Pairing-Free Blind Signatures from CDH Assumptions

Rutchathon Chairattana-Apirom, Stefano Tessaro, and Chenzhi Zhu

Paul G. Allen School of Computer Science & Engineering
University of Washington, Seattle, US
{rchairat,tessaro,zhucz20}@cs.washington.edu

Abstract. This paper presents new blind signatures for which concurrent security, in the random oracle model, can be proved from variants of the computational Diffie-Hellman (CDH) assumption in pairing-free groups *without* relying on the algebraic group model (AGM). With the exception of careful instantiations of generic non-black box techniques following Fischlin's paradigm (CRYPTO '06), prior works without the AGM in the pairing-free regime have only managed to prove security for a-priori bounded concurrency.

Our most efficient constructions rely on the chosen-target CDH assumption, which has been used to prove security of Blind BLS by Boldyreva (PKC '03), and can be seen as blind versions of signatures by Goh and Jarecki (EUROCRYPT '03) and Chevallier-Mames (CRYPTO '05). We also give a less efficient scheme with security based on (plain) CDH which builds on top of a natural pairing-free variant of Rai-Choo (Hanzlik, Loss, and Wagner, EUROCRYPT '23). Our schemes have signing protocols that consist of four (in order to achieve regular unforgeability) or five moves (for strong unforgeability).

The blindness of our schemes is either computational (assuming the hardness of the discrete logarithm problem), or statistical in the random oracle model.

1 Introduction

Blind signatures [Cha82] allow a user to obtain a signature on a message by interacting with the signer in a way that does not reveal anything about the message-signature pair to the signer. They are a fundamental building block to achieve anonymity in e-cash [Cha82, CFN90, OO92], e-voting [FOO93], and credentials [Bra94, BL13]. They have also come to use in a number of recent industry applications, such as privacy-preserving ad-click measurement [PCM], Apple's iCloud Private Relay [App], Google One's VPN Service [Goo], and various forms of anonymous tokens [HIP$^+$21, Tru].

It is natural to want to design blind signatures in pairing-free groups. On the one hand, widely adopted signatures, such as Schnorr signatures [Sch90], EdDSA [BDL$^+$12], and ECDSA [Ame05] rely on such curves. On the other hand, many of the aforementioned applications are implemented in environments such as Internet browsers where pairing-friendly curves are usually not part of the available cryptographic libraries (such as NSS and BoringSSL).

The question of designing blind signatures in pairing-free groups has turned out to be extremely challenging. The main difficulty is finding schemes secure in the sense of *one-more unforgeability* [JLO97], even when a malicious user can run several concurrent signing interactions with the signer. Pointcheval and Stern [PS00] were the first to prove security of blind Okamoto-Schnorr signatures [Oka94] under bounded concurrency, in the random oracle model (ROM) [BR93], assuming the hardness of the discrete logarithm (DL) problem. Their approach was later abstracted in [HKL19]. Blind Schnorr signatures [CP93] have also only been proved secure under bounded concurrency [FPS20, KLX22], in this case additionally assuming the Algebraic Group Model (AGM) [FKL18], along with the stronger one-more discrete logarithm (OMDL) assumption [BNPS03]. These results are also in some sense best possible, as recent ROS attacks [BLL$^+$21] yield polynomial-time forgery attacks against these schemes using $\log p$ concurrent signing sessions, where p is the group order.

One can rely on boosting techniques [Poi98, KLR21, CAHL$^+$22] to increase the number of concurrent sessions a scheme such as Okamoto-Schnorr remains secure to. The current state of the art [CAHL$^+$22] requires a signer whose complexity grows linearly in the number of signing sessions, which still has to be fixed a priori.

A concurrently secure scheme, i.e., one supporting arbitrary concurrent adversarial signing sessions, was given by Abe [Abe01], but its proof (in the ROM, assuming the hardness of DL) later turned out to be incorrect, and was only recently re-stablished in the AGM only [KLX22]. Similarly, all other provably secure solutions [FPS20, TZ22, CKM$^+$23] fundamentally rely on the AGM.

This paper addresses the fundamental question of coming up with such schemes that can be proved secure without the AGM in the random-oracle model.

Non-black box baseline. One exception to the above is a folklore instantiation of Fischlin's transform [Fis06] using generic NIZKs with online extractability in the random oracle model, such as those from the MPC in the head paradigm [IKOS07]. In the pairing-free setting, the signer uses a hash-based signature scheme (which exists under the hardness of the DL problem) [NY89] to sign a Pedersen commitment to the message, and the actual signature for a message is a proof of knowledge of a signature on a commitment to this message. While round optimal, this approach relies on heavy non-black box techniques and is much less efficient than existing AGM schemes. Also worth noting here is the recent work by Fuchsbauer and Wolf [FW22], which relies on generic NIZKs as well, and assumes Schnorr signatures to be secure for a given fixed (non random oracle) hash function.

Our contribution. We give the first natural blind signature schemes in pairing-free settings proved to be concurrently secure *without relying* on the AGM. Our schemes treat the underlying group in a black-box way, additionally using one or more hash functions to be modeled as random oracles, and their security is based on variants of the Computational Diffie-Hellman assumption.

A summary of our constructions can be found in Table 1, and we give a technical overview below. Our most efficient constructions are based on the *chosen target CDH (CT-CDH) assumption* introduced by Boldyreva [Bol03] to prove security (in the pairing setting) of Blind BLS [BLS01], which is a one-more version of CDH. The signing protocols take four and five moves, respectively, the difference being that the latter protocol achieves strong unforgeability. The starting points of these schemes are the Goh-Jarecki [GJ03] and the Chevallier-Mames [Che05, KLP17] signature schemes, respectively, with a number of modifications based on witness indistinguishable OR-proofs [CDS94] to be able to prove concurrent security.

Our third, more complex, scheme dispenses entirely with interactive assumptions, and solely relies on (plain) CDH. It applies techniques in principle similar to those we used for our two better performing schemes to the recent work by Hanzlik, Loss, and Wagner [HLW23] meant to be used in the pairing setting.

Let us mention two differences with prior works. Our schemes, in comparison with recent AGM-based schemes [TZ22, CKM$^+$23], do not offer perfect blindness. Instead, we give two types of blindness proof. For all of the schemes, we prove statistical blindness assuming bounded queries to a random oracle. We also give a slightly more efficient version of the first two schemes which is computationally blind under the discrete logarithm assumption. For the first two schemes, the advantage of our random oracle proofs is that it only requires the Fiat-Shamir heuristic [FS87] to be sound for proofs (hence, there is no rewinding). While we do not prove this formally, we expect blindness of our first two schemes to also hold against quantum adversaries in the QROM [BDF$^+$11], following e.g. [Unr17].

We also only prove a slightly weaker notion of one-more unforgeability than the more standard one from [JLO97], in that we only guarantee that a malicious user cannot come up with more signatures than the number of sessions it engages in, regardless of whether these terminate or not. This weaker notion has been used before (cf. e.g. [HKKL07]), and appears to suffice for most envisioned applications. For example, when used to build anonymous tokens [HIP$^+$21, Tru], the weaker notion means that the server needs to regard a token as issued as long as the server sent out the first-round message to the user. The only advantage of the stronger security is that it guarantees that if the signing protocol aborts, the user will not come up with a valid token. However, we do not find this to be a true concern. (In principle, the server can decide whether to potentially issue a blind signature before engaging in a signing session.) We also point out that there are some applications where the stronger notion is necessary, although these seem to rely on settings where fair two-party functionalities are possible (one example we are aware of is the recent work by [HLTW22]). In this sense, it is a good open question to close this gap. We also see no reason why our scheme would not achieve the stronger, more standard, notion, but our proof techniques fail to establish that.

Scheme	Security	Mvs.	Sig. size	Comm.	Blind Asmp.	OMUF Asmp.
BS_1 (Sec. 3)	comp./stat. blindness & OMUF	4	$1\ \mathbb{G} + 4\ \mathbb{Z}_p$	$5\ \mathbb{G} + 5$(or 7) $\mathbb{Z}_p$	DL(comp.)/ ROM(stat.)	CT-CDH
BS_2 (Sec. 4)	comp./stat. blindness & OMSUF	5	$1\ \mathbb{G} + 4\ \mathbb{Z}_p$	$5\ \mathbb{G} + 5$(or 7) $\mathbb{Z}_p$	DL(comp.)/ ROM(stat.)	CT-CDH
BS_3 (Sec. 5)	stat. blind & OMUF	4	$(\lambda+1)\ \mathbb{G} +$ $(\lambda+3)\ \mathbb{Z}_p +$ λ^2 bits	$(3\lambda+2)\ \mathbb{G} +$ $(2\lambda+5)\ \mathbb{Z}_p +$ $(\lambda+3\lambda^2)$ bits	ROM	CDH

Table 1. Overview of our results. The three schemes proposed in this paper with the achieved provable security notions, number of moves, signature size and communication cost (note: $p = |\mathbb{G}|$), and the assumptions required to achieve the mentioned security notions. All of the schemes are OMUF secure assuming the ROM. For the first two schemes $\mathsf{BS}_1, \mathsf{BS}_2$, we give two versions of the protocol, one with computational blindness and one with statistical blindness in the ROM. The latter is less efficient. We also note that the scheme BS_3 depends on two parameters N and K, and the efficiencies mentioned in this table is achieved by setting $N = 2, K = \lambda$.

On DLOG based schemes. An elusive open problem is to give similar schemes with security proofs based solely on the hardness of the DL problem (or the stronger OMDL assumption). Indeed, techniques from recent works [KLX22, TZ22, CKM+23] are not robust to rewinding in several subtle ways. One may of course argue that the qualitative improvement is not significant (for several curves, indeed, DL and CDH are somewhat equivalent [Mau94, MS23]). However, the evidence is that even in the non-blind setting, signatures with security based on DL tend to be actually more efficient, so this is certainly an avenue for further research.

We note that recent work by Barreto and Zanon [BZ23] (expanded in [BRJZ23]) claims pairing-free blind signatures with a proof of concurrent security under the OMDL assumption, which hinges upon a reduction of concurrent security under impersonation attacks (IMP-CA) to the (concurrent) one-more unforgeability of the associated signature scheme. The proof appears to have some gaps, and furthermore, we note that in general IMP-CA security does not yield concurrently secure blind signatures. For instance, Bellare and Palacio [BP02] prove the Schnorr identification scheme achieves the former notion, whereas it does not yield secure blind signatures.

Paper outline. Section 2 introduces the basic preliminaries. We then discuss the two schemes based on the CT-CDH assumption, BS_1 (achieving one-more unforgeability) and BS_2 (achieving one-more strong unforgeability), in Sections 3 and 4 respectively. Lastly, we discuss the scheme BS_3 based on the CDH assumption in Section 5.

1.1 Technical Overview

The starting point of our first and simplest scheme BS_1 is the signature by Goh and Jarecki [GJ03], which can also be thought of as a "pairing-free" variant of BLS signatures [BLS04]. Given a cyclic group $\mathbb{G}$ with prime order p and generator g, a secret key sk is a random scalar in $\mathbb{Z}_p$, and the corresponding public key is $\mathsf{pk} \leftarrow g^{\mathsf{sk}}$. The signature of a message m is $Z \leftarrow \mathsf{H}(m)^{\mathsf{sk}}$, where H is a hash function, along with a non-interactive proof π of discrete logarithm equality (DLEQ), showing that $\log_g \mathsf{pk} = \log_{\mathsf{H}(m)} Z$.

The generation of such a signature can be seen as an interactive protocol. The user first sends $h \leftarrow \mathsf{H}(m)$ to the signer. The signer then sends $Z \leftarrow h^{\mathsf{sk}}$ back and initiates an interactive version of the standard DLEQ proof [Sch91]. In particular, along with Z, the signer sends two nonces $R_g \leftarrow g^r$ and $R_h \leftarrow h^r$ to the user, where $r \leftarrow\!\!\$\ \mathbb{Z}_p$; upon receiving R_g, R_h, the user picks a challenge $c \leftarrow \mathsf{H}'(m, h, Z, R_g, R_h)$ to send to the signer, and the signer replies with $z \leftarrow r + c \cdot \mathsf{sk}$. The user accepts if and only if $R_g = g^z \mathsf{pk}^{-c}$ and $R_h \leftarrow h^z Z^{-c}$, and the signature is $\sigma \leftarrow (Z, \pi = (c, z))$. To verify the signature, with $h \leftarrow \mathsf{H}(m)$, we recover $R_g \leftarrow g^z \mathsf{pk}^{-c}$ and $R_h \leftarrow h^z Z^{-c}$ and check whether $c = \mathsf{H}'(m, h, Z, R_g, R_h)$.

Achieving one-more unforgeability. Our first goal is to make this scheme one-more unforgeable (OMUF), i.e., the adversary cannot produce signatures for $\ell + 1$ *distinct messages* after engaging in at most ℓ signing sessions. The idea is to show that OMUF of the scheme is implied by the hardness of *chosen-target computational Diffie-Hellman* (CT-CDH) problem [Bol03], where, given g^x for a uniformly random $x \in \mathbb{Z}_p$ and ℓ-time access to a DH oracle that takes any group element Y as input and outputs Y^x, the adversary's goal is to compute Y_i^x for at least $\ell + 1$ randomly sampled challenges $\{Y_i \in \mathbb{G}\}$. (Here, we assume an oracle which supplies as many challenges as needed, but the attacker just needs to solve $\ell + 1$ of these.)

The reduction idea appears simple: Given an adversary $\mathcal{A}$ that breaks OMUF of the scheme, we construct an adversary $\mathcal{B}$ playing the CT-CDH game that runs $\mathcal{A}$ with $\mathsf{pk} \leftarrow g^x$. Random oracle queries $\mathsf{H}(m_i)$ for a message m_i are answered with a challenge Y_i. When $\mathcal{A}$ starts a signing session with h as the first-round message, $\mathcal{B}$ computes $Z \leftarrow h^x$ by querying the DH oracle and simulates the rest of the signing session by itself. (Note that a DH query here is necessary, because h can be any group element.) For a valid signature (Z_i, π_i) for a message m_i, by the soundness properties of π_i, $Z_i = \mathsf{H}(m_i)^x$ is a solution to the challenge $Y_i = \mathsf{H}(m_i)$ with overwhelming probability. Therefore, if the adversary $\mathcal{A}$ forges valid signatures for $\ell + 1$ distinct messages, $\mathcal{B}$ solves the CT-CDH problem.[1]

The challenge here is that the DLEQ proof is merely honest-verifier zero-knowledge, and the adversary $\mathcal{A}$ sends an *arbitrary* challenge c in the interaction, for which $\mathcal{B}$ needs to simulate a response. This cannot be done efficiently without knowing the secret key. To address this, we transform the DLEQ proof into a witness indistinguishable (WI) OR proof [CDS94] that proves the existence of a witness sk for the DLEQ proof or knowledge of a witness $w = \log_g W$ for a public parameter $W \in \mathbb{G}$. (This parameter would be generated transparently in actual implementation.) Now the proof can be generated, indistinguishably, both with knowledge of sk (this is what the protocol does) or with knowledge of w. The former is what the actual protocol does, but the latter is what the reduction $\mathcal{B}$ would do. (The reduction clearly chooses W with a known discrete logarithm w.) The challenge of this proof will be chosen as before as a hash, and the resulting non-interactive proof π will be included in the signature $\sigma = (Z, \pi)$.

However, this brings a new issue. Witness indistinguishability only holds when $Z = h^{\mathsf{sk}}$, which is always the case when the honest signer generates it. However, it is possible, in principle, to use the witness w to generate a signature (Z, π) for m where $Z \neq \mathsf{H}(m)^{\mathsf{sk}}$. Our key observation here is that any adversary succeeding in producing such a signature can be used to compute w, and thus, to break the discrete logarithm assumption. This argument is rather involved as it requires a careful use of the Forking Lemma. In essence, π gives us *two* valid proof transcripts (R_g, R_h, d, z) and (A, e, t), where the former verifies as a valid DLEQ proof for $Z = \mathsf{H}(m)^{\mathsf{sk}}$, and the latter attests knowledge of w. Further, we have that $d + e = \mathsf{H}'(m, h, Z, R_g, R_h, A)$. If we fork on this hash query, we can obtain two extra transcripts (R_g, R_h, d', z') and (A, e', t') such that $d' + e' \neq d + e$. Still, we succeed in extracting w *only* if $e \neq e'$, but this is not necessarily guaranteed if we also have $d \neq d'$.

Here, we crucially rely on a property of the DLEQ proof, namely that by fixing (R_g, R_h) and since $Z \neq \mathsf{H}(m)^{\mathsf{sk}}$, there exists at most one bad d that can generate an accepting (R_g, R_h, d, z). Therefore, $d = d'$ must hold, and hence $e \neq e'$.

Achieving blindness. To make the signing protocol of BS_1 blind, the user additionally samples a random scalar β and computes $h \leftarrow \mathsf{H}(m)g^{\beta}$. After receiving $Z = h^{\mathsf{sk}}$, the user computes $Z' \leftarrow Z\mathsf{pk}^{-\beta}$. It is easy to verify that $Z' = \mathsf{H}(m)^{\mathsf{sk}}$. Then, the user blinds the OR proof in a way similar to Abe-Okamoto blind signatures [AO00], such that after the interaction, the user generates a proof π', the distribution of which is independent of the transcript of the proof.

However, a malicious signer can send an incorrect Z (i.e., $Z \neq h^{\mathsf{sk}}$) in one of the signing sessions, and later identify the blinded signature (Z', π') by checking whether $Z' \neq \mathsf{H}(m)^{\mathsf{sk}}$. Fortunately, for the attack to work, the signer also needs to let the user accept the OR proof during the session where $Z \neq h^{\mathsf{sk}}$. Using a similar argument as the above, by the soundness of the OR proof, the probability that this occurs is bounded by the advantage of computing $\log_g W$.

[1] We remark that to reduce to the CT-CDH assumption, we need the adversary to output more forgery than the number of started sessions; hence, the weaker notion of OMUF security.

If we do not want blindness to rely on the discrete logarithm assumption, another way to tackle this is to let the signer sends a non-interactive proof that $Z = h^{\mathsf{sk}}$ in the second move. For example, if we use the non-interactive version of the DLEQ proof, we can show blindness of BS_1 in the random oracle model. Crucially, this proof does not need to be blind (it only needs to be verified at that stage by the user).

Remaining constructions. We will now briefly discuss our two remaining protocols BS_2 and BS_3.

Strong unforgeability. Unfortunately, BS_1 is not one-more strongly unforgeable (OMSUF), i.e., we cannot guarantee that the adversary cannot produce $(\ell + 1)$ *distinct* valid message-signature pairs after ℓ signing sessions. Indeed, suppose all signing sessions start with the same first-round message $h = \mathsf{H}(m)$ for some m. Then, BS_1 shares the structure of Abe-Okamoto blind signatures [AO00], and a variant of the recent ROS attacks [BLL+21] yields an adversary that starts $\log p$ signing sessions and outputs $\log p + 1$ distinct signatures for the message m. To transform BS_1 into a OMSUF one, referred to as BS_2, the idea is to let H also take (R_g, A) as input, i.e., the group elements independent of h. In particular, we let the signer send R_g, A to the user before h is sent, adding an extra move to the signing protocol. The user then computes $h \leftarrow \mathsf{H}(m, R_g, A)$ and the rest of the protocol remains as in BS_1. The resulting signature is the same as the Chevallier-Mames signature scheme [Che05, KLP17] except that we replace the DLEQ proof with the OR proof.

CDH-based construction. To construct a scheme relying on only the non-interactive CDH assumption, instead of the stronger CT-CDH, we apply the above framework to Rai-Choo [HLW23], which is a pairing-based scheme relying only on the CDH assumption. Abstractly one can think of the CT-CDH assumption as giving access to an interactive version of the BLS scheme without efficient verification (the DH oracle gives signatures, the challenge oracle implements the random oracle). Similarly, we define a natural pairing-free version of Rai-Choo with inefficient verification (see Section 5 for details), where the key generation is the same as BS_1 and a signature is of the form $((\mathsf{pk}_i \in \mathbb{G}, \varphi_i \in \{0,1\}^\lambda)_{i \in [K]}, S \in \mathbb{G})$ for some value K. The signature is valid for a message m if and only if $\mathsf{pk} = \prod_{i=1}^{K} \mathsf{pk}_i$ and $S = \prod_{i=1}^{K} \mathsf{H}(m, \varphi_i)^{\mathsf{sk}_i}$ for some hash function H, where $\mathsf{sk}_i = \log_g \mathsf{pk}_i$. Similar to BS_1, to translate this scheme into a blind signature scheme with efficient verification, referred to as BS_3, we let the signer, after the signing protocol of Rai-Choo finished, engage in a WI OR proof protocol with the user that shows the knowledge of either the witness $\log_g W$ or the witness $\{\mathsf{sk}_i\}_{i \in [K]}$ such that $\mathsf{pk}_i = g_i^{\mathsf{sk}}$ and $S = \prod_{i=1}^{K} \mathsf{H}(m, \varphi_i)^{\mathsf{sk}_i}$. To show OMUF of BS_3, using a similar idea as BS_1, we show that if there exists an adversary that breaks OMUF of BS_3, then we can either break the OMUF of Rai-Choo or extract $\log_g W$. We refer to Section 5.4 for the full proof.

2 Preliminaries

Notation. For a positive integer n, we write $[n]$ for $\{1, \ldots, n\}$. We use λ to denote the security parameter. The probabilistic polynomial time algorithm GGen takes an input 1^λ and outputs a cyclic group $\mathbb{G}$ of λ-bit prime order p and a generator g of the group. We tacitly assume standard group operations in $\mathbb{G}$ can be performed in time polynomial in λ and adopt multiplicative notation. We will often compute over the finite field $\mathbb{Z}_p$ (for a prime p) – we usually do not write modular reduction explicitly when it is clear from the context. We write $\mathbb{Z}_p^* = \mathbb{Z}_p \backslash \{0\}$. Also, we write $a = \log_g A \in \mathbb{Z}_p$ for a group element $A \in \mathbb{G}$ such that $A = g^a$.

Cryptographic assumptions. The constructions in this paper are based on a subset of the following three hardness assumptions: the hardness of the discrete logarithm (DL) problem, the hardness of computational Diffie-Hellman (CDH) problem, and the hardness of chosen-target computational Diffie-Hellman (CT-CDH) problem (introduced by Boldyreva [Bol03]). For any adversary $\mathcal{A}$, we define the advantage of $\mathcal{A}$ playing the games $\{\mathrm{DLOG}, \mathrm{CDH}, \text{CT-CDH}\}$ (each of these games is defined in Figure 1) as

$$\mathsf{Adv}^{\mathrm{dlog/cdh/ct-cdh}}_{\mathsf{GGen}}(\mathcal{A}, \lambda) := \Pr[(\mathrm{DLOG/CDH/CT\text{-}CDH})^{\mathcal{A}}_{\mathsf{GGen}}(\lambda) = 1] \,.$$

Additionally, the hardness of the CT-CDH problem implies the hardness of the CDH problem which implies the hardness of the DL problem.

```
Game $\mathrm{DLOG}^{\mathcal{A}}_{\mathsf{GGen}}(\lambda)$ :
$(\mathbb{G}, p, g) \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda)$ ; $X \leftarrow\!\!\$\ \mathbb{G}$
$x \leftarrow \mathcal{A}(\mathbb{G}, p, g, X)$
If $g^x = X$ then return 1
Return 0
```

```
Game $\mathrm{CDH}^{\mathcal{A}}_{\mathsf{GGen}}(\lambda)$ :
$(\mathbb{G}, p, g) \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda)$ ; $x, y \leftarrow\!\!\$\ \mathbb{Z}_p$
$Z \leftarrow \mathcal{A}(\mathbb{G}, p, g, g^x, g^y)$
If $g^{xy} = Z$ then return 1
Return 0
```

```
Game $\mathrm{CT\text{-}CDH}^{\mathcal{A}}_{\mathsf{GGen}}(\lambda)$ :
$(\mathbb{G}, p, g) \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda)$ ; $x \leftarrow\!\!\$\ \mathbb{Z}_p$ ; $X \leftarrow g^x$
cid $\leftarrow 0$ ; $\ell \leftarrow 0$
$(j_i, \hat{Z}_i)_{i \in [\ell+1]} \leftarrow \mathcal{A}^{\mathrm{CHAL},\mathrm{DH}}(\mathbb{G}, p, g, X)$
If $\forall\, i \in [\ell+1] : \hat{Z}_i = Y^x_{j_i}$ then
    Return 1
Return 0
```

```
Oracle CHAL :
cid $\leftarrow$ cid $+ 1$
$Y_{\mathrm{cid}} \leftarrow\!\!\$\ \mathbb{G}$
Return $Y_{\mathrm{cid}}$

Oracle DH$(Y)$ :
$\ell \leftarrow \ell + 1$
Return $Y^x$
```

Fig. 1. The DLOG, CDH and CT-CDH games.

Blind signatures. This paper focuses on *four-move* and *five-move* blind signature schemes. Formally, a four-move (and five-move respectively) *blind signature scheme* BS is a tuple of efficient (randomized) algorithms

$$\mathsf{BS} = (\mathsf{BS.Setup}, \mathsf{BS.KG}, \mathsf{BS.S}_1, \mathsf{BS.S}_2, \mathsf{BS.U}_1, \mathsf{BS.U}_2, \mathsf{BS.U}_3, \mathsf{BS.Ver});$$
$$\mathsf{BS} = (\mathsf{BS.Setup}, \mathsf{BS.KG}, \mathsf{BS.S}_1, \mathsf{BS.S}_2, \mathsf{BS.S}_3, \mathsf{BS.U}_1, \mathsf{BS.U}_2, \mathsf{BS.U}_3, \mathsf{BS.Ver});$$

with the following behavior:

- The *parameter generation* algorithm $\mathsf{BS.Setup}(1^\lambda)$ outputs a string of parameters par, whereas the *key generation* algorithm $\mathsf{BS.KG}(\mathsf{par})$ outputs a key-pair $(\mathsf{sk}, \mathsf{pk})$, where sk is the *secret* (or *signing*) key and pk is the *public* (or *verification*) key. *All other algorithms of* BS *implicitly take* par *as input.*
- The interaction between the user and the signer to sign a message $m \in \{0,1\}^*$ with key-pair $(\mathsf{pk}, \mathsf{sk})$ is defined by the following experiments (1) for four-move and (2) for five-move blind signatures:

$$\left.\begin{array}{l} (\mathsf{st}^u_1, \mathsf{umsg}_1) \leftarrow \mathsf{BS.U}_1(\mathsf{pk}, m), (\mathsf{st}^s, \mathsf{smsg}_1) \leftarrow \mathsf{BS.S}_1(\mathsf{sk}, \mathsf{umsg}_1), \\ (\mathsf{st}^u_2, \mathsf{umsg}_2) \leftarrow \mathsf{BS.U}_2(\mathsf{st}^u_1, \mathsf{smsg}_1), \mathsf{smsg}_2 \leftarrow \mathsf{BS.S}_2(\mathsf{st}^s, \mathsf{umsg}_2), \\ \sigma \leftarrow \mathsf{BS.U}_3(\mathsf{st}^u_2, \mathsf{smsg}_2)\ . \end{array}\right\} \tag{1}$$

$$\left.\begin{array}{l} (\mathsf{st}^s_1, \mathsf{smsg}_1) \leftarrow \mathsf{BS.S}_1(\mathsf{sk}), (\mathsf{st}^u_1, \mathsf{umsg}_1) \leftarrow \mathsf{BS.U}_1(\mathsf{pk}, m, \mathsf{smsg}_1), \\ (\mathsf{st}^s_2, \mathsf{smsg}_2) \leftarrow \mathsf{BS.S}_2(\mathsf{st}^s_1, \mathsf{umsg}_1), (\mathsf{st}^u_2, \mathsf{umsg}_2) \leftarrow \mathsf{BS.U}_2(\mathsf{st}^u_1, \mathsf{smsg}_2), \\ \mathsf{smsg}_3 \leftarrow \mathsf{BS.S}_3(\mathsf{st}^s_2, \mathsf{umsg}_2), \sigma \leftarrow \mathsf{BS.U}_3(\mathsf{st}^u_2, \mathsf{smsg}_3)\ . \end{array}\right\} \tag{2}$$

  Here, σ is either the resulting *signature* or an *error message* $\perp$.
- The (deterministic) *verification algorithm* outputs a bit $\mathsf{BS.Ver}(\mathsf{pk}, m, \sigma)$.

We say that BS is (perfectly) *correct* if for every message $m \in \{0,1\}^*$, with probability one over the sampling of parameters and the key pair $(\mathsf{pk}, \mathsf{sk})$, the experiment in (1, 2) returns σ such that $\mathsf{BS.Ver}(\mathsf{pk}, m, \sigma) = 1$. All of our schemes are going to be perfectly correct.

As an intermediate object in our proof we will also need a two-move blind signature scheme, which is easily obtained from the above formalization of a four-move scheme.

One-more unforgeability. We consider variants of *one-more (strong) unforgeability* (OMUF, OMSUF), which ensure that no adversary playing the role of a user and *starting* ℓ signing interactions with the signer, in an arbitrarily concurrent fashion, can issue $\ell+1$ signatures (or more) for distinct messages, in the case of OMUF, or $\ell+1$ distinct message-signature pairs, in the case of OMSUF. The $\mathrm{OMUF}^{\mathcal{A}}_{\mathsf{BS}}$ and $\mathrm{OMSUF}^{\mathcal{A}}_{\mathsf{BS}}$ games for a blind signature scheme BS are defined in Figure 2, which we define for a general r-round protocol. The corresponding advantage of $\mathcal{A}$ is defined as $\mathsf{Adv}^{\mathrm{omuf/omsuf}}_{\mathsf{BS}}(\mathcal{A}, \lambda) := \Pr[(\mathrm{OMUF/OMSUF})^{\mathcal{A}}_{\mathsf{BS}}(\lambda) = 1]$. All of our analyses will further assume one or more random oracles, which are modeled as an additional oracle to which the adversary $\mathcal{A}$ is given access.

```
Game $\mathrm{OMUF}^{\mathcal{A}}_{\mathsf{BS}}(\lambda)$, $\boxed{\mathrm{OMSUF}^{\mathcal{A}}_{\mathsf{BS}}(\lambda)}$:
$\mathsf{par} \leftarrow \mathsf{BS.Setup}(1^\lambda)$
$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{BS.KG}(\mathsf{par})$
$\ell \leftarrow 0$ ; $\mathcal{I}_1, \ldots, \mathcal{I}_r \leftarrow \varnothing$
$\{(m^*_k, \sigma^*_k)\}_{k \in [\ell+1]} \leftarrow\$ \mathcal{A}^{\mathrm{S}_1,\ldots,\mathrm{S}_r}(\mathsf{par}, \mathsf{pk})$
If $\exists\, k_1 \neq k_2,\, m^*_{k_1} = m^*_{k_2}$ then
$\boxed{\text{If } \exists\, k_1 \neq k_2, (m^*_{k_1}, \sigma^*_{k_1}) = (m^*_{k_2}, \sigma^*_{k_2}) \text{ then}}$
    return 0
If $\exists\, k \in [\ell + 1]$ such that
        $\mathsf{BS.Ver}(\mathsf{pk}, m^*_k, \sigma^*_k) = 0$
    then return 0
Return 1
```

```
Oracle $\mathrm{S}_j(\mathrm{sid}, \mathsf{umsg})$ :                    // $j = 1, \ldots, r$
                                      // If BS is 5-move and $j = 1$,
                              // the input umsg is set as an empty string
If $\mathrm{sid} \notin \mathcal{I}_1, \ldots, \mathcal{I}_{j-1}$ or
    $\mathrm{sid} \in \mathcal{I}_j$ then return $\perp$
$\mathcal{I}_j \leftarrow \mathcal{I}_j \cup \{\mathrm{sid}\}$
If $j = 1$ then
    $\ell \leftarrow \ell + 1$
    $(\mathsf{st}^s_{\mathrm{sid}}, \mathsf{smsg}) \leftarrow \mathsf{BS.S}_1(\mathsf{sk}, \mathsf{umsg})$
If $j > 1$ then
    $(\mathsf{st}^s_{\mathrm{sid}}, \mathsf{smsg}) \leftarrow \mathsf{BS.S}_j(\mathsf{st}^s_{\mathrm{sid}}, \mathsf{umsg})$
                                          // for $j = r, \mathsf{st}^s_{\mathrm{sid}} = \perp$
Return smsg
```

Fig. 2. The OMUF and OMSUF security games for a 4-move or 5-move blind signature scheme BS, where $r = 2$ if BS is 4-move and $r = 3$ if BS is 5-move. Also, the input umsg of S_1 is set as an empty string if BS is 5-move. The OMUF game contains everything but the solid boxes, and the OMSUF game contains everything but the dashed boxes.

```
Game $\mathrm{BLIND}^{\mathcal{A}}_{\mathsf{BS}}(\lambda)$ :
$\mathsf{par} \leftarrow \mathsf{BS.Setup}(1^\lambda)$
$b \leftarrow\$ \{0, 1\}$
$b_0 \leftarrow b$ ; $b_1 \leftarrow 1 - b$
$b' \leftarrow\$ \mathcal{A}^{\mathrm{INIT},\mathrm{U}_1,\ldots,\mathrm{U}_r}(\mathsf{par})$
If $b' = b$ then return 1
Return 0

Oracle $\mathrm{INIT}(\tilde{\mathsf{pk}}, \tilde{m_0}, \tilde{m_1})$ :
$\mathsf{sess}_0 \leftarrow 1$ ; $\mathsf{sess}_1 \leftarrow 1$
$\mathsf{pk} \leftarrow \tilde{\mathsf{pk}}$
$m_0 \leftarrow \tilde{m_0}$ ; $m_1 \leftarrow \tilde{m_1}$
```

```
Oracle $\mathrm{U}_j(i, \mathsf{smsg}^{(i)})$ :                    // $j = 1, \ldots, 3$
                                      // If BS is 4-move and $j = 1$,
                              // the input $\mathsf{smsg}^{(i)}$ is set as an empty string
If $i \notin \{0, 1\}$ or $\mathsf{sess}_i \neq j$ then return $\perp$
$\mathsf{sess}_i \leftarrow \mathsf{sess}_i + 1$
If $j = 1$ then
    $(\mathsf{st}^u_i, \mathsf{umsg}^{(i)}) \leftarrow \mathsf{BS.U}_1(\mathsf{pk}, \mathsf{smsg}^{(i)})$
    Return $\mathsf{umsg}^{(i)}$
If $j = 2$ then
    $(\mathsf{st}^u_i, \mathsf{umsg}^{(i)}) \leftarrow \mathsf{BS.U}_2(\mathsf{st}^u_i, \mathsf{smsg}^{(i)})$
    Return $\mathsf{umsg}^{(i)}$
$\sigma_{b_i} \leftarrow \mathsf{BS.U}_3(\mathsf{st}^u_i, \mathsf{smsg}^{(i)})$                    // $j = 3$
If $\mathsf{sess}_0 = \mathsf{sess}_1 = 4$ and $\sigma_0 \neq \perp$ and $\sigma_1 \neq \perp$ then
    Return $(\sigma_0, \sigma_1)$
Else return $(\perp, \perp)$
```

Fig. 3. The BLIND security game for a 4-move or 5-move blind signature scheme BS. The only difference between the game defined for 4-move schemes and the game defined for 5-move schemes is that if BS is a 4-move scheme, the input umsg of U_1 is set as an empty string.

BLINDNESS. We also consider the standard notion of blindness against a malicious server that can, in particular, attempt to publish a malformed public key. The corresponding game $\mathrm{BLIND}^{\mathcal{A}}_{\mathsf{BS}}$ is defined in Figure 3, and for any adversary $\mathcal{A}$, we define its advantage as $\mathsf{Adv}^{\mathrm{blind}}_{\mathsf{BS}}(\mathcal{A}, \lambda) := \left|\Pr[\mathrm{BLIND}^{\mathcal{A}}_{\mathsf{BS}}(\lambda) = 1] - \frac{1}{2}\right|$.

GAME-PLAYING PROOFS. Several of our proofs adopt a lightweight variant of the standard "Game-Playing Framework" by Bellare and Rogaway [BR06].

