From `Selective-ID` to `Full-ID` IBS without Random Oracles

Sanjit Chatterjee and Chethan Kamath,

Indian Institute of Science, Bangalore.

{sanjit,chethan0510}@csa.iisc.ernet.in

November 2, 2013

Abstract

Since its induction, the selective-identity (sID) model for identity-based cryptosystems and its relationship with various other notions of security has been extensively studied. As a result, it is a general consensus that the sID model is much weaker than the full-identity (ID) model. In this paper, we study the sID model for the particular case of identity-based signatures (IBS). The main focus is on the problem of constructing an ID-secure IBS given an sID-secure IBS without using *random oracles*–the so-called *standard* model–and with reasonable security degradation. We accomplish this by devising a generic construction which uses as black-box: *i*) a *chameleon* hash function and *ii*) a *weakly*-secure public-key signature. We argue that the resulting IBS is ID-secure but with a tightness gap of $\mathrm{O}(q_s)$, where q_s is the upper bound on the number of signature queries that the adversary is allowed to make. To the best of our knowledge, this is the first attempt at such a generic construction.

Keywords: Identity-Based Signatures, Security Models, Selective-Iden-tity Security, Generic Chosen-Message Attack, Chameleon Hash Function.

Contents

1 Introduction

The concept of identity-based cryptosystems (IBC) was introduced by Shamir in 1984 [Sha85]. In IBC, any arbitrary string such as an e-mail address can act as the public key. In traditional

public-key cryptosystems (PKC), users have to exchange public-key certificates before being able to communicate securely. These certificates provide the external binding between the public key and the identity of a user. In some scenarios certificates can prove to be cumbersome. Using IBC one can avoid the complicated certificate management–this was Shamir's foresight.

Identity-based signatures. The notion of identity-based signatures (IBS) is an extension of the idea of digital signatures to the identity-based setting. As in traditional public-key signature (PKS) schemes, the signer uses her secret key to sign a message. However, the signature can be verified by anyone using the signer's identity and the master public key of the private-key generator[1] (PKG). IBS–or more generally, IBC–does not require any certificates to be exchanged and hence can be advantageous over the traditional PKI based systems in certain scenarios. IBS, in particular, turns out to be quite practical in wireless sensor-networks, BGP protocol, MANET routing *etc.*. Therefore, the question of designing efficient and secure IBS is an important problem in the context of applied cryptography. With the advent of pairings [BF01], the interest in IBS–and, of course, identity-based encryption (IBE) schemes–mushroomed, resulting in numerous efficient schemes [CHC02, Her05, Hes03].

The selective-identity model. The selective-identity (`sID`) model for identity-based cryptographic schemes was introduced in [CHK03]. The distinguishing feature of this model is that the adversary has to commit, beforehand, to a "target" identity–*i.e.*, the identity which it eventually forges on. Since its induction, the relationship of the `sID` notion with various other notions of security has been extensively studied [CS06, CFH$^+$09, Gal06, GH05]. One of the interesting results is the separation between the `sID` models and `ID` models for IBE in the standard model [Gal06]. Therefore, it is a general consensus that the `sID` model is much weaker than the full-identity (`ID`) model. However, it is *easier* to design efficient schemes, based on weaker assumptions, that are secure in the `sID` model compared to the `ID` model. This is, in particular, highlighted by the disparity in the construction of IBE schemes given in [BB04a] and [Wat05]. The former is simple and efficient, whereas the latter, involved. Therefore, a generic transformation from an `sID` scheme to `ID` scheme would be a problem worth pursuing. We could design efficient `sID`-secure schemes and then just bootstrap it to `ID`-security using the transformation. In fact, this is a long-standing open problem.

Existing techniques for constructing IBS. The task of constructing IBS is generally considered to be a much easier task than that of constructing IBE. [BNN04] contains a comprehensive list of such techniques, along with the security arguments. The "folklore" construction of IBS using two applications of PKS (certificate) is one of the well-known techniques. A PKS, on the other hand, can be derived from a *weakly*-secure PKS [GMR88] using a *chameleon* hash function (CHF) [KR00]. This approach was used implicitly in [ST01] and, later, formalised in [HW09]. The (existence of) two aforementioned techniques implies one can construct an IBS from a weakly-secure PKS and a CHF.

There exist techniques to construct (`ID`-secure) IBS from `sID`-secure IBS as well. An efficient (comparatively) black-box method to convert an `sID`-secure IBE scheme to `ID` security was suggested in [BF01]. But the method relies on *random oracles* [BR93]. This was followed by [BB04a], in which the problem is solved without using random oracles–in the so-called *standard* model–albeit with an *exponential* loss of tightness. Both these methods can be adapted to IBS.

Our Contribution. The primary focus of this paper is on the question of constructing an `ID`-secure IBS, given an `sID`-secure IBS, in the standard model, and with reasonable security degradation. We accomplish this through a generic transformation which uses a CHF and a

[1]The PKG is a trusted third party whose duty is to create and then communicate the secret keys to the users in the system through a secure channel.

weakly-secure PKS as black-box. We go one step further by applying the same construction technique to a *relaxed* notion of IBS security which we call the *weak* selective-identity (wID) model. The distinguishing feature of the wID model is that the adversary, apart from committing to the target identity, has to commit to a set of "query" identities–the set of identities which it wishes to query the signature and extract oracle with (see **§4** for the definition of the security model). Thus, we reduce the problem of constructing an ID-secure IBS to that of constructing wID-secure IBS, an EU-GCMA-secure PKS and a CHF. Our approach can be considered to be an alternative paradigm to the aforementioned folklore construction of IBS.

The security argument constitutes the main hurdle–the construction itself is quite straightforward. The line of argument, roughly, is: given an adversary that breaks the ID-IBS, we construct algorithms to break either the sID/wID-IBS, the PKS or the CHF. It leads to a tightness gap of $\mathrm{O}(q_s)$, where q_s is the upper bound on the number of signature queries that the adversary is allowed to make.

Organisation. We start with the formal definitions in §**2**. The generic transformation, along with its security argument and analysis, is given in **§3**. In **§4** we show that our construction can be used to construct an ID-secure IBS from a wID-secure IBS. Finally, we conclude with some remarks in **§5**.

2 Definitions

2.1 Public-Key Signatures

Definition 1 (Public-Key Signature)**.** A public-key signature (PKS) consists of three *polynomial-time non-deterministic* algorithms $\{\mathcal{K}, \mathcal{S}, \mathcal{V}\}$.

> **Key Generation**, $\mathcal{K}(\kappa)$: It takes as input the security parameter κ and outputs the public key pk and the secret key sk.
>
> **Signing**, $\mathcal{S}(m, \mathtt{sk})$: It takes as input a message m and the secret key of the user sk to generate a signature σ.
>
> **Verification**, $\mathcal{V}(\sigma, m, \mathtt{pk})$: It takes as input the signature σ, the message m and the public key of the user pk. It outputs the b which is 1 if σ is valid signature on m or 0 if the signature is invalid.

The standard *correctness* condition: $1 \leftarrow \mathcal{V}(\mathcal{S}(m, \mathtt{sk}), m, \mathtt{pk})$, should be satisfied for all m and $(\mathtt{pk}, \mathtt{sk}) \xleftarrow{\$} \mathcal{K}(\kappa)$.

Security Notions. The standard security notion for PKS schemes is *existential unforgeability under chosen-message attack* (EU-CMA) [GMR88].

Definition 2 (EU-CMA Game)**.** The security of a PKS scheme in the EU-CMA model is argued in terms of the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

> **Set-up**: $\mathcal{C}$ invokes $\mathcal{K}$ to obtain the public key pk and the secret key sk. $\mathcal{A}$ is given the public key but the secret key is kept by $\mathcal{C}$.
>
> **Signature queries**: $\mathcal{A}$ can adaptively make signature queries to an oracle $\mathcal{O}s$. For a query on a message m, $\mathcal{C}$ responds by running $\mathcal{S}$ on m to obtain a signature σ, which is forwarded to $\mathcal{A}$.
>
> **Forgery**: $\mathcal{A}$ outputs a signature $\hat{\sigma}$ on a message $\hat{m}$ and wins the game if

1. $\hat{\sigma}$ is a valid signature on $\hat{m}$.
2. $\mathcal{A}$ has not made a signature query on $\hat{m}$.

The advantage $\mathcal{A}$ has in the above game, denoted by $\mathrm{Adv}_{\mathcal{A}}^{\mathtt{EU-CMA}}(\kappa)$, is defined as the probability with which it wins the above game, *i.e.*

$$\Pr\left[1 \leftarrow \mathcal{V}(\hat{\sigma}, \hat{m}, \mathtt{pk}) \mid (\mathtt{sk}, \mathtt{pk}) \xleftarrow{\$} \mathcal{K}(\kappa); (\hat{\sigma}, \hat{m}) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}^s}(\mathtt{pk})\right]$$

provided $\hat{\sigma}$ is a valid forgery. An adversary $\mathcal{A}$ is said to be an (ϵ, t, q_s)-forger of a PKS scheme if it has advantage of at least ϵ in the above game, runs in time at most t and makes at most q_s signature queries.

A weaker notion of security for PKS is *existential forgery under generic chosen-message attack* (`EU-GCMA`) [GMR88]. In the `EU-GCMA` model, the adversary initially commits to a set of messages $m_1, \ldots, m_{q_s}$ to the challenger. Next, it is given the signatures correspond to the committed messages along with the public key to be attacked. The adversary finally outputs a forgery on a message that was not part of the committed set. A more formal definition follows.

Definition 3 (`EU-GCMA` Game)**.** The security of a PKS scheme in the `EU-GCMA` model is argued in terms of the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Commitment: $\mathcal{A}$ commits to a set of messages $\tilde{\mathbb{M}} := \{m_1, \ldots, m_{q_s}\}$.

