathit{pk}_{I_{\rm b}}, \sigma)$ be the certificate sent by $\overline{\mathsf{CP}}$ to $\overline{\mathsf{V}}$ as part of its first flow in the identification protocol. Let $\mathbf{E}$ be the event that $\mathit{pk} = \mathit{pk}_{I_{\rm b}}$. If event $\mathbf{E}$ does not happen, then $\mathit{pk}_{I_{\rm b}} \| I_{\rm b}$ was never a query of $\mathsf{F}$ to its $\mathsf{Sign}(\mathit{sk}, \cdot)$ oracle. Thus, if $\overline{\mathsf{A}}$ was successful, so is $\mathsf{F}$. Thus:

$$\mathbf{Adv}^{\text{uf-cma}}_{\mathcal{SS},F}(k) \quad \geq \quad \Pr\left[\neg\mathbf{E} \wedge \mathbf{Exp}^{\text{imp-atk}}_{\mathit{Cert}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1\right]. \tag{9}$$

Let $\mathbf{G}$ be the event that $|HU| \geq q_{\rm g}$ and $I_{\rm g} = I_{\rm b}$ in $\mathbf{Exp}^{\text{imp-atk}}_{\mathcal{SI},\mathsf{A}}(k)$. If this event happens and $\overline{\mathsf{A}}$ succeeds then $I_{\rm g}$ cannot have been queried to CORR, and thus $\mathsf{A}$'s simulation of the environment of $\overline{\mathsf{A}}$ is perfect. In that case, $\mathsf{A}$ succeeds whenever $\overline{\mathsf{A}}$ succeeds and event $\mathbf{E}$ happens, for in that case it is attacking the public key *pk*. So we have

$$\begin{aligned}
\mathbf{Adv}^{\text{imp-atk}}_{\mathcal{SI},\mathsf{A}}(k) &\geq \Pr\left[\mathbf{G} \wedge \mathbf{E} \wedge \mathbf{Exp}^{\text{imp-atk}}_{\mathit{Cert}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1\right] \\
&= \Pr\left[\,\mathbf{G}\,\right] \cdot \Pr\left[\mathbf{E} \wedge \mathbf{Exp}^{\text{imp-atk}}_{\mathit{Cert}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1\right] \\
&\geq \frac{1}{\boldsymbol{Q}^{\text{INIT}}_{\overline{\mathsf{CV}}}(k)} \cdot \Pr\left[\mathbf{E} \wedge \mathbf{Exp}^{\text{imp-atk}}_{\mathit{Cert}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1\right].
\end{aligned} \tag{10}$$

Using Equations (9) and (10) we get

$$\begin{aligned}
\mathbf{Adv}^{\text{imp-atk}}_{\mathit{Cert}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) &= \Pr\left[\mathbf{Exp}^{\text{imp-atk}}_{\mathit{Cert}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1\right] \\
&= \Pr\left[\neg\mathbf{E} \wedge \mathbf{Exp}^{\text{imp-atk}}_{\mathit{Cert}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1\right] + \Pr\left[\mathbf{E} \wedge \mathbf{Exp}^{\text{imp-atk}}_{\mathit{Cert}\text{-}\mathcal{IBI},\overline{\mathsf{A}}}(k) = 1\right] \\
&\leq \mathbf{Adv}^{\text{uf-cma}}_{\mathcal{SS},F}(k) + \boldsymbol{Q}^{\text{INIT}}_{\overline{\mathsf{CV}}}(k) \cdot \mathbf{Adv}^{\text{imp-atk}}_{\mathcal{SI},\mathsf{A}}(k)\,,
\end{aligned}$$

which is Equation (8) as desired.