FORKING LEMMA. In our proof, we utilize the general forking lemma in the version introduced by Bellare and Neven [BN06] stated below:

Lemma 2.1 (General Forking Lemma [BN06]). *Fix an integer $q \geqslant 1$ and a set H of size $h \geqslant 2$. Let $\mathcal{A}$ be a randomized algorithm that on input $x, h_1, \ldots, h_q$ returns a pair (I, aux), the first element of which is an integer in the range $1, \ldots, q$ or $\perp$ and the second element of which we refer to as a side output. Let IG be a randomized algorithm that we call the input generator. The accepting probability of $\mathcal{A}$, denoted acc, is defined as the probability that $I \neq \perp$ in the experiment*

$$x \leftarrow\$ \mathsf{IG};\ h_1, \ldots, h_q \leftarrow\$ H;\ (I, \mathsf{aux}) \leftarrow\$ \mathcal{A}(x, h_1, \ldots, h_q)$$

The forking algorithm $F_{\mathcal{A}}(x)$ associated to $\mathcal{A}$ is a randomized algorithm on input x defined as follows:

- *Pick random coins ρ for $\mathcal{A}$ and sample $h_1, \ldots, h_q \leftarrow\$ H$.*

- *Run* $(I, \mathsf{aux}) \leftarrow \mathcal{A}(x, h_1, \ldots, h_q; \rho)$
- *If* $I = \bot$, *return 0*
- *Sample* $h'_I, \ldots, h'_q \leftarrow\!\!{\scriptstyle\$}\ H$, *run* $(I', \mathsf{aux}') \leftarrow \mathcal{A}(x, h_1, \ldots, h_{I-1}, h'_I, \ldots, h'_q; \rho)$
- *If* $I = I'$ *and* $h_I \neq h'_I$, *return 1. Otherwise, return 0.*

Let $\mathsf{frk} = \mathsf{Pr}[b = 1 : x \leftarrow\!\!{\scriptstyle\$}\ \mathsf{IG}; b \leftarrow\!\!{\scriptstyle\$}\ F_{\mathcal{A}}(x)]$. *Then,*

$$\mathsf{frk} \geqslant \mathsf{acc}\left(\frac{\mathsf{acc}}{q} - \frac{1}{h}\right) \text{ , or alternatively, } \mathsf{acc} \leqslant \frac{q}{h} + \sqrt{q \cdot \mathsf{frk}} \ .$$

3 Four-Move Blind Signatures from CT-CDH

We present a four-move blind signature scheme BS_1, described in Figure 4. The scheme can be viewed as a blind version of the scheme by Goh and Jarecki [GJ03], where a signature consists of an element $Z = \mathsf{H}(m)^{\mathsf{sk}}$ with a discrete-log equality (DLEQ) proof proving that the discrete logarithms of (Z, pk) are equal to the base $(\mathsf{H}(m), g)$. However, we replace this proof with a witness-indistinguishable OR proof, which additionally accepts the discrete logarithm of a public random parameter W as a witness. Needless to say, this parameter is meant to be generated transparently, e.g., by hashing a constant, and nobody is meant to know this second witness. It is easy to show that the scheme satisfies correctness, but for completeness, we prove this in Section 3.1.

Blindness. The following theorem, proved in Section 3.2, shows that BS_1 is *statistically* blind when H'' is a random oracle. This property relies on the NIZK proof highlighted in Figure 4 to show equality of discrete logarithms to the base (g, h) of (X, Z). In Section 3.3, we also show that if we omit this NIZK proof, we still achieve *computationally* blind under the discrete logarithm assumption, without random oracles.

Theorem 3.1 (Blindness of BS_1). *Assume that* GGen *outputs the description of a group of prime order* $p = p(\lambda)$, *and let* $\mathsf{BS}_1 = \mathsf{BS}_1[\mathsf{GGen}]$. *For any adversary* $\mathcal{A}$ *playing the game* BLIND *making at most* $Q_{\mathsf{H}''} = Q_{\mathsf{H}''}(\lambda)$ *queries to* H'', *modeled as a random oracle, we have*

$$\mathsf{Adv}^{\mathrm{blind}}_{\mathsf{BS}_1}(\mathcal{A}, \lambda) \leqslant \frac{2Q_{\mathsf{H}''}}{p} \ .$$

One-more unforgeability. The next theorem establishes one-more unforgeability of BS_1 in the random oracle model under the CT-CDH assumption. We refer to Section 1.1 for a proof sketch, whereas the full proof is in Section 3.4.

Theorem 3.2 (OMUF of BS_1). *Assume that* GGen *outputs the description of a group of prime order* $p = p(\lambda)$, *and let* $\mathsf{BS}_1 = \mathsf{BS}_1[\mathsf{GGen}]$. *For any adversary* $\mathcal{A}$ *for game* OMUF *with running time* $t_{\mathcal{A}} = t_{\mathcal{A}}(\lambda)$, *making at most* $\ell = \ell(\lambda)$ *queries to* S_1, $Q_{\mathsf{H}_\star} = Q_{\mathsf{H}_\star}(\lambda)$ *queries to* $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}', \mathsf{H}''\}$, *modeled as random oracles, there exist adversaries* $\mathcal{B}$ *and* $\mathcal{B}'$ *for games* DLOG *and* CT-CDH, *respectively, such that*

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{BS}_1}(\mathcal{A}, \lambda) \leqslant \frac{\ell(\ell + Q_{\mathsf{H}''})}{p} + (\ell + 1)\left(\sqrt{Q_{\mathsf{H}'}\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p}\right) + \mathsf{Adv}^{\mathrm{ct-cdh}}_{\mathsf{GGen}}(\mathcal{B}', \lambda) \ .$$

Furthermore, $\mathcal{B}$ *runs in time* $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$ *and* $\mathcal{B}'$ *runs in time* $t_{\mathcal{B}'} \approx t_{\mathcal{A}}$, *makes* Q_{H} *challenge queries to* Chal *and* ℓ *queries to* Dh.

3.1 Correctness of BS_1

Theorem 3.3. BS_1 *satisfies correctness.*

```
Algorithm $\mathsf{BS}_1.\mathsf{Setup}(1^\lambda)$ :
$(\mathbb{G}, p, g) \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda)$ ; $W \leftarrow\!\!\$\ \mathbb{G}$
Select $\mathsf{H} : \{0,1\}^* \to \mathbb{G}$
Select $\mathsf{H}', \mathsf{H}'' : \{0,1\}^* \to \mathbb{Z}_p$
Return $\mathsf{par} = (\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \mathsf{H}'')$

Algorithm $\mathsf{BS}_1.\mathsf{KG}(\mathsf{par})$ :
$(\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \mathsf{H}'') \leftarrow \mathsf{par}$
$x \leftarrow\!\!\$\ \mathbb{Z}_p$ ; $X \leftarrow g^x$ ; $\mathsf{sk} \leftarrow x$ ; $\mathsf{pk} \leftarrow X$
Return $(\mathsf{sk}, \mathsf{pk})$

Algorithm $\mathsf{BS}_1.\mathsf{U}_1(\mathsf{par}, \mathsf{pk}, m)$ :
$X \leftarrow \mathsf{pk}$ ; $(\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \mathsf{H}'') \leftarrow \mathsf{par}$
$h' \leftarrow \mathsf{H}(m)$
$\beta \leftarrow\!\!\$\ \mathbb{Z}_p$ ; $h \leftarrow h' g^\beta$
$\mathsf{st}_1^u \leftarrow (m, \beta, X, h', h, W)$ ; $\mathsf{umsg}_1 \leftarrow h$
Return $(\mathsf{st}_1^u, \mathsf{umsg}_1)$

Algorithm $\mathsf{BS}_1.\mathsf{U}_2(\mathsf{st}_1^u, \mathsf{smsg}_1)$ :
$(m, \beta, X, h', h, W) \leftarrow \mathsf{st}_1^u$
$(Z, \boxed{\pi}, R_g, R_h, A) \leftarrow \mathsf{smsg}_1$ ; $\boxed{(\delta, s') \leftarrow \pi}$
If $\delta \neq \mathsf{H}''(h, X, Z, g^{s'} X^{-\delta}, h^{s'} Z^{-\delta})$ then
    return $\perp$
$\alpha_0, \alpha_1, \gamma_0, \gamma_1 \leftarrow\!\!\$\ \mathbb{Z}_p$
$Z' \leftarrow Z X^{-\beta}$ ; $R'_g \leftarrow R_g X^{-\gamma_0} g^{\alpha_0}$
$R'_h \leftarrow R_h R_g^{-\beta} Z'^{-\gamma_0} h'^{\alpha_0}$
$A' \leftarrow A W^{-\gamma_1} g^{\alpha_1}$
$c' \leftarrow \mathsf{H}'(m, h', Z', R'_g, R'_h, A')$
$c \leftarrow c' - \gamma_0 - \gamma_1$
$\mathsf{st}_2^u \leftarrow (c, \alpha_0, \alpha_1, \gamma_0, \gamma_1, Z, A, R_g, R_h, \mathsf{st}_1^u)$
$\mathsf{umsg}_2 \leftarrow c$
Return $(\mathsf{st}_2^u, c)$
```

```
Algorithm $\mathsf{BS}_1.\mathsf{U}_3(\mathsf{st}_2^u, \mathsf{smsg}_2)$ :
$(c, \alpha_0, \alpha_1, \gamma_0, \gamma_1, Z, A, R_g, R_h, \mathsf{st}_1^u) \leftarrow \mathsf{st}_2^u$
$(m, \beta, X, h', h, W) \leftarrow \mathsf{st}_1^u$
$(d, e, z_0, z_1) \leftarrow \mathsf{smsg}_2$
If $c \neq e + d$ or
    $(R_g X^d, R_h Z^d) \neq (g^{z_0}, h^{z_0})$ or
    $A W^e \neq g^{z_1}$ then return $\perp$
$d' \leftarrow d + \gamma_0$ ; $e' \leftarrow e + \gamma_1$
$z'_0 \leftarrow z_0 + \alpha_0$ ; $z'_1 \leftarrow z_1 + \alpha_1$
Return $\sigma \leftarrow (Z', d', e', z'_0, z'_1)$

Algorithm $\mathsf{BS}_1.\mathsf{S}_1(\mathsf{par}, \mathsf{sk}, \mathsf{umsg}_1)$ :
$x \leftarrow \mathsf{sk}$ ; $X \leftarrow g^x$ ; $h \leftarrow \mathsf{umsg}_1$
$(\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \mathsf{H}'') \leftarrow \mathsf{par}$ ; $Z \leftarrow h^x$
$z_1, e, r_0, \boxed{s} \leftarrow\!\!\$\ \mathbb{Z}_p$
$R_g \leftarrow g^{r_0}$ ; $R_h \leftarrow h^{r_0}$ ; $A \leftarrow g^{z_1} W^{-e}$
$\delta \leftarrow \mathsf{H}''(h, X, Z, g^s, h^s)$
$\pi \leftarrow (\delta, \delta \cdot x + s)$
$\mathsf{st}^s \leftarrow (x, z_1, e, r_0)$ ; $\mathsf{smsg}_1 \leftarrow (Z, \boxed{\pi}, R_g, R_h, A)$
Return $(\mathsf{st}^s, \mathsf{smsg}_1)$

Algorithm $\mathsf{BS}_1.\mathsf{S}_2(\mathsf{st}^s, \mathsf{umsg}_2)$ :
$(x, z_1, e, r_0) \leftarrow \mathsf{st}^s$
$d \leftarrow c - e$ ; $z_0 \leftarrow r_0 + d \cdot x$
Return $\mathsf{smsg}_2 \leftarrow (d, e, z_0, z_1)$

Algorithm $\mathsf{BS}_1.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m, \sigma)$ :
$(Z, d, e, z_0, z_1) \leftarrow \sigma$ ; $X \leftarrow \mathsf{pk}$
$(\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}') \leftarrow \mathsf{par}$
$h \leftarrow \mathsf{H}(m)$ ; $A \leftarrow g^{z_1} W^{-e}$
$R_g \leftarrow g^{z_0} X^{-d}$, $R_h \leftarrow h^{z_0} Z^{-d}$
If $e + d \neq \mathsf{H}'(m, h, Z, R_g, R_h, A)$ then
    return 0
Return 1
```

Fig. 4. The blind signature scheme $\mathsf{BS}_1 = \mathsf{BS}_1[\mathsf{GGen}]$. The highlighted boxes denote the NIZK proof to show the equality of discrete log to the bases (g, h) of (X, Z). We also give a protocol-style description of BS_1 in Figure 16.

Proof. Consider an honestly generated signature $\sigma = (Z', d', e', z'_0, z'_1)$ for a message m. We use variables as defined in the signing protocol.

First, we argue that the checks in $\mathsf{BS}_1.\mathsf{U}_2$ and $\mathsf{BS}_1.\mathsf{U}_3$ verifies. For the check in $\mathsf{BS}_1.\mathsf{U}_2$, since $s' = \delta x + s$ and $X = g^x, Z = h^x$, we have $g^{s'} X^{-\delta} = g^s$ and $h^{s'} Z^{-\delta} = h^s$. Thus, $\mathsf{H}''(h, X, Z, g^{s'} X^{-\delta}, h^{s'} Z^{-\delta}) = \mathsf{H}''(h, X, Z, g^s, h^s) = \delta$.

For the check in $\mathsf{BS}_1.\mathsf{U}_3$, $c = e + d$ by how the signer computes e, $A \cdot W^{-e} = g^{z_1}$ by how A is generated, and lastly $(R_g \cdot X^d, R_h \cdot Z^d) = (g^{r_0 + dx}, h^{r_0 + dx}) = (g^{z_0}, h^{z_0})$, where the first equality follows from $R_g = g^{r_0}, R_h = h^{r_0}, X = g^x, Z = h^x$ and the second equality follows from $z_0 = dx + r_0$.

Now, to argue the validity of the signature, let $h' = \mathsf{H}(m)$. Then, we have to argue the following to say that the signature is valid:

1. $c' = e' + d'$. This follows from $c = e + d$ as $c' = c + \gamma_0 + \gamma_1 = e + d + \gamma_0 + \gamma_1 = e' + d'$.
2. $g^{z'_1} \cdot W^{-e'} = A'$. This follows from $z'_1 = z_1 + \alpha_1$ and $e' = e + \gamma_1$, as $g^{z'_1} \cdot W^{-e'} = (g^{z_1} W^{-e}) \cdot (W^{-\gamma_1} g^{\alpha_1}) = A \cdot (W^{-\gamma_1} g^{\alpha_1}) = A'$.
3. $g^{z'_0} \cdot X^{-d'} = R'_g$. This follows from $z'_0 = z_0 + \alpha_0$ and $d' = d + \gamma_0$, as

$$g^{z'_0} \cdot X^{-d'} = (g^{z_0} X^{-d}) \cdot (X^{-\gamma_0} g^{\alpha_0}) = R_g \cdot (X^{-\gamma_0} g^{\alpha_0}) = R'_g ,$$

where the second equality follows from the check $R_g \cdot X^d = g^{z_0}$ in $\mathsf{BS}_1.\mathsf{U}_3$.

4. $h'^{z_0'} \cdot Z'^{-e'} = R_h'$. This follows from $z_0' = z_0 + \alpha_0, d' = d + \gamma_0, h' = h \cdot g^{-\beta}$, and $Z' = Z \cdot X^{-\beta}$ as

$$\begin{aligned} h'^{z_0'} \cdot Z'^{-d'} &= h'^{z_0} Z'^{-d} \cdot (h'^{\alpha_0} Z'^{-\gamma_0}) \\ &= h^{z_0} Z^{-d} \cdot (g^{z_0} X^{-d})^{-\beta} \cdot (h'^{\alpha_0} Z'^{-\gamma_0}) \\ &= R_h \cdot R_g^{-\beta} \cdot (h'^{\alpha_0} Z'^{-\gamma_0}) = R_h' \ , \end{aligned}$$

where the second to last equality follows from the check $R_h \cdot Z^d = g^{z_0}$ and $R_g \cdot X^d = g^{z_0}$ in $\mathsf{BS}_1.\mathsf{U}_3$.

By (2, 3, 4), we have

$$\begin{aligned} \mathsf{H}'(m, h', Z', g^{z_0'} X^{-d'}, h'^{z_0'} Z'^{-d'}, g^{z_1'} W^{-e'}) &= \mathsf{H}'(m, h', Z', R_g', R_h', A') \\ &= c' = e' + d' \ , \end{aligned}$$

proving the scheme's correctness. □

3.2 Proof of Theorem 3.1 (Blindness of BS_1)

To prove blindness, we consider the following sequence of games.
Game $\mathbf{G}_0^{\mathcal{A}}$: This game is the BLIND game of BS_1 where $\mathcal{A}$ has $Q_{\mathsf{H}''}$ queries access to the random oracle H''. We also assume w.l.o.g. that $\mathcal{A}$ has already made any query to H'' which the user oracle has to make. In particular, the query to H'' to verify π is done by $\mathcal{A}$ before $\mathcal{A}$ sends π.
Game $\mathbf{G}_1^{\mathcal{A}}$: This game made the following changes:

- In the oracle $\textsc{Init}(\tilde{\mathsf{pk}}, \tilde{m}_0, \tilde{m}_1)$, the oracle additionally parses $X \leftarrow \mathsf{pk}$ and sets $\mathsf{sk} \leftarrow x = \log_g X$ found by exhaustive search.
- When the oracle $\mathrm{U}_2(i, \mathsf{smsg}_1^{(i)})$ receives $\mathsf{smsg}_1^{(i)}$ from $\mathcal{A}$, it parses $(Z_i, \pi_i = (\delta_i, s_i), R_{g,i}, R_{h,i}, A_i) \leftarrow \mathsf{smsg}_1^{(i)}$. Then, it computes $S_{g,i} = g^{s_i} X^{-\delta_i}$, $S_{h,i} = h_i^{s_i} Z_i^{-\delta_i}$, and checks if $\delta_i \neq \mathsf{H}''(h_i, X, Z_i, S_{g,i}, S_{h,i})$. If this check passes, the game now aborts if $Z_i \neq (h_i)^{\mathsf{sk}}$ where h_i is the message replied by the first query to $\mathrm{U}_1(i, \cdot)$.

The winning probability of $\mathcal{A}$ only changes when the new abort occurs in either signing sessions, which corresponds to the following event:

$$Z_i \neq (h_i)^{\mathsf{sk}} \ \wedge \ \delta_i = \mathsf{H}''(h_i, X, Z_i, S_{g,i}, S_{h,i}) \ .$$

The soundness of the NIZK proof implies that the event occurs with negligible probability. More precisely, with how $S_{g,i}, S_{h,i}$ is defined and that $Z \neq h_i^{\mathsf{sk}} = X^{\log_g h}$, we have $(S_{g,i})^{-\log_g h_i} S_{h,i} = h_i^{-s_i} h_i^{\delta_i \mathsf{sk}} h_i^{s_i} Z_i^{-\delta_i} = \left(h_i^{-\mathsf{sk}} Z_i\right)^{-\delta_i}$. Since $h_i^{-\mathsf{sk}} Z_i \neq 1_{\mathbb{G}}$, there is only one value of $\delta_i \in \mathbb{Z}_p$ to satisfy such equation. Since δ_i is uniformly at random after fixing the query and $\mathcal{A}$ makes at most $Q_{\mathsf{H}''}$ queries to H'', with the union bound over the two signing sessions

$$\left|\mathsf{Pr}[\mathbf{G}_0^{\mathcal{A}} = 1] - \mathsf{Pr}[\mathbf{G}_1^{\mathcal{A}} = 1]\right| \leqslant \frac{2Q_{\mathsf{H}''}}{p} \ .$$

For the last step, we show that the transcript and returned signatures are distributed identically for both cases of $b = 0, 1$, which implies $\mathsf{Pr}[\mathbf{G}_1^{\mathcal{A}} = 1] = \frac{1}{2}$ concluding the proof.

To show this, first, assume w.l.o.g. that the randomness of $\mathcal{A}$ is fixed and $\mathcal{A}$ only outputs messages in the transcript where neither the game nor the user oracles abort; thus, $\mathcal{A}$ receives valid signatures (σ_0, σ_1). (If a user oracle aborts, for each signing session, the adversary will only see h_i and c_i which are both blinded to be uniformly random over $\mathbb{G}$ and $\mathbb{Z}_p$ respectively.)

Let View_A denote the set of all possible views of $\mathcal{A}$ in the game $\mathbf{G}_1^{\mathcal{A}}$. A view $\varDelta \in \mathrm{View}_A$ is of the form $\varDelta = (W, X, m_0, m_1, T_0, T_1, \sigma_0, \sigma_1)$ where for $i \in \{0, 1\}$, $T_i = (h_i, Z_i, R_{g,i}, R_{h,i}, A_i, c_i, d_i, e_i, z_{0,i}, z_{1,i})$ denotes the transcript of the interaction between $\mathcal{A}$ and the user oracle in signing session i (omitted π_i as it is distributed independently of (m_0, m_1) given (h_i, Z_i)), and $\sigma_i = (Z_i', d_i', e_i', z_{0,i}', z_{1,i}')$ denotes the valid signature for message m_i. We need to show that the actual adversarial view, denoted as v_A, is distributed identically between $b = 0, 1$. Since $\mathcal{A}$'s randomness is fixed, v_A only depends on the user randomness $\eta = (\beta_i, \alpha_{0,i}, \alpha_{1,i}, \gamma_{0,i}, \gamma_{1,i})_{i \in \{0,1\}}$. We write $v_A(\eta)$ to make this explicit.

Since we assume $\mathcal{A}$ does not make the game abort, for the signatures $\sigma_{b_i} = (Z'_{b_i}, d'_{b_i}, e'_{b_i}, z'_{0,b_i}, z'_{1,b_i})$ in any transcript $\Delta \in \mathrm{View}_A$, we have that $Z'_{b_i} = {h'_{b_i}}^{\mathsf{sk}}$ where $h'_{b_i} = \mathsf{H}(m_{b_i})$. This is because of the abort introduced in $\mathbf{G}_1^{\mathcal{A}}$ that induces $Z_i = h_i^x$ leading to $Z'_{b_i} = Z_i X^{-\beta_i} = (h_i g^{-\beta_i})^{\mathsf{sk}} = {h'_{b_i}}^{\mathsf{sk}}$.

To show that the distribution of v_A is identical between $b = 0$ and $b = 1$, consider a view $\Delta \in \mathrm{View}_A$. We now show that there exists a unique η such that $v_A(\eta) = \Delta$, regardless of whether $b = 0$ or $b = 1$. More specifically, we claim that for both $b = 0$ and $b = 1$, $v_A(\eta) = \Delta$ if and only if for $i \in \{0, 1\}$, η satisfies

$$\begin{aligned} &\beta_i = \log_g h_i - \log_g h'_{b_i} \ , \\ &\alpha_{0,i} = z'_{0,b_i} - z_{0,i}, \ \ \alpha_{1,i} = z'_{1,b_i} - z_{1,i} \ , \\ &\gamma_{0,i} = d'_{b_i} - d_i, \ \ \gamma_{1,i} = e'_{b_i} - e_i \ . \end{aligned} \tag{3}$$

For the "only if" direction, i.e., if $v_A(\eta) = \Delta$, then η satisfies Equation (3), this is true by how the user algorithm of BS_1 is defined.

To show the "if" direction, suppose η satisfies Equation (3), we show that $v_A(\eta) = \Delta$. Particularly, we have to show that the user messages and signatures from oracles $\mathrm{U}_1, \mathrm{U}_2$ and U_3 are $(h_0, h_1), (c_0, c_1)$, and (σ_0, σ_1) respectively.

Again, since we only consider a non-aborting transcript Δ, we have the following guarantees for $i \in \{0, 1\}$:

$$Z_i = h_i^{\mathsf{sk}}, \ Z'_{b_i} = {h'_{b_i}}^{\mathsf{sk}}, \tag{4}$$

$$c_i = d_i + e_i, \ R_{g,i} X^{d_i} = g^{z_{0,i}}, \ R_{h,i} Z_i^{d_i} = h_i^{z_{1,i}}, \ A_i W^{e_i} = g^{z_{1,i}} \ , \tag{5}$$

$$d'_{b_i} + e'_{b_i} = \mathsf{H}'(m_{b_i}, h'_{b_i}, Z'_{b_i}, X^{-d'_{b_i}} g^{z'_{0,b_i}}, {Z'_{b_i}}^{-d'_{b_i}} {h'_{b_i}}^{z'_{0,b_i}}, W^{-e'_{b_i}} g^{z'_{1,b_i}}) \ , \tag{6}$$

where Equation (4) follows from the discussion above, Equation (5) follows from the check in $\mathsf{BS}_1.\mathsf{U}_3$, and Equation (6) follows from the validity of the signatures.

First, we argue that h_i is the user message from $\mathrm{U}_1(i, \cdot)$ for $i \in \{0, 1\}$: recall that the user oracle outputs $\mathsf{H}(m_{b_i}) \cdot g^{\beta_i}$ and by the value of β_i from Equation (3), $\mathsf{H}(m_{b_i}) \cdot g^{\beta_i} = h'_{b_i} \cdot g^{\beta_i} = h_i$, so the first user message is consistent with Δ. Thus, the next message from $\mathcal{A}$ will be $Z_i, R_{g,i}, R_{h,i}, A_i$ from the transcript Δ.

Next, we argue that the second user message from $\mathrm{U}_2(i, \cdot)$ is c_i. To do this, we consider the blinded values of $Z_i, R_{g,i}, R_{h,i}, A_i$.

$$\begin{aligned} Z_i X^{-\beta_i} &= h_i^{\mathsf{sk}} g^{-\beta_i \mathsf{sk}} = (h_i g^{-\beta_i})^{\mathsf{sk}} = {h'_{b_i}}^{\mathsf{sk}} = Z'_{b_i}, \text{ the last equality by equation (4)} \\ R'_{g,i} &= R_{g,i} X^{-\gamma_{0,i}} g^{\alpha_{0,i}} = (X^{-d_i} g^{z_{0,i}}) X^{-\gamma_{0,i}} g^{\alpha_{0,i}}; \text{ By equation (5)} \\ &= X^{-d_i - \gamma_{0,i}} g^{z_{0,i} + \alpha_{0,i}} = X^{-d'_{b_i}} g^{z'_{0,b_i}}, \text{ By equation (3)} \\ R'_{h,i} &= R_{h,i} R_{g,i}^{-\beta_i} {Z'_{b_i}}^{-\gamma_{0,i}} {h'_i}^{\alpha_{0,i}} \\ &= \left(Z^{-d_i} h_i^{z_{0,i}}\right) \left(X^{-d_i} g^{z_{0,i}}\right)^{-\beta_i} {Z'_{b_i}}^{-\gamma_{0,i}} {h'_{b_i}}^{\alpha_{0,i}}; \text{ By equation (5)} \\ &= \left(Z X^{-\beta_i}\right)^{-d_i} \left(h_i g^{-\beta_i}\right)^{z_{0,i}} {Z'_{b_i}}^{-\gamma_{0,i}} {h'_{b_i}}^{\alpha_{0,i}} \\ &= {Z'_{b_i}}^{-d_i - \gamma_{0,i}} {h'_{b_i}}^{z_{0,i} + \alpha_{0,i}} = {Z'_{b_i}}^{-d'_{b_i}} {h'_{b_i}}^{z'_{0,b_i}}, \text{ By equation (3)} \\ A'_i &= A W^{-\gamma_{1,i}} g^{\alpha_{1,i}} = (W^{-e_i} g^{z_{1,i}}) W^{-\gamma_{1,i}} g^{\alpha_{1,i}}; \text{ By equation (5)} \\ &= W^{-e_i - \gamma_{1,i}} g^{z_{1,i} + \alpha_{1,i}} = W^{-e'_{b_i}} g^{z'_{1,b_i}}, \text{ By equation (3)} \ . \end{aligned}$$

Therefore, the message returned from $\mathrm{U}_2(i, \cdot)$ is

$$\begin{aligned} &\mathsf{H}'(m_{b_i}, h'_{b_i}, Z_i X^{-\beta_i}, R'_{g,i}, R'_{h,i}, A'_i) - \gamma_{0,i} - \gamma_{1,i} \\ &= \mathsf{H}'(m_{b_i}, h'_{b_i}, Z'_{b_i}, X^{-d'_{b_i}} g^{z'_{0,b_i}}, {Z'_{b_i}}^{-d'_{b_i}} {h'_{b_i}}^{z'_{0,b_i}}, W^{-e'_{b_i}} g^{z'_{1,b_i}}) - \gamma_{0,i} - \gamma_{1,i} \\ &= d'_{b_i} + e'_{b_i} - \gamma_{0,i} - \gamma_{1,i} = d_i + e_i = c_i \ , \end{aligned}$$

which is consistent with Δ. Thus, the next message from $\mathcal{A}$ will be $d_i, e_i, z_{0,i}, z_{1,i}$ from the transcript Δ. Lastly, the signatures from the oracle U_3 are

$$(Z_i X^{-\beta_i}, d_i + \gamma_{0,i}, e_i + \gamma_{1,i}, z_{0,i} + \alpha_{0,i}, z_{1,i} + \alpha_{1,i}) = (Z'_{b_i}, d'_{b_i}, e'_{b_i}, z'_{0,b_i}, z'_{1,b_i}) = \sigma_{b_i},$$

which are exactly the signatures in Δ. □

3.3 Computational Blindness of BS_1 without NIZK

As mentioned before, we can remove the NIZK proof from our scheme BS_1 (resulting in a scheme which we will call BS'_1) and still achieve computational blindness according to the following theorem. We stress that here we make no assumptions on the hash functions used by BS'_1.

Theorem 3.4 (Computational Blindness of BS'_1). *Assume that* GGen *outputs the description of a group of prime order $p = p(\lambda)$, and let $\mathsf{BS}'_1 = \mathsf{BS}'_1[\mathsf{GGen}]$. For any adversary $\mathcal{A}$ for game* BLIND *running in time $t_\mathcal{A} = t_\mathcal{A}(\lambda)$, there exists an adversary $\mathcal{B}$ for game* DLOG *running in time $t_\mathcal{B} \approx 2t_\mathcal{A}$ such that*

$$\mathsf{Adv}^{\mathrm{blind}}_{\mathsf{BS}'_1}(\mathcal{A}, \lambda) \leqslant 2\sqrt{\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{2}{p} .$$

Proof. The proof for this theorem mainly follows the proof of Theorem 3.1 with the only difference being the game $\mathbf{G}^\mathcal{A}_1$ and its transition from $\mathbf{G}^\mathcal{A}_0$. We define the game $\mathbf{G}^\mathcal{A}_1$ as follows:
Game $\mathbf{G}^\mathcal{A}_1$: This game made the following changes:

- In the oracle $\textsc{Init}(\tilde{\mathsf{pk}}, \tilde{m}_0, \tilde{m}_1)$, the oracle additionally parses $X \leftarrow \mathsf{pk}$ and sets $\mathsf{sk} \leftarrow x = \log_g X$ found by exhaustive search.
- For both $i \in \{0,1\}$, in the oracle $\mathrm{U}_3(i, \mathsf{smsg}^{(i)}_2)$ after it receives $\mathsf{smsg}^{(i)}_1, \mathsf{smsg}^{(i)}_2$ from $\mathcal{A}$ and parses $(Z_i, R_{g,i}, R_{h,i}, A_i) \leftarrow \mathsf{smsg}^{(i)}_1$ and $(d_i, e_i,\ z_{0,i}, z_{1,i}) \leftarrow \mathsf{smsg}^{(i)}_2$. Then, if the user algorithm $\mathsf{BS}'_1.\mathsf{U}_3$ does not abort but $Z_i \neq h_i^{\mathsf{sk}}$ where h_i is the first message $\mathrm{U}_1(i, \cdot)$ replied to $\mathcal{A}$, the game aborts.

Fix a signing session $i \in \{0,1\}$ and let Bad_i be the event where the abort described occurs in signing session i, i.e., $Z_i \neq h_i^{\mathsf{sk}}$ but the user algorithm does not abort. This gives

$$|\Pr[\mathbf{G}^\mathcal{A}_1 = 1] - \Pr[\mathbf{G}^\mathcal{A}_0 = 1]| \leqslant \Pr[\mathsf{Bad}_0 \vee \mathsf{Bad}_1] .$$

Note that the event Bad_i only depends on the two user messages in the signing protocol, i.e., (h_i, c_i) (since the event occurs before the signatures are returned).

To argue the probability of event Bad_i occurring, we will give a reduction $\mathcal{B}$ rewinding the adversary $\mathcal{A}$ and argue that if Bad_i occurs in both runs, $\mathcal{B}$ can extract $\log_g W$.

Before giving $\mathcal{B}$, we make the following observation that h_i and c_i are uniformly random in $\mathbb{G}$ and $\mathbb{Z}_p$ respectively. To see this, first consider that $h_i = h'_i g^{\beta_i}$ and $c_i = \mathsf{H}'(m_{b_i}, h'_i, Z'_i, R'_{g,i}, R'_{h,i}, A'_i) - \gamma_{0,i} - \gamma_{1,i}$ where $h'_i = \mathsf{H}(m_{b_i})$ and $Z'_i, R'_{g,i}, R'_{h,i}, A'_i$ are the blinded values of $Z_i, R_{g,i}, R_{h,i}, A_i$ respectively. We specifically note that $A'_i = A_i g^{\alpha_{1,i}} W^{-\gamma_{1,i}}$ is uniform over $\mathbb{G}$ and is independent of $\gamma_{1,i}$. This is because conditioning on a value for $\gamma_{1,i}$, A'_i takes on any element in $\mathbb{G}$ with probability $1/p$ due to $\alpha_{1,i}$ being uniform over $\mathbb{Z}_p$ and independent of $\gamma_{1,i}$. Then, the distribution of (h_i, c_i) can now be seen as dependent only on the signer messages $R_{g,i}, A_i, R_{h,i}, Z_i$, the blinding randomness $\beta_i, \alpha_{0,i}, \gamma_{0,i}, \gamma_{1,i}$ and A'_i. Conditioning on every values other than β_i and $\gamma_{1,i}$, we can see that h_i is uniform over $\mathbb{G}$ as β_i is uniform over $\mathbb{Z}_p$ and c_i is uniform over $\mathbb{Z}_p$ as $\gamma_{1,i}$ is uniform over $\mathbb{Z}_p$. This means that the probability of Bad_i stays the same even if h_i and c_i are uniformly randomly sampled instead of generated by following the protocol.

Then, using the above observation, consider the following reduction $\mathcal{B}$ playing the DLOG game and running $\mathcal{A}$.

1. The reduction $\mathcal{B}$ takes as input $(\mathbb{G}, p, g, W)$ and runs $\mathcal{A}$ on input $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W)$. It also fixes the randomness to be used in the signing session $1-i$ and the first round message h_i of signing session i in advance.

2. The oracle Init, $\mathrm{U}_1(1-i,\cdot), \mathrm{U}_2(1-i,\cdot)$, and $\mathrm{U}_3(1-i,\cdot)$ are answered as in the game $\mathbf{G}_0^{\mathcal{A}}$. The oracle $\mathrm{U}_1(i,\cdot)$ instead of computing the values as usual answers with h_i instead. While for $\mathrm{U}_2(i,\cdot)$, $\mathcal{B}$ answers with a freshly sampled $c_i \leftarrow_{\$} \mathbb{Z}_p$.
3. For the call to $\mathrm{U}_3(i, \mathsf{smsg}_3^{(i)})$, if the user algorithm does not abort $\mathcal{B}$ rewinds the adversary $\mathcal{A}$ to when it queries $\mathrm{U}_2(i, \mathsf{smsg}_2^{(i)})$ and replies with a fresh $c_i' \leftarrow_{\$} \mathbb{Z}_p$. The oracles for the signing session $1-i$ still use the same randomness from the previous run.
4. After the rewinding, for the call to $\mathrm{U}_3(i, \mathsf{smsg'}_3^{(i)})$, if the user algorithm does not abort, we have $(d_i, e_i, z_{0,i}, z_{1,i}) \leftarrow \mathsf{smsg}_3^{(i)}$ and $(d_i', e_i', z_{0,i}', z_{1,i}') \leftarrow \mathsf{smsg'}_3^{(i)}$. If $e_i \neq e_i'$, return $(z_{1,i} - z_{1,i}')(e_i - e_i')^{-1}$. Otherwise, abort.

It is clear that the running time of $\mathcal{B}$ is about that of $\mathcal{A}$. Then, we argue the success probability of the reduction $\mathcal{B}$ by considering the event Bad_i. We note that the event Bad_i cannot be detected efficiently; however, here we show that if such event occurs in both runs (even without $\mathcal{B}$ detecting Bad_i), the reduction $\mathcal{B}$ will find $\log_g W$. More specifically, we consider the following event frk such that the event Bad_i occurs in both the first and the rewound run of $\mathcal{A}$ in the reduction $\mathcal{B}$ and that the outputs of $\mathrm{U}_2(i,\cdot)$ over the two runs are different (i.e., $c_i' \neq c_i$). If this event occurs, then $\mathcal{A}$ has sent $(Z_i, R_{g,i}, R_{h,i}, A_i)$ and $(d_i, e_i, z_{0,i}, z_{1,i}), (d_i', e_i', z_{0,i}', z_{1,i}')$ such that

(i) $Z_i \neq h_i^x$.
(ii) $d_i + e_i = c_i \neq c_i' = d_i' + e_i'$.
(iii) $(R_{g,i}, R_{h,i}) = (g^{z_{0,i}} X^{-d_i}, h_i^{z_{0,i}} Z_i^{-d_i}) = (g^{z_{0,i}'} X^{-d_i'}, h_i^{z_{0,i}'} Z_i^{-d_i'})$.
(iv) $A_i = g^{z_{1,i}} W^{-e_i} = g^{z_{1,i}'} W^{-e_i'}$.