Set-up: $\mathcal{C}$ invokes $\mathcal{G}$ to obtain the public key `pk` and the secret key `sk`. $\mathcal{A}$ is given the public key but the secret key is kept by $\mathcal{C}$.

Signing: $\mathcal{C}$ invokes the signing algorithm $\mathcal{S}$ on each $m_i \in \tilde{\mathbb{M}}$ to generate signatures $\sigma_1, \ldots, \sigma_{q_s}$ and passes them on to $\mathcal{A}$.

Forgery: $\mathcal{A}$ outputs a signature $\hat{\sigma}$ on a message $\hat{m}$ and wins the game if

1. $\hat{\sigma}$ is a valid signature on $\hat{m}$.
2. $\hat{m}$ was not a part of the committed set $\tilde{\mathbb{M}}$.

The advantage $\mathcal{A}$ has in the above game, denoted by $\mathrm{Adv}_{\mathcal{A}}^{\mathtt{EU-GCMA}}(\kappa)$, is defined as the probability with which it wins the above game, *i.e.*

$$\Pr\big[1 \leftarrow \mathcal{V}(\hat{\sigma}, \hat{m}, \mathtt{pk}) \wedge \hat{m} \notin \tilde{\mathbb{M}} \mid \tilde{\mathbb{M}} \xleftarrow{\$} \mathcal{A}(q_s); (\mathtt{sk}, \mathtt{pk}) \xleftarrow{\$} \mathcal{K}(\kappa); (\hat{\sigma}, \hat{m}) \xleftarrow{\$} \mathcal{A}(\mathtt{pk}, \sigma_1, \ldots, \sigma_{q_s})\big]$$

An adversary $\mathcal{A}$ is said to be an (ϵ, t, q_s)-forger of a PKS scheme if it has advantage of at least ϵ in the above game, runs in time at most t, after initially committing to a set of q_s messages.

2.2 Identity-Based Signatures

Definition 4 (Identity-Based Signature)**.** An IBS scheme consists of four *polynomial-time non-deterministic* algorithms $\{\mathcal{G}, \mathcal{E}, \mathcal{S}, \mathcal{V}\}$ described below.

Set-up, $\mathcal{G}(\kappa)$: It takes as input the security parameter κ. It outputs the master secret key `msk` and the master public key `mpk`.

Key Extraction, $\mathcal{E}(\mathtt{id}, \mathtt{msk})$: It takes as input the user's identity `id`, the master secret key `msk` to generate the secret key `usk` of a user.

Signing, $\mathcal{S}(\mathtt{id}, m, \mathtt{usk})$: It takes as input the user's identity $\mathtt{id}$, a message m and the user's secret key $\mathtt{usk}$ to generate a signature σ.

Verification, $\mathcal{V}(\sigma, \mathtt{id}, m, \mathtt{mpk})$: It takes as input a signature σ, a message m, an identity $\mathtt{id}$ and master public key $\mathtt{mpk}$. It outputs the result bit $\mathtt{b}$ which is 1 if σ is a valid signature on $(\mathtt{id}, m)$ or 0 if the signature is invalid.

The standard *correctness* condition: $1 \leftarrow \mathcal{V}(\mathcal{S}(\mathtt{id}, m, \mathtt{usk}), \mathtt{id}, m, \mathtt{mpk})$, should be satisfied for all $\mathtt{id}$, m, $(\mathtt{msk}, \mathtt{mpk}) \xleftarrow{\$} \mathcal{G}(\kappa)$ and $\mathtt{usk} \xleftarrow{\$} \mathcal{E}(\mathtt{id}, \mathtt{msk})$.

Security Notions. We use the standard security notion for IBS schemes given in [BNN04]. In addition, we also describe the weaker selective-identity notion of IBS security.

Definition 5 ($\mathtt{EU\text{-}ID\text{-}CMA}$ Game[2])**.** The security of an IBS scheme in the $\mathtt{EU\text{-}ID\text{-}CMA}$ model is argued in terms of the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Set-up: $\mathcal{C}$ invokes $\mathcal{G}$ to obtain master public key $\mathtt{mpk}$ and the master secret key $\mathtt{msk}$. $\mathcal{A}$ is given master public key but the master secret key is kept by $\mathcal{C}$.

Queries: $\mathcal{A}$ can adaptively make extract queries to an oracle $\mathcal{O}\varepsilon$ and signature queries to an oracle $\mathcal{O}s$. These queries are handled as follows.

Extract query, $\mathcal{O}\varepsilon(\mathtt{id})$: $\mathcal{A}$ asks for the secret key of a user with identity $\mathtt{id}$. If there has already been an extract query on $\mathtt{id}$, $\mathcal{C}$ returns the user secret key that was generated during the earlier query. Otherwise, $\mathcal{C}$ uses the knowledge of $\mathtt{msk}$ to run $\mathcal{E}$ and generate the user secret key $\mathtt{usk}$, which is passed on to $\mathcal{A}$.

Signature query, $\mathcal{O}s(\mathtt{id}, m)$: $\mathcal{A}$ asks for the signature of a user with identity $\mathtt{id}$ on a message m. $\mathcal{C}$ first generates a user secret key for $\mathtt{id}$, as in the extract query. Next, it uses the knowledge of $\mathtt{usk}$ to run $\mathcal{S}$ and generate a signature σ, which is passed to $\mathcal{A}$.

Forgery: $\mathcal{A}$ outputs a signature $\hat{\sigma}$ on an identity $\hat{\mathtt{id}}$ and a message $\hat{m}$, and wins the game if

1. $\hat{\sigma}$ is a valid signature on $\hat{m}$ by $\hat{\mathtt{id}}$.
2. $\mathcal{A}$ has not made an extract query on $\hat{\mathtt{id}}$.
3. $\mathcal{A}$ has not made a signature query on $(\hat{\mathtt{id}}, \hat{m})$.

The advantage $\mathcal{A}$ has in the above game, denoted by $\mathrm{Adv}_{\mathcal{A}}^{\mathtt{EU-ID-CMA}}(\kappa)$, is defined as the probability with which it wins the above game, *i.e.*

$$\Pr\left[1 \leftarrow \mathcal{V}(\hat{\sigma}, \hat{\mathtt{id}}, \hat{m}, \mathtt{mpk}) \mid (\mathtt{msk}, \mathtt{mpk}) \xleftarrow{\$} \mathcal{G}(\kappa); (\hat{\sigma}, \hat{\mathtt{id}}, \hat{m}) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}\varepsilon, \mathcal{O}s}(\mathtt{mpk})\right]$$

provided $\hat{\sigma}$ is a valid forgery on $(\hat{\mathtt{id}}, \hat{m})$. An adversary $\mathcal{A}$ is said to be an $(\epsilon, t, q_\varepsilon, q_s)$-forger of an IBS scheme if it has advantage of at least ϵ in the above game, runs in time at most t and makes at most q_ε and q_s extract and signature queries respectively. If the security argument uses the random oracle methodology [BR93], the adversary is also allowed to make queries to the random oracle(s).

Definition 6 ($\mathtt{EU\text{-}sID\text{-}CMA}$ Game)**.** The security of an IBS scheme in the $\mathtt{EU\text{-}sID\text{-}CMA}$ model is argued in terms of the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

[2]The security game in [BNN04], *i.e.* $\mathrm{Exp}^{\text{uf-cma}}_{\text{IBS},\text{F}}$, is explained in terms of the three oracles: INIT, CORR and SIGN. But we use a slightly simpler, but equivalent, formulation using the two oracles: $\mathcal{O}\varepsilon$ and $\mathcal{O}s$.

Commitment: $\mathcal{A}$ commits to a target identity $\tilde{\texttt{id}}$.

Set-up: $\mathcal{C}$ runs the set-up algorithm $\mathcal{G}$ to obtain the master keys (mpk,msk). It passes mpk as the challenge master public key to $\mathcal{A}$.

Queries: $\mathcal{A}$ can adaptively make extract queries to an oracle $\mathcal{O}\varepsilon$ and signature queries to an oracle $\mathcal{O}s$. It is restricted though from making extract query on the target identity $\hat{\texttt{id}}$. These queries are handled as follows.

Extract query, $\mathcal{O}\varepsilon(\texttt{id})$: $\mathcal{A}$ asks for the secret key of a user with identity id. $\mathcal{C}$ responds by running $\mathcal{E}$ and passes the secret key usk to $\mathcal{A}$.

Signature query, $\mathcal{O}s(\texttt{id}, m)$: $\mathcal{A}$ asks for the signature of a user with identity id on a message m. $\mathcal{C}$ responds by first running $\mathcal{E}$ on id to obtain the secret key usk of the user and then running $\mathcal{S}$ to obtain a signature σ, which is forwarded to $\mathcal{A}$.

Forgery: $\mathcal{A}$ outputs a signature $\hat{\sigma}$ on a message $\hat{m}$ by identity $\hat{\texttt{id}}$, and wins the game if

1. $\hat{\sigma}$ is a valid signature on $\hat{m}$ by $\tilde{\texttt{id}}$.
2. $\mathcal{A}$ has not made a signature query on $(\tilde{\texttt{id}}, \hat{m})$.

The advantage $\mathcal{A}$ has in the above game, denoted by $\text{Adv}_{\mathcal{A}}^{\texttt{EU-sID-CMA}}(\kappa)$, is defined as the probability with which it wins the game, *i.e.*

$$\Pr\big[1 \leftarrow \mathcal{V}(\hat{\sigma}, \hat{\texttt{id}}, \hat{m}, \texttt{mpk}) \mid \tilde{\texttt{id}} \xleftarrow{\$} \mathcal{A}; (\texttt{msk}, \texttt{mpk}) \xleftarrow{\$} \mathcal{G}(\kappa); (\hat{\sigma}, \hat{\texttt{id}}, \hat{m}) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}\varepsilon, \mathcal{O}s}(\texttt{mpk})\big]$$

An adversary $\mathcal{A}$ is said to be an $(\epsilon, t, q_\varepsilon, q_s)$-forger of an IBS scheme in the EU-sID-CMA model if it has advantage of at least ϵ in the above game, runs in time at most t and makes at most q_ε and q_s extract and signature queries respectively.

2.3 Chameleon Hash Function

A chameleon hash function (CHF) is a randomised trapdoor hash function. Apart from the *collision resistance* property, it has an additional "chameleon" property which enables anyone with the trapdoor information to efficiently generate collisions.

Definition 7 (Chameleon Hash Function [BCC88, KR00, Moh11])**.** A family of CHF $\mathfrak{H}$ consists of three *polynomial-time* algorithms $\{\mathcal{G}, \text{h}, \text{h}^{-1}\}$ described below.

Key Generation, $\mathcal{G}(\kappa)$: It takes as input the security parameter κ. It outputs the evaluation key ek and the trapdoor key td.

Hash Evaluation, $\text{h}(\texttt{ek}, m, r)$: It takes as input the evaluation key ek, a message m from the message-space $\mathbb{M}$ and a randomiser r from the domain $\mathbb{R}$. It outputs the hash value y from the range $\mathbb{Y}$.

Collision Generation, $\text{h}^{-1}(\texttt{td}, m, r, m')$: It takes as input the trapdoor key td, two messages $m, m' \in \mathbb{M}$ and $r \in \mathbb{R}$. It outputs $r' \in \mathbb{R}$ such that $\text{h}(\texttt{ek}, m, r) = \text{h}(\texttt{ek}, m', r')$; in other words, (m, r) and (m', r') is a collision.

Any CHF should satisfy the following two properties.

(i) *Uniformity.* The distribution induced by $\text{h}(\texttt{ek}, m, r)$ for all messages m and a randomly chosen r should be the same. In other words, the distributions $(\texttt{ek}, \text{h}(\texttt{ek}, m, r))$ and $(\texttt{ek}, y)$ should be computationally indistinguishable, where $(\texttt{ek}, \texttt{td}) \xleftarrow{\$} \mathcal{G}(\kappa), r \in_R \mathbb{R}$ and $y \in_R \mathbb{Y}$.

(ii) *Collision Resistance.* Given the evaluation key $\mathtt{ek}$, it should be hard to compute a pair $(m, r) \neq (m', r')$ such that $\mathrm{h}(\mathtt{ek}, m, r) = \mathrm{h}(\mathtt{ek}, m', r')$, *i.e.* the probability given below should be negligible for all *polynomial-time non-deterministic* adversaries $\mathcal{A}$.

$$\Pr\left[\mathrm{h}(\mathtt{ek}, m, r) = \mathrm{h}(\mathtt{ek}, m', r') \wedge (m, r) \neq (m', r') \mid (\mathtt{ek}, \mathtt{td}) \xleftarrow{\$} \mathcal{G}(\kappa); (m, r, m', r') \xleftarrow{\$} \mathcal{A}(\mathtt{ek})\right]$$

3 The Generic Transformation

The transformation takes as input: *i*) an $\mathtt{sID}$-secure IBS $\mathfrak{I}_s := \{\mathcal{G}_s, \mathcal{E}_s, \mathcal{S}_s, \mathcal{V}_s\}$; *ii*) an $\mathtt{EU\text{-}GCMA}$-secure PKS $\mathfrak{P} := \{\mathcal{K}, \mathcal{S}_p, \mathcal{V}_p\}$; and *iii*) a CHF $\mathfrak{H} := \{\mathcal{G}_h, \mathrm{h}, \mathrm{h}^{-1}\}$, to output an $\mathtt{ID}$-secure IBS $\mathfrak{I} := \{\mathcal{G}, \mathcal{E}, \mathcal{S}, \mathcal{V}\}$. The basic idea is to *map* an identity $\mathtt{id}$ in $\mathfrak{I}$ to an identity $\mathtt{id}_s$ in $\mathfrak{I}_s$ using the CHF. These two identities are then *bound* by using the PKS. A formal description follows.

Assumptions. We denote the identity-space of $\mathfrak{I}_s$ (and that of resulting $\mathfrak{I}$) by $\mathbb{I}$ and its message-space by $\mathbb{M}$. For simplicity, we assume that *i*) the message-space of $\mathfrak{P}$ *ii*) the message-space of $\mathfrak{H}$ (denoted by $\mathbb{M}_h$) and *iii*) the range of $\mathfrak{H}$ (denoted by $\mathbb{Y}$) *are all* the same set $\mathbb{I}$, *i.e.*, $\mathbb{M}_h = \mathbb{Y} = \mathbb{I}$.[3] In addition, the randomness space of $\mathfrak{H}$ is denoted by $\mathbb{R}$. Therefore, for a particular evaluation key $\mathtt{ek}$, the hash evaluation algorithm can be considered as a function $\mathrm{h} : \mathbb{I} \times \mathbb{R} \rightarrow \mathbb{I}$. The description of the transformation is given in **Figure 1**. It is followed by the argument that the resultant IBS $\mathfrak{I}$ is secure in the $\mathtt{ID}$ model.

$\mathfrak{I} \leftarrow \mathfrak{T}(\mathfrak{I}_s, \mathfrak{P}, \mathfrak{H})$

Set-up, $\mathcal{G}(\kappa)$: Invoke the algorithms $\mathcal{G}_s$, $\mathcal{K}$ and $\mathcal{G}_h$ (all) on κ to obtain $(\mathtt{msk}_s, \mathtt{mpk}_s)$, $(\mathtt{sk}, \mathtt{pk})$ and $(\mathtt{ek}, \mathtt{td})$ respectively. Return $\mathtt{msk} := (\mathtt{msk}_s, \mathtt{sk})$ as the master secret key and $\mathtt{mpk} := (\mathtt{mpk}_s, \mathtt{pk}, \mathtt{ek})$ as the master public key.

Key Extraction, $\mathcal{E}(\mathtt{id}, \mathtt{msk})$: Select $r \in_R \mathbb{R}$ and compute $\mathtt{id}_s \leftarrow \mathrm{h}(\mathtt{ek}, \mathtt{id}, r)$. Next, run $\mathcal{E}_s(\mathtt{id}_s, \mathtt{msk}_s)$ and $\mathcal{S}_p(\mathtt{id}_s, \mathtt{sk})$ to obtain $\mathtt{usk}_s$ and σ_p respectively. Finally, return $\mathtt{usk} := (\mathtt{usk}_s, r, \sigma_p)$ as the user secret key.

Signing, $\mathcal{S}(\mathtt{id}, m, \mathtt{usk})$: Parse the user secret key $\mathtt{usk}$ as $(\mathtt{usk}_s, r, \sigma_p)$ and compute $\mathtt{id}_s \leftarrow \mathrm{h}(\mathtt{ek}, \mathtt{id}, r)$. Next, run $\mathcal{S}_s(\mathtt{id}_s, m, \mathtt{usk}_s)$ to obtain σ_s. Finally, return $\sigma := (\sigma_s, r, \sigma_p)$ as the signature.

Verification, $\mathcal{V}(\sigma, \mathtt{id}, m, \mathtt{mpk})$: Parse σ as (σ_s, r, σ_p) and compute $\mathtt{id}_s \leftarrow \mathrm{h}(\mathtt{ek}, \mathtt{id}, r)$. Return 1 only if σ_p is a valid signature on $\mathtt{id}_s$ and σ_s is a valid signature on $(\mathtt{id}_s, m)$. In other words, if $\mathtt{b}_p \leftarrow \mathcal{V}_p(\sigma_p, \mathtt{id}_s, \mathtt{pk})$ and $\mathtt{b}_s \leftarrow \mathcal{V}_s(\sigma_s, \mathtt{id}_s, m, \mathtt{mpk})$, return $(\mathtt{b}_p \wedge \mathtt{b}_s)$.

Figure 1: Constructing $\mathtt{ID}$-secure IBS from an $\mathtt{sID}$-secure IBS.

The **Hash Evaluation** function h is used to map an identity $\mathtt{id}$ in $\mathfrak{I}$ on to an identity $\mathtt{id}_s$ in $\mathfrak{I}_s$. The mapped identities are then bound using σ_p. This is reflected in the structure of the user secret key for $\mathtt{id}$ which is of the form $(\mathtt{usk}_s, r, \sigma_p)$.

[3] This assumption can be relaxed–to accommodate a CHF with $\mathbb{M}_h \neq \mathbb{Y} \neq \mathbb{I}$–using two collision resistant hash functions H and G defined as follows:

$$\mathrm{H} : \mathbb{I} \rightarrow \mathbb{M}_h \quad \text{and} \quad \mathrm{G} : \mathbb{Y} \rightarrow \mathbb{I}$$

These hash functions can be used in the protocol, and also in the security argument, to couple the CHF with the IBS.

Remark 1. Note that we have omitted td from the master secret key. The trapdoor key td–hence the collision generation function h^{-1}–is not used *per se* in the transformation. However, it *does* play a crucial role in its security argument.

3.1 Security Argument

For simplicity, we consider the security of the specific case of EU-sID-CMA model; we argue that the resulting IBS is EU-ID-CMA-secure. The details of both the security models is given in **§2.2**. The line of argument can be easily extended to other models as well[4].