By considering (iii),

$$Z_i^{d_i - d_i'} = h_i^{z_{0,i} - z_{0,i}'} = g^{(z_{0,i} - z_{0,i}') \log_g h_i} = X^{(d_i - d_i') \log_g h_i} = h_i^{\mathsf{sk}(d_i - d_i')}\ .$$

Then, $d_i = d_i'$ follows from $Z_i \neq h_i^{\mathsf{sk}}$. Thus, $e_i \neq e_i'$ and $(z_{1,i} - z_{1,i}')(e_i - e_i')^{-1} = \log_g W$ by (iv). This shows that if frk occurs $\mathcal{B}$ wins the DLOG game, i.e., $\Pr[\mathsf{frk}] \leqslant \mathsf{Adv}_{\mathsf{GGen}}^{\mathsf{dlog}}(\mathcal{B}, \lambda)$.

Now, we bound $\Pr[\mathsf{frk}]$ using the forking lemma (Lemma 2.1). To this end, we define a wrapper $\mathcal{A}_i$ over $\mathcal{A}$ where $\mathcal{A}_i$ takes as input the instance $(\mathbb{G}, p, g, W)$, the challenge c_i, and a randomness ρ which is used to derive the random tape for $\mathcal{A}$, h_i, and the randomness used in signing session $1-i$. The wrapper $\mathcal{A}_i$ then simulates the user oracles as $\mathcal{B}$ does and returns $I = 1$ when Bad_i occurs. Otherwise $\mathcal{A}_i$ returns $\perp$. This means that the probability that $I = 1 \neq \perp$ is $\Pr[\mathsf{Bad}_i]$. Also, we can see that the event frk corresponds to the event where $\mathcal{A}_i$ is run twice with the same inputs except the two different $c_i \neq c_i'$, and both runs return I and I' such that $I = I' \neq \perp$. Thus by the forking lemma, we have

$$\Pr[\mathsf{Bad}_i] \leqslant \sqrt{\Pr[\mathsf{frk}]} + \frac{1}{p} \leqslant \sqrt{\mathsf{Adv}_{\mathsf{GGen}}^{\mathsf{dlog}}(\mathcal{B}, \lambda)} + \frac{1}{p}\ .$$

Applying the union bound over $i \in \{0, 1\}$ concludes the proof. □

3.4 Proof of Theorem 3.2 (OMUF of BS_1)

To prove one-more unforgeability for BS_1, we consider the following sequence of games. Here, we describe the sequence of games in text, while the pseudocode version of the games can be found in Figure 5.

Game $\mathbf{G}_0^{\mathcal{A}}$: The game first generates the public parameters $\mathsf{par} \leftarrow_{\$} \mathsf{BS}_1.\mathsf{Setup}(1^\lambda)$ and the public and secret keys $(\mathsf{pk}, \mathsf{sk}) \leftarrow_{\$} \mathsf{BS}_1.\mathsf{KG}(\mathsf{par})$. Then, the game interacts with an adversary $\mathcal{A}(\mathsf{par}, \mathsf{pk})$ with access to the signing oracles $\mathrm{S}_1, \mathrm{S}_2$ and the random oracles $\mathsf{H}, \mathsf{H}', \mathsf{H}''$ which are simulated by lazy sampling. The adversary $\mathcal{A}$ (w.l.o.g.) queries the signing oracle S_1 for ℓ times and the random oracles $\mathsf{H}, \mathsf{H}', \mathsf{H}''$ for $Q_{\mathsf{H}}, Q_{\mathsf{H}'}, Q_{\mathsf{H}''}$ times respectively. At the end of the game, $\mathcal{A}$ outputs $\ell + 1$ message-signature pairs (m_k^*, σ_k^*) for $k \in [\ell + 1]$. The

```
Game $\mathbf{G}_0^{\mathcal{A}}$, $\boxed{\mathbf{G}_1^{\mathcal{A}}}$, $\boxed{\mathbf{G}_2^{\mathcal{A}}}$, $\boxed{\mathbf{G}_3^{\mathcal{A}}}$, $\boxed{\mathbf{G}_4^{\mathcal{A}}}$:
$(\mathbb{G}, p, g) \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda)$
$W \leftarrow\!\!\$\ \mathbb{G}$                                   // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_1^{\mathcal{A}}$
$w \leftarrow\!\!\$\ \mathbb{Z}_p$ ; $W \leftarrow g^w$                 // $\mathbf{G}_2^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$\mathsf{par} \leftarrow (\mathbb{G}, p, g, W)$
$x \leftarrow\!\!\$\ \mathbb{Z}_p$ ; $X \leftarrow g^x$
$\mathsf{sk} \leftarrow x$ ; $\mathsf{pk} \leftarrow X$
$\ell \leftarrow 0$ ; $\mathcal{I}_1, \mathcal{I}_2 \leftarrow \varnothing$
$\{(m_k^*, \sigma_k^*)\}_{k \in [\ell+1]} \leftarrow\!\!\$\ \mathcal{A}^{\mathrm{S}_1, \mathrm{S}_2}(\mathsf{par}, \mathsf{pk})$
If $\exists\ k_1 \neq k_2$ such that $m_{k_1}^* = m_{k_2}^*$
  then return 0
If $\exists\ k \in [\ell + 1]$ such that
  $\mathsf{BS}_1.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m_k^*, \sigma_k^*) = 0$
    or $((Z_k^*, d_k^*, e_k^*, z_{0,k}^*, z_{1,k}^*) \leftarrow \sigma_k^*$
    and $Z_k^* \neq \mathsf{H}(m_k^*)^{\mathsf{sk}})$ // $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
      then return 0
Return 1

Oracle $\mathsf{H}(m)$:
If $\mathsf{H}(m) \neq \perp$ then
  return $\mathsf{H}(m)$
$\mathsf{H}(m) \leftarrow\!\!\$\ \mathbb{G}$ Return $\mathsf{H}(m)$

Oracle $\mathsf{H}_\star(\mathrm{str})$ for $\mathsf{H}_\star \in \{\mathsf{H}', \mathsf{H}''\}$:
If $\mathsf{H}_\star(\mathrm{str}) \neq \perp$ then
  return $\mathsf{H}_\star(\mathrm{str})$
$\mathsf{H}_\star(\mathrm{str}) \leftarrow\!\!\$\ \mathbb{Z}_p$
Return $\mathsf{H}_\star(\mathrm{str})$
```

```
Oracle $\mathrm{S}_1(\mathrm{sid}, h_{\mathrm{sid}})$:
If $\mathrm{sid} \in \mathcal{I}_1$ then return $\perp$
$\ell \leftarrow \ell + 1$ ; $\mathcal{I}_1 \leftarrow \mathcal{I}_1 \cup \{\mathrm{sid}\}$
$Z_{\mathrm{sid}} \leftarrow h_{\mathrm{sid}}^x$
$z_{1,\mathrm{sid}}, e_{\mathrm{sid}}, r_{0,\mathrm{sid}} \leftarrow\!\!\$\ \mathbb{Z}_p$
$R_{g,\mathrm{sid}} \leftarrow g^{r_{0,\mathrm{sid}}}$ ; $R_{h,\mathrm{sid}} \leftarrow h^{r_{0,\mathrm{sid}}}$
$A_{\mathrm{sid}} \leftarrow g^{z_{1,\mathrm{sid}}} W^{-e_{\mathrm{sid}}}$ // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_3^{\mathcal{A}}$
$z_{0,\mathrm{sid}}, d_{\mathrm{sid}}, r_{1,\mathrm{sid}} \leftarrow\!\!\$\ \mathbb{Z}_p$
$R_{g,\mathrm{sid}} \leftarrow g^{z_{0,\mathrm{sid}}} X^{-d}$
$R_{h,\mathrm{sid}} \leftarrow h^{z_{0,\mathrm{sid}}} Z^{-d}$
$A_{\mathrm{sid}} \leftarrow g^{r_{1,\mathrm{sid}}}$    // $\mathbf{G}_4^{\mathcal{A}}$
$s \leftarrow\!\!\$\ \mathbb{Z}_p$ ; $\delta \leftarrow \mathsf{H}''(h, X, Z, g^s, h^s)$
$\pi \leftarrow (\delta, \delta \cdot x + s)$    // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_2^{\mathcal{A}}$
$\delta, s' \leftarrow\!\!\$\ \mathbb{Z}_p, \pi \leftarrow (\delta, s')$          // $\mathbf{G}_3^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
If $\mathsf{H}''(h_{\mathrm{sid}}, X, Z_{\mathrm{sid}}, g^{s'} X^{-\delta}, h_{\mathrm{sid}}^s Z_{\mathrm{sid}}^{-\delta}) \neq \perp$
  then **abort game**
$\mathsf{H}''(h_{\mathrm{sid}}, X, Z_{\mathrm{sid}}, g^{s'} X^{-\delta}, h_{\mathrm{sid}}^s Z_{\mathrm{sid}}^{-\delta}) \leftarrow \delta$
Return $(Z_{\mathrm{sid}}, \pi, R_{g,\mathrm{sid}}, R_{h,\mathrm{sid}}, A_{\mathrm{sid}})$

Oracle $\mathrm{S}_2(\mathrm{sid}, c_{\mathrm{sid}})$ :
If $\mathrm{sid} \notin \mathcal{I}_1$ or
  $\mathrm{sid} \in \mathcal{I}_2$ then return $\perp$
$\mathcal{I}_2 \leftarrow \mathcal{I}_2 \cup \{\mathrm{sid}\}$
$d_{\mathrm{sid}} \leftarrow c_{\mathrm{sid}} - e_{\mathrm{sid}}$
$z_{0,\mathrm{sid}} \leftarrow r_{0,\mathrm{sid}} + d_{\mathrm{sid}} \cdot x$                // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_3^{\mathcal{A}}$
$e_{\mathrm{sid}} \leftarrow c_{\mathrm{sid}} - d_{\mathrm{sid}}$
$z_{1,\mathrm{sid}} \leftarrow r_{1,\mathrm{sid}} + e_{\mathrm{sid}} \cdot w$                // $\mathbf{G}_4^{\mathcal{A}}$
Return $(d_{\mathrm{sid}}, e_{\mathrm{sid}}, z_{0,\mathrm{sid}}, z_{1,\mathrm{sid}})$
```

Fig. 5. The OMUF $= \mathbf{G}_0^{\mathcal{A}}$ security game for BS_1 and the subsequent games $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$. We assume that the adversary $\mathcal{A}$ makes ℓ queries to the signing oracle S_1. We remark that $\mathsf{H}, \mathsf{H}', \mathsf{H}''$ are modeled as random oracles and $\mathcal{A}$ has access to them. Each box type containing the game name indicates the changes made in that game and to make things clearer, for each box, we indicate which game contains the box by a comment by the side of it. Also, we omitted the signer state and instead use variable names with subscript sid to indicate the corresponding values in the signer state.

adversary $\mathcal{A}$ succeeds if for all $k_1 \neq k_2, m_{k_1}^* \neq m_{k_2}^*$ and $\mathsf{BS}_1.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m_k^*, \sigma_k^*) = 1$ for all $k \in [\ell + 1]$. We additionally assume w.l.o.g. that $\mathcal{A}$ does not make the same random oracle query twice and it already makes the queries to H and H' that would be called in $\mathsf{BS}_1.\mathsf{Ver}$ when the game checks the forgeries. The probability of $\mathcal{A}$ winning in game $\mathbf{G}_0^{\mathcal{A}}$ is exactly its advantage in the game OMUF, i.e.,

$$\mathsf{Adv}_{\mathsf{BS}_1}^{\mathsf{omuf}}(\mathcal{A}, \lambda) = \Pr[\mathbf{G}_0^{\mathcal{A}} = 1] .$$

Game $\mathbf{G}_1^{\mathcal{A}}$: This game is identical to $\mathbf{G}_0^{\mathcal{A}}$ except that the game adds another winning condition for $\mathcal{A}$ where for all forgeries (m_k^*, σ_k^*) for $k \in [\ell + 1]$, after parsing $\sigma_k^* = (Z_k^*, d_k^*, e_k^*, z_{k,0}^*, z_{k,1}^*)$, the game requires that $Z_k^* = \mathsf{H}(m_k^*)^{\mathsf{sk}}$.

To argue the advantage change, we refer to Lemma 3.5, stating that there exists an adversary $\mathcal{B}$ playing DLOG game running in time $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$ such that

$$\Pr[\mathbf{G}_1^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_0^{\mathcal{A}} = 1] - (\ell + 1)\left(\sqrt{Q_{\mathsf{H}'} \mathsf{Adv}_{\mathsf{GGen}}^{\mathsf{dlog}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p}\right) .$$

Game $\mathbf{G}_2^{\mathcal{A}}$: This game is identical to $\mathbf{G}_1^{\mathcal{A}}$ except that when generating the component W in par, the game generates $w \leftarrow\!\!\$\ \mathbb{Z}_p$ and sets $W \leftarrow g^w$. Since W still has the same distribution, the success probability of $\mathcal{A}$ is exactly as in $\mathbf{G}_1^{\mathcal{A}}$.

$$\Pr[\mathbf{G}_2^{\mathcal{A}} = 1] = \Pr[\mathbf{G}_1^{\mathcal{A}} = 1] .$$

Game $\mathbf{G}_3^{\mathcal{A}}$: This game is identical to $\mathbf{G}_2^{\mathcal{A}}$ except that the signing oracle S_1 generates the proof π as follows: it samples $s', \delta \leftarrow\$\, \mathbb{Z}_p$ and programs $\mathsf{H}''(h, X, Z, g^{s'}X^{-\delta}, h^{s'}Z^{-\delta})$ as δ. If H'' is already defined at $(h, X, Z, g^{s'}X^{-\delta}, h^{s'}Z^{-\delta})$, the game aborts.

The view of $\mathcal{A}$ is identical to $\mathbf{G}_2^{\mathcal{A}}$ if the game does not abort. Moreover, the game only aborts if $(h, X, Z, g^{s'}X^{-\delta}, h^{s'}Z^{-\delta})$ has been queried or programmed beforehand, but $g^{s'}X^{-\delta}$ and $h^{s'}Z^{-\delta}$ are uniformly random and independent of the view of $\mathcal{A}$ and previous programming attempts of H'' as s' is uniformly random and independent at the time that the oracle tries to program H''. Thus, by applying the union bound over all pairs of queries to oracle S_1 and queries to both H'' and S_1 (accounting for attempts to program H''),

$$\Pr[\mathbf{G}_3^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_2^{\mathcal{A}} = 1] - \frac{\ell(\ell + Q_{\mathsf{H}''})}{p} \ .$$

Game $\mathbf{G}_4^{\mathcal{A}}$: This game is identical to $\mathbf{G}_3^{\mathcal{A}}$ except that the signing oracles are modified to use w instead of x to generate the signature. More specifically, $A, R_g, R_h, d, e, z_0, z_1$ are now generated as follows:

1. $r_1, d, z_0 \leftarrow\$\, \mathbb{Z}_p$.
2. $A \leftarrow g^{r_1}, (R_g, R_h) \leftarrow (g^{z_0}X^{-d}, h^{z_0}Z^{-d})$.
3. After receiving c, set $e \leftarrow c - d$ and $z_1 \leftarrow e \cdot w + r_1$.

Since the joint distributions of $(A, R_g, R_h, d, e, z_0, z_1)$ in the games $\mathbf{G}_3^{\mathcal{A}}$ and $\mathbf{G}_4^{\mathcal{A}}$ are identical, the view of $\mathcal{A}$ remains the same. Thus,

$$\Pr[\mathbf{G}_4^{\mathcal{A}} = 1] = \Pr[\mathbf{G}_3^{\mathcal{A}} = 1] \ .$$

Lastly, we give a reduction $\mathcal{B}'$ playing the CT-CDH game using the adversary $\mathcal{A}$ as a subroutine. The reduction $\mathcal{B}'$ is defined as follows:

1. The reduction $\mathcal{B}'$ takes as input a CT-CDH instance $(\mathbb{G}, p, g, X)$. It samples $w \leftarrow\$\, \mathbb{Z}_p$ and sets $W \leftarrow g^w$. Lastly, it sends $\mathsf{par} = (\mathbb{G}, p, g, W), \mathsf{pk} = X$ to $\mathcal{A}$.
2. The simulation of $\mathsf{H}', \mathsf{H}''$ are done as in $\mathbf{G}_4^{\mathcal{A}}$. However, for queries to H (labeling each with $1 \leqslant j \leqslant Q_{\mathsf{H}}$), the reduction $\mathcal{B}'$ queries the challenge oracle $\textsc{Chal}$ and receives a random group element Y_j which it returns as the random oracle output. (This means that $\mathcal{B}'$ makes Q_{H} queries to $\textsc{Chal}$.)
3. The signing oracles are also simulated as in $\mathbf{G}_4^{\mathcal{A}}$ except for the computation of $Z = h^x$ in S_1 which is done by querying its $\textsc{Dh}$ oracle, i.e., $Z \leftarrow \textsc{Dh}(h)$.
4. After receiving the forgery $(m_k^*, \sigma_k^*)_{k \in [\ell+1]}$ from $\mathcal{A}$, $\mathcal{B}'$ checks if all the messages are distinct and all the pairs are valid. If not, it aborts. Next, $\mathcal{B}'$ identifies j_k for $k \in [\ell + 1]$ where j_k is the index of the hash query $\mathsf{H}(m_k^*)$ made by $\mathcal{A}$. Since m_k^* are distinct, there are exactly $\ell + 1$ distinct j_k. Lastly, $\mathcal{B}'$ returns $(j_k, Z_k^*)_{k \in [\ell+1]}$ where Z_k^* is the corresponding value in σ_k^*.

It is clear that the running time of $\mathcal{B}'$ is about that of $\mathcal{A}$. For the success probability of the reduction, we can see that $\mathcal{B}'$ simulates the oracles identically to the game $\mathbf{G}_4^{\mathcal{A}}$. Then, if $\mathcal{A}$ wins in game $\mathbf{G}_4^{\mathcal{A}}$, then $\mathcal{A}$ returns $Z_k^* = \mathsf{H}(m_k^*)^{\mathsf{sk}} = Y_{j_k}^x$ for all $k \in [\ell + 1]$ where $x = \log_g X$. Thus, $\mathcal{B}'$ wins the game CT-CDH, i.e., returning $\ell + 1$ correct CT-CDH solutions while only querying the oracle $\textsc{Dh}$ for ℓ times. Therefore, $\Pr[\mathbf{G}_4^{\mathcal{A}} = 1] \leqslant \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{ct-cdh}}(\mathcal{B}', \lambda)$. Then, by combining all the advantage changes,

$$\begin{aligned}\mathsf{Adv}_{\mathsf{BS}_1}^{\mathrm{omuf}}(\mathcal{A}, \lambda) \leqslant \frac{\ell(\ell + Q_{\mathsf{H}''})}{p} + (\ell + 1)\left(\sqrt{Q_{\mathsf{H}'}\mathsf{Adv}_{\mathsf{GGen}}^{\mathsf{dlog}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p}\right) \\ + \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{ct-cdh}}(\mathcal{B}', \lambda) \ .\square\end{aligned}$$

Lemma 3.5. *There exists an adversary $\mathcal{B}$ playing* DLOG *game such that its running time is $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$ and*

$$\Pr[\mathbf{G}_1^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_0^{\mathcal{A}} = 1] - (\ell + 1)\left(\sqrt{Q_{\mathsf{H}'}\mathsf{Adv}_{\mathsf{GGen}}^{\mathsf{dlog}}(\mathcal{B}, \lambda)} + Q_{\mathsf{H}'}/p\right) \ .$$

Proof. Let Bad be the event where $\mathbf{G}_0^{\mathcal{A}}$ outputs 1 but $\mathbf{G}_1^{\mathcal{A}}$ outputs 0. That is the following event: $\mathcal{A}$ outputs $\ell+1$ message-signature pairs (m_k^*, σ_k^*) for $k \in [\ell+1]$ such that (1) for all $k_1 \neq k_2, m_{k_1}^* \neq m_{k_2}^*$, (2) for all $k \in [\ell+1]$, $\mathsf{BS}_1.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m_k^*, \sigma_k^*) = 1$, and (3) *there exists some* $k \in [\ell+1]$ *where parsing the signature* $\sigma_k^* = (Z_k^*, d_k^*, e_k^*, z_{k,0}^*, z_{k,1}^*)$ *we have that* $Z_k^* \neq \mathsf{H}(m_k^*)^{\mathsf{sk}}$. Then, we can write $\mathsf{Pr}[\mathbf{G}_1^{\mathcal{A}} = 1] \geqslant \mathsf{Pr}[\mathbf{G}_0^{\mathcal{A}} = 1] - \mathsf{Pr}[\mathsf{Bad}]$.

Also, define the event Bad_k for $k \in [\ell+1]$ which is event Bad with the condition (3) specified only for the k-th pair (m_k^*, σ_k^*). This gives $\mathsf{Bad} = \bigcup_{k=1}^{\ell+1} \mathsf{Bad}_k$.

Now, define a deterministic wrapper $\mathcal{A}_k$ over the adversary $\mathcal{A}$ where $\mathcal{A}_k$ receives the following inputs: instance $(\mathbb{G}, p, g, W)$, outputs $(c_1, \ldots, c_{Q_{\mathsf{H}'}})$ of H', and a random tape ρ.

1. Extract $(x, (s_i, r_{0,i}, e_i, z_{1,i})_{i\in[\ell]}, (h_i)_{i\in[Q_{\mathsf{H}}]}, (\delta_i)_{i\in Q_{\mathsf{H}''}}, \rho')$ from the random tape ρ where $x, s_i, r_{0,i}, e_i, z_{1,i}, \delta_i \in \mathbb{Z}_p$ and $h_i \in \mathbb{G}$.
2. Set $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W), \mathsf{pk} \leftarrow g^x, \mathsf{sk} \leftarrow x$.
3. Run $(m_k^*, \sigma_k^*)_{k\in[\ell+1]} \leftarrow \mathcal{A}^{\mathrm{S}_1, \mathrm{S}_2, \mathsf{H}, \mathsf{H}', \mathsf{H}''}(\mathsf{par}, \mathsf{pk}; \rho')$ where each oracle is answered as follows:
   - For the i-th query ($i \in [\ell]$) to S_1 use $x, (s_i, r_{0,i}, e_i, z_{1,i})$ to answer the query as in $\mathsf{BS}_1.\mathsf{S}_1$. For the query to S_2 of the same session id, also use $x, (s_i, r_{0,i}, e_i, z_{1,i})$ as in $\mathsf{BS}_1.\mathsf{S}_2$.
   - For the i-th query ($i \in [Q_{\mathsf{H}}]$) to H, answer with h_i.
   - For the i-th query ($i \in [Q_{\mathsf{H}'}]$) to H', answer with c_i.
   - For the i-th query ($i \in [Q_{\mathsf{H}''}]$) to H'', answer with δ_i. (In these queries, we w.l.o.g. accounted for the queries that the wrapper made to generate π. Moreover, when the same queries are queried more than once, the oracle will answer with the first value initialized.)
4. If event Bad_k does not occur, return $(\perp, \perp)$. Otherwise, return $(I, (m_k^*, \sigma_k^*))$ where I is the index of the query to H' from $\mathcal{A}$ that corresponds to the verification of (m_k^*, σ_k^*). More specifically, after parsing $\sigma_k^* = (Z^*, d^*, e^*, z_0^*, z_1^*)$, I is the index corresponding to the query (m, h, Z, R_g, R_h, A) where $m = m_k^*$, $h = \mathsf{H}(m_k^*), Z = Z^*, R_g = g^{z_0^*} X^{-d^*}, R_h = h^{z_0^*} Z^{-d^*}, A = g^{z_1^*} W^{-e^*}$. Note that I is well-defined as we assume that all random oracle queries in forgery verification are made by $\mathcal{A}$ beforehand. Also, it is easy to see that the running time of $\mathcal{A}_k$ is roughly the running time of $\mathcal{A}$.

Next, we consider the following reduction $\mathcal{B}$ playing the discrete logarithm game defined as follows:

1. On the input $(\mathbb{G}, p, g, W)$, $\mathcal{B}$ samples $c_1, \ldots, c_{Q_{\mathsf{H}'}} \leftarrow\!\!{\scriptstyle\$}\ \mathbb{Z}_p$ along with random coins ρ for $\mathcal{A}_k$.
2. Run $(I, (m, \sigma)) \leftarrow\!\!{\scriptstyle\$}\ \mathcal{A}_k((\mathbb{G}, p, g, W), (c_1, \ldots, c_{Q_{\mathsf{H}'}}); \rho)$.
3. If $I = \perp$, abort. If not, sample $c'_I, \ldots, c'_{Q'_{\mathsf{H}}} \leftarrow\!\!{\scriptstyle\$}\ \mathbb{Z}_p$ and
   run $(I', (m', \sigma')) \leftarrow\!\!{\scriptstyle\$}\ \mathcal{A}_k((\mathbb{G}, p, g, W), (c_1, \ldots, c_{I-1}, c'_I, \ldots, c'_{Q_{\mathsf{H}'}}); \rho)$.
4. If $I = I'$ and $c'_I \neq c_I$, parse $\sigma = (Z, d, e, z_0, z_1), \sigma' = (Z', d', e', z'_0, z'_1)$, and return $(z_1 - z'_1)(e - e')^{-1}$. Otherwise, abort.

Since $\mathcal{B}$ runs $\mathcal{A}_k$ twice and the running time of $\mathcal{A}_k$ is about that of $\mathcal{A}$, $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$. Then, we argue the success probability of the reduction. More precisely, we show that if $\mathcal{B}$ does not abort (i.e., $I = I' \neq \perp$ and $c_I \neq c'_I$), then it returns a discrete logarithm of W. Since $I = I' \neq \perp$, the signatures σ, σ' are: (a) valid signatures corresponding to the I-th query from $\mathcal{A}$ to H' and (b) satisfying $Z \neq \mathsf{H}(m)^{\mathsf{sk}} = h^x$ and $Z' \neq \mathsf{H}(m')^{\mathsf{sk}} = h'^x$. By (a), we know the following

(i) $m = m', \mathsf{H}(m) = h = h' = \mathsf{H}(m'), Z = Z'$.
(ii) $c_I = d + e, c'_I = d' + e'$.
(iii) $g^{z_0} X^{-d} = g^{z'_0} X^{-d'}, h^{z_0} Z^{-d} = h^{z'_0} Z^{-d'}$.
(iv) $A = g^{z_1} W^{-e} = g^{z'_1} W^{-e'}$.

We will argue that $d = d'$. The equations in (iii) give the following equation

$$Z^{d-d'} = h^{z_0 - z'_0} = g^{(z_0 - z'_0)\log_g h} = X^{(d-d')\log_g h} = h^{x(d-d')} \,.$$

Since $Z \neq h^x$, only $d = d'$ satisfies the equation. Since $d + e = c_I \neq c'_I = d' + e'$, we have $e \neq e'$. Thus, the returned value $(z_1 - z'_1)(e - e')^{-1} = \log_g W$ by (iv). Hence,

$$\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda) = \mathsf{Pr}[\mathcal{B} \text{ does not abort}] = \mathsf{Pr}[I = I' \wedge I \neq \perp \wedge c_I \neq c'_I] \,.$$

Lastly, by the fact that $\mathcal{B}$ rewinds $\mathcal{A}_k$ which only outputs $I \neq \perp$ when Bad_k occurs, we can apply the forking lemma (Lemma 2.1),

$$\Pr[\mathsf{Bad}_k] \leqslant \sqrt{Q_{\mathsf{H}'}\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p} \ .$$

The lemma statement follows from the union bound over Bad_k for $k \in [\ell + 1]$. □

4 Strong Unforgeability from CT-CDH

It turns out that the scheme BS_1 from the prior section is not one-more *strongly* unforgeable. We omit a formal proof, but the basic idea is to consider an adversary attempting to produce $\ell + 1$ signatures on the *same* message m by starting ℓ signing sessions with $h = \mathsf{H}(m)$, fixing h and $Z = h^{\mathsf{sk}}$ in all of them. After this, the structure of the signing protocol becomes essentially equivalent to that of the Abe-Okamoto blind signature [AO00], which is subject to a variant of ROS attacks [BLL$^+$21].

To obtain a *strongly* unforgeable scheme, we modify BS_1 by adding a first move where the signer sends the nonces R_g and A (note that these do not depend on h in BS_1), and the user then sets $h \leftarrow \mathsf{H}(m, R_g, A)$ instead of $\mathsf{H}(m)$ as in BS_1. The resulting five-move scheme BS_2 is described in Figure 6, and we will show it indeed satisfies OMSUF under the CT-CDH assumption. This scheme can be seen as a blind version of Chevallier-Mames signatures [Che05, KLP17]. It is easy to show that the scheme satisfies correctness (see Section 4.1 for a proof).

Blindness. As with BS_1, the scheme can be shown *computationally* blind under the DL assumption, without any further assumption on the hash functions used by the scheme, or *statistically* blind by modeling H'' as a random oracle, once again using the highlighted NIZK proof. We only state a theorem for the latter property, and give both proofs for statistical and computational blindness in Sections 4.2 and 4.3, respectively.

Theorem 4.1 (Blindness of BS_2). *Assume that* GGen *outputs the description of a group of prime order* $p = p(\lambda)$, *and let* $\mathsf{BS}_2 = \mathsf{BS}_2[\mathsf{GGen}]$. *For any adversary* $\mathcal{A}$ *for game* BLIND *making at most* $Q_{\mathsf{H}''} = Q_{\mathsf{H}''}(\lambda)$ *queries to* H'', *modeled as a random oracle, we have*

$$\mathsf{Adv}^{\mathrm{blind}}_{\mathsf{BS}_2}(\mathcal{A}, \lambda) \leqslant \frac{2Q_{\mathsf{H}''}}{p} \ .$$

One-more unforgeability. The next theorem establishes one-more unforgeability of BS_2 in the random oracle model under the CT-CDH assumption. We give a proof sketch below, whereas the full proof can be found in Section 4.4.

Theorem 4.2 (OMUF of BS_2). *Assume that* GGen *outputs the description of a group of prime order* $p = p(\lambda)$, *and let* $\mathsf{BS}_2 = \mathsf{BS}_2[\mathsf{GGen}]$. *For any adversary* $\mathcal{A}$ *for game* OMSUF *with running time* $t_{\mathcal{A}} = t_{\mathcal{A}}(\lambda)$, *making at most* $\ell = \ell(\lambda)$ *queries to* S_1, $Q_{\mathsf{H}_\star} = Q_{\mathsf{H}_\star}(\lambda)$ *queries to* $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}', \mathsf{H}''\}$, *modeled as random oracles, there exist adversaries* $\mathcal{B}, \mathcal{B}_2$ *for game* DLOG, *and* $\mathcal{B}', \mathcal{B}_3$ *for game* CT-CDH, *such that*

$$\mathsf{Adv}^{\mathrm{omsuf}}_{\mathsf{BS}_2}(\mathcal{A}, \lambda) \leqslant \frac{\ell(\ell + Q_{\mathsf{H}''})}{p} + (\ell + 1)\left(\sqrt{Q_{\mathsf{H}'}\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p}\right) + \mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}_2, \lambda) + \mathsf{Adv}^{\mathrm{ct-cdh}}_{\mathsf{GGen}}(\mathcal{B}_3, \lambda) + \mathsf{Adv}^{\mathrm{ct-cdh}}_{\mathsf{GGen}}(\mathcal{B}', \lambda) \ .$$

Furthermore, $\mathcal{B}$ *and* $\mathcal{B}_2$ *run in time* $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$ *and* $t_{\mathcal{B}_2} \approx t_{\mathcal{A}}$ *respectively, whereas* $\mathcal{B}'$ *runs in time* $t_{\mathcal{B}'} \approx t_{\mathcal{A}}$, *makes* Q_{H} *queries to* Chal, *and* ℓ *queries to* Dh, *and, lastly,* $\mathcal{B}_3$ *runs in time* $t_{\mathcal{B}_3} \approx t_{\mathcal{A}}$, *makes and* $\ell + 1$ *queries to* Chal, *and* ℓ *queries to* Dh.

The proof below builds on top of the approach for proving OMUF of BS_1. Specifically, we show that after starting ℓ signing sessions, no adversary can forge $\ell + 1$ valid message-signature pairs $\{(m_i, (Z_i, d_i, e_i, z_{0,i}, z_{1,i}))\}$ with *distinct* $(m_i, R_{g,i}, A_i)$, where $R_{g,i} = g^{z_{0,i}}\mathsf{pk}^{-d_i}$ and $A_i = g^{z_{1,i}}W^{-e_i}$. To see this implies that BS_2 is OMSUF, we only need to show that no adversary can output two *distinct* pairs $(m_i, (Z_i, d_i, e_i, z_{0,i}, z_{1,i}))$ and

```
Algorithm $\mathsf{BS}_2.\mathsf{Setup}(1^\lambda)$ :
$(\mathbb{G}, p, g) \leftarrow\$ \mathsf{GGen}(1^\lambda)$ ; $W \leftarrow\$ \mathbb{G}$
Select $\mathsf{H} : \{0,1\}^* \to \mathbb{G}$
Select $\mathsf{H}', \mathsf{H}'' : \{0,1\}^* \to \mathbb{Z}_p$
Return $\mathsf{par} = (\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \mathsf{H}'')$

Algorithm $\mathsf{BS}_2.\mathsf{KG}(\mathsf{par})$ :
$(\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \mathsf{H}'') \leftarrow \mathsf{par}$
$x \leftarrow\$ \mathbb{Z}_p$ ; $X \leftarrow g^x$ ; $\mathsf{sk} \leftarrow x$ ; $\mathsf{pk} \leftarrow X$
Return $(\mathsf{sk}, \mathsf{pk})$

Algorithm $\mathsf{BS}_2.\mathsf{S}_1(\mathsf{par}, \mathsf{sk})$ :
$x \leftarrow \mathsf{sk}$ ; $(\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \mathsf{H}'') \leftarrow \mathsf{par}$
$z_1, e, r_0 \leftarrow\$ \mathbb{Z}_p$
$R_g \leftarrow g^{r_0}$ ; $A \leftarrow g^{z_1} W^{-e}$
$\mathsf{st}_1^s \leftarrow (x, X, z_1, e, r_0)$ ; $\mathsf{smsg}_1 \leftarrow (R_g, A)$
Return $(\mathsf{st}_1^s, \mathsf{smsg}_1)$

Algorithm $\mathsf{BS}_2.\mathsf{S}_2(\mathsf{st}_1^s, \mathsf{umsg}_1)$ :
$(x, z_1, e, r_0) \leftarrow \mathsf{st}_1^s$ ; $h \leftarrow \mathsf{umsg}_1$
$Z \leftarrow h^x$ ; $R_h \leftarrow h^{r_0}$
$\boxed{s \leftarrow\$ \mathbb{Z}_p \ ; \ \delta \leftarrow \mathsf{H}''(h, g^x, Z, g^s, h^s)}$
$\boxed{\pi \leftarrow (\delta, \delta \cdot x + s)}$
$\mathsf{st}_2^s \leftarrow (x, z_1, e, r_0)$ ; $\mathsf{smsg}_2 \leftarrow (Z, \boxed{\pi}, R_h)$
Return $(\mathsf{st}_2^s, \mathsf{smsg}_1)$

Algorithm $\mathsf{BS}_2.\mathsf{S}_3(\mathsf{st}_2^s, \mathsf{umsg}_2)$ :
$(x, z_1, e, r_0) \leftarrow \mathsf{st}_2^s$
$d \leftarrow c - e$ ; $z_0 \leftarrow r_0 + d \cdot x$
Return $\mathsf{smsg}_2 \leftarrow (d, e, z_0, z_1)$

Algorithm $\mathsf{BS}_2.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m, \sigma)$ :
$(\mathbb{G}, p, g, W) \leftarrow \mathsf{par}$ ; $(Z, d, e, z_0, z_1) \leftarrow \sigma$
$X \leftarrow \mathsf{pk}$ ; $A \leftarrow g^{z_1} W^{-e}$ ; $R_g \leftarrow g^{z_0} X^{-d}$
$h \leftarrow \mathsf{H}(m, R_g, A)$ ; $R_h \leftarrow h^{z_0} Z^{-d}$
If $e + d \neq \mathsf{H}'(m, h, Z, R_g, R_h, A)$ then
    return 0
Return 1
```

```
Algorithm $\mathsf{BS}_2.\mathsf{U}_1(\mathsf{par}, \mathsf{pk}, m, \mathsf{smsg}_1)$ :
$X \leftarrow \mathsf{pk}$ ; $(\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \mathsf{H}'') \leftarrow \mathsf{par}$
$(R_g, A) \leftarrow \mathsf{smsg}_1$
$\alpha_0, \alpha_1, \gamma_0, \gamma_1 \leftarrow\$ \mathbb{Z}_p$
$R'_g \leftarrow R_g X^{-\gamma_0} g^{\alpha_0}$ ; $A' \leftarrow A W^{-\gamma_1} g^{\alpha_1}$
$h' \leftarrow \mathsf{H}(m, R'_g, A')$
$\beta \leftarrow\$ \mathbb{Z}_p$ ; $h \leftarrow h' g^\beta$
$\mathsf{st}_1^u \leftarrow (m, \beta, \alpha_0, \alpha_1, \gamma_0, \gamma_1, X, h', h, W, R_g, R'_g, A, A')$
$\mathsf{umsg}_1 \leftarrow h$
Return $(\mathsf{st}_1^u, \mathsf{umsg}_1)$

Algorithm $\mathsf{BS}_2.\mathsf{U}_2(\mathsf{st}_1^u, \mathsf{smsg}_2)$ :
$(m, \beta, \alpha_0, \alpha_1, \gamma_0, \gamma_1, X, h', h, W, R_g, R'_g, A, A') \leftarrow \mathsf{st}_1^u$
$(Z, \boxed{\pi}, R_h) \leftarrow \mathsf{smsg}_2$; $\boxed{(\delta, s') \leftarrow \pi}$
$\boxed{\text{If } \delta \neq \mathsf{H}''(h, X, Z, g^{s'} X^{-\delta}, h^{s'} Z^{-\delta}) \text{ then}}$
$\boxed{\quad \text{return } \perp}$
$Z' \leftarrow Z X^{-\beta}$
$R'_h \leftarrow R_h R_g^{-\beta} Z'^{-\gamma_0} h'^{\alpha_0}$
$c' \leftarrow \mathsf{H}'(m, h', Z', R'_g, R'_h, A')$
$c \leftarrow c' - \gamma_0 - \gamma_1$
$\mathsf{st}_2^u \leftarrow (c, Z, Z', R_h, \mathsf{st}_1^u)$ ; $\mathsf{umsg}_2 \leftarrow c$
Return $(\mathsf{st}_2^u, \mathsf{umsg}_2)$

Algorithm $\mathsf{BS}_2.\mathsf{U}_3(\mathsf{st}_2^u, \mathsf{smsg}_2)$ :
$(c, Z, Z', R_h, \mathsf{st}_1^u) \leftarrow \mathsf{st}_2^u$
$\mathsf{st}_1^u \leftarrow (m, \beta, \alpha_0, \alpha_1, \gamma_0, \gamma_1, X, h', h, W, R_g, R'_g, A, A')$
$(d, e, z_0, z_1) \leftarrow \mathsf{smsg}_2$
If $c \neq e + d$ or
    $(R_g X^d, R_h Z^d) \neq (g^{z_0}, h^{z_0})$ or
    $A W^e \neq g^{z_1}$ then return $\perp$
$d' \leftarrow d + \gamma_0$ ; $e' \leftarrow e + \gamma_1$
$z'_0 \leftarrow z_0 + \alpha_0$ ; $z'_1 \leftarrow z_1 + \alpha_1$
Return $\sigma \leftarrow (Z', d', e', z'_0, z'_1)$
```

Fig. 6. The blind signature scheme $\mathsf{BS}_2 = \mathsf{BS}_2[\mathsf{GGen}]$. The highlighted boxes denote the NIZK proof to show the equality of discrete log to the bases (g, h) of (X, Z). We also give a protocol-style description of BS_2 in Figure 17.