Theorem 1. *Given an $(\epsilon, t, q_\varepsilon, q_s)$-adversary $\mathcal{A}$, in the* EU-ID-CMA *model, agai-nst the IBS $\mathfrak{I}$, we can construct either*

(i) Algorithm $\mathcal{B}_s$ which $(\epsilon_s, t_s, q_\varepsilon, q_s)$-breaks $\mathfrak{I}_s$ in the EU-sID-CMA *model, where*

$$\epsilon_s \geq \frac{1}{3q_s}\epsilon \quad \textit{and} \quad t_s \leq t + (q_\varepsilon + q_s)\tau_1, \quad \textit{or}$$

(ii) Algorithm $\mathcal{B}_p$ which $(\epsilon_p, t_p, q_\varepsilon + q_s)$-breaks $\mathfrak{P}$ in the EU-GCMA *model, where*

$$\epsilon_p = \frac{1}{3}\epsilon \quad \textit{and} \quad t_p \leq t + (q_\varepsilon\tau_2 + q_s\tau_3), \quad \textit{or}$$

(iii) Algorithm $\mathcal{B}_h$ which (ϵ_h, t_h)-breaks $\mathfrak{H}$, where

$$\epsilon_h = \frac{1}{3}\epsilon \textit{ and } \quad t_h \leq t + (q_\varepsilon + q_s)\tau_1 + (q_\varepsilon\tau_2 + q_s\tau_3).$$

Here, q_ε ([resp.]q_s) denotes the upper bound on the number of extract ([resp. s] ignature) queries that $\mathcal{A}$ can make. τ_1 is the time taken for generating a signature in $\mathfrak{P}$; τ_2 ([resp.]τ_3) denotes the time taken to generate a user secret key ([resp. s] ignature) in $\mathfrak{I}_s$.

Proof. We classify the forgeries (mutually-exclusively and exhaustively) into three: type 1,type 2 and type 3. A forgery qualifies as type 1 if the adversary makes *at least* one signature query on the "target" identity–the identity which $\mathcal{A}$ eventually forges on–and produces a forgery with the binding (provided by the simulator) *intact*. In both type 2 and type 3 forgeries, the binding is violated by the adversary by some means. The strategy adopted in each of the three cases is different; we give a reduction $\mathcal{B}_s$ for type 1, $\mathcal{B}_p$ for type 2 and $\mathcal{B}_h$ for type 3 adversary. The details follow.

Classifying the forgery. Consider an adversary $\mathcal{A}$ in the EU-ID-CMA model. At the beginning of the security game, $\mathcal{A}$ is given the challenge master public key mpk by (its challenger) $\mathcal{C}$. $\mathcal{A}$ produces a forgery after making a series of queries–extract and signature–adaptively with $\mathcal{C}$. Let $\dot{\mathtt{id}}_i$ denote the i^{th} extract query made by $\mathcal{A}$, which is responded to with $\mathtt{usk}_i = (\mathtt{usk}_{s,i}, \dot{r}_i, \dot{\sigma}_{p,i})$ by $\mathcal{C}$. Similarly, $(\mathtt{id}_i, m_i)$ denotes the i^{th} signature query by $\mathcal{A}$, which is responded to with $\sigma_i = (\sigma_{s,i}, r_i, \sigma_{p,i})$ by $\mathcal{C}$. Note that the number of extract ([resp. s] ignature) queries is bounded by q_ε ([resp.]q_s). Finally, let $\hat{\sigma} = (\hat{\sigma}_s, \hat{r}, \hat{\sigma}_p)$ be the forgery produced by $\mathcal{A}$ on $(\hat{\mathtt{id}}, \hat{m})$. The identity $\hat{\mathtt{id}}$ is the so-called target identity. The forgeries can be partitioned into three types, *viz.*:

(i) type 1 forgery. $\mathcal{A}$ produces the forgery with $(\hat{\mathtt{id}}, \hat{r}) = (\mathtt{id}_i, r_i)$ for some $i \in \{1, \ldots, q_s\}$.

[4] *e.g.*, consider the sM-sID-CMA model–the *selective-message, selective-identity chosen-message attack* model. It is similar to the EU-sID-CMA model, except that the adversary–in addition to committing to the target identity–has to commit to the target message too. If we start from sM-sID-CMA-secure IBS, we end up with an sM-ID-CMA-secure IBS.

(ii) type 2 forgery. $\mathcal{A}$ produces the forgery with $(\hat{\mathtt{id}}, \hat{r}) \neq (\mathtt{id}_i, r_i)$ for all $i \in \{1, \ldots, q_s\}$, and with

(a) $\mathrm{h}(\mathtt{ek}, \hat{\mathtt{id}}, \hat{r}) \neq \mathrm{h}(\mathtt{ek}, \dot{\mathtt{id}}_i, \dot{r}_i)$ for all $i \in \{1, \ldots, q_\varepsilon\}$, and

(b) $\mathrm{h}(\mathtt{ek}, \hat{\mathtt{id}}, \hat{r}) \neq \mathrm{h}(\mathtt{ek}, \mathtt{id}_i, r_i)$ for all $i \in \{1, \ldots, q_s\}$.

(iii) type 3 forgery. $\mathcal{A}$ produces the forgery with $(\hat{\mathtt{id}}, \hat{r}) \neq (\mathtt{id}_i, r_i)$ for all $i \in \{1, \ldots, q_s\}$, but with

(a) $\mathrm{h}(\mathtt{ek}, \hat{\mathtt{id}}, \hat{r}) = \mathrm{h}(\mathtt{ek}, \dot{\mathtt{id}}_i, \dot{r}_i)$ for some $i \in \{1, \ldots, q_\varepsilon\}$, or

(b) $\mathrm{h}(\mathtt{ek}, \hat{\mathtt{id}}, \hat{r}) = \mathrm{h}(\mathtt{ek}, \mathtt{id}_i, r_i)$ for some $i \in \{1, \ldots, q_s\}$.

If $\mathcal{A}$ produces a forgery of type 1, we construct an algorithm $\mathcal{B}_s$ which breaks the IBS scheme $\mathfrak{I}_s$; whereas, in case of type 2 forgery, we construct an algorithm $\mathcal{B}_p$ which breaks the PKS scheme $\mathfrak{P}$; and finally, in case of type 3 forgery, we construct an algorithm $\mathcal{B}_h$ that breaks collision resistance property of the CHF $\mathfrak{H}$. We describe these reductions in the subsequent sections. □

3.1.1 Reduction $\mathcal{B}_s$.

Recall that in type 1 forgeries, $\mathcal{A}$ makes at least one signature query on the target identity $\hat{\mathtt{id}}$. The strategy is to guess the index of this identity and map it to the identity that $\mathcal{B}_s$ *commits* to (initially) in the EU-sID-CMA game. This leads to a degradation of $\mathrm{O}(q_s)$.

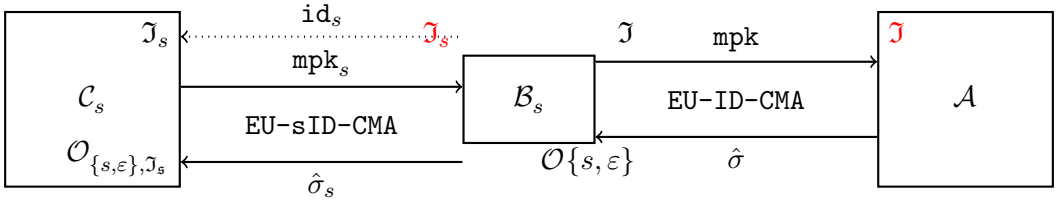

Figure 2: Reduction $\mathcal{B}_s$

Let $\mathcal{C}_s$ be the challenger in the EU-sID-CMA game. $\mathcal{B}_s$ plays the role of the adversary in the EU-sID-CMA game and, at the same time, the role of the challenger to $\mathcal{A}$ in the EU-ID-CMA game (see **Figure 2**). $\mathcal{B}_s$ starts by running the **Key Generation** algorithms $\mathcal{K}$ and $\mathcal{G}_h$ to obtain $(\mathtt{pk}, \mathtt{sk})$ and $(\mathtt{ek}, \mathtt{td})$ respectively. In order to initiate the EU-sID-CMA game, $\mathcal{B}_s$ has to commit to a target identity. It does so by selecting an identity $\tilde{\mathtt{id}} \in_R \mathbb{I}$ and a randomiser $\tilde{r} \in_R \mathbb{R}$, and committing $\tilde{\mathtt{id}}_s \leftarrow \mathrm{h}(\mathtt{ek}, \tilde{\mathtt{id}}, \tilde{r})$ to $\mathcal{C}_s$. As a result, $\mathcal{C}_s$ releases the challenge master public key $\mathtt{mpk}_s$ to $\mathcal{B}_s$. $\mathcal{B}_s$ is also allowed access to a signature oracle $\mathcal{O}_{s,\mathfrak{I}_s}$. Now, $\mathcal{B}_s$ passes $\mathtt{mpk} := (\mathtt{mpk}_s, \mathtt{pk}, \mathtt{ek})$ as its own challenge master public key to $\mathcal{A}$. Next, $\mathcal{B}_s$ guesses $1 \leq \tilde{\ell} \leq q_s$ as the index of the target identity.

Mapping the identities. In order to track the mapping between the identities in $\mathfrak{I}$ and $\mathfrak{I}_s$, $\mathcal{B}_s$ maintains a table $\mathfrak{L}$. It also maintains a counter ℓ (initially 1) to track the index of these identities. $\mathfrak{L}$ contains tuples of the form $\langle \mathtt{id}, \mathtt{id}_s, \ell, \mathtt{usk} \rangle$. Here, $\mathtt{id}$ and $\mathtt{id}_s$ are the related identities from $\mathfrak{I}$ and $\mathfrak{I}_s$ respectively; ℓ is the index of the identity $\mathtt{id}$. The $\mathtt{usk}$-field stores the user secret key for $\mathtt{id}$ and hence contains elements of the form $(\mathtt{usk}_s, r, \sigma_p)$. If any component of the $\mathtt{usk}$-field is yet to be generated, it is indicated by a '$\perp$'.