$(m_j, (Z_j, d_j, e_j, z_{0,j}, z_{1,j}))$ with $(m_i, R_{g,i}, A_i) = (m_j, R_{g,j}, A_j)$. Suppose such an adversary exists. Then, there are three cases: (1) $Z_i \neq Z_j$, (2) $(d_i, z_{0,i}) \neq (d_j, z_{0,j})$, and (3) $(e_i, z_{1,i}) \neq (e_j, z_{1,j})$. If $Z_i \neq Z_j$, one of Z_i and Z_j is not equal to $\mathsf{H}(m_i, R_{g,i}, A_i)^{\mathsf{sk}}$ and thus, we can follow the same argument as BS_1 to extract the discrete logarithm of W. If $(d_i, z_i) \neq (d_j, z_j)$, since $g^{z_i} \mathsf{pk}^{-d_i} = R_{g,i} = R_{g,j} = g^{z_j} \mathsf{pk}^{-d_j}$, we can extract sk. If $(e_i, t_i) \neq (e_j, t_j)$, since $g^{t_i} W^{-e_i} = A_i = A_j = g^{t_j} W^{-e_j}$, we can extract $\log_g W$. Therefore, such an adversary contradicts the discrete logarithm assumption.

Although we do not show this, we note that our scheme BS_2 satisfies a slightly stronger notion of one-more unforgeability where the adversary cannot output $\ell + 1$ forgeries with only ℓ queries access to the oracle S_2 no matter how many S_1 queries it made (instead of ℓ queries access to S_1 as we have stated above). This is immediate from the reduction to the CT-CDH assumption as the reduction only has to query its own DH oracle when the adversary queries S_2.

4.1 Correctness of BS_2

Theorem 4.3. BS_2 *satisfies correctness.*

Proof. Consider an honestly generated signature $\sigma = (Z', d', e', z'_0, z'_1)$ for a message m and the variables as defined in the signing protocol.

First, we argue that the checks in $\mathsf{BS}_2.\mathsf{U}_2$ and $\mathsf{BS}_2.\mathsf{U}_3$ verifies. For the check in $\mathsf{BS}_2.\mathsf{U}_2$, since $s' = \delta x + s$ and $X = g^x, Z = h^x$, we have $g^{s'} X^{-\delta} = g^s$ and $h^{s'} Z^{-\delta} = h^s$. Thus, $\mathsf{H}''(h, X, Z, g^{s'} X^{-\delta}, h^{s'} Z^{-\delta}) = \mathsf{H}''(h, X, Z, g^s, h^s) = \delta$.

For the check in $\mathsf{BS}_2.\mathsf{U}_3$, $c = e + d$ by how the signer computes e, $A \cdot W^{-e} = g^{z_1}$ by how A is generated, and lastly $(R_g \cdot X^d, R_h \cdot Z^d) = (g^{r_0+dx}, h^{r_0+dx}) = (g^{z_0}, h^{z_0})$, where the first equality follows from $R_g = g^{r_0}, R_h = h^{r_0}, X = g^x, Z = h^x$ and the second equality follows from $z_0 = dx + r_0$.

Now, to argue the validity of the signature, let $h' = \mathsf{H}(m, R'_g, A')$ where $R'_g = R_g X^{-\gamma_0} g^{\alpha_0}, A' = AW^{-\gamma_1} g^{\alpha_1}$. Then, we have to argue the following to say that the signature is valid:

1. $c' = e' + d'$. This follows from $c = e + d$ as $c' = c + \gamma_0 + \gamma_1 = e + d + \gamma_0 + \gamma_1 = e' + d'$.
2. $g^{z'_1} \cdot W^{-e'} = A'$. This follows from $z'_1 = z_1 + \alpha_1$ and $e' = e + \gamma_1$, as $g^{z'_1} \cdot W^{-e'} = (g^{z_1} W^{-e}) \cdot (W^{-\gamma_1} g^{\alpha_1}) = A \cdot (W^{-\gamma_1} g^{\alpha_1}) = A'$.
3. $g^{z'_0} \cdot X^{-d'} = R'_g$. This follows from $z'_0 = z_0 + \alpha_0$ and $d' = d + \gamma_0$, as

   $$g^{z'_0} \cdot X^{-d'} = (g^{z_0} X^{-d}) \cdot (X^{-\gamma_0} g^{\alpha_0}) = R_g \cdot (X^{-\gamma_0} g^{\alpha_0}) = R'_g \ ,$$

   where the second equality follows from the check $R_g \cdot X^d = g^{z_0}$ in $\mathsf{BS}_2.\mathsf{U}_3$.
4. $h'^{z'_0} \cdot Z'^{-e'} = R'_h$. This follows from $z'_0 = z_0 + \alpha_0, d' = d + \gamma_0, h' = h \cdot g^{-\beta}$, and $Z' = Z \cdot X^{-\beta}$

   $$\begin{aligned} h'^{z'_0} \cdot Z'^{-d'} &= h'^{z_0} Z'^{-d} \cdot (h'^{\alpha_0} Z'^{-\gamma_0}) \\ &= h^{z_0} Z^{-d} \cdot (g^{z_0} X^{-d})^{-\beta} \cdot (h'^{\alpha_0} Z'^{-\gamma_0}) \\ &= R_h \cdot R_g^{-\beta} \cdot (h'^{\alpha_0} Z'^{-\gamma_0}) = R'_h \ , \end{aligned}$$

   where the second to last equality follows from the check $R_h \cdot Z^d = g^{z_0}$ and $R_g \cdot X^d = g^{z_0}$ in $\mathsf{BS}_2.\mathsf{U}_3$.

By the points above, we have $\mathsf{H}(m, g^{z'_0} X^{-d'}, g^{z'_1} W^{-e'}) = \mathsf{H}(m, R'_g, A') = h'$

$$\begin{aligned} \mathsf{H}'(m, h', Z', g^{z'_0} X^{-d'}, h'^{z'_0} Z'^{-d'}, g^{z'_1} W^{-e'}) &= \mathsf{H}'(m, h', Z', R'_g, R'_h, A') \\ &= c' = e' + d' \ , \end{aligned}$$

proving the scheme's correctness. □

4.2 Proof of Theorem 4.1

To prove blindness, we consider the following sequence of games.

Game $\mathbf{G}_0^{\mathcal{A}}$: This game is the BLIND game of BS_2 where $\mathcal{A}$ has $Q_{\mathsf{H}''}$ queries access to the random oracle H''. We also assume w.l.o.g. that any query to H'' which the user oracle has to make has already been made by $\mathcal{A}$. In particular, the query to H'' to verify π is done by $\mathcal{A}$ before $\mathcal{A}$ sends π.

Game $\mathbf{G}_1^{\mathcal{A}}$: This game made the following changes:

- In the oracle $\textsc{Init}(\widetilde{\mathsf{pk}}, \tilde{m}_0, \tilde{m}_1)$, the oracle additionally parses $X \leftarrow \mathsf{pk}$ and sets $\mathsf{sk} \leftarrow x = \log_g X$ found by exhaustive search.
- When the oracle $\mathrm{U}_2(i, \mathsf{smsg}_2^{(i)})$ receives $\mathsf{smsg}_2^{(i)}$ from $\mathcal{A}$, it parses $(Z_i, \pi_i = (\delta_i, s_i), R_{h,i}) \leftarrow \mathsf{smsg}_2^{(i)}$. Then, it computes $S_{g,i} = g^{s_i} X^{-\delta_i}$, $S_{h,i} = h_i^{s_i} Z_i^{-\delta_i}$, and checks if $\delta_i \neq \mathsf{H}''(h_i, X, Z_i, S_{g,i}, S_{h,i})$. If this check passes, the game now aborts if $Z_i \neq (h_i)^{\mathsf{sk}}$ where h_i is the message replied by the first query to $\mathrm{U}_1(i, \cdot)$.

The winning probability of $\mathcal{A}$ only changes when the new abort occurs in either signing sessions which corresponds to the following event:

$$Z_i \neq (h_i)^{\mathsf{sk}} \ \wedge \ \delta_i = \mathsf{H}''(h_i, X, Z_i, S_{g,i}, S_{h,i}).$$

The soundness of the NIZK proof implies that the event occurs with negligible probability. More precisely, with how $S_{g,i}, S_{h,i}$ is defined and that $Z_i \neq h_i^{\mathsf{sk}} = X^{\log_g h_i}$, we have $(S_{g,i})^{-\log_g h_i} S_{h,i} = h_i^{-s_i} h_i^{\delta_i \mathsf{sk}} h_i^{s_i} Z_i^{-\delta_i} = \left(h_i^{-\mathsf{sk}} Z_i\right)^{-\delta_i}$. Since $h_i^{-\mathsf{sk}} Z_i \neq 1_{\mathbb{G}}$, there is only one value of $\delta_i \in \mathbb{Z}_p$ to satisfy such equation. Since δ_i is uniformly at random after fixing the query and $\mathcal{A}$ makes at most $Q_{\mathsf{H}''}$ queries to H'', with the union bound over the two signing sessions

$$\left|\Pr[\mathbf{G}_0^{\mathcal{A}} = 1] - \Pr[\mathbf{G}_1^{\mathcal{A}} = 1]\right| \leqslant \frac{2Q_{\mathsf{H}''}}{p} \ .$$

Lastly, we show that the transcript and returned signatures are distributed identically for both cases of $b = 0, 1$, which implies $\Pr[\mathbf{G}_1^{\mathcal{A}} = 1] = \frac{1}{2}$.

To show this, first, assume without loss of generality that we fix the randomness of $\mathcal{A}$ and that $\mathcal{A}$ only outputs messages in the transcript where neither the game nor the user oracles abort which means $\mathcal{A}$ receives valid signatures (σ_0, σ_1). (Note that if the user oracle aborts at some point, for each signing session, the adversary will only see h_i and c_i which are both blinded and uniformly random over $\mathbb{G}$ and $\mathbb{Z}_p$ respectively.)

Let View_A denote the set of all possible views of $\mathcal{A}$ that can occur in the game $\mathbf{G}_1^{\mathcal{A}}$. A view $\varDelta \in \mathrm{View}_A$ is of the form $\varDelta = (W, X, m_0, m_1, T_0, T_1, \sigma_0, \sigma_1)$ where for $i \in \{0, 1\}$, T_i denotes the transcript of the interaction between $\mathcal{A}$ and the user oracle in signing session i, and σ_i denotes the valid signature for message m_i. In particular, $T_i = (h_i, Z_i, R_{g,i}, R_{h,i}, A_i, c_i, d_i, e_i, z_{0,i}, z_{1,i})$ (note that we omitted π_i as it is distributed independently of (m_0, m_1) given (h_i, Z_i)), and $\sigma_i = (Z'_i, d'_i, e'_i, z'_{0,i}, z'_{1,i})$. We need to show that the distribution of the actual adversarial view, which we denote as v_A, is the same between $b = 0$ and $b = 1$. Since we fix the randomness of $\mathcal{A}$, v_A only depends on the randomness of the user algorithm which we denote $\eta = (\beta_i, \alpha_{0,i}, \alpha_{1,i}, \gamma_{0,i}, \gamma_{1,i})_{i \in \{0,1\}}$ and write $v_A(\eta)$ to make this explicit.

Since we assume $\mathcal{A}$ does not make the game abort, for the signatures $\sigma_{b_i} = (Z'_{b_i}, d'_{b_i}, e'_{b_i}, z'_{0,b_i}, z'_{1,b_i})$ in any transcript $\varDelta \in \mathrm{View}_A$, we have $Z'_{b_i} = {h'_{b_i}}^{\mathsf{sk}}$ where $h'_{b_i} = \mathsf{H}(m_{b_i}, X^{-d'_{b_i}} g^{z'_{0,b_i}}, W^{-e'_{b_i}} g^{z'_{1,b_i}})$. This is because of the abort introduced in $\mathbf{G}_1^{\mathcal{A}}$ that induces $Z_i = h_i^x$ leading to $Z'_{b_i} = Z_i X^{-\beta_i} = (h_i g^{-\beta_i})^{\mathsf{sk}} = {h'_{b_i}}^{\mathsf{sk}}$.

To show that the distribution of v_A is identical between $b = 0$ and $b = 1$, consider a view $\varDelta \in \mathrm{View}_A$. We now show that there exists a unique η such that $v_A(\eta) = \varDelta$, regardless of whether $b = 0$ or $b = 1$. More specifically, we claim that for both $b = 0$ and $b = 1$, $v_A(\eta) = \varDelta$ if and only if for $i \in \{0, 1\}$, η satisfies

$$\begin{aligned} &\beta_i = \log_g h_i - \log_g h'_{b_i} \\ &\alpha_{0,i} = z'_{0,b_i} - z_{0,i}, \ \ \alpha_{1,i} = z'_{1,b_i} - z_{1,i} \\ &\gamma_{0,i} = d'_{b_i} - d_i, \ \ \gamma_{1,i} = e'_{b_i} - e_i \ . \end{aligned} \tag{7}$$

For the "only if" direction, i.e., if $v_A(\eta) = \varDelta$, then η satisfies Equation (7), this is true by how the user algorithm of BS_2 is defined.

To show the "if" direction, suppose η satisfies Equation (7), we need to show that $v_A(\eta) = \varDelta$. Particularly, we have to show that the user messages from oracles $\mathrm{U}_1, \mathrm{U}_2$ and the signatures from oracle U_3 are $(h_0, h_1), (c_0, c_1)$, and (σ_0, σ_1) respectively.

Again, since we only consider non-aborting transcript $\varDelta$, we have the following guarantees for $i \in \{0, 1\}$:

$$Z_i = h_i^{\mathsf{sk}}, \ Z'_{b_i} = {h'_{b_i}}^{\mathsf{sk}}, \tag{8}$$

$$c_i = d_i + e_i, \ R_{g,i} X^{d_i} = g^{z_{0,i}}, \ R_{h,i} Z_i^{d_i} = h_i^{z_{1,i}}, \ A_i W^{e_i} = g^{z_{1,i}} \tag{9}$$

$$d'_{b_i} + e'_{b_i} = \mathsf{H}'(m_{b_i}, h'_{b_i}, Z'_{b_i}, X^{-d'_{b_i}} g^{z'_{0,b_i}}, {Z'_{b_i}}^{-d'_{b_i}} {h'_{b_i}}^{z'_{0,b_i}}, W^{-e'_{b_i}} g^{z'_{1,b_i}}) \ , \tag{10}$$

where Equation (8) follows from the discussion above, Equation (9) follows from the check in $\mathsf{BS}_2.\mathsf{U}_3$, and Equation (10) follows from the validity of the signatures.

First, we argue that h_i is the user message from $\mathrm{U}_1(i,\cdot)$ for $i \in \{0,1\}$. Since we assume the randomness of $\mathcal{A}$ is fixed, $\mathcal{A}$'s first message will be consistent with $R_{g,i}, A_i$. Consider the blinded values of $R_{g,i}, A_i$

$$\begin{aligned} R'_{g,i} &= R_{g,i} X^{-\gamma_{0,i}} g^{\alpha_{0,i}} \\ &= (X^{-d_i} g^{z_{0,i}}) X^{-\gamma_{0,i}} g^{\alpha_{0,i}}; \text{ By equation (9)} \\ &= X^{-d_i-\gamma_{0,i}} g^{z_{0,i}+\alpha_{0,i}} = X^{-d'_{b_i}} g^{z'_{0,b_i}}, \text{ By equation (7)} \\ A'_i &= AW^{-\gamma_{1,i}} g^{\alpha_{1,i}} \\ &= (W^{-e_i} g^{z_{1,i}}) W^{-\gamma_{1,i}} g^{\alpha_{1,i}}; \text{ By equation (9)} \\ &= W^{-e_i-\gamma_{1,i}} g^{z_{1,i}+\alpha_{1,i}} = W^{-e'_{b_i}} g^{z'_{1,b_i}}, \text{ By equation (7)} \end{aligned}$$

Then, by the value of β_i from Equation (7), the first user message is

$$\begin{aligned} \mathsf{H}(m_{b_i}, R'_{g,i}, A'_i) \cdot g^{\beta_i} &= \mathsf{H}(m_{b_i}, X^{-d'_{b_i}} g^{z'_{0,b_i}}, W^{-e'_{b_i}} g^{z'_{1,b_i}}) \cdot g^{\beta_i} \\ &= h'_{b_i} \cdot g^{\beta_i} = h_i \ , \end{aligned}$$

which is consistent with Δ. Thus, the next message from $\mathcal{A}$ will be $Z_i, R_{h,i}$ from the transcript Δ.

Next, we argue that the second user message from $\mathrm{U}_2(i,\cdot)$ will be c_i. To do this, we consider the blinded values of $Z_i, R_{h,i}$ (the blinded values of $R_{g,i}$ and A_i is already argued above).

$$\begin{aligned} Z_i X^{-\beta_i} &= h_i^{\mathsf{sk}} g^{-\beta_i \mathsf{sk}} = (h_i g^{-\beta_i})^{\mathsf{sk}} = {h'_{b_i}}^{\mathsf{sk}} = Z'_{b_i}, \text{ Last equality by equation (8)} \\ R'_{h,i} &= R_{h,i} R_{g,i}^{-\beta_i} {Z'_{b_i}}^{-\gamma_{0,i}} {h'_i}^{\alpha_{0,i}} \\ &= \left(Z^{-d_i} h_i^{z_{0,i}}\right) \left(X^{-d_i} g^{z_{0,i}}\right)^{-\beta_i} {Z'_{b_i}}^{-\gamma_{0,i}} {h'_{b_i}}^{\alpha_{0,i}}; \text{ By equation (9)} \\ &= \left(ZX^{-\beta_i}\right)^{-d_i} \left(h_i g^{-\beta_i}\right)^{z_{0,i}} {Z'_{b_i}}^{-\gamma_{0,i}} {h'_{b_i}}^{\alpha_{0,i}} \\ &= {Z'_{b_i}}^{-d_i-\gamma_{0,i}} {h'_{b_i}}^{z_{0,i}+\alpha_{0,i}} = {Z'_{b_i}}^{-d'_{b_i}} {h'_{b_i}}^{z'_{0,b_i}}, \text{ By equation (7)} \end{aligned}$$

Therefore, the message returned from $\mathrm{U}_2(i,\cdot)$ is

$$\begin{aligned} &\mathsf{H}'(m_{b_i}, h'_{b_i}, Z_i X^{-\beta_i}, R'_{g,i}, R'_{h,i}, A'_i) - \gamma_{0,i} - \gamma_{1,i} \\ &= \mathsf{H}'(m_{b_i}, h'_{b_i}, Z'_{b_i}, X^{-d'_{b_i}} g^{z'_{0,b_i}}, {Z'_{b_i}}^{-d'_{b_i}} {h'_{b_i}}^{z'_{0,b_i}}, W^{-e'_{b_i}} g^{z'_{1,b_i}}) - \gamma_{0,i} - \gamma_{1,i} \\ &= d'_{b_i} + e'_{b_i} - \gamma_{0,i} - \gamma_{1,i} = d_i + e_i = c_i \ , \end{aligned}$$

so the second user message is consistent with Δ. Thus, the next message from $\mathcal{A}$ will be $d_i, e_i, z_{0,i}, z_{1,i}$ from the transcript Δ. Lastly, the final signatures output by the oracle U_3 are

$$(Z_i X^{-\beta_i}, d_i + \gamma_{0,i}, e_i + \gamma_{1,i}, z_{0,i} + \alpha_{0,i}, z_{1,i} + \alpha_{1,i}) = (Z'_{b_i}, d'_{b_i}, e'_{b_i}, z'_{0,b_i}, z'_{1,b_i}) = \sigma_{b_i},$$

which are exactly the signatures in Δ. □

4.3 Computational Blindness of BS_2 without NIZK

As mentioned before, we can remove the NIZK proof from our scheme BS_2 (resulting in a scheme which we will call BS'_2) and still achieve computational blindness according to the following theorem. We stress that here we make no assumptions on the hash functions used by BS'_2.

Theorem 4.4 (Computational Blindness of BS'_2). *Assume that* GGen *outputs the description of a group of prime order* $p = p(\lambda)$*, and let* $\mathsf{BS}'_2 = \mathsf{BS}'_2[\mathsf{GGen}]$*. For any adversary* $\mathcal{A}$ *for* BLIND *running in time* $t_{\mathcal{A}} = t_{\mathcal{A}}(\lambda)$*, there exists an adversary* $\mathcal{B}$ *for* DLOG *with* $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$ *such that*

$$\mathsf{Adv}^{\mathrm{blind}}_{\mathsf{BS}'_2}(\mathcal{A}, \lambda) \leqslant 2\sqrt{\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{2}{p} \ .$$

Proof. The proof for this theorem mainly follows the proof for Theorem 4.1 with the only difference being the game $\mathbf{G}_1^{\mathcal{A}}$ and its transition from $\mathbf{G}_0^{\mathcal{A}}$. We define the game $\mathbf{G}_1^{\mathcal{A}}$ as follows:
Game $\mathbf{G}_1^{\mathcal{A}}$: This game made the following changes:

- In the oracle $\textsc{Init}(\tilde{\mathsf{pk}}, \tilde{m}_0, \tilde{m}_1)$, the oracle additionally parses $X \leftarrow \mathsf{pk}$ and sets $\mathsf{sk} \leftarrow x = \log_g X$ found by exhaustive search.
- For both $i \in \{0,1\}$, in the oracle $\mathrm{U}_3(i, \mathsf{smsg}_3^{(i)})$ after it receives $\mathsf{smsg}_1^{(i)}, \mathsf{smsg}_2^{(i)}, \mathsf{smsg}_3^{(i)}$ from $\mathcal{A}$ and parses $(R_{g,i}, A_i) \leftarrow \mathsf{smsg}_1^{(i)}, (Z_i, R_{h,i}) \leftarrow \mathsf{smsg}_2^{(i)}$ and $(d_i, e_i,\ z_{0,i}, z_{1,i}) \leftarrow \mathsf{smsg}_3^{(i)}$. Then, if the user algorithm $\mathsf{BS}_2'.\mathsf{U}_3$ does not abort but $Z_i \neq h_i^{\mathsf{sk}}$ where h_i is the message $\mathrm{U}_1(i, \cdot)$ replied to $\mathcal{A}$, the game aborts.

Fix a signing session $i \in \{0,1\}$ and let Bad_i be the event where the abort described occurs in signing session i, i.e., $Z_i \neq h_i^{\mathsf{sk}}$ but the user algorithm does not abort. This gives

$$|\Pr[\mathbf{G}_1^{\mathcal{A}} = 1] - \Pr[\mathbf{G}_0^{\mathcal{A}} = 1]| \leqslant \Pr[\mathsf{Bad}_0 \vee \mathsf{Bad}_1] .$$

Note that the event Bad_i only depends on the two user messages in the signing protocol, i.e., (h_i, c_i) (since the event occurs before the signatures are returned).

To argue the probability of event Bad_i occurring, we will give a reduction $\mathcal{B}$ rewinding the adversary $\mathcal{A}$ and argue that if Bad_i occurs in both runs, $\mathcal{B}$ can extract $\log_g W$.

Before giving $\mathcal{B}$, we make the following observation that h_i and c_i are uniformly random in $\mathbb{G}$ and $\mathbb{Z}_p$ respectively. To see this, first consider that $h_i = h_i' g^{\beta_i}$ and $c_i = \mathsf{H}'(m_{b_i}, h_i', Z_i', R_{g,i}', R_{h,i}', A_i') - \gamma_{0,i} - \gamma_{1,i}$ where $h_i' = \mathsf{H}(m_{b_i}, R_{g,i}', A_i')$ and $Z_i', R_{g,i}', R_{h,i}', A_i'$ are the blinded values of $Z_i, R_{g,i}, R_{h,i}, A_i$ respectively. We specifically note that $A_i' = A_i g^{\alpha_{1,i}} W^{-\gamma_{1,i}}$ is uniform over $\mathbb{G}$ and is independent of $\gamma_{1,i}$. This is because conditioning on a value for $\gamma_{1,i}$, A_i' takes on any element in $\mathbb{G}$ with probability $1/p$ due to $\alpha_{1,i}$ being uniform over $\mathbb{Z}_p$ and independent of $\gamma_{1,i}$. Then, the distribution of (h_i, c_i) can now be seen as dependent only on the signer messages $R_{g,i}, A_i, R_{h,i}, Z_i$, the blinding randomness $\beta_i, \alpha_{0,i}, \gamma_{0,i}, \gamma_{1,i}$ and A_i'. Conditioning on every values other than β_i and $\gamma_{1,i}$, we can see that h_i is uniform over $\mathbb{G}$ as β_i is uniform over $\mathbb{Z}_p$ and c_i is uniform over $\mathbb{Z}_p$ as $\gamma_{1,i}$ is uniform over $\mathbb{Z}_p$. This means that the probability of Bad_i stays the same even if h_i and c_i are uniformly randomly sampled instead of generated by following the protocol.

Then, using the above observation, consider the following reduction $\mathcal{B}$ playing the DLOG game and running $\mathcal{A}$.

1. The reduction $\mathcal{B}$ takes as input $(\mathbb{G}, p, g, W)$ and runs $\mathcal{A}$ on input $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W)$. It also fixes the randomness to be used in the signing session $1-i$ and the first round message h_i of signing session i in advance.
2. The oracle Init, $\mathrm{U}_1(1-i, \cdot), \mathrm{U}_2(1-i, \cdot)$, and $\mathrm{U}_3(1-i, \cdot)$ are answered as in the game $\mathbf{G}_0^{\mathcal{A}}$. The oracle $\mathrm{U}_1(i, \cdot)$ instead of computing the values as usual answers with h_i instead. While for $\mathrm{U}_2(i, \cdot)$, $\mathcal{B}$ answers with a freshly sampled $c_i \leftarrow\!\!\$\, \mathbb{Z}_p$.
3. For the call to $\mathrm{U}_3(i, \mathsf{smsg}_3^{(i)})$, if the user algorithm does not abort $\mathcal{B}$ rewinds the adversary $\mathcal{A}$ to when it queries $\mathrm{U}_2(i, \mathsf{smsg}_2^{(i)})$ and replies with a fresh $c_i' \leftarrow\!\!\$\, \mathbb{Z}_p$. The oracles for the signing session $1-i$ still use the same randomness from the previous run.
4. For the call (after the rewinding) to $\mathrm{U}_3(i, \mathsf{smsg}_3'^{(i)})$, if the user algorithm does not abort, we have $(d_i, e_i, z_{0,i}, z_{1,i}) \leftarrow \mathsf{smsg}_3^{(i)}$ and $(d_i', e_i', z_{0,i}', z_{1,i}') \leftarrow \mathsf{smsg}_3'^{(i)}$. If $e_i \neq e_i'$, return $(z_{1,i} - z_{1,i}')(e_i - e_i')^{-1}$. Otherwise, abort.

It is clear that the running time of $\mathcal{B}$ is about that of $\mathcal{A}$. Then, we argue the success probability of the reduction $\mathcal{B}$ by considering the event Bad_i. We note that the event Bad_i cannot be detected efficiently; however, here we show that if such event occurs in both runs (even without $\mathcal{B}$ detecting Bad_i), the reduction $\mathcal{B}$ will find $\log_g W$. More specifically, we consider the following event frk such that the event Bad_i occurs in both the first and the rewound run of $\mathcal{A}$ in the reduction $\mathcal{B}$ and that the outputs of $\mathrm{U}_2(i, \cdot)$ over the two runs are different (i.e., $c_i' \neq c_i$). If this event occurs, then $\mathcal{A}$ has sent $(Z_i, R_{g,i}, R_{h,i}, A_i)$ and $(d_i, e_i, z_{0,i}, z_{1,i}), (d_i', e_i', z_{0,i}', z_{1,i}')$ such that

(i) $Z_i \neq h_i^x$.

(ii) $d_i + e_i = c_i \neq c_i' = d_i' + e_i'$.

(iii) $(R_{g,i}, R_{h,i}) = (g^{z_{0,i}} X^{-d_i}, h_i^{z_{0,i}} Z_i^{-d_i}) = (g^{z'_{0,i}} X^{-d'_i}, h_i^{z'_{0,i}} Z_i^{-d'_i})$

(iv) $A_i = g^{z_{1,i}} W^{-e_i} = g^{z'_{1,i}} W^{-e'_i}$

By considering (iii),

$$Z_i^{d_i - d'_i} = h_i^{z_{0,i} - z'_{0,i}} = g^{(z_{0,i} - z'_{0,i}) \log_g h_i} = X^{(d_i - d'_i) \log_g h_i} = h_i^{\mathsf{sk}(d_i - d'_i)}$$

Then, $d_i = d'_i$ follows from $Z_i \neq h_i^x$. Thus, $e_i \neq e'_i$ and $(z_{1,i} - z'_{1,i})(e_i - e'_i)^{-1} = \log_g W$ by (iv). This shows that if frk occurs, $\mathcal{B}$ wins the DLOG game, i.e., $\Pr[\mathsf{frk}] \leqslant \mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)$.

Now, we bound $\Pr[\mathsf{frk}]$ using the forking lemma (Lemma 2.1). To this end, we define a wrapper $\mathcal{A}_i$ over $\mathcal{A}$ where $\mathcal{A}_i$ takes as input the instance $(\mathbb{G}, p, g, W)$, the challenge c_i, and a randomness ρ which is used to derive the random tape for $\mathcal{A}$, h_i, and the randomness used in signing session $1 - i$. The wrapper $\mathcal{A}_i$ then simulates the signing oracles as $\mathcal{B}$ does and returns $I = 1$ when Bad_i occurs. Otherwise $\mathcal{A}_i$ returns $\perp$. This means that the probability that $I = 1 \neq \perp$ is $\Pr[\mathsf{Bad}_i]$. Also, we can see that the event frk corresponds to the event where $\mathcal{A}_i$ is run twice with the same inputs except the two different $c_i \neq c'_i$, and both runs return I and I' such that $I = I' \neq \perp$. Thus by the forking lemma, we have

$$\Pr[\mathsf{Bad}_i] \leqslant \sqrt{\Pr[\mathsf{frk}]} + \frac{1}{p} \leqslant \sqrt{\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{1}{p} \ .$$

Applying the union bound over $i \in \{0, 1\}$ concludes the proof. □

4.4 Proof of Theorem 4.2

To prove one-more strong unforgeability for BS_2, we consider the following sequence of games (pseudocode description of the games can be found in Figure 7).

Game $\mathbf{G}_0^{\mathcal{A}}$: The game first generates the public parameters $\mathsf{par} \leftarrow\!\!\$\ \mathsf{BS}_2.\mathsf{Setup}(1^\lambda)$ and the public and secret keys $(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\!\$\ \mathsf{BS}_2.\mathsf{KG}(\mathsf{par})$. Then, the game interacts with an adversary $\mathcal{A}(\mathsf{par}, \mathsf{pk})$ with access to the signing oracles $\mathrm{S}_1, \mathrm{S}_2, \mathrm{S}_3$ and the random oracles $\mathsf{H}, \mathsf{H}', \mathsf{H}''$ which are simulated by lazy sampling. The adversary $\mathcal{A}$ (w.l.o.g.) queries the signing oracle S_1 for ℓ times and the random oracle $\mathsf{H}, \mathsf{H}', \mathsf{H}''$ for $Q_{\mathsf{H}}, Q_{\mathsf{H}'}, Q_{\mathsf{H}''}$ times respectively. At the end of the game, $\mathcal{A}$ outputs $\ell + 1$ message-signature pairs (m_k^*, σ_k^*) for $k \in [\ell + 1]$. The adversary $\mathcal{A}$ succeeds if for all $k_1 \neq k_2, (m_{k_1}^*, \sigma_{k_1}^*) \neq (m_{k_2}^*, \sigma_{k_2}^*)$ and $\mathsf{BS}_2.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m_k^*, \sigma_k^*) = 1$ for all $k \in [\ell + 1]$. We additionally assume w.l.o.g. that $\mathcal{A}$ does not make the same random oracle query twice and already makes the queries to H and H' that would otherwise be called in $\mathsf{BS}_2.\mathsf{Ver}$ when the game checks the forgeries. The probability of $\mathcal{A}$ winning in game $\mathbf{G}_0^{\mathcal{A}}$ is exactly its advantage in game OMSUF, i.e.,

$$\mathsf{Adv}^{\mathrm{omsuf}}_{\mathsf{BS}_2}(\mathcal{A}, \lambda) = \Pr[\mathbf{G}_0^{\mathcal{A}} = 1] \ .$$

Game $\mathbf{G}_1^{\mathcal{A}}$: This game is identical to $\mathbf{G}_0^{\mathcal{A}}$ except that it adds a new winning condition for $\mathcal{A}$ where *for all $k \in [\ell + 1]$, parsing the signature $\sigma_k^* = (Z_k^*, d_k^*, e_k^*, z_{k,0}^*, z_{k,1}^*)$ and letting $R_{g,k}^* = g^{z_{k,0}^*} X^{-d_k^*}, A_k^* = g^{z_{1,k}^*} W^{-e_k^*}$, the game requires that $Z_k^* = \mathsf{H}(m_k^*, R_{g,k}^*, A_k^*)^{\mathsf{sk}}$.*

By Lemma 4.5, there exists an adversary $\mathcal{B}$ playing the game DLOG running in time $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$ such that

$$\Pr[\mathbf{G}_1^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_0^{\mathcal{A}} = 1] - (\ell + 1) \left(\sqrt{Q_{\mathsf{H}'} \mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p} \right) \ .$$

Game $\mathbf{G}_2^{\mathcal{A}}$: This game is identical to $\mathbf{G}_1^{\mathcal{A}}$ except that the signing oracle S_2 generates the proof π by programming the random oracle H'', i.e., it samples $s', \delta \leftarrow\!\!\$\ \mathbb{Z}_p$ and programs H'' at $(h, X, Z, g^{s'} X^{-\delta}, h^{s'} Z^{-\delta})$ as δ. H'' is already defined at $(h, X, Z, g^{s'} X^{-\delta}, h^{s'} Z^{-\delta})$, the game aborts.