An identity $\mathtt{id}$ has already been mapped if there exists $\langle \mathtt{id}_i, \mathtt{id}_{s,i}, \ell_i, \mathtt{usk} \rangle$ in $\mathfrak{L}$ such that $\mathtt{id}_i = \mathtt{id}$. For mapping a *fresh* identity $\mathtt{id}$, $\mathcal{B}_s$ chooses $r \in_R \mathbb{R}$ and sets $\mathtt{id}_s \leftarrow \mathrm{h}(\mathtt{ek}, \mathtt{id}, r)$.[5]

[5]If there already exists a tuple $\langle \mathtt{id}_i, r_i, \mathtt{id}_{s,i}, \ell_i, \mathtt{usk}_i \rangle$ such that $\mathtt{id}_{s,i} = \mathtt{id}_s$, to maintain injection in the mapping, $\mathcal{B}_s$ repeats the process with a fresh r.

Finally, it adds $\langle \mathtt{id}, \mathtt{id}_s, \ell, (\perp, r, \perp)\rangle$ to $\mathfrak{L}$ and increments ℓ by one. A more formal description of the mapping function is given below.

$\mathrm{M}_s(\mathtt{id})$:
if $\exists$ a tuple $\langle \mathtt{id}_i, \mathtt{id}_{s,i}, \ell_i, \mathtt{usk}_i\rangle \in \mathfrak{L}$ such that $(\mathtt{id}_i = \mathtt{id})$ **then**
  Set $\tau := (\mathtt{id}_{s,i}, \ell_i, \mathtt{usk}_i)$
else
  if $(\ell = \tilde{\ell})$ **then return** set $r \leftarrow \mathrm{h}^{-1}(\mathtt{td}, \tilde{\mathtt{id}}, \tilde{r}, \mathtt{id})$
  else return choose $r \in_R \mathbb{R}$
  Compute $\mathtt{id}_s \leftarrow \mathrm{h}(\mathtt{ek}, \mathtt{id}, r)$ and set $\tau := (\mathtt{id}_s, \ell, (\perp, r, \perp))$
  Add $\langle \mathtt{id}, \mathtt{id}_s, \ell, (\perp, r, \perp)\rangle$ to $\mathfrak{L}$ and increment ℓ by one
end if
return τ

Queries. The extract and signature queries by $\mathcal{A}$ are answered as per the following specifications.

Extract query, $\mathcal{O}_{\varepsilon,\mathfrak{I}}(\mathtt{id})$: Invoke $\mathrm{M}_s(\mathtt{id})$ to obtain $(\mathtt{id}_s, \ell, (\mathtt{usk}_s, r, \sigma_p))$.

(i) If $(\ell = \tilde{\ell})$ then $\mathcal{B}_s$ *aborts* (abort_1).

(ii) Otherwise, if $(\mathtt{usk}_s \neq \perp)$ then return $\mathtt{usk} := (\mathtt{usk}_s, r, \sigma_p)$ as the user secret key.

(iii) Otherwise, $\mathcal{B}_s$ makes an extract query with $\mathcal{O}_{\varepsilon,\mathfrak{I}_s}$ on $\mathtt{id}_s$ to obtain $\mathtt{usk}_s$. Next, it uses the knowledge of $\mathtt{sk}$ to compute $\sigma_p := \mathcal{S}_p(\mathtt{id}_s, \mathtt{sk})$. Finally, it returns $\mathtt{usk} := (\mathtt{usk}_s, r, \sigma_p)$ as the user secret key and updates the $\mathtt{usk}$-field of the tuple corresponding to $\mathtt{id}$ in $\mathfrak{L}$.

Signature query, $\mathcal{O}_{s,\mathfrak{I}}(\mathtt{id}, m)$: Invoke $\mathrm{M}_s(\mathtt{id})$ to get $(\mathtt{id}_s, \ell, (\mathtt{usk}_s, r, \sigma_p))$.

(i) If $((\ell = \tilde{\ell}) \vee (\mathtt{usk}_s = \perp))$ then $\mathcal{B}_s$ makes a signature query with $\mathcal{O}_{s,\mathfrak{I}_s}$ on $(\mathtt{id}_s, m)$ to obtain σ_s. It uses the knowledge of $\mathtt{sk}$ to compute $\sigma_p := \mathcal{S}_p(\mathtt{id}_s, \mathtt{sk})$. Finally, it returns $\sigma := (\sigma_s, r, \sigma_p)$ as the signature.

(ii) Otherwise, $\mathcal{B}_s$ uses the knowledge of the user secret key $\mathtt{usk}$ to generate the signature, *i.e.* it returns $\sigma := \mathcal{S}(\mathtt{id}, m, \mathtt{usk})$.

Forgery. At the end of the simulation, $\mathcal{A}$ produces a $\mathsf{type\ 1}$ forgery $\hat{\sigma} = (\hat{\sigma}_s, \hat{r}, \hat{\sigma}_p)$ on $(\hat{\mathtt{id}}, \hat{m})$. Let $\langle \mathtt{id}_{\hat{i}}, \mathtt{id}_{s,\hat{i}}, l_{\hat{i}}, \mathtt{usk}_{\hat{i}}\rangle$ be the tuple in $\mathfrak{L}$ such that $\mathtt{id}_{\hat{i}} = \hat{\mathtt{id}}$. If $\ell_{\hat{i}}$ matches $\mathcal{B}_s$'s initial guess for the target index (*i.e.* $\ell_{\hat{i}} = \tilde{\ell}$), it wins the $\mathtt{EU\text{-}sID\text{-}CMA}$ game with $\mathcal{C}_s$ by passing $\hat{\sigma}_s$ as a forgery on $(\tilde{\mathtt{id}}_s, \hat{m})$ to $\mathcal{C}_s$; otherwise it *aborts* (abort_2).

Analysis. The probability of success of the reduction $\mathcal{B}_s$ is governed by the two events abort_1 and abort_2. To be precise,

$$\begin{aligned}\epsilon_s &= \Pr\left[\neg\mathsf{abort}_1 \wedge \neg\mathsf{abort}_2\right]\epsilon \\ &= \Pr\left[\neg\mathsf{abort}_1 \mid \neg\mathsf{abort}_2\right]\Pr\left[\neg\mathsf{abort}_2\right]\epsilon.\end{aligned}$$

Since $\tilde{\ell}$ is hidden from the adversary, it is easy to see that

$$\Pr\left[\neg\mathsf{abort}_2\right] = \Pr\left[\ell_{\hat{i}} = \tilde{\ell}\right] = 1/q_s.$$

On the other hand, $\Pr\left[\neg\mathsf{abort}_1 \mid \neg\mathsf{abort}_2\right] = 1$. This follows from the fact that if the simulator's guess of the target index was indeed correct ($\neg\mathsf{abort}_2$), then the adversary *would not* have made

an extract query on that identity (which causes abort_1). Thus, $\epsilon_s = \epsilon/q_s$. As for the time complexity, if τ_1 is the time taken for generating a signature in $\mathfrak{P}$, then the time taken by $\mathcal{B}_s$ can be easily seen as $t_s \leq t + (q_\varepsilon + q_s)\tau_1$.

3.1.2 Reduction $\mathcal{B}_p$.

The strategy adopted in $\mathcal{B}_p$ is similar to that in security arguments of [HW09, ST01]. It is also, on a high level, related to the technique used in [BB04b] for proving the security of the EU-CMA-secure PKS scheme constructed from EU-GCMA-secure PKS scheme (using CHF implicitly). The details follow.

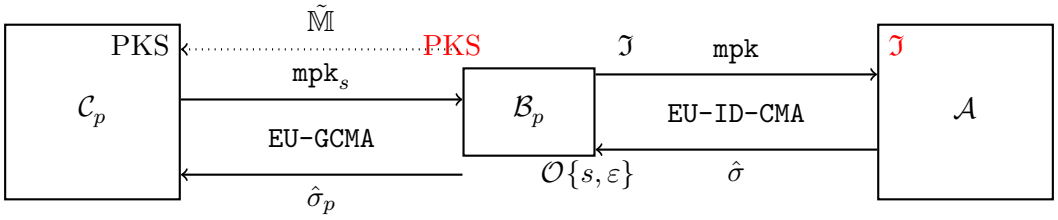

Figure 3: Reduction $\mathcal{B}_p$

Let $\mathcal{C}_p$ be the challenger in the EU-GCMA game. $\mathcal{B}_p$ plays the role of the adversary in the EU-GCMA game and, at the same time, the role of the challenger to $\mathcal{A}$ in the EU-ID-CMA game (see **Figure 3**). It starts by running the **Key Generation** algorithms $\mathcal{G}_s$ and $\mathcal{G}_h$ to obtain $(\mathsf{mpk}_s, \mathsf{msk}_s)$ and $(\mathsf{ek}, \mathsf{td})$ respectively. In order to initiate the EU-GCMA game, $\mathcal{B}_p$ has to commit to a set of q_s messages to $\mathcal{C}_p$. On the other hand, it also has to answer the adaptive queries by $\mathcal{A}$. Let's see how this is accomplished using the CHF. $\mathcal{B}_p$ first selects pairs $(\tilde{\mathsf{id}}_1, \tilde{r}_1), \ldots, (\tilde{\mathsf{id}}_{q_s}, \tilde{r}_{q_s})$ independently and uniformly at random from $\mathbb{I} \times \mathbb{R}$. Next, it commits $\tilde{\mathbb{M}} := \{\tilde{\mathsf{id}}_{s,1}, \ldots, \tilde{\mathsf{id}}_{s,q_s}\}$ to $\mathcal{C}_p$, where $\tilde{\mathsf{id}}_{s,i} \leftarrow \mathrm{h}(\mathsf{ek}, \tilde{\mathsf{id}}_i, \tilde{r}_i)$. As a result, $\mathcal{C}_p$ releases the challenge public key pk to $\mathcal{B}_p$ along with the set of signatures $\{\sigma_{p,1}, \ldots, \sigma_{p,q_s}\}$ on the (respective) committed messages. All this information is stored in a table $\mathfrak{C}$ as tuples $\langle \tilde{\mathsf{id}}_i, \tilde{r}_i, \tilde{\mathsf{id}}_{s,i}, \sigma_{p,i} \rangle$. Now, $\mathcal{B}_p$ initiates the EU-ID-CMA game by passing $\mathsf{mpk} := (\mathsf{mpk}_s, \mathsf{pk}, \mathsf{ek})$ as the challenge master public key to $\mathcal{A}$.

Mapping the identities. $\mathcal{B}_p$ too maintains the table $\mathfrak{L}$; but, it's structure is slightly different from that in $\mathcal{B}_s$. $\mathfrak{L}$ contains tuples of the form $\langle \mathsf{id}, \tilde{\mathsf{id}}_s, \mathsf{usk} \rangle$. Here, id and id_s are the related identities from $\mathfrak{I}$ and $\mathfrak{I}_s$ respectively. The usk-field stores the user secret key for id and hence contains elements of the form $(\mathsf{usk}_s, r, \sigma_p)$. If any component of the usk-field is yet to be generated, it is indicated by a '$\perp$'.

The way in which the mapping is maintained between the identities is somewhat different from that in $\mathcal{B}_s$. For mapping a *fresh* identity id, $\mathcal{B}_p$ first picks a tuple $\mathfrak{t} = \langle \tilde{\mathsf{id}}_s, \tilde{\mathsf{id}}, \tilde{r}, \sigma_p \rangle$ randomly from $\mathfrak{C}$. It then computes $r \leftarrow \mathrm{h}^{-1}(\mathsf{td}, \tilde{\mathsf{id}}, \tilde{r}, \mathsf{id})$, and adds the tuple $\langle \mathsf{id}, \tilde{\mathsf{id}}_s, (\perp, r, \sigma_p) \rangle$ to $\mathfrak{L}$. Finally it removes the tuple $\mathfrak{t}$ from $\mathfrak{C}$. As a result of these actions, id is effectively mapped to $\tilde{\mathsf{id}}_s$ since $\mathrm{h}(\mathsf{ek}, \mathsf{id}, r) = \mathrm{h}(\mathsf{ek}, \tilde{\mathsf{id}}, \tilde{r}) = \tilde{\mathsf{id}}_s$. A more formal description follows.

$\mathrm{M}_p(\mathsf{id})$:
if $\exists$ a tuple $\langle \mathsf{id}_i, \mathsf{id}_{s,i}, \mathsf{usk}_i \rangle \in \mathfrak{L}$ such that $(\mathsf{id}_i = \mathsf{id})$ **then**
  Set $\tau := (\mathsf{id}_{s,i}, \mathsf{usk}_i)$
else
  Pick $\mathfrak{t} \xleftarrow{\$} \mathfrak{C}$ and parse it as $\langle \tilde{\mathsf{id}}, \tilde{r}, \tilde{\mathsf{id}}_s, \sigma_p \rangle$
  Compute $r \leftarrow \mathrm{h}^{-1}(\mathsf{td}, \tilde{\mathsf{id}}, \tilde{r}, \mathsf{id})$ and set $\tau := (\mathsf{id}_s, (\perp, r, \sigma_p))$
  Add $\langle \mathsf{id}, \mathsf{id}_s, (\perp, r, \sigma_p) \rangle$ to $\mathfrak{L}$ and remove $\mathfrak{t}$ from $\mathfrak{C}$

end if
return τ

Queries: The extract and signature queries by $\mathcal{A}$ are answered as follows.

Extract query, $\mathcal{O}_{\varepsilon,\mathfrak{J}}(\mathtt{id})$: Invoke $\mathrm{M}_p(\mathtt{id})$ to obtain $(\tilde{\mathtt{id}}_s, \ell, (\mathtt{usk}_s, r, \sigma_p))$.

(i) If $(\mathtt{usk}_s \neq \perp)$ then return $\mathtt{usk} := (\mathtt{usk}_s, r, \sigma_p)$ as the user secret key.

(ii) Otherwise, $\mathcal{B}_s$ uses the knowledge of the master secret key $\mathtt{msk}_s$ to generate the user secret key $\mathtt{usk}_s := \mathcal{E}_s(\tilde{\mathtt{id}}_s, \mathtt{msk}_s)$ for $\tilde{\mathtt{id}}_s$. It returns $\mathtt{usk} := (\mathtt{usk}_s, r, \sigma_p)$ as the user secret key for $\mathtt{id}$ and updates the $\mathtt{usk}_s$-field of the tuple corresponding to $\mathtt{id}$ in $\mathfrak{L}$.

Signature query, $\mathcal{O}_{s,\mathfrak{J}}(\mathtt{id}, m)$: Invoke $\mathrm{M}_p(\mathtt{id})$ to get $(\tilde{\mathtt{id}}_s, \ell, (\mathtt{usk}_s, r, \sigma_p))$.

(i) If $(\mathtt{usk}_s \neq \perp)$ then $\mathcal{B}_p$ uses the knowledge of $\mathtt{usk}$ to return the signature $\sigma := \mathcal{S}(\mathtt{id}, m, \mathtt{usk})$

(ii) Otherwise, $\mathcal{B}_s$ uses step (ii) of **Extract query** to generate a user secret key $\mathtt{usk}$ for $\mathtt{id}$ and then uses this $\mathtt{usk}$ to return a signature $\sigma := \mathcal{S}(\mathtt{id}, m, \mathtt{usk})$.

Forgery. Finally, $\mathcal{A}$ produces a forgery $\hat{\sigma} = (\hat{\sigma}_s, \hat{r}, \hat{\sigma}_p)$ on $(\hat{\mathtt{id}}, \hat{m})$. As the forgery is of type 2, it implies $\hat{\mathtt{id}}_s := \mathrm{h}(\mathtt{ek}, \hat{\mathtt{id}}, \hat{r}) \notin \tilde{\mathbb{M}}$. Therefore $\hat{\sigma}_p$ is a valid forgery in the EU-GCMA game and $\mathcal{B}_p$ passes it to $\mathcal{C}_s$ to win the game.

Analysis. Since no abort is involved in $\mathcal{S}_p$, there is no degradation involved either. Thus, its advantage in attacking $\mathfrak{P}$ is $\epsilon_p = \epsilon$. If τ_2 and τ_3 denote the time taken for generating a secret key and a signature respectively in $\mathfrak{I}_s$, then the time taken by $\mathcal{B}_p$ is $t_p \leq t + (q_\varepsilon \tau_2 + q_s \tau_3)$.

3.1.3 Reduction $\mathcal{B}_h$.

$\mathcal{B}_h$ first obtains the challenge evaluation key $\mathtt{ek}$ for $\mathfrak{H}$ from its challenger $\mathcal{C}_h$. Then it invokes the algorithms $\mathcal{G}_s$ and $\mathcal{K}$ to generate $(\mathtt{msk}_s, \mathtt{mpk}_s)$ and $(\mathtt{sk}, \mathtt{pk})$ respectively. Finally, it passes $(\mathtt{mpk}_s, \mathtt{pk}, \mathtt{ek})$ as the challenge master public key to $\mathcal{A}$ (see **Figure 4**).

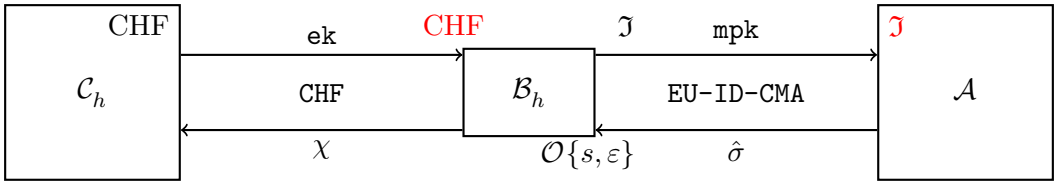

Figure 4: Reduction $\mathcal{B}_h$

Mapping the identities. The table used for maintaining the mapping has the same structure as in $\mathcal{B}_p$. However, the actual method used for mapping identities is far simpler than in $\mathcal{B}_p$ as shown below.

$\mathrm{M}_h(\mathtt{id})$:
if $\exists$ a tuple $\langle \mathtt{id}_i, \mathtt{id}_{s,i}, \mathtt{usk}_i \rangle \in \mathfrak{L}$ such that $(\mathtt{id}_i = \mathtt{id})$ **then**
  Set $\tau := (\mathtt{id}_{s,i}, \mathtt{usk}_i)$
else

Pick $r \in_R \mathbb{R}$ and compute $\mathtt{id}_s := \mathrm{h}(\mathtt{ek}, \mathtt{id}, r)$
Set $\tau := (\mathtt{id}_s, (\perp, r, \sigma_p))$ add $\langle \mathtt{id}, \mathtt{id}_s, (\perp, r, \perp) \rangle$ to $\mathfrak{L}$
end if
return τ

Queries: The extract and signature queries by $\mathcal{A}$ are answered as follows.

Extract query, $\mathcal{O}_{\varepsilon,\mathfrak{I}}(\mathtt{id})$: Invoke $\mathrm{M}_h(\mathtt{id})$ to obtain $(\mathtt{id}_s, (\mathtt{usk}_s, r, \sigma_p))$.

(i) If $(\mathtt{usk}_s \neq \perp)$ then return $\mathtt{usk} := (\mathtt{usk}_s, r, \sigma_p)$ as the user secret key.