The view of $\mathcal{A}$ is identical to $\mathbf{G}_1^{\mathcal{A}}$ if the game does not abort. Moreover, the game only aborts if $(h, X, Z, g^{s'} X^{-\delta}, h^{s'} Z^{-\delta})$ has been queried beforehand, but $g^{s'} X^{-\delta}$ and $h^{s'} Z^{-\delta}$ are uniformly random and

```
Game $\mathbf{G}_0^{\mathcal{A}}, \mathbf{G}_1^{\mathcal{A}}, \mathbf{G}_2^{\mathcal{A}}, \mathbf{G}_3^{\mathcal{A}}, \mathbf{G}_4^{\mathcal{A}}, \mathbf{G}_5^{\mathcal{A}}$:
$(\mathbb{G}, p, g) \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda)$
$W \leftarrow\!\!\$\ \mathbb{G}$                                          // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_3^{\mathcal{A}}$
$w \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ W \leftarrow g^w$                          // $\mathbf{G}_4^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
$\mathsf{par} \leftarrow (\mathbb{G}, p, g, W)$
$x \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ X \leftarrow g^x$
$\mathsf{sk} \leftarrow x\ ;\ \mathsf{pk} \leftarrow X$
$\ell \leftarrow 0\ ;\ \mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3 \leftarrow \varnothing$
$\{(m_k^*, \sigma_k^*)\}_{k \in [\ell+1]} \leftarrow\!\!\$\ \mathcal{A}^{\mathrm{S}_1, \mathrm{S}_2, \mathrm{S}_3}(\mathsf{par}, \mathsf{pk})$
For $k \in [\ell + 1]$:                // parsing
    $(Z_k^*, d_k^*, e_k^*, z_{0,k}^*, z_{1,k}^*) \leftarrow \sigma_k^*$
    $R_{g,k}^* \leftarrow g^{z_{0,k}^*} X^{-d_k^*}$
    $A^* \leftarrow g^{z_{1,k}^*} W^{-e_k^*}$                                  // $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
If $\exists\ k_1 \neq k_2$, $(m_{k_1}^*, \sigma_{k_1}^*) = (m_{k_2}^*, \sigma_{k_2}^*)$
    or $(m_{k_1}^*, R_{g,k_1}^*, A_{k_1}^*) = (m_{k_2}^*, R_{g,k_2}^*, A_{k_2}^*)$
                                                               // $\mathbf{G}_3^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
    then return 0
If $\exists\ k \in [\ell + 1]$ such that
    $\mathsf{BS}_2.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m_k^*, \sigma_k^*) = 0$
    or $Z_k^* \neq \mathsf{H}(m_k^*, R_{g,k}^*, A_k^*)^{\mathsf{sk}}$                       // $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
        then return 0
Return 1

Oracle $\mathsf{H}(m, R_g, A)$:
If $\mathsf{H}(m, R_g, A) \neq \perp$ then
    return $\mathsf{H}(m, R_g, A)$
$\mathsf{H}(m, R_g, A) \leftarrow\!\!\$\ \mathbb{G}$
Return $\mathsf{H}(m, R_g, A)$

Oracle $\mathsf{H}_\star(\mathrm{str})$ for $\mathsf{H}_\star \in \{\mathsf{H}', \mathsf{H}''\}$:
If $\mathsf{H}_\star(\mathrm{str}) \neq \perp$ then return $\mathsf{H}_\star(\mathrm{str})$
$\mathsf{H}_\star(\mathrm{str}) \leftarrow\!\!\$\ \mathbb{Z}_p$
Return $\mathsf{H}_\star(\mathrm{str})$
```

```
Oracle $\mathrm{S}_1(\mathrm{sid})$:
If $\mathrm{sid} \in \mathcal{I}_1$ then return $\perp$
$\ell \leftarrow \ell + 1\ ;\ \mathcal{I}_1 \leftarrow \mathcal{I}_1 \cup \{\mathrm{sid}\}$
$z_{1,\mathrm{sid}}, e_{\mathrm{sid}}, r_{0,\mathrm{sid}} \leftarrow\!\!\$\ \mathbb{Z}_p$
$R_{g,\mathrm{sid}} \leftarrow g^{r_{0,\mathrm{sid}}}$
$A_{\mathrm{sid}} \leftarrow g^{z_{1,\mathrm{sid}}} W^{-e_{\mathrm{sid}}}$   // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$z_{0,\mathrm{sid}}, d_{\mathrm{sid}}, r_{1,\mathrm{sid}} \leftarrow\!\!\$\ \mathbb{Z}_p$
$R_{g,\mathrm{sid}} \leftarrow g^{z_{0,\mathrm{sid}}} X^{-d_{\mathrm{sid}}}$
$A_{\mathrm{sid}} \leftarrow g^{r_{1,\mathrm{sid}}}$        // $\mathbf{G}_5^{\mathcal{A}}$
Return $(R_{g,\mathrm{sid}}, A_{\mathrm{sid}})$

Oracle $\mathrm{S}_2(\mathrm{sid}, h_{\mathrm{sid}})$ :
If $\mathrm{sid} \notin \mathcal{I}_1$ or $\mathrm{sid} \in \mathcal{I}_2$ then return $\perp$
$\mathcal{I}_2 \leftarrow \mathcal{I}_2 \cup \{\mathrm{sid}\}$; $Z_{\mathrm{sid}} \leftarrow h_{\mathrm{sid}}^x$
$R_{h,\mathrm{sid}} \leftarrow h_{\mathrm{sid}}^{r_{0,\mathrm{sid}}}$                                  // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$R_{h,\mathrm{sid}} \leftarrow h_{\mathrm{sid}}^{z_{0,\mathrm{sid}}} Z_{\mathrm{sid}}^{-d_{\mathrm{sid}}}$                                      // $\mathbf{G}_5^{\mathcal{A}}$
$s \leftarrow\!\!\$\ \mathbb{Z}_p\ ;\ \delta \leftarrow \mathsf{H}''(h_{\mathrm{sid}}, X, Z_{\mathrm{sid}}, g^s, h_{\mathrm{sid}}^s)$
$\pi \leftarrow (\delta, \delta \cdot x + s)$            // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_1^{\mathcal{A}}$
$\delta, s' \leftarrow\!\!\$\ \mathbb{Z}_p, \pi \leftarrow (\delta, s')$          // $\mathbf{G}_2^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
If $\mathsf{H}''(h_{\mathrm{sid}}, X, Z_{\mathrm{sid}}, g^{s'} X^{-\delta}, h_{\mathrm{sid}}^s Z_{\mathrm{sid}}^{-\delta}) \neq \perp$
    then **abort game**
$\mathsf{H}''(h_{\mathrm{sid}}, X, Z_{\mathrm{sid}}, g^{s'} X^{-\delta}, h_{\mathrm{sid}}^s Z_{\mathrm{sid}}^{-\delta}) \leftarrow \delta$
Return $(Z_{\mathrm{sid}}, \pi, R_{h,\mathrm{sid}})$

Oracle $\mathrm{S}_3(\mathrm{sid}, c_{\mathrm{sid}})$ :
If $\mathrm{sid} \notin \mathcal{I}_1, \mathcal{I}_2$ or $\mathrm{sid} \in \mathcal{I}_3$ then return $\perp$
$\mathcal{I}_3 \leftarrow \mathcal{I}_3 \cup \{\mathrm{sid}\}$
$d_{\mathrm{sid}} \leftarrow c_{\mathrm{sid}} - e_{\mathrm{sid}}$
$z_{0,\mathrm{sid}} \leftarrow r_{0,\mathrm{sid}} + d_{\mathrm{sid}} \cdot x$                        // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$e_{\mathrm{sid}} \leftarrow c_{\mathrm{sid}} - d_{\mathrm{sid}}$
$z_{1,\mathrm{sid}} \leftarrow r_{1,\mathrm{sid}} + e_{\mathrm{sid}} \cdot w$                            // $\mathbf{G}_5^{\mathcal{A}}$
Return $(d_{\mathrm{sid}}, e_{\mathrm{sid}}, z_{0,\mathrm{sid}}, z_{1,\mathrm{sid}})$
```

Fig. 7. The OMSUF = $\mathbf{G}_0^{\mathcal{A}}$ security game for BS_2 and the subsequent games $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$. We assume that the adversary $\mathcal{A}$ makes ℓ queries to the signing oracle S_1. We remark that $\mathsf{H}, \mathsf{H}', \mathsf{H}''$ are modeled as random oracles and $\mathcal{A}$ has access to them. Each box type containing the game name indicates the changes made in that game and to make things clearer, for each box, we indicate which game contains the box by a comment by the side of it. Also, we omitted the signer state and instead use variable names with subscript sid to indicate the corresponding values in the signer state.

independent of the view of $\mathcal{A}$ and previous programming attempts of H'' as s' is uniformly random and independent at the time that the oracle tries to program H''. Thus, by applying the union bound over possible collision events, i.e., all pairs of queries to oracle S_2 and queries to both H'' and S_2 (accounting for attempts to program H'').

$$\Pr[\mathbf{G}_2^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_1^{\mathcal{A}} = 1] - \frac{\ell(\ell + Q_{\mathsf{H}''})}{p}\ .$$

Game $\mathbf{G}_3^{\mathcal{A}}$: This game is identical to $\mathbf{G}_2^{\mathcal{A}}$ except that for $\mathcal{A}$ to succeed, the game additionally requires that each random oracle call to H corresponding to the verification of (m_k^*, σ_k^*) for $k \in [\ell+1]$ are all distinct. More specifically, this means that after parsing $\sigma_k^* = (Z_k^*, d_k^*, e_k^*, z_{k,0}^*, z_{k,1}^*)$ and setting $R_{g,k}^* = g^{z_{k,0}^*} X^{-d_k^*}, A_k^* = g^{z_{1,k}^*} W^{-e_k^*}$ for all $k \in [\ell + 1]$, for any $k_1 \neq k_2$,

$$(m_{k_1}^*, R_{g,k_1}^*, A_{k_1}^*) \neq (m_{k_2}^*, R_{g,k_2}^*, A_{k_2}^*)\ .$$

The change in success probability of $\mathcal{A}$ corresponds to the event where $\mathcal{A}$ outputs $\ell+1$ distinct valid message-signature pairs, but there exists $k_1 \neq k_2$ such that $(m^*_{k_1}, R^*_{g,k_1}, A^*_{k_1}) = (m^*_{k_2}, R^*_{g,k_2}, A^*_{k_2})$. Consider all the cases where this occurs:

1. Case E_1: $(d^*_{k_1}, e^*_{k_1}) = (d^*_{k_2}, e^*_{k_2})$. Consider that $R^*_{g,k_1} = R^*_{g,k_2}$ and $A^*_{k_1} = A^*_{k_2}$. By how $R^*_{g,k}$ and A^*_k are defined and that $(d^*_{k_1}, e^*_{k_1}) = (d^*_{k_2}, e^*_{k_2})$, we can infer that $z^*_{0,k_1} = z^*_{0,k_2}$ and $z^*_{1,k_1} = z^*_{1,k_2}$. Moreover, since $(m^*_{k_1}, \sigma^*_{k_1}) \neq (m^*_{k_2}, \sigma^*_{k_2})$ and $m^*_{k_1} = m^*_{k_2}$, we have $Z^*_{k_1} \neq Z^*_{k_2}$. However, by the change in $\mathbf{G}_1^{\mathcal{A}}$,
$$Z^*_{k_1} = \mathsf{H}(m^*_{k_1}, R^*_{g,k_1}, A^*_{k_1})^{\mathsf{sk}} = \mathsf{H}(m^*_{k_2}, R^*_{g,k_2}, A^*_{k_2})^{\mathsf{sk}} = Z^*_{k_2} \ .$$
Thus, this event cannot occur.
2. Case E_2: $e^*_{k_1} \neq e^*_{k_2}$. As a result of $A^*_{k_1} = A^*_{k_2}$, we can extract the discrete logarithm of W as $(z^*_{1,k_2} - z^*_{1,k_1})(e^*_{k_2} - e^*_{k_1})^{-1}$. Then, we can bound the probability of event E_2, by a direct reduction $\mathcal{B}_2$ receiving inputs $(\mathbb{G}, p, g, W)$, simulating the game $\mathbf{G}_2^{\mathcal{A}}$ against $\mathcal{A}$ and returning $(z^*_{1,k_2} - z^*_{1,k_1})(e^*_{k_2} - e^*_{k_1})^{-1}$ when E_2 occurs. Thus, the probability of E_2 occurring is bounded by $\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}_2, \lambda)$. We can also see that the running time of $\mathcal{B}_2$ is about that of $\mathcal{A}$.
3. Case E_3: $d^*_{k_1} \neq d^*_{k_2}$. With the same argument and $R^*_{g,k_1} = R^*_{g,k_2}$, this allows us to extract the discrete logarithm of X. However, the reduction here would need to answer h^x to the adversary without knowing x. To achieve this, we give the following reduction $\mathcal{B}_3$ to CT-CDH assumption instead.
   - At the beginning, $\mathcal{B}_3$ receives the CT-CDH instance $(\mathbb{G}, p, g, X)$. It then generates $W \leftarrow g^w$ where $w \leftarrow\$\ \mathbb{Z}_p$. Moreover, $\mathcal{B}_3$ queries Chal for $\ell+1$ challenges $Y_1, \ldots, Y_{\ell+1}$. The random oracles are simulated with lazy sampling as in $\mathbf{G}_3^{\mathcal{A}}$.
   - For each S_1 query, $\mathcal{B}_3$ samples $z_0, d, r_1 \leftarrow\$\ \mathbb{Z}_p$ and returns $R_g \leftarrow g^{z_0}X^{-d}$, $A \leftarrow g^{r_1}$ to $\mathcal{A}$.
   - For each S_2 query, $\mathcal{B}_3$ forwards its query h to its own Dh oracle instead of using the secret key to receive $Z = h^x$, and simulates the protocol on by setting $R_h \leftarrow h^{z_0}Z^{-d}$. It then returns Z, R_h.
   - For each S_3 query, $\mathcal{B}_3$ returns $d, e \leftarrow c - d, z_0, z_1 \leftarrow e \cdot w + r_1$. (Note here that the simulation of the oracle $\mathrm{S}_1, \mathrm{S}_2, \mathrm{S}_3$ does not require the reduction to know x)
   - At the end when E_3 occurs, $\mathcal{B}_3$ extracts the dlog of X as $x = (z^*_{0,k_2} - z^*_{0,k_1})(d^*_{k_2} - d^*_{k_1})^{-1}$ and returns the CT-CDH solutions as $Y_1^x, \ldots, Y_{\ell+1}^x$.

   Since the distribution of $(A, R_g, R_h, d, e, z_0, z_1)$ in this reduction is still identical to signing with sk, the probability of E_3 occurring in the game simulated by $\mathcal{B}_3$ is exactly the same as in $\mathbf{G}_3^{\mathcal{A}}$. With $X = g^x$, $\mathcal{B}_3$ wins the CT-CDH game if E_3 occurs. Thus, the probability of event E_3 is bounded by $\mathsf{Adv}^{\mathrm{ct-cdh}}_{\mathsf{GGen}}(\mathcal{B}_3, \lambda)$. We can also see that the running time of $\mathcal{B}_3$ is about that of $\mathcal{A}$.

Hence combining the probability of each case,
$$\Pr[\mathbf{G}_3^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_2^{\mathcal{A}} = 1] - \mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}_2, \lambda) - \mathsf{Adv}^{\mathrm{ct-cdh}}_{\mathsf{GGen}}(\mathcal{B}_3, \lambda) \ .$$

Game $\mathbf{G}_4^{\mathcal{A}}$: This game is identical to $\mathbf{G}_3^{\mathcal{A}}$ except that when generating the component W in par, the game generates the discrete logarithm $w \leftarrow\$\ \mathbb{Z}_p$ and set $W \leftarrow g^w$.

Since the game runs the oracles in the same way and W still has the same distribution, $\mathcal{A}$'s success probability is exactly the same as in $\mathbf{G}_1^{\mathcal{A}}$.
$$\Pr[\mathbf{G}_4^{\mathcal{A}} = 1] = \Pr[\mathbf{G}_3^{\mathcal{A}} = 1] \ .$$

Game $\mathbf{G}_5^{\mathcal{A}}$: This game is identical to $\mathbf{G}_4^{\mathcal{A}}$ except that the signing oracles are modified to use w instead of x in the signing protocol. More specifically, $A, R_g, R_h, d, e, z_0, z_1$ are now generated as follows:

1. $r_1, d, z_0 \leftarrow\$\ \mathbb{Z}_p$.
2. $A \leftarrow g^{r_1}$, $(R_g, R_h) \leftarrow (g^{z_0}X^{-d}, h^{z_0}Z^{-d})$.
3. After receiving c, set $e \leftarrow c - d$ and $z_1 \leftarrow ew + r_1$.

Since the joint distributions of $(A, R_g, R_h, d, e, z_0, z_1)$ in game $\mathbf{G}_4^{\mathcal{A}}$ and game $\mathbf{G}_5^{\mathcal{A}}$ are identical, the view of $\mathcal{A}$ remains the same. Thus,

$$\Pr[\mathbf{G}_5^{\mathcal{A}} = 1] = \Pr[\mathbf{G}_4^{\mathcal{A}} = 1] \; .$$

Lastly, we give a reduction $\mathcal{B}'$ playing the CT-CDH game using the adversary $\mathcal{A}$ as a subroutine. $\mathcal{B}'$ is defined as follows:

1. The reduction $\mathcal{B}'$ takes as input a CT-CDH instance $(\mathbb{G}, p, g, X)$. It samples $w \leftarrow\!\!\$\ \mathbb{Z}_p$ and sets $W \leftarrow g^w$. Lastly, it sends $\mathsf{par} = (\mathbb{G}, p, g, W), \mathsf{pk} = X$ to $\mathcal{A}$.
2. The simulations of random oracles $\mathsf{H}', \mathsf{H}''$ are done as in $\mathbf{G}_5^{\mathcal{A}}$. However, for queries to H (labeling each with $1 \leqslant j \leqslant Q_{\mathsf{H}}$), the reduction $\mathcal{B}'$ queries the challenge oracle $\textsc{Chal}$ and receives a random group element Y_j which it returns as the random oracle output. (This means that $\mathcal{B}'$ makes Q_{H} queries to $\textsc{Chal}$.)
3. The signing oracles are also simulated as in $\mathbf{G}_5^{\mathcal{A}}$ except for the computation of $Z = h^x$ in S_1 as the reduction does not know x, Z is computed by $\mathcal{B}$ querying its $\textsc{Dh}$ oracle, i.e., $Z \leftarrow \textsc{Dh}(h)$.
4. At last after receiving the forgery $(m_k^*, \sigma_k^*)_{k\in[\ell+1]}$ from $\mathcal{A}$, $\mathcal{B}'$ parses $\sigma_k^* = (Z_k^*, e_k^*, d_k^*, z_{k,0}^*, z_{k,1}^*)$, sets $R_{g,k}^* = g^{z_{k,0}^*} X^{-d_k^*}, A_k^* = g^{z_{1,k}^*} W^{-e_k^*}$, and checks if (m_k^*, σ_k^*) and $(m_k^*, R_{g,k}^*, A_k^*)$ are distinct for all $k \in [\ell+1]$ and that all the message-signature pairs are valid. If not, it aborts.
   Next, $\mathcal{B}'$ identifies j_k for $k \in [\ell+1]$ such that j_k is the index of the hash query $\mathsf{H}(m_k^*, R_{g,k}^*, A_k^*)$ made by $\mathcal{A}$. Since $(m_k^*, R_{g,k}^*, A_k^*)$ are distinct for all k, j_k are all distinct, meaning there is exactly $\ell+1$ such indices. Lastly, $\mathcal{B}'$ returns the CT-CDH solutions $(j_k, Z_k^*)_{k\in[\ell+1]}$.

It is clear that the running time of $\mathcal{B}'$ is about that of $\mathcal{A}$. For the success probability of the reduction, we can see that $\mathcal{B}'$ simulates the oracles identical to the game $\mathbf{G}_5^{\mathcal{A}}$. Let x be the discrete log of X base g. If $\mathcal{A}$ wins in game $\mathbf{G}_5^{\mathcal{A}}$, then $\mathcal{A}$ returns $Z_k^* = \mathsf{H}(m_k^*, R_{g,k}^*, A_k^*)^{\mathsf{sk}} = Y_{j_k}^x$ for all $k \in [\ell+1]$. Thus, $\mathcal{B}'$ returns $\ell+1$ correct CT-CDH solutions while only querying the oracle $\textsc{Dh}$ for ℓ times. Hence, if $\mathcal{A}$ wins in game $\mathbf{G}_5^{\mathcal{A}}$, $\mathcal{B}'$ wins in game CT-CDH. Thus,

$$\Pr[\mathbf{G}_5^{\mathcal{A}} = 1] \leqslant \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{ct-cdh}}(\mathcal{B}', \lambda) \; .$$

By combining all the advantage changes,

$$\begin{aligned}\mathsf{Adv}_{\mathsf{BS}_2}^{\mathrm{omsuf}}(\mathcal{A}, \lambda) \leqslant \frac{\ell(\ell + Q_{\mathsf{H}''})}{p} + (\ell+1)\left(\sqrt{Q_{\mathsf{H}'}\mathsf{Adv}_{\mathsf{GGen}}^{\mathsf{dlog}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p}\right) \\ + \mathsf{Adv}_{\mathsf{GGen}}^{\mathsf{dlog}}(\mathcal{B}_2, \lambda) + \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{ct-cdh}}(\mathcal{B}_3, \lambda) + \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{ct-cdh}}(\mathcal{B}', \lambda) \; .\end{aligned}$$

□

Lemma 4.5. *There exists an adversary $\mathcal{B}$ playing* DLOG *game such that its running time is $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$ and*

$$\Pr[\mathbf{G}_1^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_0^{\mathcal{A}} = 1] - (\ell+1)\left(\sqrt{Q_{\mathsf{H}'}\mathsf{Adv}_{\mathsf{GGen}}^{\mathsf{dlog}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p}\right) \; .$$

Proof. Let event Bad be the event where $\mathbf{G}_0^{\mathcal{A}}$ outputs 1 but $\mathbf{G}_1^{\mathcal{A}}$ outputs 0 which corresponds to the following event: $\mathcal{A}$ outputs $\ell+1$ message-signature pairs (m_k^*, σ_k^*) for $k \in [\ell+1]$ which we parse $\sigma_k^* = (Z_k^*, d_k^*, e_k^*, z_{k,0}^*, z_{k,1}^*)$ and set $R_{g,k}^* = g^{z_{k,0}^*} X^{-d_k^*}, A_k^* = g^{z_{1,k}^*} W^{-e_k^*}$; then, (1) for all $k_1 \neq k_2, (m_{k_1}^*, \sigma_{k_1}^*) \neq (m_{k_2}^*, \sigma_{k_2}^*)$, (2) for all $k \in [\ell+1]$, $\mathsf{BS}_2.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m_k^*, \sigma_k^*) = 1$, and (3) *there exists some $k \in [\ell+1]$ where $Z_k^* \neq$* $\mathsf{H}(m_k^*, R_{g,k}^*, A_k^*)^{\mathsf{sk}}$. Then, we can write

$$\Pr[\mathbf{G}_1^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_0^{\mathcal{A}} = 1] - \Pr[\mathsf{Bad}] \; .$$

Also, define the event Bad_k for $k \in [\ell+1]$ which is event Bad with the condition (3) modified to be for only the k-th pair (m_k^*, σ_k^*) where we have $Z_k^* \neq \mathsf{H}(m_k^*, R_{g,k}^*, A_k^*)^{\mathsf{sk}}$. This gives $\mathsf{Bad} = \bigcup_{k=1}^{\ell+1} \mathsf{Bad}_k$.

Now, define a deterministic wrapper $\mathcal{A}_k$ over the adversary $\mathcal{A}$ where $\mathcal{A}_k$ receives the following inputs: instance $(\mathbb{G}, p, g, W)$, outputs $(c_1, \ldots, c_{Q_{\mathsf{H}'}})$ of H', and a random tape ρ. $\mathcal{A}_k$ is defined as follows:

1. Extract $(x, (r_{0,i}, e_i, z_{1,i}, s_i)_{i \in [\ell]}, (h_i)_{i \in [Q_{\mathsf{H}}]}, (\delta_i)_{i \in Q_{\mathsf{H}''}}, \rho')$ from the random tape ρ where $x, r_{0,i}, e_i, z_{1,i}, s_i, \delta_i \in \mathbb{Z}_p$ and $h_i \in \mathbb{G}$.
2. Set $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W), \mathsf{pk} \leftarrow g^x, \mathsf{sk} \leftarrow x$.
3. Run $(m_k^*, \sigma_k^*)_{k \in [\ell+1]} \leftarrow \mathcal{A}^{\mathrm{S}_1, \mathrm{S}_2, \mathsf{H}, \mathsf{H}', \mathsf{H}''}(\mathsf{par}, \mathsf{pk}; \rho')$ where each oracle is answered as follows:
   - For the i-th query ($i \in [\ell]$) to S_1 use $x, (r_{0,i}, e_i, z_{1,i}, s_i)$ to answer the query as in $\mathsf{BS}_2.\mathsf{S}_1$. For the query to S_2 and S_3 of the same session id, also use $x, (r_{0,i}, e_i, z_{1,i}, s_i)$ as in $\mathsf{BS}_2.\mathsf{S}_2$ and $\mathsf{BS}_2.\mathsf{S}_3$ respectively.
   - For the i-th query ($i \in [Q_{\mathsf{H}}]$) to H, answer with h_i.
   - For the i-th query ($i \in [Q_{\mathsf{H}'}]$) to H', answer with c_i.
   - For the i-th query ($i \in [Q_{\mathsf{H}''}]$) to H'', answer with δ_i. (In these queries, we w.l.o.g. accounted for the queries that the wrapper made to generate π. Moreover, when the same queries are queried more than once, the oracle will answer with the first value initialized.)
4. If event Bad_k does not occur, return $(\perp, \perp)$. Otherwise, return $(I, (m_k^*, \sigma_k^*))$ where I is the index of the query to H' from $\mathcal{A}$ that corresponds to the verification of (m_k^*, σ_k^*). More specifically, after parsing $(Z^*, d^*, e^*, z_0^*, z_1^*)$ from σ_k^*, I is the index that corresponds to the query (m, h, Z, R_g, R_h, A) where

$$m = m_k^*,\ R_g = g^{z_0^*} X^{-d^*},\ A = g^{z_1^*} W^{-e^*},$$
$$h = \mathsf{H}(m_k^*, R_g, A),\ Z = Z^*,\ R_h = h^{z_0^*} Z^{-d^*} .$$

   Note that I is well-defined as we assume that all random oracle queries in forgery verification are made by $\mathcal{A}$ beforehand. Also, it is easy to see that the running time of $\mathcal{A}_k$ is roughly the running time of $\mathcal{A}$.

Next, we consider the following reduction $\mathcal{B}$ playing the discrete logarithm game defined as follows:

1. On the input $(\mathbb{G}, p, g, W)$, $\mathcal{B}$ samples $c_1, \ldots, c_{Q_{\mathsf{H}'}} \leftarrow\$\ \mathbb{Z}_p$ along with random coins ρ for $\mathcal{A}_k$.
2. Run $(I, (m, \sigma)) \leftarrow\$\ \mathcal{A}_k((\mathbb{G}, p, g, W), (c_1, \ldots, c_{Q_{\mathsf{H}'}}); \rho)$.
3. If $I = \perp$, abort. If not, sample $c'_I, \ldots, c'_{Q'_{\mathsf{H}}} \leftarrow\$\ \mathbb{Z}_p$ and run $(I', (m', \sigma')) \leftarrow\$\ \mathcal{A}_k((\mathbb{G}, p, g, W), (c_1, \ldots, c'_I, \ldots, c'_{Q_{\mathsf{H}'}}); \rho)$.
4. If $I = I'$ and $c'_I \neq c_I$, parse $\sigma = (Z, d, e, z_0, z_1), \sigma' = (Z', d', e', z'_0, z'_1)$, and return $(z_1 - z'_1)(e - e')^{-1}$. Otherwise, abort.

Since $\mathcal{B}$ runs $\mathcal{A}_k$ twice and the running time of $\mathcal{A}_k$ is about that of $\mathcal{A}$, $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$. Then, we argue the correctness of the reduction, i.e., we show that if $\mathcal{B}$ does not abort (i.e., $I = I' \neq \perp$ and $c_I \neq c'_I$), then it returns a discrete logarithm of W. Since $I = I' \neq \perp$, the signatures σ, σ' are: (a) valid signatures which correspond to the I-th query from $\mathcal{A}$ to H' and (b) satisfying $Z \neq \mathsf{H}(m, R_g, A)^{\mathsf{sk}}$ and $Z' \neq \mathsf{H}(m', R'_g, A')^{\mathsf{sk}}$ with $R_g = g^{z_0} X^{-d}, A = g^{z_1} W^{-e}$ and $R'_g = g^{z'_0} X^{-d'}, A' = g^{z'_1} W^{-e'}$. By (a), we know the following

(i) $m = m', \mathsf{H}(m, R_g, A) = h = h' = \mathsf{H}(m', R'_g, A'), Z = Z'$.
(ii) $c_I = d + e, c'_I = d' + e'$.
(iii) $g^{z_0} X^{-d} = g^{z'_0} X^{-d'}, h^{z_0} Z^{-d} = h^{z'_0} Z^{-d'}$.
(iv) $A = g^{z_1} W^{-e} = g^{z'_1} W^{-e'}$.

We will argue that $d = d'$. The equations in (iii) gives the following equations

$$g^{z_0 - z'_0} = X^{d-d'} \text{ and } h^{z_0 - z'_0} = Z^{d-d'}$$
$$Z^{d-d'} = X^{(d-d') \log_g(h)} = h^{x(d-d')} .$$

Since $Z \neq \mathsf{H}(m, R_g, A)^{\mathsf{sk}} = h^x$, only $d = d'$ satisfies the equation. Since $d + e = c_I \neq c'_I = d' + e'$, we have $e \neq e'$. Thus, the returned value $(z_1 - z'_1)(e - e')^{-1}$ is well defined and from (iv), this value is the discrete log of W. Hence,

$$\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda) = \Pr[\mathcal{B} \text{ does not abort}] = \Pr[I = I' \wedge I \neq \perp \wedge c_I \neq c'_I] .$$

Lastly, by the fact that $\mathcal{B}$ rewinds $\mathcal{A}_k$ which only outputs $I \neq \perp$ when Bad_k occurs, we can apply the forking lemma (Lemma 2.1),

$$\Pr[\mathsf{Bad}_k] \leqslant \sqrt{Q_{\mathsf{H}'} \mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p} .$$

The lemma statement follows from the union bound over Bad_k for $k \in [\ell + 1]$. □

```
$\mathit{\Pi}.\mathsf{Prove}^{\mathsf{H}_{\mathit{\Pi}}}((g, (h_i, \mathsf{pk}_i)_{i \in [K]}, \bar{S}), (\mathsf{sk}_i)_{i \in [K]})$ :
$r_1, \ldots, r_K \leftarrow\$ \mathbb{Z}_p$
For $i \in [K]$: $R_i \leftarrow g^{r_i}$
$\bar{R} \leftarrow \prod_{i=1}^{K} h_i^{r_i}$
$c \leftarrow \mathsf{H}_{\mathit{\Pi}}(g, (h_i, \mathsf{pk}_i)_{i \in [K]}, \bar{S}, (R_i)_{i \in [K]}, \bar{R})$
For $i \in [K]$ : $s_i \leftarrow r_i + c \cdot \mathsf{sk}_i$
Return $\pi \leftarrow (c, (s_i)_{i \in [K]})$
```

```
$\mathit{\Pi}.\mathsf{Ver}^{\mathsf{H}_{\mathit{\Pi}}}((g, (h_i, \mathsf{pk}_i)_{i \in [K]}, \bar{S}), \pi)$ :
$(c, (s_i)_{i \in [K]}) \leftarrow \pi$
For $i \in [K]$: $R_i \leftarrow g^{s_i} \mathsf{pk}_i^{-c}$
$\bar{R} \leftarrow \bar{S}^{-c} \prod_{i=1}^{K} h_i^{s_i}$
If $c \neq \mathsf{H}_{\mathit{\Pi}}(g, (h_i, \mathsf{pk}_i)_{i \in [K]}, \bar{S}, (R_i)_{i \in [K]}, \bar{R})$
    then return 0
Return 1
```

Fig. 8. Description of the proof system $\mathit{\Pi}$

5 Four-Move Blind Signatures from CDH

We present a four-move blind signature scheme BS_3, described across Figures 9 and 10. Our starting point is Rai-Choo [HLW23], a two-move blind signature scheme which is OMUF secure based on the CDH assumption in a pairing group. To better abstract our ideas, we consider a pairing-free analogue of their scheme, which we call BS_R (formally described in Figure 13), producing signatures of the form $((\mathsf{pk}_i, \varphi_i)_{i \in [K]}, \bar{S})$ with *inefficient verification* checking $\mathsf{pk} = \prod_{i=1}^{K} \mathsf{pk}_i$ and $\bar{S} = \prod_{i=1}^{K} \mathsf{H}(\mathsf{H}_\mu(m, \varphi_i))^{\log_g \mathsf{pk}_i}$. Then, to make the scheme efficiently verifiable, we add a witness-indistinguishable OR proof showing that the signature is a valid BS_R signature, or that we know the discrete logarithm of a public parameter W. It is easy to show that the scheme satisfies correctness, but for completeness, we prove this in Section 5.2.

We remark that the complexity of scheme depends on two parameters N, K of which N^{-K} needs to be negligible for the OMUF proof; for simplicity, we can set $N = 2$ and $K = \lambda$ to achieve the mentioned signature size and communication in Table 1. Also, any trade-offs in the choice of N and K (e.g. as discussed in [HLW23]) also apply to our scheme.

Blindness. To preserve blindness, we have to ensure that the signer cannot reply to the user's first message $(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]})$ with $((\mathsf{pk}_i)_{i \in [K]}, \bar{S})$ such that $\bar{S} \neq \prod_{i=1}^{K} h_{i,\vec{J}_i}^{\log_g \mathsf{pk}_i}$. Otherwise, a malicious signer can link the signatures back to the signing sessions by checking whether one of the signatures contains the values $((\mathsf{pk}'_i, \varphi_i)_{i \in [K]}, \bar{S}')$ with $\bar{S}' \neq \prod_{i=1}^{K} \mathsf{H}(\mathsf{H}_\mu(m, \varphi_i))^{\log_g \mathsf{pk}'_i}$. To avoid this, we include a proof π in the first signer response attesting that $((\mathsf{pk}_i)_{i \in [K]}, \bar{S})$ is honestly generated. For this, we use the NIZK proof system $\mathit{\Pi} = (\mathit{\Pi}.\mathsf{Prove}^{\mathsf{H}_{\mathit{\Pi}}}, \mathit{\Pi}.\mathsf{Ver}^{\mathsf{H}_{\mathit{\Pi}}})$, described in Figure 8, with access to hash function $\mathsf{H}_{\mathit{\Pi}} : \{0,1\}^* \to \mathbb{Z}_p$ which we model as a random oracle in the security proofs. We require that $\mathit{\Pi}$ satisfies completeness, soundness, and zero-knowledge in the random oracle model. The formal definitions and proofs are given to Section 5.1.

Similar to BS_1 and BS_2, one could also not include $\mathit{\Pi}$ in the protocol, and show computational blindness under the DL assumption. However, this proof would still depend on the random oracle model since the original blindness proof of Rai-Choo also required random oracles. Thus, we only consider the variant with $\mathit{\Pi}$ included, and prove the following theorem in Section 5.3.

Theorem 5.1 (Blindness of BS_3). *Assume that* GGen *outputs the description of a group of prime order* $p = p(\lambda)$, *and let* $\mathsf{BS}_3 = \mathsf{BS}_3[\mathsf{GGen}]$ *and* $K = K(\lambda), N = N(\lambda)$ *be positive integer inputs to* $\mathsf{BS}_3.\mathsf{Setup}$. *For any adversary* $\mathcal{A}$ *playing the game* BLIND *making at most* $Q_{\mathsf{H}_\star} = Q_{\mathsf{H}_\star}(\lambda)$ *queries to* $\mathsf{H}_\star \in \{\mathsf{H}_{\mathit{\Pi}}, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}\}$, *modeled as random oracles, we have*

$$\mathsf{Adv}^{\mathrm{blind}}_{\mathsf{BS}_3}(\mathcal{A}, \lambda) \leqslant \frac{2Q_{\mathsf{H}_{\mathit{\Pi}}}}{p} + \frac{2KNQ_{\mathsf{H}_\mu}}{2^\lambda} + \frac{2KQ_{\mathsf{H}_\beta}}{2^\lambda} + \frac{2KQ_{\mathsf{H}_{\mathsf{com}}}}{2^\lambda} .$$

One-more unforgeability. The following theorem shows one-more unforgeability of BS_3. The proof is given below in Section 5.4.