(ii) Otherwise, $\mathcal{B}_h$ uses the knowledge of the master secret key $\mathtt{msk}_s$ to generate the user secret key $\mathtt{usk}_s := \mathcal{E}_s(\mathtt{id}_s, \mathtt{msk}_s)$ for $\mathtt{id}_s$. It also uses $\mathtt{sk}$ to generate $\sigma_p := \mathcal{S}_p(\mathtt{id}_s)$. Finally, $\mathcal{B}_h$ returns $\mathtt{usk} := (\mathtt{usk}_s, r, \sigma_p)$ as the user secret key for $\mathtt{id}$ and updates the $\mathtt{usk}_s$-field and σ_p-field of the tuple corresponding to $\mathtt{id}$ in $\mathfrak{L}$.

Signature query, $\mathcal{O}_{s,\mathfrak{I}}(\mathtt{id}, m)$: Invoke $\mathrm{M}_h(\mathtt{id})$ to obtain $(\mathtt{id}_s, (\mathtt{usk}_s, r, \sigma_p))$.

(i) If $(\mathtt{usk}_s \neq \perp)$ then $\mathcal{B}_h$ uses the knowledge of $\mathtt{usk}$ to return the signature $\sigma := \mathcal{S}(\mathtt{id}, m, \mathtt{usk})$.

(ii) Otherwise, $\mathcal{B}_h$ uses step (ii) of **Extract query** to generate a user secret key $\mathtt{usk}$ for $\mathtt{id}$ and then use this $\mathtt{usk}$ to return a signature $\sigma := \mathcal{S}(\mathtt{id}, m, \mathtt{usk})$.

Forgery. Finally, $\mathcal{A}$ produces a type 3 forgery $\hat{\sigma} = (\hat{\sigma}_s, \hat{r}, \hat{\sigma}_p)$ on $(\hat{\mathtt{id}}, \hat{m})$. Recall that this implies $\mathcal{A}$ produces the forgery with $(\hat{\mathtt{id}}, \hat{r}) \neq (\mathtt{id}_i, r_i)$ for all $i \in \{1, \ldots, q_s\}$, but with

(a) $\mathrm{h}(\mathtt{ek}, \hat{\mathtt{id}}, \hat{r}) = \mathrm{h}(\mathtt{ek}, \dot{\mathtt{id}}_i, \dot{r}_i)$ for some $i \in \{1, \ldots, q_\varepsilon\}$, or

(b) $\mathrm{h}(\mathtt{ek}, \hat{\mathtt{id}}, \hat{r}) = \mathrm{h}(\mathtt{ek}, \mathtt{id}_i, r_i)$ for some $i \in \{1, \ldots, q_s\}$.

Both the cases tantamount to breaking the collision resistance property of $\mathfrak{H}$. In case (a) ([resp. c] ase (b)) $\mathcal{B}_h$ passes $((\hat{\mathtt{id}}, \hat{r}), (\dot{\mathtt{id}}_i, \dot{r}_i))$ ([resp.]$((\hat{\mathtt{id}}, \hat{r}), (\mathtt{id}_i, r_i))$) as a collision to the challenger $\mathcal{C}_h$ to win the game.

Analysis. As in $\mathcal{B}_p$, there is no abort involved in $\mathcal{B}_h$. Therefore, its advantage in attacking $\mathfrak{H}$ is $\epsilon_p = \epsilon$. Again, if τ_2 and τ_3 denote the time taken for generating a secret key and a signature respectively in $\mathfrak{I}_s$, then the time taken by $\mathcal{B}_h$ is $t_p \leq t + (q_\varepsilon \tau_2 + q_s \tau_3)$.

4 Transforming from the EU-wID-CMA model

The construction technique described in the previous section can as well be used with a *relaxed* version of the selective-identity model which we call the *weak selective-identity* (wID) model. In this model the adversary, apart from committing to the "target" identity $\tilde{\mathtt{id}}$, has to commit a set of "query" identities $\tilde{\mathbb{I}}$. The adversary is allowed to query the extract oracle only on identities belonging to $\tilde{\mathbb{I}}$; whereas, it is allowed to query the signature oracle with identities from $\tilde{\mathbb{I}}$ *as well as* the target identity. Finally, as in the sID model, the adversary has to produce a forgery on $\tilde{\mathtt{id}}$. One may see the analogy between the EU-GCMA model for PKS and the wID model–both involve the adversary committing, beforehand, to the identities/messages that it wants to query. The only change involved is in the security argument–the way in which mapping is handled by the simulator. We elaborate on this later. But first, let's formally define the EU-wID-CMA model for IBS.

Definition 8 (EU-wID-CMA Game)**.** The security of an IBS scheme in the EU-wID-CMA model is argued in terms of the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Commitment: $\mathcal{A}$ commits to a target identity $\tilde{\mathtt{id}}$ and a set of query identities $\tilde{\mathbb{I}} := \{\tilde{\mathtt{id}}_1, \ldots, \tilde{\mathtt{id}}_{\tilde{q}}\} \subset \mathbb{I} \setminus \{\tilde{\mathtt{id}}\}$.

Set-up: $\mathcal{C}$ runs the set-up algorithm $\mathcal{G}$ to obtain the master keys (mpk,msk). It passes mpk as the challenge master public key to $\mathcal{A}$.

Queries: $\mathcal{A}$ can adaptively make extract queries on identities from $\tilde{\mathbb{I}}$ to an oracle $\mathcal{O}\varepsilon$ and signature queries involving identities from $\tilde{\mathbb{I}} \cup \{\tilde{\mathtt{id}}\}$ to an oracle $\mathcal{O}s$. These queries are handled as follows.

Extract query, $\mathcal{O}\varepsilon(\mathtt{id})$: $\mathcal{A}$ asks for the secret key of a user with identity $\mathtt{id} \in \tilde{\mathbb{I}}$. $\mathcal{C}$ computes $\mathtt{usk} := \mathcal{E}(\mathtt{id})$ and passes it to $\mathcal{A}$.

Signature query, $\mathcal{O}s(\mathtt{id}, m)$: $\mathcal{A}$ asks for the signature of a user with identity $\mathtt{id} \in \tilde{\mathbb{I}} \cup \{\tilde{\mathtt{id}}\}$ on a message m. $\mathcal{C}$ first runs $\mathcal{E}$ on id to obtain the user secret key usk. Next, it computes $\sigma := \mathcal{S}(\mathtt{id}, m, \mathtt{usk})$ and forwards it to $\mathcal{A}$.

Forgery: $\mathcal{A}$ outputs a signature $\hat{\sigma}$ on a message $\hat{m}$ and the target identity $\tilde{\mathtt{id}}$. $\mathcal{A}$ wins the game if:

1. $\hat{\sigma}$ is a valid signature on $\hat{m}$ by $\tilde{\mathtt{id}}$.
2. $\mathcal{A}$ has not made a signature query on $(\tilde{\mathtt{id}}, \hat{m})$.

The advantage $\mathcal{A}$ has in the above game, denoted by $\mathrm{Adv}_{\mathcal{A}}^{\mathtt{EU-wID-CMA}}(\kappa)$, is defined as the probability with which it wins the game, *i.e.*

$$\Pr\big[1 \leftarrow \mathcal{V}(\hat{\sigma}, \hat{\mathtt{id}}, \hat{m}, \mathtt{mpk}) \mid (\tilde{\mathtt{id}}, \tilde{\mathbb{I}}) \xleftarrow{\$} \mathcal{A}; (\mathtt{msk}, \mathtt{mpk}) \xleftarrow{\$} \mathcal{G}(\kappa); (\hat{\sigma}, \hat{\mathtt{id}}, \hat{m}) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}\varepsilon, \mathcal{O}s}(\mathtt{mpk})\big]$$

where the oracles $\mathcal{O}\varepsilon$ and $\mathcal{O}s$ are restricted to answering queries involving iden-tities from $\tilde{\mathbb{I}}$ and $\tilde{\mathbb{I}} \cup \{\tilde{\mathtt{id}}\}$ respectively. An adversary is said to be an $(\epsilon, t, q_\varepsilon, q_s, \tilde{q})$-forger of an IBS scheme in the EU-wID-CMA model if it has advantage of at least ϵ in the above game, runs in time at most t and makes at most q_ε and q_s extract and signature queries respectively, provided the number of identities involved in the signature and extract queries, excluding the target identity, is at most $\tilde{q}$. It is easy to see that $\tilde{q} \leq q_\varepsilon + q_s$. As we pointed out, the same transformation technique applies; the only change is in the security argument.

Theorem 2. *Given an $(\epsilon, t, q_\varepsilon, q_s)$-adversary $\mathcal{A}$, in the* EU-ID-CMA *model, agai-nst the IBS $\mathfrak{I}$, we can construct either*

(i) Algorithm $\mathcal{B}_w$ which $(\epsilon_w, t_w, q_\varepsilon, q_s, q_\varepsilon + q_s)$-breaks $\mathfrak{I}_w$ in the EU-wID-CMA *model, where*

$$\epsilon_w \geq \frac{1}{3q_s}\epsilon \quad \textit{and} \quad t_w \leq t + (q_\varepsilon + q_s)\tau_1, \quad \textit{or}$$

(ii) Algorithm $\mathcal{B}_p$ which $(\epsilon_p, t_p, q_\varepsilon + q_s)$-breaks $\mathfrak{P}$ in the EU-GCMA *model, where*

$$\epsilon_p = \frac{1}{3}\epsilon \quad \textit{and} \quad t_p \leq t + (q_\varepsilon\tau_2 + q_s\tau_3), \quad \textit{or}$$

(iii) Algorithm $\mathcal{B}_h$ which (ϵ_h, t_h)-breaks $\mathfrak{H}$, where

$$\epsilon_h = \frac{1}{3}\epsilon \textit{ and } \quad t_h \leq t + (q_\varepsilon + q_s)\tau_1 + (q_\varepsilon\tau_2 + q_s\tau_3).$$

Here, q_{ε} ([resp.]q_s) denotes the upper bound on the number of extract ([resp. s] ignature) queries that $\mathcal{A}$ can make. τ_1 is the time taken for generating a signature in $\mathfrak{P}$; τ_2 ([resp.]τ_3) denotes the time taken to generate a user secret key ([resp. s] ignature) in $\mathfrak{I}_s$.