Theorem 5.2 (OMUF of BS_3). *Assume that* GGen *outputs the description of a group of prime order* $p = p(\lambda)$, *and let* $\mathsf{BS}_3 = \mathsf{BS}_3[\mathsf{GGen}]$ *and* $K = K(\lambda), N = N(\lambda)$ *be positive integer inputs to* $\mathsf{BS}_3.\mathsf{Setup}$. *For any adversary* $\mathcal{A}$ *for game* OMUF *with running time* $t_{\mathcal{A}} = t_{\mathcal{A}}(\lambda)$, *making at most* $\ell = \ell(\lambda)$ *queries to* S_1, *and*

```
Algorithm $\mathsf{BS}_3.\mathsf{U}_1(\mathsf{par}, \mathsf{pk}, m)$ :
$(\mathbb{G}, p, g, W, K, N, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}, \mathsf{H}', \mathsf{H}_{cc}, \mathsf{H}_\Pi) \leftarrow \mathsf{par}$
For $(i, j) \in [K] \times [N]$:
    $\varphi_{i,j} \leftarrow\$ \{0,1\}^\lambda$ ; $\mu_{i,j} \leftarrow \mathsf{H}_\mu(m, \varphi_{i,j})$
    $\varepsilon_{i,j} \leftarrow\$ \{0,1\}^\lambda$ ; $\beta_{i,j} \leftarrow \mathsf{H}_\beta(\varepsilon_{i,j})$
    $\mathsf{r}_{i,j} \leftarrow (\mu_{i,j}, \varepsilon_{i,j})$ ; $\mathsf{com}_{i,j} \leftarrow \mathsf{H}_{\mathsf{com}}(\mathsf{r}_{i,j})$
    $h'_{i,j} \leftarrow \mathsf{H}(\mu_{i,j})$ ; $h_{i,j} \leftarrow h'_{i,j} g^{\beta_{i,j}}$
$\mathsf{com} \leftarrow (\mathsf{com}_{i,j})_{i \in [K], j \in [N]}$
$h \leftarrow (h_{i,j})_{i \in [K], j \in [N]}$
$\vec{J} \leftarrow \mathsf{H}_{cc}(\mathsf{com}, h)$
$\mathsf{umsg}_1 \leftarrow (\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]})$
$\mathsf{st}_1^u \leftarrow (m, \mathsf{pk}, W,$
    $(h_{i,\vec{J}_i}, h'_{i,\vec{J}_i}, \beta_{i,\vec{J}_i}, \varphi_{i,\vec{J}_i})_{i \in [K]})$
Return $(\mathsf{st}_1^u, \mathsf{umsg}_1)$
```

```
Algorithm $\mathsf{BS}_3.\mathsf{U}_2(\mathsf{st}_1^u, \mathsf{smsg}_1)$ :
$(m, \mathsf{pk}, W, (h_{i,\vec{J}_i}, h'_{i,\vec{J}_i}, \beta_{i,\vec{J}_i}, \varphi_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{st}_1^u$
$((\mathsf{pk}_i)_{i \in [K-1]}, \bar{S}, \vec{R}, \bar{R}, A, \pi) \leftarrow \mathsf{smsg}_1$
$\mathsf{pk}_K \leftarrow \mathsf{pk} \prod_{i \in [K]} \mathsf{pk}_i^{-1}$
If $\Pi.\mathsf{Ver}^{\mathsf{H}_\Pi}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}), \pi) = 0$
    then return $\perp$
$((\mathsf{pk}'_i)_{i \in [K]}, \bar{S}', \vec{\tau}) \leftarrow\$$
    $\mathsf{ReRa}((\mathsf{pk}_i, h'_{i,\vec{J}_i})_{i \in [K]}, \bar{S} \prod_{i=1}^K \mathsf{pk}_i^{-\beta_{i,\vec{J}_i}})$
$\alpha_1, \gamma_0, \gamma_1 \leftarrow\$ \mathbb{Z}_p, \vec{\alpha}_0 \leftarrow\$ \mathbb{Z}_p^K$
Let $\vec{R}' \in \mathbb{G}^K$
For $i \in [K]$ : $\vec{R}'_i \leftarrow \vec{R}_i \mathsf{pk}_i'^{-\gamma_0} g^{\vec{\alpha}_{0,i}}$
$\bar{R}' \leftarrow \bar{R} \bar{S}'^{-\gamma_0} \prod_{i=1}^K \vec{R}_i^{-\beta_{i,\vec{J}_i}} h'^{\vec{\alpha}_{0,i}}_{i,\vec{J}_i}$
$A' \leftarrow A W^{-\gamma_1} g^{\alpha_1}$
$c' \leftarrow \mathsf{H}'(m, (h'_{i,\vec{J}_i}, \mathsf{pk}'_i)_{i \in [K]}, \bar{S}', \vec{R}', \bar{R}', A')$
$c \leftarrow c' - \gamma_0 - \gamma_1$
$\mathsf{st}_2^u \leftarrow (c, \vec{\alpha}_0, \alpha_1, \gamma_0, \gamma_1, \vec{\tau}$
    $(\mathsf{pk}_i, \mathsf{pk}'_i)_{i \in [K]}, \bar{S}, \bar{S}', \vec{R}, \bar{R}, A, \mathsf{st}_1^u)$
$\mathsf{umsg}_2 \leftarrow c$
Return $(\mathsf{st}_2^u, \mathsf{umsg}_2)$
```

```
Algorithm $\mathsf{BS}_3.\mathsf{U}_3(\mathsf{st}_2^u, \mathsf{smsg}_2)$ :
$(c, \vec{\alpha}_0, \alpha_1, \gamma_0, \gamma_1, \vec{\tau}, (\mathsf{pk}_i, \mathsf{pk}'_i)_{i \in [K]},$
        $\bar{S}, \bar{S}', \vec{R}, \bar{R}, A, \mathsf{st}_1^u) \leftarrow \mathsf{st}_2^u$
$(m, \mathsf{pk}, W, (h_{i,\vec{J}_i}, h'_{i,\vec{J}_i}, \beta_{i,\vec{J}_i}, \varphi_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{st}_1^u$
$(d, e, \vec{z}_0, z_1) \leftarrow \mathsf{smsg}_2$
If $c \neq e + d$ or
    $\exists i \in [K], \vec{R}_i \mathsf{pk}_i^d \neq g^{\vec{z}_{0,i}}$ or
    $\bar{R} \bar{S}^d \neq \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{z}_{0,i}}$ or
    $A W^e \neq g^{z_1}$ then return $\perp$
$d' \leftarrow d + \gamma_0$ ; $e' \leftarrow e + \gamma_1$
$\vec{z}'_0 \leftarrow \vec{z}_0 + \vec{\alpha}_0 + d \cdot \vec{\tau}$ ; $z'_1 \leftarrow z_1 + \alpha_1$
$\sigma \leftarrow ((\mathsf{pk}'_i, \varphi_{i,\vec{J}_i})_{i \in [K]}, \bar{S}', d', e', \vec{z}'_0, z'_1)$
Return $\sigma$
```

```
Algorithm $\mathsf{BS}_3.\mathsf{S}_1(\mathsf{par}, \mathsf{sk}, \mathsf{umsg}_1)$ :
$(\mathbb{G}, p, g, W, K, N, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}, \mathsf{H}', \mathsf{H}_{cc}, \mathsf{H}_\Pi) \leftarrow \mathsf{par}$
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{umsg}_1$
If $\mathsf{Check}(\mathsf{umsg}_1) = 0$ then return $\perp$
For $i \in [K-1]$ : $\mathsf{sk}_i \leftarrow\$ \mathbb{Z}_p$
$\mathsf{sk}_K \leftarrow \mathsf{sk} - \sum_{i=1}^{K-1} \mathsf{sk}_i$
For $i \in [K]$ : $\mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i}$
$\bar{S} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_i}$
$z_1, e \leftarrow\$ \mathbb{Z}_p, \vec{r}_0 \leftarrow\$ \mathbb{Z}_p^K$
$A \leftarrow g^{z_1} W^{-e}$ ; $\vec{R} \leftarrow (g^{\vec{r}_{0,1}}, \ldots, g^{\vec{r}_{0,K}})$
$\bar{R} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{r}_{0,i}}$
$\pi \leftarrow\$ \Pi.\mathsf{Prove}^{\mathsf{H}_\Pi}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}), (\mathsf{sk}_i)_{i \in [K]})$
$\mathsf{smsg}_1 \leftarrow ((\mathsf{pk}_i)_{i \in [K-1]}, \bar{S}, \vec{R}, \bar{R}, A, \pi)$
$\mathsf{st}^s \leftarrow ((\mathsf{sk}_i)_{i \in [K]}, \vec{r}_0, e, z_1)$
Return $(\mathsf{st}^s, \mathsf{smsg}_1)$
```

```
Algorithm $\mathsf{BS}_3.\mathsf{S}_2(\mathsf{st}^s, \mathsf{umsg}_2)$ :
$((\mathsf{sk}_i)_{i \in [K]}, \vec{r}_0, e, z_1) \leftarrow \mathsf{st}^s$
$c \leftarrow \mathsf{umsg}_1$ ; $d = c - e$
For $i \in [K]$: $\vec{z}_{0,i} = d \cdot \mathsf{sk}_i + \vec{r}_{0,i}$
Return $\mathsf{smsg}_2 \leftarrow (d, e, \vec{z}_0, z_1)$
```

Fig. 9. The signing protocol of the blind signature scheme $\mathsf{BS}_3 = \mathsf{BS}_3[\mathsf{GGen}]$ with the algorithms $\mathsf{BS}_3.\mathsf{Setup}, \mathsf{BS}_3.\mathsf{KG}, \mathsf{BS}_3.\mathsf{Ver}, \mathsf{Check}$ and ReRa defined in Figure 10 and the proof system Π (in highlighted boxes) defined in Figure 8. Also, we give a protocol-style description of BS_3 in Figure 18.

$Q_{\mathsf{H}_\star} = Q_{\mathsf{H}_\star}(\lambda)$ queries to $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}', \mathsf{H}_\Pi, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}\}$, modeled as random oracles, there exist adversaries $\mathcal{B}$ and $\mathcal{B}'$ for games DLOG *and* CDH, *respectively, such that*

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{BS}_3}(\mathcal{A}, \lambda) \leqslant (\ell + 1)\left(\sqrt{Q_{\mathsf{H}'} \mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p}\right) + \frac{\ell(\ell + Q_{\mathsf{H}_\Pi})}{p}$$
$$+ \frac{\ell}{N^K} + \frac{Q^2_{\mathsf{H}_{\mathsf{com}}} + Q^2_{\mathsf{H}_\mu} + Q_{\mathsf{H}_{\mathsf{com}}} Q_{\mathsf{H}_{cc}} + Q_{\mathsf{H}} Q_{\mathsf{H}_\mu}}{2^\lambda} + 4\ell \cdot \mathsf{Adv}^{\mathrm{cdh}}_{\mathsf{GGen}}(\mathcal{B}', \lambda) .$$

Furthermore, $\mathcal{B}$ runs in time $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$ and $\mathcal{B}'$ runs in time $t_{\mathcal{B}'} \approx t_{\mathcal{A}}$.

The best way to visualize our proof below is that it follows a similar blueprint to the earlier OMUF proofs, but where in lieu of reducing to CT-CDH, we reduce to the OMUF of the pairing-free Rai-Choo scheme BS_R with inefficient verification. To do so, we look at the produced $\ell + 1$ signatures, and see whether the $(\mathsf{pk}_i, \varphi_i)_{i \in [K]}$ and $\bar{S}$ portion satisfies $\bar{S} = \prod_{i=1}^K \mathsf{H}(\mathsf{H}_\mu(m, \varphi_i))^{\log_g \mathsf{pk}_i}$ in *all of them*. If this does *not* occur, we can rewind $\mathcal{A}$ to extract $\log_g W$. If this *does* occur, then we reduce a one-more forgery for BS_3 to one for BS_R.

```
Algorithm $\mathsf{BS}_3.\mathsf{Setup}(1^\lambda, K, N)$ :
$(\mathbb{G}, p, g) \leftarrow\!\$\ \mathsf{GGen}(1^\lambda)$ ; $W \leftarrow\!\$\ \mathbb{G}$
Select $\mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}} : \{0,1\}^* \to \{0,1\}^\lambda$
Select $\mathsf{H} : \{0,1\}^* \to \mathbb{G}$
Select $\mathsf{H}_\beta, \mathsf{H}', \mathsf{H}_\Pi : \{0,1\}^* \to \mathbb{Z}_p$
Select $\mathsf{H}_{cc} : \{0,1\}^* \to [N]^K$
$\mathsf{par} \leftarrow (\mathbb{G}, p, g, W, K, N,$
          $\mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}, \mathsf{H}', \mathsf{H}_{cc}, \mathsf{H}_\Pi)$
Return $\mathsf{par}$

Algorithm $\mathsf{Check}(\mathsf{open})$:
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{open}$
For $i \in [K]$ and $j \in [K] \backslash \{\vec{J}_i\}$:
    $\mathsf{com}_{i,j} \leftarrow \mathsf{H}_{\mathsf{com}}(\mathsf{r}_{i,j})$
    $(\mu_{i,j}, \varepsilon_{i,j}) \leftarrow \mathsf{r}_{i,j}$ ; $\beta_{i,j} \leftarrow \mathsf{H}_\beta(\varepsilon_{i,j})$
    $h_{i,j} \leftarrow \mathsf{H}(\mu_{i,j}) g^{\beta_{i,j}}$
$\mathsf{com} \leftarrow (\mathsf{com}_{i,j})_{(i,j) \in [K] \times [N]}$
$h \leftarrow (h_{i,j})_{(i,j) \in [K] \times [N]}$
If $\vec{J} \neq \mathsf{H}_{cc}(\mathsf{com}, h)$ then return 0.
Return 1
```

```
Algorithm $\mathsf{BS}_3.\mathsf{KG}(\mathsf{par})$ :
$(\mathbb{G}, p, g, W, K, N, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}, \mathsf{H}', \mathsf{H}_{cc}, \mathsf{H}_\Pi) \leftarrow \mathsf{par}$
$x \leftarrow\!\$\ \mathbb{Z}_p$ ; $X \leftarrow g^x$ ; $\mathsf{sk} \leftarrow x$ ; $\mathsf{pk} \leftarrow X$
Return $(\mathsf{sk}, \mathsf{pk})$

Algorithm $\mathsf{BS}_3.\mathsf{Ver}(\mathsf{par}, pk, m, \sigma)$ :
$(\mathbb{G}, p, g, W, K, N, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}, \mathsf{H}', \mathsf{H}_{cc}, \mathsf{H}_\Pi) \leftarrow \mathsf{par}$
$((\mathsf{pk}_i, \varphi_i)_{i \in [K]}, \bar{S}, d, e, \vec{z}_0, z_1) \leftarrow \sigma$
For $i \in [K]$ :
    $h_i \leftarrow \mathsf{H}(\mathsf{H}_\mu(m, \varphi_i))$ ; $\vec{R}_i \leftarrow g^{\vec{z}_{0,i}} \mathsf{pk}_i^{-d}$
$\bar{R} \leftarrow \bar{S}^{-d} \prod_{i=1}^K h_i^{\vec{z}_{0,i}}$ ; $A \leftarrow W^{-e} g^{z_1}$
If $\mathsf{pk} \neq \prod_{i \in [K]} \mathsf{pk}_i$ or
    $d + e \neq \mathsf{H}'(m, (h_i, \mathsf{pk}_i)_{i \in [K]}, \bar{S}, \vec{R}, \bar{R}, A)$
    then return 0
Return 1

Algorithm $\mathsf{ReRa}((\mathsf{pk}_i, h_i)_{i \in [K]}, S)$:
$\tau_1, \ldots, \tau_{K-1} \leftarrow\!\$\ \mathbb{Z}_p$; $\tau_K \leftarrow -\sum_{i=1}^{K-1} \tau_K$
For $i \in [K]$: $\mathsf{pk}'_i \leftarrow \mathsf{pk}_i g^{\tau_i}$
$S' \leftarrow S \prod_{i=1}^K h_i^{\tau_i}$
Return $((\mathsf{pk}_i)_{i \in [K]}, S', (\tau_i)_{i \in [K]})$
```

Fig. 10. Description for algorithms $\mathsf{BS}_3.\mathsf{Setup}, \mathsf{BS}_3.\mathsf{KG}, \mathsf{BS}_3.\mathsf{Ver}, \mathsf{Check}$ and ReRa

5.1 Security Properties of Proof System Π

The lemma below establishes the security properties for the proof system Π (defined in Figure 8) with the hash function $\mathsf{H}_\Pi : \{0,1\}^* \to \mathbb{Z}_p$ modeled as a random oracle.

Lemma 5.3. *Let $\mathbb{G}$ be a group of prime order $p = p(\lambda)$ with generator g and $K = K(\lambda)$ be a positive integer. Denote $\mathcal{L}_{\mathbb{G},K}$ as the following language:*

$$\mathcal{L}_{\mathbb{G},K} := \left\{ (g, (h_i, \mathsf{pk}_i)_{i \in [K]}, \bar{S}) : \bar{S} = \prod_{i=1}^{K} h_i^{\log_g \mathsf{pk}_i} \right\} .$$

The proof system Π (defined in Figure 8) satisfies the following properties with regard to $\mathcal{L}_{\mathbb{G},K}$ where the corresponding security games are defined in Figure 11:

- ***Completeness**: For any $\mathsf{st} = (g, (h_i, \mathsf{pk}_i)_{i \in [K]}, \bar{S}) \in \mathcal{L}_{\mathbb{G},K}$ and $\mathsf{sk}_i = \log_g \mathsf{pk}_i$ for $i \in [K]$,*

$$\mathsf{Pr}[\Pi.\mathsf{Ver}^{\mathsf{H}_\Pi}(\mathsf{st}, \pi) = 1 | \pi \leftarrow\!\$\ \Pi.\mathsf{Prove}^{\mathsf{H}_\Pi}(\mathsf{st}, (\mathsf{sk}_i)_{i \in [K]})] = 1 .$$

- ***Soundness**: For any adversary $\mathcal{A}$ playing the game* Sound *and making $Q_{\mathsf{H}_\Pi} = Q_{\mathsf{H}_\Pi}(\lambda)$ queries to H_Π,*

$$\mathsf{Pr}[\mathrm{Sound}_\Pi^{\mathcal{A}}(\lambda) = 1] \leqslant \frac{Q_{\mathsf{H}_\Pi}}{p} .$$

- ***Zero-Knowledge**: There exists a simulator Sim, which can program the random oracle H_Π, such that for any statement $\mathsf{st} \in \mathcal{L}_{\mathbb{G},K}$ and any adversary $\mathcal{A}$ playing the game* ZK *and making $Q_{\mathsf{H}_\Pi} = Q_{\mathsf{H}_\Pi}(\lambda)$ query to the random oracle H_Π and $Q_{\mathrm{CHAL}} = Q_{\mathrm{CHAL}}(\lambda)$ to* CHAL*, we have*

$$|\mathsf{Pr}[\mathrm{ZK}_{\Pi,0}^{\mathcal{A}}(\lambda) = 1] - \mathsf{Pr}[\mathrm{ZK}_{\Pi,1}^{\mathcal{A}}(\lambda) = 1]| \leqslant \frac{Q_{\mathrm{CHAL}}(Q_{\mathrm{CHAL}} + Q_{\mathsf{H}_\Pi})}{p} .$$

Proof (of Lemma 5.3). We consider each of the listed properties.

- **Completeness.** Completeness follows by inspection.

```
Game $\mathrm{Sound}_{\Pi}^{\mathcal{A}}(\lambda)$
$(\mathbb{G}, p, g) \leftarrow\$ \mathsf{GGen}(1^\lambda)$
$(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}, \pi) \leftarrow\$ \mathcal{A}^{\mathsf{H}_\Pi}(\mathbb{G}, p, g)$
If $\Pi.\mathsf{Ver}((g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}), \pi) = 1$
    and $(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}) \notin \mathcal{L}_{\mathbb{G},K}$
        then return 1
Return 0

Oracle $\mathsf{H}_\Pi(\mathrm{str})$ :
If $\mathsf{H}_\Pi(\mathrm{str}) \neq \perp$ then return $\mathsf{H}_\Pi(\mathrm{str})$
$\mathsf{H}_\Pi(\mathrm{str}) \leftarrow\$ \mathbb{Z}_p$
Return $\mathsf{H}_\Pi(\mathrm{str})$
```

```
Game $\mathrm{ZK}_{\Pi,b}^{\mathcal{A}}(\lambda)$
$(\mathbb{G}, p, g) \leftarrow\$ \mathsf{GGen}(1^\lambda)$
$b' \leftarrow\$ \mathcal{A}^{\textsc{Chal},\mathsf{H}_\Pi}(\mathbb{G}, p, g)$
Return $b'$

Oracle $\textsc{Chal}(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}, (\mathsf{sk}_i)_{i\in[K]})$ :
If $\exists i \in [K], \mathsf{pk}_i \neq g^{\mathsf{sk}_i}$ or $\bar{S} \neq \prod_{i=1}^K h_i^{\mathsf{sk}_i}$
    then return $\perp$
$\pi_0 \leftarrow\$ \Pi.\mathsf{Prove}^{\mathsf{H}_\Pi}((g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}), (\mathsf{sk}_i)_{i\in[K]})$
$\pi_1 \leftarrow\$ \mathsf{Sim}(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S})$
Return $\pi_b$

Algorithm $\mathsf{Sim}(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S})$ :
$c, s_1, \ldots, s_K \leftarrow\$ \mathbb{Z}_p$
For $i \in [K]$: $R_i \leftarrow g^{s_i}\mathsf{pk}_i^{-c}$
$\bar{R} \leftarrow \bar{S}^{-c}\prod_{i=1}^K h_i^{s_i}$
If $\mathsf{H}_\Pi(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}, (R_i)_{i\in[K]}, \bar{R}) \neq \perp$
    then return $\perp$
Program $\mathsf{H}_\Pi(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}, (R_i)_{i\in[K]}, \bar{R}) \leftarrow c$
Return $\pi \leftarrow (c, (s_i)_{i\in[K]})$
```

Fig. 11. The security games $\mathrm{Sound}_{\Pi}^{\mathcal{A}}$ and $\mathrm{ZK}_{\Pi,b}^{\mathcal{A}}$ for the proof system Π.

- **Soundness.** Let $\mathcal{A}$ be an adversary playing the soundness game which outputs a statement $(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}) \notin \mathcal{L}_{\mathbb{G},K}$ and a proof $\pi = (c, (s_i)_{i\in[K]})$ where $s_i \in \mathbb{Z}_p$ for $i \in [K]$. Since the statement is not in the language, $\bar{S} \neq \prod_{i=1}^K h_i^{\log_g \mathsf{pk}_i}$. Also, because π is a valid proof for $(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S})$,

$$c = \mathsf{H}_\Pi\left(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}, (g^{s_i}\mathsf{pk}_i^{-c})_{i\in[K]}, \bar{S}^{-c}\prod_{i=1}^K h_i^{s_i}\right) .$$

Here, w.l.o.g., assume that $\mathcal{A}$ already made this query as it is done when checking the validity of π anyways. Then, consider any query $(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}, (R_i)_{i\in[K]}, \bar{R})$ to H_Π where $\bar{S} \neq \prod_{i=1}^K h_i^{\log_g \mathsf{pk}_i}$. We will show that there is exactly one $c \in \mathbb{Z}_p$ which allows the existence of $(s_1, \ldots, s_K) \in \mathbb{Z}_p^K$ such that

$$\mathsf{pk}_i^c R_i = g^{s_i} \text{ for } i \in [K], \text{ and } \bar{S}^c\bar{R} = \prod_{i=1}^K h_i^{s_i} .$$

We consider such c, which gives us the above equations. Then, by raising $\mathsf{pk}_i^c R_i = g^{s_i}$ to $\log_g h_i$, we have $h_i^{c\log_g \mathsf{pk}_i} R_i^{\log_g h_i} = h_i^{s_i}$ for all $i \in [K]$. Thus, we have that $\prod_{i=1}^K h_i^{c\log_g \mathsf{pk}_i} R_i^{\log_g h_i} = \prod_{i=1}^K h_i^{s_i} = \bar{S}^c\bar{R}$, implying $\bar{R}\prod_{i=1}^K R_i^{-\log_g h_i} = \left(\prod_{i=1}^K h_i^{\log_g \mathsf{pk}_i}\bar{S}^{-1}\right)^c$. Since $\prod_{i=1}^K h_i^{\log_g \mathsf{pk}_i}\bar{S}^{-1} \neq 1_{\mathbb{G}}$, there exists only one c satisfying this property. Then, for any query to H_Π involving a statement not in the language, the probability of getting c which allows the adversary to give a valid proof is at most $1/p$. Hence, since $\mathcal{A}$ makes Q_{H_Π} queries to H_Π,

$$\mathsf{Pr}[\mathrm{Sound}_{\Pi}^{\mathcal{A}}(\lambda) = 1] \leqslant \frac{Q_{\mathsf{H}_\Pi}}{p} .$$

- **Zero-knowledge.** Consider the simulator Sim as described in Figure 11 which programs the random oracle H_Π. First, we can see that if the simulator Sim does not abort, then the adversary's view is exactly the same as when the proofs are generated honestly. Then, to see the abort probability, the simulator aborts if it tries to program the oracle at a point which was queried or programmed before. Since the simulator programs at tuple which includes $R_1 = g^{s_1}\mathsf{pk}_1^{-c}$ for $s_1 \leftarrow\$ \mathbb{Z}_p$ which is uniformly random over $\mathbb{G}$, the probability that a tuple including R_1 has been initialized on H_Π before is at most $(Q_{\mathsf{H}_\Pi} + Q_{\textsc{Chal}})/p$ (counting the random oracle queries and the programming attempts). Thus, bounding this over $Q_{\textsc{Chal}}$

queries to $\textsc{Chal}$,

$$|\mathsf{Pr}[\mathrm{ZK}^{\mathcal{A}}_{\Pi,0}(\lambda) = 1] - \mathsf{Pr}[\mathrm{ZK}^{\mathcal{A}}_{\Pi,1}(\lambda) = 1]| \leqslant \frac{Q_{\textsc{Chal}}(Q_{\textsc{Chal}} + Q_{\mathsf{H}_{\Pi}})}{p} .$$

□

5.2 Correctness of BS_3

Theorem 5.4. BS_3 *satisfies correctness.*

Proof. To show correctness, we show that the signing protocol does not abort and that the final signature is valid via the verification algorithm $\mathsf{BS}_3.\mathsf{Ver}$. Hence, we consider each step in the signing protocol and the signature verification as follows:

- The first user algorithm $\mathsf{BS}_3.\mathsf{U}_1$: For $i \in [K], j \in [N]$, we have the following values defined
  - $\mu_{i,j} = \mathsf{H}_{\mu}(m, \varphi_{i,j})$
  - $\beta_{i,j} = \mathsf{H}_{\beta}(\varepsilon_{i,j})$
  - $\mathsf{r}_{i,j} = (\mu_{i,j}, \varepsilon_{i,j})$ and $\mathsf{com}_{i,j} = \mathsf{H}_{\mathsf{com}}(\mathsf{r}_{i,j})$
  - $h'_{i,j} = \mathsf{H}(\mu_{i,j}), h_{i,j} = h'_{i,j} g^{\beta_{i,j}}$

  Also, $\vec{J} = \mathsf{H}_{cc}(\mathsf{com}, h)$ where $\mathsf{com} = (\mathsf{com}_{i,j})_{i,j}, h = (h_{i,j})_{i,j}$.
- The first signer algorithm $\mathsf{BS}_3.\mathsf{S}_1$: The algorithm runs Check, retracing the same computation in $\mathsf{BS}_3.\mathsf{U}_1$ for $i \in [K]$ and $j \in [K] \backslash \{\vec{J}_i\}$, and getting the same com and h which pass the check $\vec{J} = \mathsf{H}_{cc}(\mathsf{com}, h)$. Then, the signer first message consists of $((\mathsf{pk}_i)_{i \in [K-1]}, \bar{S}, \vec{R}, \bar{R}, A, \pi)$ each defined as follows:
  - $\mathsf{pk}_i = g^{\mathsf{sk}_i}$ for $i \in [K]$ and $\mathsf{sk}_K = \mathsf{sk} - \sum_{i=1}^{K-1} \mathsf{sk}_i$.
  - $\bar{S} = \prod_{i=1}^{K} h^{\mathsf{sk}_i}_{i,\vec{J}_i}$
  - $\vec{R} = (g^{\vec{r}_{0,1}}, \ldots, g^{\vec{r}_{0,K}}), \bar{R} = \prod_{i=1}^{K} h^{\vec{r}_{0,i}}_{i,\vec{J}_i}$
  - $A = g^{z_1} W^{-e}$
  - $\pi \leftarrow\!\!\$\ \Pi.\mathsf{Prove}^{\mathsf{H}_{\Pi}}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}), (\mathsf{sk}_i)_{i \in [K]})$
- The second user algorithm $\mathsf{BS}_3.\mathsf{U}_2$: The algorithm checks that the $\Pi.\mathsf{Ver}^{\mathsf{H}_{\Pi}}$ on π returns 1 which is always true by the completeness of Π. Then, the blinded values of $\mathsf{pk}_i, \bar{S}, R_i, \bar{R}, A$ are as follows:
  - By the definition of ReRa,

  $$\mathsf{pk}'_i = \mathsf{pk}_i g^{\tau_i} \text{ for } i \in [K], \prod_{i=1}^{K} \mathsf{pk}'_i = \mathsf{pk} \text{ and } \bar{S}' = \bar{S} \prod_{i=1}^{K} \mathsf{pk}_i^{-\beta_{i,\vec{J}_i}} {h'_{i,\vec{J}_i}}^{\tau_i}$$

  - $\vec{R}'_i = \vec{R}_i {\mathsf{pk}'_i}^{-\gamma_0} g^{\vec{\alpha}_{0,i}}$ for $i \in [K]$ and $\bar{R}' = \bar{R} {\bar{S}'}^{-\gamma_0} \prod_{i=1}^{K} \vec{R}_i^{-\beta_{i,\vec{J}_i}} {h'_{i,\vec{J}_i}}^{\vec{\alpha}_{0,i}}$
  - $A' = A W^{-\gamma_1} g^{\alpha_1}$
- The third user algorithm $\mathsf{BS}_3.\mathsf{U}_3$: On the signer message $(d, e, \vec{z}_0, z_1)$, the following checks pass:
  - $c = d + e$ because e is defined as $c - d$ by the second signer algorithm.
  - For all $i \in [K], \vec{R}_i \mathsf{pk}_i^d = g^{\vec{r}_{0,i} + d \cdot \mathsf{sk}_i} = g^{\vec{z}_{0,i}}$. Also, $\bar{R}\bar{S}^d = \prod_{i=1}^{K} h^{\vec{r}_{0,i} + d \cdot \mathsf{sk}_i}_{i,\vec{J}_i} = \prod_{i=1}^{K} h^{\vec{z}_{0,i}}_{i,\vec{J}_i}$.
  - $AW^e = W^{-e} g^{z_1} W^e = g^{z_1}$.
- Signature verification: The final signature is $\sigma = ((\mathsf{pk}'_i, \varphi_{i,\vec{J}_i})_{i \in [K]}, \bar{S}', d', e', \vec{z}'_0, z'_1)$, and following from the checks in the third user algorithm, we have
  - $d' + e' = d + e + \gamma_0 + \gamma_1 = c + \gamma_0 + \gamma_1 = c'$.

- For $i \in [K]$, $g^{\vec{z}'_{0,i}}\mathsf{pk}_i'^{-d'} = g^{\vec{z}_{0,i}+\vec{\alpha}_{0,i}+d\cdot\vec{\tau}_i}(\mathsf{pk}_i g^{\tau_i})^{-d}\mathsf{pk}_i'^{-\gamma_0} = \vec{R}_i g^{\vec{\alpha}_{0,i}}\mathsf{pk}_i'^{-\gamma_0} = \vec{R}'_i$. Also,

$$\begin{aligned}
\bar{S}'^{-d'}\prod_{i=1}^{K} h'_{i,\vec{J}_i}{}^{\vec{z}'_{0,i}} &= (\bar{S}\prod_{i=1}^{K}\mathsf{pk}_i^{-\beta_{i,\vec{J}_i}} h'_{i,\vec{J}_i}{}^{\tau_i})^{-d}\bar{S}'^{-\gamma_0}\prod_{i=1}^{K} h'_{i,\vec{J}_i}{}^{\vec{z}_{0,i}+\vec{\alpha}_{0,i}+d\cdot\vec{\tau}_i} \\
&= \bar{S}^{-d}\prod_{i=1}^{K}\mathsf{pk}_i^{d\beta_{i,\vec{J}_i}} h'_{i,\vec{J}_i}{}^{\vec{z}_{0,i}}\bar{S}'^{-\gamma_0}\prod_{i=1}^{K} h'_{i,\vec{J}_i}{}^{\vec{\alpha}_{0,i}} \\
&= \bar{S}^{-d}\prod_{i=1}^{K}\mathsf{pk}_i^{d\beta_{i,\vec{J}_i}}(h_{i,\vec{J}_i} g^{-\beta_{i,\vec{J}_i}})^{\vec{z}_{0,i}}\bar{S}'^{-\gamma_0}\prod_{i=1}^{K} h'_{i,\vec{J}_i}{}^{\vec{\alpha}_{0,i}} \\
&= \bar{S}^{-d}\prod_{i=1}^{K} h_{i,\vec{J}_i}^{\vec{z}_{0,i}}(\mathsf{pk}_i^{-d} g^{\vec{z}_{0,i}})^{-\beta_{i,\vec{J}_i}}\bar{S}'^{-\gamma_0}\prod_{i=1}^{K} h'_{i,\vec{J}_i}{}^{\vec{\alpha}_{0,i}} \\
&= \bar{R}\bar{S}'^{-\gamma_0}\prod_{i=1}^{K}\vec{R}_i^{-\beta_{i,\vec{J}_i}} h'_{i,\vec{J}_i}{}^{\vec{\alpha}_{0,i}} = \bar{R}'
\end{aligned}$$

- $g^{z'_1}W^{-e'} = g^{z_1+\alpha_1}W^{-e-\gamma_1} = Ag^{\alpha_1}W^{-\gamma_1} = A'$

Thus, the verification algorithm returns 1, because $\prod_{i=1}^{K}\mathsf{pk}'_i = \mathsf{pk}$ and

$$\begin{aligned}
d' + e' = c' &= \mathsf{H}'(m, (h'_{i,\vec{J}_i}, \mathsf{pk}'_i)_{i\in[K]}, \bar{S}', \vec{R}', \bar{R}', A') \\
&= \mathsf{H}'(m, (h'_{i,\vec{J}_i}, \mathsf{pk}'_i)_{i\in[K]}, \bar{S}', (g^{\vec{z}'_{0,i}}\mathsf{pk}_i'^{-d'})_{i\in[K]}, \bar{S}'^{-d'}\prod_{i=1}^{K} h'_{i,\vec{J}_i}{}^{\vec{z}'_{0,i}}, g^{z'_1}W^{-e'})\ .
\end{aligned}$$

□

5.3 Proof of Theorem 5.1

To show blindness of BS_3, we consider the following sequence of games.

Game $\mathbf{G}_0^{\mathcal{A}}$: This game is identical to the game BLIND of BS_3 where $\mathcal{A}$ makes at most $Q_{\mathsf{H}_\star}$ queries to the random oracles $\mathsf{H}_\star \in \{\mathsf{H}_\Pi, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}\}$. For $k \in \{0,1\}$, we denote the superscript $(\cdot)^{(k)}$ as the corresponding value in the user oracles $\mathrm{U}_j(k,\cdot), j = 1,2,3$. (The superscript notation is chosen for readability of the proof as the scheme BS_3 contains many values with subscripts.)

Game $\mathbf{G}_1^{\mathcal{A}}$: This game adds an abort in the oracle $\mathrm{U}_2(k,\cdot)$ such that on the signer message $((\mathsf{pk}_i^{(k)})_{i\in[K-1]}, \bar{S}^{(k)}, \vec{R}^{(k)}, \bar{R}^{(k)}, A^{(k)}, \pi^{(k)})$, the oracle computes $\mathsf{pk}_K^{(k)} \leftarrow \mathsf{pk}\cdot\prod_{i=1}^{K-1}\mathsf{pk}_i^{(k)^{-1}}$ and the game aborts if the proof $\pi^{(k)}$ verifies, but $\bar{S}^{(k)} \neq \prod_{i=1}^{K}(h_{i,\vec{J}_i^{(k)}}^{(k)})^{\log_g \mathsf{pk}_i^{(k)}}$. Notice that the view of $\mathcal{A}$ only changes when the abort occurs, i.e., the event where $\mathcal{A}$ queries $\mathrm{U}_2(k,\cdot)$ for $k \in \{0,1\}$ with a valid proof $\pi^{(k)}$ for a statement $(g, (h_{i,\vec{J}_i^{(k)}}^{(k)}, \mathsf{pk}_i^{(k)})_{i\in[K]}, \bar{S}^{(k)})$ with $\bar{S}^{(k)} \neq \prod_{i=1}^{K}(h_{i,\vec{J}_i^{(k)}}^{(k)})^{\log_g \mathsf{pk}_i^{(k)}}$. This corresponds to breaking the soundness property of Π. By Lemma 5.3, any adversary with Q_{H_Π}-query access to H_Π breaks the soundness of Π only with probability Q_{H_Π}/p. Thus, bounding over both signing sessions $k \in \{0,1\}$, we have

$$|\Pr[\mathbf{G}_0^{\mathcal{A}} = 1] - \Pr[\mathbf{G}_1^{\mathcal{A}} = 1]| \leqslant \frac{2Q_{\mathsf{H}_\Pi}}{p}\ .$$

Game $\mathbf{G}_2^{\mathcal{A}}$: This game adds another abort such that for all $k \in \{0,1\}, i \in [K]$, and $j \in [N]\backslash\{\vec{J}_i^{(k)}\}$, if $\mathsf{H}_\mu(\cdot, \varphi_{i,j}^{(k)})$ has been queried by $\mathcal{A}$ at any point throughout the game, the game aborts. Since $\varphi_{i,j}^{(k)}$ for $j \neq \vec{J}_i^{(k)}$ is uniformly random from $\{0,1\}^\lambda$ and hidden from the view of $\mathcal{A}$ throughout the game,

$$|\Pr[\mathbf{G}_1^{\mathcal{A}} = 1] - \Pr[\mathbf{G}_2^{\mathcal{A}} = 1]| \leqslant \frac{2KNQ_{\mathsf{H}_\mu}}{2^\lambda}\ .$$

Game $\mathbf{G}_3^{\mathcal{A}}$: This game adds another abort such that for all $k \in \{0,1\}, i \in [K]$, if $\mathsf{H}_\beta(\varepsilon^{(k)}_{i,\vec{J}_i^{(k)}})$ or $\mathsf{H}_{\mathsf{com}}(\cdot, \varepsilon^{(k)}_{i,\vec{J}_i^{(k)}})$ has been queried by $\mathcal{A}$ at any point throughout the game, the game aborts. Since $\varepsilon^{(k)}_{i,\vec{J}_i^{(k)}}$ is uniformly random from $\{0,1\}^\lambda$ and hidden from the view of $\mathcal{A}$ throughout the game,

$$|\mathsf{Pr}[\mathbf{G}_2^{\mathcal{A}} = 1] - \mathsf{Pr}[\mathbf{G}_3^{\mathcal{A}} = 1]| \leqslant \frac{2KQ_{\mathsf{H}_\beta}}{2^\lambda} + \frac{2KQ_{\mathsf{H}_{\mathsf{com}}}}{2^\lambda} \ .$$

Game $\mathbf{G}_4^{\mathcal{A}}$: In this game, the game samples $\hat{\vec{J}}^{(k)} \leftarrow_\$ [N]^K$ for both $k \in \{0,1\}$ at the start of the game and aborts if $\hat{\vec{J}}^{(k)} \neq \vec{J}^{(k)}$ later in the game. The view of $\mathcal{A}$ does not change unless the game aborts, so conditioning on the event that the game does not abort, we have

$$\mathsf{Pr}[\mathbf{G}_4^{\mathcal{A}} = 1] = \frac{1}{N^{2K}} \mathsf{Pr}[\mathbf{G}_3^{\mathcal{A}} = 1] \ .$$

Game $\mathbf{G}_5^{\mathcal{A}}$: This game changes how $\mu^{(k)}_{i,j}$ is computed for $k \in \{0,1\}, i \in [K], j \in [N] \backslash \{\hat{\vec{J}}_i^{(k)}\}$. Previously, it was defined as $\mathsf{H}_\mu(m_{b_k}, \varphi^{(k)}_{i,j})$, however, now it is only uniformly sampled from $\{0,1\}^\lambda$. By the changes in games $\mathbf{G}_2^{\mathcal{A}}$ and $\mathbf{G}_4^{\mathcal{A}}$, $\hat{\vec{J}}_i^{(k)} = \vec{J}_i^{(k)}$ and $\mathsf{H}_\mu(\cdot, \varphi^{(k)}_{i,j})$ is never queried by $\mathcal{A}$. Therefore, since $\mu^{(k)}_{i,j}$ is distributed identically as before,

$$\mathsf{Pr}[\mathbf{G}_5^{\mathcal{A}} = 1] = \mathsf{Pr}[\mathbf{G}_4^{\mathcal{A}} = 1] \ .$$

Game $\mathbf{G}_6^{\mathcal{A}}$: This game changes how $\beta^{(k)}_{i,\vec{J}_i^{(k)}}$ and $\mathsf{com}^{(k)}_{i,\vec{J}_i^{(k)}}$ are computed for $k \in \{0,1\}, i \in [K]$. Previously, it was defined as $\mathsf{H}_\beta(\varepsilon^{(k)}_{i,\vec{J}_i^{(k)}})$ and $\mathsf{H}_{\mathsf{com}}(\mathsf{r}^{(k)}_{i,\vec{J}_i^{(k)}})$; however, now it is only uniformly sampled from $\mathbb{Z}_p$ and $\{0,1\}^\lambda$ respectively. By the changes in games $\mathbf{G}_3^{\mathcal{A}}$ and $\mathbf{G}_4^{\mathcal{A}}$, $\hat{\vec{J}}_i^{(k)} = \vec{J}_i^{(k)}$ and $\mathsf{H}_\beta(\varepsilon^{(k)}_{i,\vec{J}_i^{(k)}})$ nor $\mathsf{H}_{\mathsf{com}}(\cdot, \varepsilon^{(k)}_{i,\vec{J}_i^{(k)}})$ has been queried by $\mathcal{A}$. Since $\beta^{(k)}_{i,\vec{J}_i^{(k)}}$ and $\mathsf{com}^{(k)}_{i,\vec{J}_i^{(k)}}$ are distributed identically as before,

$$\mathsf{Pr}[\mathbf{G}_6^{\mathcal{A}} = 1] = \mathsf{Pr}[\mathbf{G}_5^{\mathcal{A}} = 1] \ .$$

Lastly, we claim (in the lemma below) that when $\mathbf{G}_6^{\mathcal{A}}$ does not abort, the view of $\mathcal{A}$ is identical for both cases $b = 0$ and $b = 1$. This results in $\mathsf{Pr}[\mathbf{G}_6^{\mathcal{A}} = 1] = 1/(2N^{2K})$ (as there is only $1/N^{2K}$ chance of the game not aborting from the change in $\mathbf{G}_6^{\mathcal{A}}$). By combining the advantage changes

$$|\mathsf{Pr}[\mathrm{BLIND}^{\mathcal{A}}_{\mathsf{BS}_3}(\lambda) = 1] - \frac{1}{2}| \leqslant \frac{2Q_{\mathsf{H}_\Pi}}{p} + \frac{2KNQ_{\mathsf{H}_\mu}}{2^\lambda} + \frac{2KQ_{\mathsf{H}_\beta}}{2^\lambda} + \frac{2KQ_{\mathsf{H}_{\mathsf{com}}}}{2^\lambda} \ ,$$

concluding the proof.

Lemma 5.5. *In $\mathbf{G}_6^{\mathcal{A}}$, if the game does not abort, the view of $\mathcal{A}$ is identical between both cases of $b = 0$ and $b = 1$.*

Proof. To show this, first, assume w.l.o.g. that the randomness of $\mathcal{A}$ is fixed and that $\mathcal{A}$ only outputs messages in the transcript where neither the game nor the user oracles abort which makes $\mathcal{A}$ receives valid signatures (σ_0, σ_1). Also, let View_A denote the set of all possible views of $\mathcal{A}$ that can occur in the game $\mathbf{G}_6^{\mathcal{A}}$. A view $\Delta \in \mathrm{View}_A$ is of the form

$$\Delta = (W, X, m_0, m_1, T_0, T_1, \sigma_0, \sigma_1) \ ,$$

where for $k \in \{0,1\}$: T_k denotes the transcript of the interaction between $\mathcal{A}$ and the user oracle in signing session k and σ_k denotes the valid signature for message m_k. They are of the form:

$$T_k = ((h^{(k)}_{i,\vec{J}_i^{(k)}}, \mathsf{pk}_i^{(k)})_{i \in [K]}, \bar{S}^{(k)}, \vec{R}^{(k)}, \bar{R}^{(k)}, A^{(k)}, c^{(k)}, d^{(k)}, e^{(k)}, \vec{z}_0^{(k)}, z_1^{(k)}),$$

$$\sigma_k = ((\mathsf{pk}'^{(k)}_i, \varphi'^{(k)}_i)_{i \in [K]}, \bar{S}'^{(k)}, d'^{(k)}, e'^{(k)}, \vec{z}'^{(k)}_0, z'^{(k)}_1) \ .$$

Note that we omit $\pi^{(k)}$ as it is distributed independently of (m_0, m_1) given $((h^{(k)}_{i,\vec{J}^{(k)}_i}, \mathsf{pk}^{(k)}_i)_{i\in[K]}, \bar{S}^{(k)})$, and also omit the $(\vec{J}^{(k)}, ((\mathsf{r}^{(k)}_{i,j})_{j\neq\vec{J}^{(k)}_i}, \mathsf{com}^{(k)}_{i,\vec{J}^{(k)}_i})_{i\in[K]})$ portion of $\mathsf{umsg}^{(k)}_1$ because they are now independent of the messages (m_0, m_1) by the changes introduced to the games $\mathbf{G}^{\mathcal{A}}_1 - \mathbf{G}^{\mathcal{A}}_6$. Also, we rename some variables from the signing protocol as follows,

$$\begin{gathered}\beta^{(k)}_i = \beta^{(k)}_{i,\vec{J}^{(k)}_i},\ \ \varphi^{(k)}_i = \varphi^{(k)}_{i,\vec{J}^{(k)}_i},\ \ \mu^{(k)}_i = \mu^{(k)}_{i,\vec{J}^{(k)}_i} = \mathsf{H}_\mu(m_{b_k}, \varphi^{(k)}_i),\\ h^{(k)}_i = {h_{i,\vec{J}^{(k)}_i}}^{(k)},\ \ {h'_i}^{(k)} = {h'_{i,\vec{J}^{(k)}_i}}^{(k)} = \mathsf{H}(\mu^{(k)}_i)\ .\end{gathered} \tag{11}$$

We need to show that the distribution of the actual adversarial view, which we denote as v_A, is the same between $b = 0$ and $b = 1$. Since we fix the randomness of $\mathcal{A}$, v_A only depends on the randomness of the user algorithm which we denote

$$\eta = ((\beta^{(k)}_i, \varphi^{(k)}_i)_{i\in[K]}, \vec{\tau}^{(k)}, \vec{\alpha}^{(k)}_0, \alpha^{(k)}_1, \gamma^{(k)}_0, \gamma^{(k)}_1)_{k\in\{0,1\}}$$

and write $v_A(\eta)$ to make this explicit.

Before continuing, we note that because of the change in $\mathbf{G}^{\mathcal{A}}_1$ any non-aborting transcript should have

$$\left.\begin{aligned}\bar{S}^{(k)} &= \textstyle\prod_{i=1}^K \left(h^{(k)}_i\right)^{\mathsf{sk}^{(k)}_i} \text{ which induces}\\ \bar{S'}^{(b_k)} &= \bar{S}^{(k)} \textstyle\prod_{i=1}^K (\mathsf{pk}^{(k)}_i)^{-\beta^{(k)}_i} ({h'_i}^{(k)})^{\vec{\tau}^{(k)}_i}\\ &= \textstyle\prod_{i=1}^K (h^{(k)}_i g^{-\beta^{(k)}_i})^{\mathsf{sk}^{(k)}_i} ({h'_i}^{(k)})^{\vec{\tau}^{(k)}_i}\\ &= \textstyle\prod_{i=1}^K ({h'_i}^{(k)})^{\mathsf{sk}^{(k)}_i + \vec{\tau}^{(k)}_i}\ .\end{aligned}\right\} \tag{12}$$

To show that the distribution of v_A is identical between $b = 0$ and $b = 1$, consider a view $\Delta \in \mathrm{View}_A$. We now show that there exists a unique η such that $v_A(\eta) = \Delta$, regardless of whether $b = 0$ or $b = 1$. More specifically, we claim that for both $b = 0$ and $b = 1$, $v_A(\eta) = \Delta$ if and only if for $i \in \{0, 1\}$, η satisfies

$$\begin{aligned}&\left.\begin{aligned}\varphi^{(k)}_i &= {\varphi'_i}^{(b_k)}\\ \beta^{(k)}_i &= \log_g h^{(k)}_i - \log_g {h'_i}^{(k)}\\ \vec{\tau}^{(k)}_i &= \log_g {\mathsf{pk}'_i}^{(b_k)} - \log_g \mathsf{pk}^{(k)}_i\end{aligned}\right\} \text{ for } i \in [K],\\ &\vec{\alpha}^{(k)}_0 = \vec{z'}^{(b_k)}_0 - \vec{z}^{(k)}_0 - d^{(k)} \cdot \vec{\tau}^{(k)},\ \ \alpha^{(k)}_1 = {z'}^{(b_k)}_1 - z^{(k)}_1,\\ &\gamma^{(k)}_0 = {d'}^{(b_k)} - d^{(k)},\ \ \gamma^{(k)}_1 = {e'}^{(b_k)} - e^{(k)}\ .\end{aligned} \tag{13}$$

For the "only if" direction, i.e., if $v_A(\eta) = \Delta$, then η satisfies Equation (13), this is true by how the user algorithm of BS_3 is defined.

To show the "if" direction, suppose η satisfies Equation (13), we need to show that $v_A(\eta) = \Delta$. Particularly, we have to show that the user messages from oracles $\mathrm{U}_1, \mathrm{U}_2$ and the signatures from oracle U_3 are $((h^{(0)}_i)_{i\in[K]}, (h^{(1)}_i)_{i\in[K]})$, $(c^{(0)}, c^{(1)})$, and (σ_0, σ_1) respectively.

Again, since we only consider non-aborting transcript Δ, we have the following guarantees for $i \in \{0, 1\}$:

$$\left.\begin{aligned}\vec{R}^{(k)}_i &= (\mathsf{pk}^{(k)}_i)^{-d^{(k)}} g^{\vec{z}^{(k)}_{0,i}} \text{ for } i \in [K],\\ \bar{R}^{(k)} &= (\bar{S}^{(k)})^{-d^{(k)}} \textstyle\prod_{i=1}^K {h^{(k)}_i}^{\vec{z}^{(k)}_{0,i}},\\ A^{(k)} &= W^{-e^{(k)}} g^{z^{(k)}_1},\quad c^{(k)} = d^{(k)} + e^{(k)},\end{aligned}\right\} \tag{14}$$

$$\begin{aligned}{d'}^{(b_k)} + {e'}^{(b_k)} = \mathsf{H}'(m_{b_k}, ({h'_i}^{(k)}, {\mathsf{pk}'_i}^{(b_k)})_{i\in[K]}, \bar{S'}^{(b_k)}, (g^{\vec{z}^{(b_k)}_{0,i}} {{\mathsf{pk}'_i}^{(b_k)}}^{-{d'}^{(b_k)}})_{i\in[K]},\\ (\bar{S'}^{(k)})^{-{d'}^{(k)}} \prod_{i=1}^K {{h'_i}^{(k)}}^{\vec{z'}^{(k)}_{0,i}}, W^{-{e'}^{(b_k)}} g^{{z'_1}^{(b_k)}})\ .\end{aligned} \tag{15}$$

where Equation (14) follows from the checks in $\mathsf{BS}_3.\mathsf{U}_3$, and Equation (15) follows from the validity of the signatures.

First, we argue that the first user message $h_i^{(k)}$ of both signing sessions corresponds to the values in Δ. This is due to

$$h_i^{(k)} = {h'}_i^{(k)} g^{\beta_i^{(k)}} = \mathsf{H}(\mu_i^{(k)}) g^{\beta_i^{(k)}} = \mathsf{H}(\mathsf{H}_\mu(m_{b_k}, \varphi_i^{(k)})) g^{\beta_i^{(k)}} \ .$$

The first equality is from the value of $\beta_i^{(k)}$ in Equation (13). The other equalities follow from how we renamed the values in Equation (11). The right-hand side of the equation is exactly the value in $\mathsf{umsg}_1^{(k)}$. Thus, the next message from $\mathcal{A}$ will be $((\mathsf{pk}_i^{(k)})_{i\in[K]}, \bar{S}^{(k)}, \vec{R}^{(k)}, \bar{R}^{(k)}, A^{(k)})$.

Next, we argue that the second user message from $\mathrm{U}_2(k,\cdot)$ will be $c^{(k)}$. To do this, we consider the blinded values of $(\mathsf{pk}_i^{(k)}, R_i^{(k)})_{i\in[K]}, \bar{S}^{(k)}, \bar{R}^{(k)}, A^{(k)}$ which will be the input to H' to compute $c^{(k)}$.

$$\begin{aligned}
\mathsf{pk}_i^{(k)} g^{\vec{\tau}_i^{(k)}} &= {\mathsf{pk}'}_i^{(b_k)} \text{ for } i \in [K], \text{ By } \vec{\tau}^{(k)} \text{ in Equation (13)} \\
{\bar{S}'}^{(b_k)} &= \bar{S}^{(k)} \prod_{i=1}^{K} (\mathsf{pk}_i^{(k)})^{-\beta_i^{(k)}} ({h'}_i^{(k)})^{\vec{\tau}_i^{(k)}}, \text{ By Equation (12)} \\
{\vec{R}'}_i^{(k)} &= \vec{R}_i^{(k)} {{\mathsf{pk}'}_i^{(b_k)}}^{-\gamma_0^{(k)}} g^{\vec{\alpha}_{0,i}^{(k)}} \\
&= (\mathsf{pk}_i^{(k)})^{-d^{(k)}} g^{\vec{z}_{0,i}^{(k)}} {{\mathsf{pk}'}_i^{(b_k)}}^{-\gamma_0^{(k)}} g^{\vec{\alpha}_{0,i}^{(k)}} \text{ By Equation (14)} \\
&= ({\mathsf{pk}'}_i^{(b_k)})^{-d^{(k)}} g^{\vec{z}_{0,i}^{(k)} + \alpha_{0,i}^{(k)} + d^{(k)} \vec{\tau}_i^{(k)}} {{\mathsf{pk}'}_i^{(b_k)}}^{-\gamma_0^{(k)}} \\
&= g^{\vec{z'}_{0,i}^{(b_k)}} {{\mathsf{pk}'}_i^{(b_k)}}^{-{d'}^{(b_k)}}, \text{ By } \vec{\alpha}_0^{(k)} \text{ in Equation (13)} \\
{A'}^{(k)} &= A^{(k)} W^{-\gamma_1^{(k)}} g^{\alpha_1^{(k)}} \\
&= (W^{-e^{(k)}} g^{z_1^{(k)}}) W^{-\gamma_1^{(k)}} g^{\alpha_1^{(k)}}, \text{ By Equation (14)} \\
&= W^{-{e'}^{(b_k)}} g^{{z'_1}^{(b_k)}}, \text{ By } \alpha_1^{(k)} \text{ in Equation (13)} \\
{\bar{R}'}^{(k)} &= \bar{R}^{(k)} ({\bar{S}'}^{(b_k)})^{-\gamma_0^{(k)}} \prod_{i=1}^{K} (\vec{R}_i^{(k)})^{-\beta_i^{(k)}} ({h'}_i^{(k)})^{\vec{\alpha}_{0,i}} \\
&= \left((\bar{S}^{(k)})^{-d^{(k)}} \prod_{i=1}^{K} {h_i^{(k)}}^{\vec{z}_{0,i}^{(k)}} \right) ({\bar{S}'}^{(b_k)})^{-\gamma_0^{(k)}} \prod_{i=1}^{K} (\vec{R}_i^{(k)})^{-\beta_i^{(k)}} ({h'}_i^{(k)})^{\vec{\alpha}_{0,i}^{(k)}} \\
&= \left({\bar{S}'}^{(b_k)} \prod_{i=1}^{K} (\mathsf{pk}_i^{(k)})^{\beta_i^{(k)}} ({h'}_i^{(k)})^{-\vec{\tau}_i^{(k)}} \right)^{-d^{(k)}} \prod_{i=1}^{K} {h_i^{(k)}}^{\vec{z}_{0,i}^{(k)}} \\
&\qquad ({\bar{S}'}^{(b_k)})^{-\gamma_0^{(k)}} \prod_{i=1}^{K} (\vec{R}_i^{(k)})^{-\beta_i^{(k)}} ({h'}_i^{(k)})^{\vec{\alpha}_{0,i}^{(k)}} \\
&= ({\bar{S}'}^{(b_k)})^{-\gamma_0^{(k)}} \left({\bar{S}'}^{(b_k)} \prod_{i=1}^{K} (\mathsf{pk}_i^{(k)})^{\beta_i^{(k)}} ({h'}_i^{(k)})^{-\vec{\tau}_i^{(k)}} \right)^{-d^{(k)}} \\
&\qquad \prod_{i=1}^{K} (\vec{R}_i^{(k)})^{-\beta_i^{(k)}} ({h'}_i^{(k)})^{\vec{\alpha}_{0,i}^{(k)}} ({h'}_i^{(k)} g^{\beta_i^{(k)}})^{\vec{z}_{0,i}^{(k)}} \\
&= ({\bar{S}'}^{(b_k)})^{-{d'}^{(b_k)}} \prod_{i=1}^{K} (\vec{R}_i^{(k)} (\mathsf{pk}_i^{(k)})^{d^{(k)}} g^{-\vec{z}_{0,i}^{(k)}})^{-\beta_i^{(k)}} ({h'}_i^{(k)})^{\vec{z'}_{0,i}^{(b_k)}} \\
&= ({\bar{S}'}^{(k)})^{-{d'}^{(b_k)}} \prod_{i=1}^{K} {{h'}_i^{(k)}}^{\vec{z'}_{0,i}^{(b_k)}} \ .
\end{aligned}$$

For the value of $\bar{R}'^{(k)}$: the first equality follows from how the value is defined; the second equality follows from Equation (14); the third equality follows from Equation (12); the fourth equality follows from rearranging the terms and $h_i^{(k)} = h_i'^{(k)} g^{\beta_i^{(k)}}$; the fifth equality follows from rearranging the terms and the values of $\gamma_0^{(k)}, \vec{\alpha}_0^{(k)}$ in Equation (13); and the last equality follows from the value of $\vec{R}_i^{(k)}$ in Equation (14). With these equalities, we have

$$\begin{aligned}
&\mathsf{H}'(m_{b_k}, (h_i'^{b_k}, \mathsf{pk}_i'^{(b_k)})_{i\in[K]}, \bar{S}'^{(b_k)}, \vec{R}'^{(k)}, \bar{R}'^{(k)}, A'^{(k)}) - \gamma_0^{(k)} - \gamma_1^{(k)} \\
&= d'^{(b_k)} + e'^{(b_k)} - \gamma_0^{(k)} - \gamma_1^{(k)} \\
&= d^{(k)} + e^{(k)} = c^{(k)},
\end{aligned}$$

where the first equality follows from Equation (15), the second to last equality follows from the values of $\gamma_0^{(k)}, \gamma_1^{(k)}$ in Equation (13), and the last equality follows from Equation (14). Thus, the next message from $\mathcal{A}$ will be $d^{(k)}, e^{(k)}, \vec{z}_0^{(k)}, z_1^{(k)}$ from the transcript $\varDelta$. Lastly, the final signatures output by the oracle U_3 will be σ_0, σ_1 by how the randomness η is defined in Equation (13). □

5.4 Proof of Theorem 5.2 (OMUF of BS_3)

Let $\mathcal{A}$ be an adversary playing one-more unforgeability game of BS_3. We consider the following sequence of games described in text, while the pseudocode version of the games can be found in Figure 12.

Game $\mathbf{G}_0^{\mathcal{A}}$: The game first generates the parameters $\mathsf{par} \xleftarrow{\$} \mathsf{BS}_3.\mathsf{Setup}(1^\lambda, N, K)$, $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{BS}_3.\mathsf{KG}(\mathsf{par})$. Then, the game interacts with an adversary $\mathcal{A}(\mathsf{par}, \mathsf{pk})$ with access to signing oracles $\mathrm{S}_1, \mathrm{S}_2$ and the random oracles $\mathsf{H}, \mathsf{H}', \mathsf{H}_\Pi, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}$ each simulated by lazy sampling. The adversary $\mathcal{A}$ (w.l.o.g.) queries the signing oracle S_1 for ℓ times and the random oracles $\mathsf{H}_\star$ for $Q_{\mathsf{H}_\star}$ times for $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}', \mathsf{H}_\Pi, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}\}$. At the end of the game, $\mathcal{A}$ outputs $\ell + 1$ message-signature pairs (m_k^*, σ_k^*) for $k \in [\ell + 1]$. The adversary $\mathcal{A}$ succeeds if for all $k_1 \neq k_2, m_{k_1}^* \neq m_{k_2}^*$ and for all $k \in [\ell + 1], \mathsf{BS}_3.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m_k^*, \sigma_k^*) = 1$. We also assume w.l.o.g. that $\mathcal{A}$ does not make the same random oracle query twice and already makes the random oracle queries that would be called in $\mathsf{BS}_3.\mathsf{Ver}$ when the game checks the forgeries. The probability of $\mathcal{A}$ winning in game $\mathbf{G}_0^{\mathcal{A}}$ is exactly its advantage in OMUF i.e.

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{BS}_3}(\mathcal{A}, \lambda) = \Pr[\mathbf{G}_0^{\mathcal{A}} = 1] \,.$$

Game $\mathbf{G}_1^{\mathcal{A}}$: This game is identical to $\mathbf{G}_0^{\mathcal{A}}$ except that queries of the form $\mu \in \{0,1\}^\lambda$ to H is answered by introducing a map $t[\mu] \in \mathbb{Z}_p$ where the oracle simulation first outputs $\mathsf{H}(\mu)$ if it is already defined, otherwise it samples $t[\mu] \xleftarrow{\$} \mathbb{Z}_p$ and sets $\mathsf{H}(\mu) \leftarrow g^{t[\mu]}$. Since the distribution of $\mathsf{H}(\mu)$ remains identical, the success probability of $\mathcal{A}$ remains as in $\mathbf{G}_0^{\mathcal{A}}$ as its view is identical in both games.

$$\Pr[\mathbf{G}_1^{\mathcal{A}} = 1] = \Pr[\mathbf{G}_0^{\mathcal{A}} = 1] \,.$$

Game $\mathbf{G}_2^{\mathcal{A}}$: This game is identical to $\mathbf{G}_1^{\mathcal{A}}$ except that it introduces the following check: namely, when $\mathcal{A}$ outputs forgeries $(m_k^*, \sigma_k^*)_{k\in[\ell+1]}$, where for each $k \in [\ell + 1]$, $\sigma_k^* = ((\mathsf{pk}_{k,i}^*, \varphi_{k,i}^*)_{i\in[K]}, \bar{S}_k^*, d_k^*, e_k^*, \vec{z}_{k,0}^*, z_{k,1}^*)$, the game checks that

$$\bar{S}_k^* = \prod_{i=1}^{K} \mathsf{H}(\mu_{k,i}^*)^{\mathsf{sk}_{k,i}^*} = \prod_{i=1}^{K} g^{\mathsf{sk}_{k,i}^* t[\mu_{k,i}^*]} = \prod_{i=1}^{K} \mathsf{pk}_{k,i}^{*\;t[\mu_{k,i}^*]} \,,$$

with $\mu_{k,i}^* = \mathsf{H}_\mu(m_k^*, \varphi_{k,i}^*), \mathsf{sk}_{k,i}^* = \log_g \mathsf{pk}_{k,i}^*$. If this check does not pass, the game aborts. Note that the check here can be done efficiently by the change in $\mathbf{G}_1^{\mathcal{A}}$ since all the values in the rightmost side of the equation are known to the game. (This fact will be used in the proof of Lemma 5.7.) By Lemma 5.7, there exists an adversary $\mathcal{B}$ playing the DLOG game with running time $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$ such that

$$\Pr[\mathbf{G}_2^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_1^{\mathcal{A}} = 1] - (\ell + 1)\left(\sqrt{Q_{\mathsf{H}'}\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p}\right) \,.$$

```
Game $\mathbf{G}_0^{\mathcal{A}}, \mathbf{G}_1^{\mathcal{A}}, \mathbf{G}_2^{\mathcal{A}}, \boxed{\mathbf{G}_3^{\mathcal{A}}}, \mathbf{G}_4^{\mathcal{A}}, \mathbf{G}_5^{\mathcal{A}}$:
$(\mathbb{G}, p, g) \leftarrow\$ \mathsf{GGen}(1^\lambda)$
$W \leftarrow\$ \mathbb{G}$                                   // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_2^{\mathcal{A}}$
$w \leftarrow\$ \mathbb{Z}_p$ ; $W \leftarrow g^w$                        // $\mathbf{G}_3^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
$\mathsf{par} \leftarrow (\mathbb{G}, p, g, W)$
$x \leftarrow\$ \mathbb{Z}_p$ ; $X \leftarrow g^x$
$\mathsf{sk} \leftarrow x$ ; $\mathsf{pk} \leftarrow X$
$\ell \leftarrow 0$ ; $\mathcal{I}_1, \mathcal{I}_2 \leftarrow \varnothing$
$\{(m_k^*, \sigma_k^*)\}_{k \in [\ell+1]} \leftarrow\$ \mathcal{A}^{\mathrm{S}_1, \mathrm{S}_2}(\mathsf{par}, \mathsf{pk})$
If $\exists\ k_1 \neq k_2, m_{k_1}^* = m_{k_2}^*$ then
   return 0
If $\exists\ k \in [\ell+1]$ such that
   $\mathsf{BS}_3.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m_k^*, \sigma_k^*) = 0$ then
      return 0
For $k \in [\ell+1]$:
   $((\mathsf{pk}_{k,i}^*, \varphi_{k,i}^*)_{i \in [K]}, \bar{S}_k^*, d_k^*, e_k^*, \vec{z}_{k,0}^*, z_{k,1}^*) \leftarrow \sigma_k^*$
   For $i \in [K] : \mu_{k,i}^* \leftarrow \mathsf{H}_\mu(m_k^*, \varphi_{k,i}^*)$
   If $\bar{S}_k^* \neq \prod_{i=1}^K {\mathsf{pk}_{k,i}^*}^{t[\mu_{k,i}^*]}$
      then return 0                                 // $\mathbf{G}_2^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
Return 1

Oracle $\mathsf{H}(\mu)$:
If $\mathsf{H}(\mu) \neq \bot$ then
   return $\mathsf{H}(\mu)$
$\boxed{\mathsf{H}(\mu) \leftarrow\$ \mathbb{G}}$                                 // $\mathbf{G}_0^{\mathcal{A}}$
$t[\mu] \leftarrow\$ \mathbb{Z}_p$
$\mathsf{H}(\mu) \leftarrow g^{t[\mu]}$                                // $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
Return $\mathsf{H}(m)$

Oracle $\mathrm{S}_2(\mathrm{sid}, c_{\mathrm{sid}})$ :
If $\mathrm{sid} \notin \mathcal{I}_1$ or $\mathrm{sid} \in \mathcal{I}_2$ then return $\bot$
$\mathcal{I}_2 \leftarrow \mathcal{I}_2 \cup \{\mathrm{sid}\}$
$d_{\mathrm{sid}} \leftarrow c_{\mathrm{sid}} - e_{\mathrm{sid}}$                          // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
For $i \in [K] : \vec{z}_{0,\mathrm{sid},i} \leftarrow \vec{r}_{0,\mathrm{sid},i} + d_{\mathrm{sid}} \cdot \mathsf{sk}_{i,\mathrm{sid}}$
$e_{\mathrm{sid}} \leftarrow c_{\mathrm{sid}} - d_{\mathrm{sid}}$
$z_{1,\mathrm{sid}} \leftarrow r_{1,\mathrm{sid}} + e_{\mathrm{sid}} \cdot w$                           // $\mathbf{G}_5^{\mathcal{A}}$
Return $(d_{\mathrm{sid}}, e_{\mathrm{sid}}, \vec{z}_{0,\mathrm{sid}}, z_{1,\mathrm{sid}})$
```

```
Oracle $\mathrm{S}_1(\mathrm{sid}, h_{\mathrm{sid}})$:
If $\mathrm{sid} \in \mathcal{I}_1$ then return $\bot$
$\ell \leftarrow \ell + 1$ ; $\mathcal{I}_1 \leftarrow \mathcal{I}_1 \cup \{\mathrm{sid}\}$
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{umsg}_1$
If $\mathsf{Check}(\mathsf{umsg}_1) = 0$ then return $\bot$
For $i \in [K-1] : \mathsf{sk}_{i,\mathrm{sid}} \leftarrow\$ \mathbb{Z}_p$
$\mathsf{sk}_{K,\mathrm{sid}} \leftarrow \mathsf{sk} - \sum_{i=1}^{K-1} \mathsf{sk}_{i,\mathrm{sid}}$
For $i \in [K] : \mathsf{pk}_{i,\mathrm{sid}} \leftarrow g^{\mathsf{sk}_{i,\mathrm{sid}}}$
$\bar{S}_{\mathrm{sid}} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_{i,\mathrm{sid}}}$
$z_{1,\mathrm{sid}}, e_{\mathrm{sid}} \leftarrow\$ \mathbb{Z}_p, \vec{r}_{0,\mathrm{sid}} \leftarrow\$ \mathbb{Z}_p^K$
For $i \in [K] : \vec{R}_{i,\mathrm{sid}} \leftarrow g^{\vec{r}_{0,\mathrm{sid},i}}$
$\bar{R}_{\mathrm{sid}} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{r}_{0,\mathrm{sid},i}}$
$A_{\mathrm{sid}} \leftarrow g^{z_{1,\mathrm{sid}}} W^{-e_{\mathrm{sid}}}$  // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$d_{\mathrm{sid}}, r_{1,\mathrm{sid}} \leftarrow\$ \mathbb{Z}_p, \vec{z}_{0,\mathrm{sid}} \leftarrow\$ \mathbb{Z}_p^K$
For $i \in [K] : \vec{R}_{i,\mathrm{sid}} \leftarrow \mathsf{pk}_{i,\mathrm{sid}}^{-d_{\mathrm{sid}}} g^{\vec{z}_{0,\mathrm{sid},i}}$
$\bar{R} \leftarrow \bar{S}^{-d_{\mathrm{sid}}} \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{z}_{0,\mathrm{sid},i}}$
$A_{\mathrm{sid}} \leftarrow g^{r_{1,\mathrm{sid}}}$                                  // $\mathbf{G}_5^{\mathcal{A}}$
$\pi \leftarrow \mathit{\Pi}.\mathsf{Prove}^{\mathsf{H}_\mathit{\Pi}}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}), (\mathsf{sk}_i)_{i \in [K]})$
                                                  // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_3^{\mathcal{A}}$
$\pi \leftarrow \mathsf{Sim}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}))$
If $\pi = \bot$ then **abort game**// $\mathbf{G}_4^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
Return $((\mathsf{pk}_{i,\mathrm{sid}})_{i \in [K-1]}, \bar{S}, \vec{R}, \bar{R}, A, \pi)$

Oracle $\mathsf{H}_\star(\mathrm{str})$:
                              // $\mathsf{H}_\star \in \{\mathsf{H}', \mathsf{H}_\mathit{\Pi}, \mathsf{H}_\mu, \mathsf{H}_\mathsf{r}, \mathsf{H}_{cc}\}$
If $\mathsf{H}_\star(\mathrm{str}) \neq \bot$ then
   return $\mathsf{H}_\star(\mathrm{str})$
$\mathsf{H}_\star(\mathrm{str}) \leftarrow\$ \mathbb{Z}_p$                          // $\mathsf{H}_\star \in \{\mathsf{H}', \mathsf{H}_\mathit{\Pi}\}$
$\mathsf{H}_\star(\mathrm{str}) \leftarrow\$ \{0,1\}^\lambda$                       // $\mathsf{H}_\star \in \{\mathsf{H}_\mu, \mathsf{H}_\mathsf{r}\}$
$\mathsf{H}_\star(\mathrm{str}) \leftarrow\$ [N]^K$                            // $\mathsf{H}_\star = \mathsf{H}_{cc}$
Return $\mathsf{H}_\star(\mathrm{str})$
```

Fig. 12. The OMUF $= \mathbf{G}_0^{\mathcal{A}}$ security game for BS_3 and the subsequent games $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$. We assume that the adversary $\mathcal{A}$ makes ℓ queries to the signing oracle S_1. We remark that $\mathsf{H}, \mathsf{H}', \mathsf{H}_\mathit{\Pi}, \mathsf{H}_\mu, \mathsf{H}_\mathsf{r}, \mathsf{H}_{cc}$ are modeled as random oracles and $\mathcal{A}$ has access to them. Each box type indicates the changes made in the game contained in the box and to make things clearer, for each box, we indicate which game contains the box by a comment by the side of it. The signer state is omitted and variable names with subscript sid are instead used to denote the corresponding signer state values.