Proof. The security argument is similar to the one discussed in **§3.1**. The only difference lies in the way in which the mapping of identities is handled in $\mathcal{B}_w$ as described below. Let $\mathcal{C}_w$ be the challenger in the `EU-wID-CMA` game. $\mathcal{B}_w$ plays the role of the adversary in the `EU-wID-CMA` game and, at the same time, the role of the challenger to $\mathcal{A}$ in the `EU-ID-CMA` game. In order to initiate the `EU-wID-CMA` game, $\mathcal{B}_w$ has to commit to a target identity and a target set. It selects an identity $\tilde{\mathtt{id}} \in_R \mathbb{I}$ and a randomiser $\tilde{r} \in_R \mathbb{R}$, and commits $\tilde{\mathtt{id}}_w \leftarrow \mathrm{h}(\mathtt{ek}, \tilde{\mathtt{id}}, \tilde{r})$ as the target identity to $\mathcal{C}_w$. Similarly, it selects $\{\tilde{\mathtt{id}}_i, \ldots, \tilde{\mathtt{id}}_{\tilde{q}}\} \xleftarrow{\$} \mathbb{I}$, $\{\tilde{r}_1, \ldots, \tilde{r}_{\tilde{q}}\} \xleftarrow{\$} \mathbb{R}$ and commits $\hat{\mathbb{I}} := \{\tilde{\mathtt{id}}_{1,w}, \ldots, \tilde{\mathtt{id}}_{\tilde{q},w}\}$, where $\tilde{\mathtt{id}}_{i,w} \leftarrow \mathrm{h}(\mathtt{ek}, \tilde{\mathtt{id}}_i, \tilde{r}_i)$, as the target set to $\mathcal{C}_w$. As a result, $\mathcal{C}_w$ releases the challenge master public key $\mathtt{mpk}_w$ to $\mathcal{B}_s$. All this information is stored in a table, denoted by $\mathfrak{D}$, as tuples $\langle \tilde{\mathtt{id}}_i, \tilde{r}_i, \tilde{\mathtt{id}}_{w,i} \rangle$.

Mapping. $\mathcal{B}_w$ maintains a table $\mathfrak{L}$ with structure the same as that in reduction $\mathcal{B}_p$. For mapping a *fresh* identity $\mathtt{id}$, $\mathcal{B}_w$ chooses a tuple $\mathfrak{t} = \langle \tilde{\mathtt{id}}, \tilde{r}, \tilde{\mathtt{id}}_w \rangle$ randomly from $\mathfrak{D}$. Next, it computes $r := \mathrm{h}^{-1}(\mathtt{td}, \tilde{\mathtt{id}}, \tilde{r}, \mathtt{id})$ and adds $\langle \mathtt{id}, \tilde{\mathtt{id}}_w, (\perp, r, \perp) \rangle$ to $\mathfrak{L}$. Finally, it removes the tuple $\mathfrak{t}$ from $\mathfrak{D}$. As a result of these actions, $\mathtt{id}$ is effectively mapped to $\tilde{\mathtt{id}}_w$ as $\mathrm{h}(\mathtt{ek}, \mathtt{id}, r) = \mathrm{h}(\mathtt{ek}, \tilde{\mathtt{id}}, \tilde{r}) = \tilde{\mathtt{id}}_w$. A more formal description follows.

$\mathrm{M}_w(\mathtt{id})$:
if $\exists$ a tuple $\langle \mathtt{id}_i, \mathtt{id}_{w,i}, \mathtt{usk}_i \rangle \in \mathfrak{L}$ such that $(\mathtt{id}_i = \mathtt{id})$ **then**
  Set $\tau := (\mathtt{id}_{w,i}, \mathtt{usk}_i)$
else
  Pick $\mathfrak{t} \xleftarrow{\$} \mathfrak{C}$ and parse it as $\langle \tilde{\mathtt{id}}, \tilde{r}, \tilde{\mathtt{id}}_w \rangle$
  Compute $r \leftarrow \mathrm{h}^{-1}(\mathtt{td}, \tilde{\mathtt{id}}, \tilde{r}, \mathtt{id})$ and set $\tau := (\mathtt{id}_w, (\perp, r, \perp))$
  Add $\langle \mathtt{id}, \mathtt{id}_w, (\perp, r, \perp) \rangle$ to $\mathfrak{L}$ and remove $\mathfrak{t}$ from $\mathfrak{C}$
end if
return τ

□

Remark 2 (Comparison with the folklore paradigm)**.** The (identities-based) signature of an IBS scheme constructed using the folklore technique consists of *two* (public-key) signatures and *one* public key of the underlying (fully-secure) PKS. In contrast, the signature of an IBS scheme using our approach consists of *one* signature each of the underlying (`wID`-secure) IBS and (weakly-secure) PKS and *one* randomiser from the CHF. The time taken for signing and verification is comparable, bar the time taken to compute the hash value.

5 Conclusion

In this paper, we described a generic transformation from `sID`/`wID` IBS to full-identity IBS using a chameleon hash function and an `EU-GCMA`-secure PKS scheme. We also argued, without using random oracles, that the resulting IBS is secure in the full-identity model with only linear degradation incurred. An interesting problem would be to replace the `EU-GCMA` PKS with a more primitive construct. Extending the transformation for Hierarchical IBS could be yet another challenging task.

Acknowledgements. We would like to thank Rishiraj Bhattacharyya for some insightful discussions and Vikas Kumar for his role in the initial phase of the work. We would also like to thank the anonymous reviewers for the constructive comments.

References

[BB04a] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer Berlin / Heidelberg, 2004. (Cited on page 2.)

[BB04b] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73. Springer Berlin / Heidelberg, 2004. (Cited on page 11.)

[BCC88] Gilles Brassard, David Chaum, and Claude Crépeau. Minimum disclosure proofs of knowledge. *Journal of Computer and System Sciences*, 37(2):156 – 189, 1988. (Cited on page 6.)

[BF01] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In Joe Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer Berlin / Heidelberg, 2001. (Cited on page 2.)

[BNN04] Mihir Bellare, Chanathip Namprempre, and Gregory Neven. Security proofs for identity-based identification and signature schemes. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 268–286. Springer Berlin / Heidelberg, 2004. (Cited on pages 2 and 5.)

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computer and communications security*, CCS '93, pages 62–73, New York, NY, USA, 1993. ACM. (Cited on pages 2 and 5.)

[CFH+09] Yang Cui, Eiichiro Fujisaki, Goichiro Hanaoka, Hideki Imai, and Rui Zhang. Formal security treatments for ibe-to-signature transformation: Relations among security notions. *IEICE Transactions*, 92-A(1):53–66, 2009. (Cited on page 2.)

[CHC02] Jae Choon and Jung Hee Cheon. An identity-based signature from gap Diffie-Hellman groups. In Yvo Desmedt, editor, *Public Key Cryptography — PKC 2003*, volume 2567 of *Lecture Notes in Computer Science*, pages 18–30. Springer Berlin / Heidelberg, 2002. (Cited on page 2.)

[CHK03] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. In Eli Biham, editor, *Advances in Cryptology — EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 646–646. Springer Berlin / Heidelberg, 2003. (Cited on page 2.)

[CS06] Sanjit Chatterjee and Palash Sarkar. Generalization of the selective-id security model for hibe protocols. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography - PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 241–256. Springer Berlin / Heidelberg, 2006. (Cited on page 2.)

[Gal06] David Galindo. A separation between selective and full-identity security notions for identity-based encryption. In Marina Gavrilova, Osvaldo Gervasi, Vipin Kumar, C. Tan, David Taniar, Antonio Laganá, Youngsong Mun, and Hyunseung Choo, editors, *Computational Science and Its Applications - ICCSA 2006*, volume 3982 of *Lecture Notes in Computer Science*, pages 318–326. Springer Berlin / Heidelberg, 2006. (Cited on page 2.)

[GH05] David Galindo and Ichiro Hasuo. Security notions for identity based encryption. Cryptology ePrint Archive, Report 2005/253, 2005. (Cited on page 2.)

[GMR88] Shafi Goldwasser, Silvio Micali, and Ron Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988. (Cited on pages 2, 3 and 4.)

[Her05] Javier Herranz. Deterministic identity-based signatures for partial aggregation. *The Computer Journal*, 49(3):322–330, 2005. (Cited on page 2.)

[Hes03] Florian Hess. Efficient identity based signature schemes based on pairings. In Kaisa Nyberg and Howard Heys, editors, *Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 310–324. Springer Berlin / Heidelberg, 2003. (Cited on page 2.)

[HW09] Susan Hohenberger and Brent Waters. Short and stateless signatures from the rsa assumption. In Shai Halevi, editor, *Advances in Cryptology - CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 654–670. Springer Berlin / Heidelberg, 2009. (Cited on pages 2 and 11.)

[KR00] Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS*. The Internet Society, 2000. (Cited on pages 2 and 6.)

[Moh11] Payman Mohassel. One-time signatures and chameleon hash functions. In Alex Biryukov, Guang Gong, and Douglas Stinson, editors, *Selected Areas in Cryptography*, volume 6544 of *Lecture Notes in Computer Science*, pages 302–319. Springer Berlin / Heidelberg, 2011. (Cited on page 6.)

[Sha85] Adi Shamir. Identity-based cryptosystems and signature schemes. In George Blakley and David Chaum, editors, *Advances in Cryptology*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer Berlin / Heidelberg, 1985. (Cited on page 1.)

[ST01] Adi Shamir and Yael Tauman. Improved online/offline signature schemes. In Joe Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 355–367. Springer Berlin / Heidelberg, 2001. (Cited on pages 2 and 11.)

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 557–557. Springer Berlin / Heidelberg, 2005. (Cited on page 2.)