Game $\mathbf{G}_3^{\mathcal{A}}$: This game is identical to $\mathbf{G}_2^{\mathcal{A}}$ except that when generating the component W in par, the game generates $w \leftarrow_\$ \mathbb{Z}_p$ and sets $W \leftarrow g^w$. Since W still has the same distribution, $\mathcal{A}$'s success probability is exactly as in $\mathbf{G}_2^{\mathcal{A}}$.

$$\Pr[\mathbf{G}_3^{\mathcal{A}} = 1] = \Pr[\mathbf{G}_2^{\mathcal{A}} = 1] .$$

Game $\mathbf{G}_4^{\mathcal{A}}$: This game is identical to $\mathbf{G}_3^{\mathcal{A}}$ except that in the signing oracle S_1, the NIZK proof π is generated by using a simulator Sim on the input $(g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S})$. Following Lemma 5.3, by the zero-knowledge property of Π, such simulator exists, and since $\mathcal{A}$ makes ℓ and Q_{H_Π} queries to S_1 and H_Π, we have

$$\Pr[\mathbf{G}_4^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_3^{\mathcal{A}} = 1] - \frac{\ell(\ell + Q_{\mathsf{H}_\Pi})}{p} .$$

Game $\mathbf{G}_5^{\mathcal{A}}$: This game is identical to $\mathbf{G}_4^{\mathcal{A}}$ except that the signing oracles $\mathrm{S}_1, \mathrm{S}_2$ now generate $\vec{R}, \bar{R}, A, d, e, \vec{z}_0, z_1$ as follows: ($h_{i,\vec{J}_i}$ are the values sent by $\mathcal{A}$ in the first user message.)

1. $r_1, d \leftarrow_\$ \mathbb{Z}_p$, $\vec{z}_0 \leftarrow_\$ \mathbb{Z}_p^K$.
2. $A \leftarrow g^{r_1}$, $\vec{R} \leftarrow (g^{\vec{z}_{0,1}} \cdot \mathsf{pk}_1^{-d}, \ldots, g^{\vec{z}_{0,K}} \cdot \mathsf{pk}_K^{-d})$, $\bar{R} \leftarrow \bar{S}^{-d} \cdot \prod h_{i,\vec{J}_i}^{\vec{z}_{0,i}}$.
3. After receiving c, set $e \leftarrow c - d$ and $z_1 \leftarrow e \cdot w + r_1$.

Since the joint distributions of $(\vec{R}, \bar{R}, A, d, e, \vec{z}_0, z_1)$ in the games $\mathbf{G}_4^{\mathcal{A}}$ and $\mathbf{G}_5^{\mathcal{A}}$ are identical, the view of $\mathcal{A}$ remains the same. Thus,

$$\Pr[\mathbf{G}_5^{\mathcal{A}} = 1] = \Pr[\mathbf{G}_4^{\mathcal{A}} = 1] .$$

After this point, we show a reduction $\mathcal{B}_R$ to the OMUF security of BS_R (described in Figure 13) which as stated earlier in this section is a pairing-free version of Rai-Choo scheme [HLW23] with inefficient verification.

More formally, there exists an adversary $\mathcal{B}_R$ playing the OMUF game of BS_R, running in time $t_{\mathcal{B}_R} \approx t_{\mathcal{A}}$, and making ℓ queries to oracle S and $Q_{\mathsf{H}_\star}$ to random oracle $\mathsf{H}_\star$ for $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}\}$ such that $\Pr[\mathbf{G}_5^{\mathcal{A}} = 1] \leqslant \mathsf{Adv}_{\mathsf{BS}_R}^{\mathrm{omuf}}(\mathcal{B}_R, \lambda)$. Then, the reduction $\mathcal{B}_R$ is described as follows:

1. The reduction $\mathcal{B}_R$ receives the inputs $\mathsf{par} = (\mathbb{G}, p, g, N, K)$ and pk, samples $w \leftarrow_\$ \mathbb{Z}_p$, and sets $W \leftarrow g^w, \mathsf{par}' \leftarrow (\mathbb{G}, p, g, W, N, K)$. It then runs $\mathcal{A}$ on the inputs $\mathsf{par}', \mathsf{pk}$.
2. For random oracle queries to $\mathsf{H}, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}$, and H_{cc}, it forwards the queries to its own oracle $\mathsf{H}, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}$, and H_{cc} respectively. For the random oracle queries to H' and H_Π, it answers as in $\mathbf{G}_5^{\mathcal{A}}$.
3. For the queries to signing oracle S_1, it first forwards the query umsg_1 to its own oracle S which returns $(\mathsf{pk}_i)_{i \in [K]}, \bar{S}$. Then, it proceeds to simulate the computation of $(R_i)_{i \in [K]}, \bar{R}, A, \pi$ as in $\mathbf{G}_5^{\mathcal{A}}$. For the queries to signing oracle S_2, it also proceeds to compute $(d, e, \vec{z}_0, z_1)$ as in $\mathbf{G}_5^{\mathcal{A}}$. Note that by the change in $\mathbf{G}_4^{\mathcal{A}}$ and $\mathbf{G}_5^{\mathcal{A}}$, $\mathcal{B}_R$ does not need sk to compute these values.
4. At the end of the simulation, it receives the output $(m_k^*, \sigma_k^*)_{k \in [\ell+1]}$ and returns to its own game $(m_k^*, \hat{\sigma}_k^*)_{k \in [\ell+1]}$ where $\hat{\sigma}_k^* = ((\mathsf{pk}_{k,i}^*, \varphi_{k,i}^*)_{i \in [K]}, \bar{S}_k^*)$ with $\mathsf{pk}_{k,i}^*, \varphi_{k,i}^*, \bar{S}_k^*$ being the corresponding values in σ_k^* for $k \in [\ell + 1]$.

It is clear that the running time of $\mathcal{B}_R$ is about that of $\mathcal{A}$. For the success probability of the reduction, we can see that the simulation of the oracles gives the same distribution of outputs as in $\mathbf{G}_5^{\mathcal{A}}$. Then, consider when $\mathcal{A}$ wins in game $\mathbf{G}_5^{\mathcal{A}}$, for all $k \in [\ell + 1]$, $\mathsf{pk} = \prod_{i=1}^K \mathsf{pk}_{k,i}^*$ and $\bar{S}_k^* = \prod_{i=1}^K \mathsf{H}(\mathsf{H}_\mu(m_k^*, \varphi_{k,i}^*))^{\mathsf{sk}_{k,i}}$ where $\mathsf{sk}_{k,i} = \log_g \mathsf{pk}_{k,i}$. Thus, if $\mathcal{A}$ wins in game $\mathbf{G}_5^{\mathcal{A}}$, then $\mathcal{B}_R$ also wins in game OMUF, so $\Pr[\mathbf{G}_5^{\mathcal{A}} = 1] \leqslant \mathsf{Adv}_{\mathsf{BS}_R}^{\mathrm{omuf}}(\mathcal{B}_R, \lambda)$. Lastly, combining the advantage changes and Lemma 5.6 (stated below), we have that

$$\begin{aligned}\mathsf{Adv}_{\mathsf{BS}_3}^{\mathrm{omuf}}(\mathcal{A}, \lambda) \leqslant (\ell + 1)\left(\sqrt{Q_{\mathsf{H}'}\mathsf{Adv}_{\mathsf{GGen}}^{\mathsf{dlog}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p}\right) + \frac{\ell(\ell + Q_{\mathsf{H}_\Pi})}{p} \\ + \frac{\ell}{N^K} + \frac{Q_{\mathsf{H}_{\mathsf{com}}}^2 + Q_{\mathsf{H}_\mu}^2 + Q_{\mathsf{H}_{\mathsf{com}}}Q_{\mathsf{H}_{cc}} + Q_{\mathsf{H}}Q_{\mathsf{H}_\mu}}{2^\lambda} + 4\ell \cdot \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{cdh}}(\mathcal{B}', \lambda) ,\end{aligned}$$

where $\mathcal{B}'$ is an adversary playing the game CDH running in time $t_{\mathcal{B}'} \approx t_{\mathcal{B}_R} \approx t_{\mathcal{A}}$. This concludes the proof for Theorem 5.2. □

```
Algorithm $\mathsf{BS}_R.\mathsf{Setup}(1^\lambda, K, N)$ :
$(\mathbb{G}, p, g) \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda)$
Select $\mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}} : \{0,1\}^* \to \{0,1\}^\lambda$
Select $\mathsf{H} : \{0,1\}^* \to \mathbb{G}$
Select $\mathsf{H}_\beta : \{0,1\}^* \to \mathbb{Z}_p$
Select $\mathsf{H}_{cc} : \{0,1\}^* \to [N]^K$
$\mathsf{par} \leftarrow (\mathbb{G}, p, g, K, N, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}, \mathsf{H}_{cc})$
Return $\mathsf{par}$

Algorithm $\mathsf{BS}_R.\mathsf{S}(\mathsf{par}, \mathsf{sk}, \mathsf{open})$ :
$(\mathbb{G}, p, g, K, N, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}, \mathsf{H}_{cc}) \leftarrow \mathsf{par}$
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{open}$
If $\mathsf{Check}(\mathsf{open}) = 0$ then return $\perp$
For $i \in [K-1]$ : $\mathsf{sk}_i \leftarrow\!\!\$\ \mathbb{Z}_p$
$\mathsf{sk}_K \leftarrow \mathsf{sk} - \sum_{i=1}^{K-1} \mathsf{sk}_i$
For $i \in [K]$ : $\mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i}$
$\bar{S} \leftarrow \prod_{i=1}^{K} h_{i,\vec{J}_i}^{\mathsf{sk}_i}$
Return $\mathsf{smsg} \leftarrow ((\mathsf{pk}_i)_{i \in [K-1]}, \bar{S})$

Algorithm $\mathsf{Check}(\mathsf{open})$:
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{open}$
For $i \in [K]$ and $j \in [K] \backslash \{\vec{J}_i\}$:
    $\mathsf{com}_{i,j} \leftarrow \mathsf{H}_{\mathsf{com}}(\mathsf{r}_{i,j})$
    $(\mu_{i,j}, \varepsilon_{i,j}) \leftarrow \mathsf{r}_{i,j}, \beta_{i,j} \leftarrow \mathsf{H}_\beta(\varepsilon_{i,j})$
    $h_{i,j} \leftarrow \mathsf{H}(\mu_{i,j}) g^{\beta_{i,j}}$
$\mathsf{com} \leftarrow (\mathsf{com}_{i,j})_{(i,j) \in [K] \times [N]}$
$h \leftarrow (h_{i,j})_{(i,j) \in [K] \times [N]}$
If $\vec{J} \neq \mathsf{H}_{cc}(\mathsf{com}, h)$ then return 0.
Return 1

Algorithm $\mathsf{ReRa}((\mathsf{pk}_i, h_i)_{i \in [K]}, S)$:
$\tau_1, \ldots, \tau_{K-1} \leftarrow\!\!\$\ \mathbb{Z}_p$; $\tau_K \leftarrow -\sum_{i=1}^{K-1} \tau_K$
For $i \in [K]$: $\mathsf{pk}'_i \leftarrow \mathsf{pk}_i g^{\tau_i}$
$S' \leftarrow S \prod_{i=1}^{K} h_i^{\tau_i}$
Return $((\mathsf{pk}_i)_{i \in [K]}, S', (\tau_i)_{i \in [K]})$
```

```
Algorithm $\mathsf{BS}_R.\mathsf{KG}(\mathsf{par})$ :
$(\mathbb{G}, p, g, K, N, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}, \mathsf{H}_{cc}) \leftarrow \mathsf{par}$
$x \leftarrow\!\!\$\ \mathbb{Z}_p$ ; $X \leftarrow g^x$ ; $\mathsf{sk} \leftarrow x$ ; $\mathsf{pk} \leftarrow X$
Return $(\mathsf{sk}, \mathsf{pk})$

Algorithm $\mathsf{BS}_R.\mathsf{U}_1(\mathsf{par}, \mathsf{pk}, m)$ :
$(\mathbb{G}, p, g, K, N, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}, \mathsf{H}_{cc}) \leftarrow \mathsf{par}$
For $(i,j) \in [K] \times [N]$:
    $\varphi_{i,j} \leftarrow\!\!\$\ \{0,1\}^\lambda$ ; $\mu_{i,j} \leftarrow \mathsf{H}_\mu(m, \varphi_{i,j})$
    $\varepsilon_{i,j} \leftarrow\!\!\$\ \{0,1\}^\lambda$ ; $\beta_{i,j} \leftarrow \mathsf{H}_\beta(\varepsilon_{i,j})$
    $\mathsf{r}_{i,j} \leftarrow (\mu_{i,j}, \varepsilon_{i,j})$ ; $\mathsf{com}_{i,j} \leftarrow \mathsf{H}_{\mathsf{com}}(\mathsf{r}_{i,j})$
    $h'_{i,j} \leftarrow \mathsf{H}(\mu_{i,j})$ ; $h_{i,j} \leftarrow h'_{i,j} g^{\beta_{i,j}}$
$\mathsf{com} \leftarrow (\mathsf{com}_{i,j})_{i \in [K], j \in [N]}$
$h \leftarrow (h_{i,j})_{i \in [K], j \in [N]}$
$\vec{J} \leftarrow \mathsf{H}_{cc}(\mathsf{com}, h)$
$\mathsf{open} \leftarrow (\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]})$
$\mathsf{st}^u \leftarrow (m, \mathsf{pk}, (h_{i,\vec{J}_i}, h'_{i,\vec{J}_i}, \beta_{i,\vec{J}_i}, \varphi_{i,\vec{J}_i})_{i \in [K]})$
Return $(\mathsf{st}^u, \mathsf{open})$

Algorithm $\mathsf{BS}_R.\mathsf{U}_2(\mathsf{st}^u, \mathsf{smsg})$ :
$(m, \mathsf{pk}, (h_{i,\vec{J}_i}, h'_{i,\vec{J}_i}, \beta_{i,\vec{J}_i}, \varphi_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{st}^u$
$((\mathsf{pk}_i)_{i \in [K-1]}, \bar{S}) \leftarrow \mathsf{smsg}$
$\mathsf{pk}_K \leftarrow \mathsf{pk} \prod_{i \in [K]} \mathsf{pk}_i^{-1}$
$((\mathsf{pk}'_i)_{i \in [K]}, \bar{S}', \vec{\tau}) \leftarrow\!\!\$$
    $\mathsf{ReRa}((\mathsf{pk}_i, h'_{i,\vec{J}_i})_{i \in [K]}, \bar{S} \prod_{i=1}^{K} \mathsf{pk}_i^{-\beta_{i,\vec{J}_i}})$
Return $\sigma \leftarrow ((\mathsf{pk}'_i, \varphi_{i,\vec{J}_i})_{i \in [K]}, \bar{S}')$

Algorithm $\mathsf{BS}_R.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m, \sigma)$
$(\mathbb{G}, p, g, K, N, \mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}, \mathsf{H}_{cc}) \leftarrow \mathsf{par}$
$((\mathsf{pk}_i, \varphi_i)_{i \in [K]}, \bar{S}) \leftarrow \sigma$
For $i \in [K]$ :
    $\boxed{\mathsf{sk}_i \leftarrow \log_g \mathsf{pk}_i}$ ; $h_i \leftarrow \mathsf{H}(\mathsf{H}_\mu(m, \varphi_i))$
If $\prod_{i=1}^{K} \mathsf{pk}_i \neq \mathsf{pk}$ or $\prod_{i=1}^{K} h_i^{\mathsf{sk}_i} \neq \bar{S}$
    then return 0
Return 1
```

Fig. 13. The signing protocol of the blind signature scheme $\mathsf{BS}_R = \mathsf{BS}_R[\mathsf{GGen}]$ with the inefficient verification $\mathsf{BS}_R.\mathsf{Ver}$. (The inefficiency is highlighted in the box)

Lemma 5.6 (OMUF of BS_R). *Assume that* GGen *outputs the description of a group of prime order $p = p(\lambda)$, and let $\mathsf{BS}_R = \mathsf{BS}_R[\mathsf{GGen}]$ and $K = K(\lambda), N = N(\lambda)$ be positive integer inputs to $\mathsf{BS}_R.\mathsf{Setup}$. For any adversary $\mathcal{A}$ for game* OMUF *with running time $t_\mathcal{A} = t_\mathcal{A}(\lambda)$, making at most $\ell = \ell(\lambda)$ queries to* S_1, *and $Q_{\mathsf{H}_\star} = Q_{\mathsf{H}_\star}(\lambda)$ queries to $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}\}$, modeled as random oracles, there exist an adversary $\mathcal{B}'$ for the game* CDH *running in time $t_{\mathcal{B}'} \approx t_\mathcal{A}$ such that*

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{BS}_R}(\mathcal{A}, \lambda) \leqslant \frac{Q^2_{\mathsf{H}_{\mathsf{com}}} + Q^2_{\mathsf{H}_\mu} + Q_{\mathsf{H}_{\mathsf{com}}} Q_{\mathsf{H}_{cc}} + Q_{\mathsf{H}} Q_{\mathsf{H}_\mu}}{2^\lambda} + \frac{\ell}{N^K} + 4\ell \cdot \mathsf{Adv}^{\mathrm{cdh}}_{\mathsf{GGen}}(\mathcal{B}', \lambda) \; .$$

We prove the above lemma following similar arguments as in the OMUF proof of Rai-Choo [HLW23], but for completeness, we present a full proof in Section 5.5.

Lemma 5.7. *There exists an adversary $\mathcal{B}$ playing the* DLOG *game with running time $t_\mathcal{B} \approx 2t_\mathcal{A}$ such that*

$$\Pr[\mathbf{G}_2^\mathcal{A} = 1] \geqslant \Pr[\mathbf{G}_1^\mathcal{A} = 1] - (\ell + 1)\left(\sqrt{Q_{\mathsf{H}'} \mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p}\right) \; .$$

Proof (of Lemma 5.7). Let event Bad be the event where $\mathbf{G}_1^\mathcal{A}$ outputs 1 but $\mathbf{G}_2^\mathcal{A}$ outputs 0 which corresponds to the following event: $\mathcal{A}$ outputs $\ell + 1$ message-signature pairs $(m_k^*, \sigma_k^*)_{k \in [\ell+1]}$ such that (1) for all $k_1 \neq k_2, m_{k_1}^* \neq m_{k_2}^*$, (2) for all $k \in [\ell+1]$, $\mathsf{BS}_3.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, \sigma_k^*, m_k^*) = 1$, and (3) there exists some $k \in [\ell+1]$ where parsing the signature $\sigma_k^* = ((\mathsf{pk}_{k,i}^*, \varphi_{k,i}^*)_{i\in[K]}, \bar{S}_k^*, d_k^*, e_k^*, \vec{z}_{k,0}^*, z_{k,1}^*)$, and letting $\mu_{k,i}^* \leftarrow \mathsf{H}_\mu(m_k^*, \varphi_{k,i}^*), \mathsf{sk}_{k,i}^* = \log_g \mathsf{pk}_{k,i}^*$,

$$\bar{S}_k^* \neq \prod_{i=1}^{K} \mathsf{H}(\mu_{k,i}^*)^{\mathsf{sk}_{k,i}^*} = \prod_{i=1}^{K} \mathsf{pk}_{k,i}^{*\ t[\mu_{k,i}^*]} \ .$$

Then, we can bound $\mathsf{Pr}[\mathbf{G}_2^{\mathcal{A}} = 1] \geqslant \mathsf{Pr}[\mathbf{G}_1^{\mathcal{A}} = 1] - \mathsf{Pr}[\mathsf{Bad}]$.

Now, we focus on upperbounding $\mathsf{Pr}[\mathsf{Bad}]$. Define the event Bad_k for $k \in [\ell+1]$ which is event Bad with the condition (3) specified only for the k-th message-signature pair (m_k^*, σ_k^*). We can see that $\mathsf{Bad} = \bigcup_{k=1}^{\ell+1} \mathsf{Bad}_k$.

To do this, define the following deterministic wrapper $\mathcal{A}_k$ over $\mathcal{A}$, which takes inputs: the instance $(\mathbb{G}, p, g, W)$, the outputs $(c_1, \ldots, c_{Q_{\mathsf{H}'}})$ of H', and a random tape ρ.

1. Extract from the random tape ρ, the following

$$(x, ((\mathsf{sk}_{j,i})_{i\in[K-1]}, \vec{r}_{0,j}, e_j, z_{1,j}, \rho_{\Pi,j})_{j\in[\ell]}, (t_i)_{i\in[Q_\mathsf{H}]}, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_\Pi, \mathsf{H}_{cc}, \rho')$$

   where $x, t_i, \mathsf{sk}_{j,i}, e_j, z_{1,j} \in \mathbb{Z}_p, \vec{r}_{0,j} \in \mathbb{Z}_p^K$, while $\rho_{\Pi,j}$ denotes the randomness used to generate π in the j-th signing session and $\mathsf{H}_\star \in \{\mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_\Pi, \mathsf{H}_{cc}\}$ denotes a list of $Q_{\mathsf{H}_\star}$ values in the codomain of $\mathsf{H}_\star$. Additionally, we denote $\mathsf{H}_\star[i]$ as the i-th entry in the list.
2. Set $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W), \mathsf{pk} \leftarrow g^x, \mathsf{sk} \leftarrow x$.
3. Run $(m_k^*, \sigma_k^*)_{k\in[\ell+1]} \leftarrow \mathcal{A}^{\mathrm{S}_1,\mathrm{S}_2,\mathsf{H},\mathsf{H}',\mathsf{H}_\Pi,\mathsf{H}_\mu,\mathsf{H}_{\mathsf{com}},\mathsf{H}_{cc}}(\mathsf{par}, \mathsf{pk}; \rho')$ where each oracle is answered as follows:
   - For the j-th query ($j \in [\ell]$) to S_1 use $x, ((\mathsf{sk}_{j,i})_{i\in[K-1]}, \vec{r}_{0,i}, e_i, z_{1,i}, \rho_{\Pi,j})$ to answer the query as in $\mathsf{BS}_3.\mathsf{S}_1$. For the query to S_2 of the same session id, also use $\mathsf{sk}, ((\mathsf{sk}_{j,i})_{i\in[K-1]}, \vec{r}_{0,i}, e_i, z_{1,i})$ as in the game $\mathsf{BS}_3.\mathsf{S}_2$.
   - For the i-th query to H ($i \in [Q_\mathsf{H}]$), answer with g^{t_i} and set $t[\cdot] = t_i$ accordingly.
   - For the i-th query to H' ($i \in [Q_{\mathsf{H}'}]$), answer with c_i.
   - For the i-th query to $\mathsf{H}_\star \in \{\mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_\Pi, \mathsf{H}_{cc}\}$ ($i \in [Q_{\mathsf{H}_\star}]$), answer with $\mathsf{H}_\star[i]$. (Among the queries to H_Π, we w.l.o.g. accounted for the queries that the wrapper made to generate π. Moreover, when the same queries are queried more than once, the oracle will answer with the first value initialized.)
4. If the event Bad_k does not occur, return $(\perp, \perp)$.
   Otherwise, return $(I, (m_k^*, \sigma_k^*))$ where I is the index of the query to H' from $\mathcal{A}$ corresponding to the verification of (m_k^*, σ_k^*). More specifically, after parsing $\sigma_k^* = ((\mathsf{pk}_i^*, \varphi_i^*)_{i\in[K]}, \bar{S}^*, d^*, e^*, \vec{z}_0^*, z_1^*)$, I is the index of the query

$$(m_k^*, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}^*, (R_i)_{i\in[K]}, \bar{R}, A) \ ,$$

   where $h_i = \mathsf{H}(\mathsf{H}_\mu(m_k^*, \varphi_i^*)), \mathsf{pk}_i = \mathsf{pk}_i^*, R_i = g^{\vec{z}_{0,i}^*}\mathsf{pk}_i^{-d^*}$ for $i \in [K]$, and $\bar{R} = (\bar{S}^*)^{-d^*} \prod_{i=1}^K h_i^{\vec{z}_{0,i}^*}$, $A = g^{z_1^*} W^{-e^*}$. Note that I and all the values above are well-defined as we assume that all RO queries done in forgery verification are made by $\mathcal{A}$ beforehand. Also, the way we program H allows us to check for event Bad_k efficiently, i.e., by checking $\bar{S}_k^* \neq \prod_{i=1}^K \mathsf{pk}_{k,i}^{*\ t[\mu_{k,i}^*]}$, which means that the running time of $\mathcal{A}_k$ is roughly that of $\mathcal{A}$.

Now, consider a reduction $\mathcal{B}$ playing the discrete logarithm game defined as follows:

1. On the input $(\mathbb{G}, p, g, W)$, $\mathcal{B}$ samples $c_1, \ldots, c_{Q_{\mathsf{H}'}} \leftarrow\!\!\$\ \mathbb{Z}_p$ along with random coins ρ.
2. Run $(I, (m, \sigma)) \leftarrow\!\!\$\ \mathcal{A}_k((\mathbb{G}, p, g, W), (c_1, \ldots, c_{Q_{\mathsf{H}'}}); \rho)$.
3. If $I = 0$, abort. If not, sample $c_I', \ldots, c_{Q_\mathsf{H}'}' \leftarrow\!\!\$\ \mathbb{Z}_p$ and
   run $(I', (m', \sigma')) \leftarrow\!\!\$\ \mathcal{A}_k((\mathbb{G}, p, g, W, \mathsf{pk}, \mathsf{sk}), (c_1, \ldots, c_{I-1}, c_I', \ldots, c_{Q_{\mathsf{H}'}}'); \rho)$.
4. If $I = I'$ and $c_I' \neq c_I$, parse $\sigma = ((\mathsf{pk}_i, \varphi_i)_{i\in[K]}, \bar{S}, d, e, \vec{z}_0, z_1)$ and $\sigma' = ((\mathsf{pk}_i', \varphi_i')_{i\in[K]}, \bar{S}', d', e', \vec{z'}_0, z_1')$, and return $(z_1 - z_1')(e - e')^{-1}$. Else, abort.

Since $\mathcal{B}$ runs $\mathcal{A}_k$ twice and the running time of $\mathcal{A}_k$ is about that of $\mathcal{A}$, we have $t_\mathcal{B} \approx 2t_\mathcal{A}$. Then, we argue the success probability of the reduction, i.e., we show that if $\mathcal{B}$ does not abort then it returns a discrete log of W. Notice that $I = I' \neq \perp$, so the message-signature pairs $(m, \sigma), (m', \sigma')$ are: (a) valid signatures corresponding to the I-th query of $\mathcal{A}$ to H', and (b) for $i \in [K]$, let $\mu_i \leftarrow \mathsf{H}_\mu(m, \varphi_i), \mu_i' \leftarrow \mathsf{H}_\mu(m', \varphi_i'), \mathsf{sk}_i \leftarrow \log_g \mathsf{pk}_i, \mathsf{sk}_i' \leftarrow \log_g \mathsf{pk}_i'$, we have $\bar{S} \neq \prod_{i=1}^K \mathsf{H}(\mu_i)^{\mathsf{sk}_i}$ and $\bar{S}' \neq \prod_{i=1}^K \mathsf{H}(\mu_i')^{\mathsf{sk}_i'}$. By (a), we know the following

(i) $m = m', \bar{S} = \bar{S}'$.
(ii) For $i \in [K], \mathsf{pk}_i = \mathsf{pk}'_i$, and $\mathsf{H}(\mu_i) = h_i = h'_i = \mathsf{H}(\mu'_i)$.
(iii) $c_I = d + e, c'_I = d' + e'$.
(iv) For $i \in [K], \mathsf{pk}_i{}^{-d} g^{\vec{z}_{0,i}} = \mathsf{pk}'_i{}^{-d'} g^{\vec{z}'_{0,i}}$ and $\bar{S}^{-d} \prod_{i=1}^{K} h_i{}^{\vec{z}_{0,i}} = \bar{S}'^{-d'} \prod_{i=1}^{K} h'_i{}^{\vec{z}'_{0,i}}$.
(v) $A = g^{z_1} W^{-e} = g^{z'_1} W^{-e'}$.

Next, we will argue that $d = d'$. As a result from (i, ii, iv), for all $i \in [K]$

$$\mathsf{pk}_i^{(d-d')\log h_i} = (\mathsf{pk}_i^d \mathsf{pk}'_i{}^{-d'})^{\log_g h_i} = g^{(\vec{z}_{0,i} - \vec{z}'_{0,i})\log_g h_i} = h_i^{\vec{z}_{0,i} - \vec{z}'_{0,i}} \text{ , and}$$
$$\bar{S}^{d-d'} = \bar{S}^d \bar{S}'^{-d'} = \prod_{i=1}^{K} h_i{}^{\vec{z}_{0,i}} h'_i{}^{-\vec{z}'_{0,i}} = \prod_{i=1}^{K} h_i^{\vec{z}_{0,i} - \vec{z}'_{0,i}} = \prod_{i=1}^{K} \mathsf{pk}_i^{(d-d')\log h_i} .$$

Since $\bar{S} \neq \prod_{i=1}^{K} \mathsf{pk}_i^{\log_g h_i}$, only $d = d'$ satisfies the equation. Since $d + e = c_I \neq c'_I = d' + e'$, we have $e \neq e'$. Thus, with (v), $\mathcal{B}$ returns $(z_1 - z'_1)(e - e')^{-1} = \log_g W$. Hence,

$$\mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda) = \mathsf{Pr}[\mathcal{B} \text{ does not abort}] = \mathsf{Pr}[I = I' \wedge I \neq \bot \wedge c_I \neq c'_I] .$$

Thus, by the forking lemma (Lemma 2.1) and that $\mathcal{B}$ rewinds $\mathcal{A}_k$ which only outputs $I \neq \bot$ when Bad_k occurs,

$$\mathsf{Pr}[\mathsf{Bad}_k] \leqslant \sqrt{Q_{\mathsf{H}'} \mathsf{Adv}^{\mathsf{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{Q_{\mathsf{H}'}}{p} .$$

The lemma statement follows from the union bound over Bad_k for $k \in [\ell + 1]$. □

5.5 Proof of Lemma 5.6 (OMUF of BS_R)

Let $\mathcal{A}$ be an adversary playing one-more unforgeability game of BS_R. We consider the following sequence of games (pseudocode game sequence is described in Figures 14 and 15).

Game $\mathbf{G}_0^{\mathcal{A}}$: The game first generates the public parameters $\mathsf{par} \leftarrow\!\!\$\ \mathsf{BS}_R.\mathsf{Setup}(1^\lambda, N, K)$ and the public and secret keys $(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\!\$\ \mathsf{BS}_R.\mathsf{KG}(\mathsf{par})$. Then, the game interacts with an adversary $\mathcal{A}(\mathsf{par}, \mathsf{pk})$ with access to signing oracle S and the random oracles $\mathsf{H}, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}$ where each of them is simulated by lazy sampling. The adversary $\mathcal{A}$ (w.l.o.g.) queries the signing oracle S for ℓ times and the random oracles $\mathsf{H}_\star$ for $Q_{\mathsf{H}_\star}$ times for $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}\}$. At the end of the game, $\mathcal{A}$ outputs $\ell + 1$ message-signature pairs (m_k^*, σ_k^*) for $k \in [\ell + 1]$. The adversary $\mathcal{A}$ succeeds if $m_{k_1}^* \neq m_{k_2}^*$ for all $k_1 \neq k_2$ and $\mathsf{BS}_R.\mathsf{Ver}(\mathsf{par}, \mathsf{pk}, m_k^*, \sigma_k^*) = 1$ for all $i \in [\ell + 1]$. We additionally assume w.l.o.g. that $\mathcal{A}$ already makes the queries to $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}\}$ that would otherwise be called in $\mathsf{BS}_R.\mathsf{Ver}$ when the game checks the forgeries. The probability of $\mathcal{A}$ winning in game $\mathbf{G}_0^{\mathcal{A}}$ is exactly its advantage in OMUF i.e.

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{BS}_R}(\mathcal{A}, \lambda) = \mathsf{Pr}[\mathbf{G}_0^{\mathcal{A}} = 1] .$$

Game $\mathbf{G}_1^{\mathcal{A}}$: This game is identical to $\mathbf{G}_0^{\mathcal{A}}$ except that it aborts if there exists two queries $x \neq x'$ to $\mathsf{H}_\star(\cdot)$ for $\mathsf{H}_\star \in \{\mathsf{H}_{\mathsf{com}}, \mathsf{H}_\mu\}$ such that $\mathsf{H}_\star(x) = \mathsf{H}_\star(x')$. This change is unnoticed by $\mathcal{A}$ except when the abort occurs which by $\mathsf{H}_\star(x)$ sampled uniformly from $\{0,1\}^\lambda$,

$$\mathsf{Pr}[\mathbf{G}_1^{\mathcal{A}} = 1] \geqslant \mathsf{Pr}[\mathbf{G}_0^{\mathcal{A}} = 1] - \frac{Q_{\mathsf{H}_{\mathsf{com}}}^2 + Q_{\mathsf{H}_\mu}^2}{2^\lambda} .$$

Game $\mathbf{G}_2^{\mathcal{A}}$: This game is identical to $\mathbf{G}_1^{\mathcal{A}}$ except that when $\mathcal{A}$ queries H_{cc} on (com, h) with $\mathsf{com} = (\mathsf{com}_{i,j})_{(i,j) \in [K] \times [N]}$ and $h = (h_{i,j})_{(i,j) \in [K] \times [N]}$, the game does the following: For each $(i,j) \in [K] \times [N]$, check if there exists a query r' to $\mathsf{H}_{\mathsf{com}}$ such that $\mathsf{H}_{\mathsf{com}}(\mathsf{r}') = \mathsf{com}_{i,j}$. If there is one, set $\hat{\mathsf{r}}_{i,j} \leftarrow \mathsf{r}'$. If not, set $\hat{\mathsf{r}}_{i,j} \leftarrow \bot$ and abort if later there is a query r' to $\mathsf{H}_{\mathsf{com}}$ where $\mathsf{H}_{\mathsf{com}}(\mathsf{r}') = \mathsf{com}_{i,j}$.

```
Game $\mathbf{G}_0^{\mathcal{A}}$, $\mathbf{G}_1^{\mathcal{A}}$, $\mathbf{G}_2^{\mathcal{A}}$, $\mathbf{G}_3^{\mathcal{A}}$, $\mathbf{G}_4^{\mathcal{A}}$:
$(\mathbb{G}, p, g) \leftarrow\!\$\ \mathsf{GGen}(1^\lambda)$; $\mathsf{par} \leftarrow (\mathbb{G}, p, g)$
$x \leftarrow\!\$\ \mathbb{Z}_p$; $X \leftarrow g^x$; $\mathsf{sk} \leftarrow x$; $\mathsf{pk} \leftarrow X$
$\ell' \leftarrow 0$ ; $\mathcal{I} \leftarrow \varnothing$
Map $\hat{\mathsf{r}}[\cdot] : \{0,1\}^\lambda \rightarrow \{0,1\}^{2\lambda}$ // $\mathbf{G}_2^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$\{(m_k^*, \sigma_k^*)\}_{k \in [\ell+1]} \leftarrow\!\$\ \mathcal{A}^{\mathrm{S}}(\mathsf{par}, \mathsf{pk})$
If $\exists\ k_1 \neq k_2, m_{k_1}^* = m_{k_2}^*$ then
    return 0
If $\exists\ k \in [\ell + 1]$ such that
    $\mathsf{BS}_R.\mathsf{Ver}(\mathsf{pk}, m_k^*, \sigma_k^*) = 0$ then
        return 0
Return 1

Oracle $\mathsf{H}_\mu(\mathrm{str})$:
If $\mathsf{H}_\mu(\mathrm{str}) \neq \perp$ then return $\mathsf{H}_\mu(\mathrm{str})$
$\mathsf{H}_\mu(\mathrm{str}) \leftarrow\!\$\ \{0,1\}^\lambda$
If $\exists \mathrm{str}' \neq \mathrm{str}, \mathsf{H}_\mu(\mathrm{str}) = \mathsf{H}_\mu(\mathrm{str}')$
    then abort game. // $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
Return $\mathsf{H}_\mu(\mathrm{str})$

Oracle $\mathsf{H}_{\mathsf{com}}(\mathrm{str})$:
If $\mathsf{H}_{\mathsf{com}}(\mathrm{str}) \neq \perp$ then
    return $\mathsf{H}_{\mathsf{com}}(\mathrm{str})$
$\mathsf{com} \leftarrow\!\$\ \{0,1\}^\lambda$
If $\exists \mathrm{str}' \neq \mathrm{str}, \mathsf{com} = \mathsf{H}_{\mathsf{com}}(\mathrm{str}')$
    then abort game. // $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
If $\hat{\mathsf{r}}[\mathsf{com}] = \perp$ and    // $\mathbf{G}_2^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
    $\exists(\mathsf{com}', h'), (\mathsf{H}_{cc}(\mathsf{com}', h') \neq \perp$
        and $\exists(i, j), \mathsf{com}'_{i,j} = \mathsf{com})$
    then abort game.
Return $\mathsf{H}_{\mathsf{com}}(\mathrm{str}) \leftarrow \mathsf{com}$
```

```
Oracle $\mathrm{S}(\mathrm{sid}, \mathsf{open})$:
If $\mathrm{sid} \in \mathcal{I}$ then return $\perp$
$\ell' \leftarrow \ell' + 1$ ; $\mathcal{I}_1 \leftarrow \mathcal{I}_1 \cup \{\mathrm{sid}\}$
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{open}$
If $\mathsf{Check}(\mathsf{open}) = 0$ then return $\perp$
$i^* \leftarrow\!\$\ \mathsf{DetectCheat}(\mathsf{open})$ // $\mathbf{G}_3^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
If $i^* = \perp$ then abort game
For $i \in [K-1]$ : $\mathsf{sk}_{i,\mathrm{sid}} \leftarrow\!\$\ \mathbb{Z}_p$
$\mathsf{sk}_{K,\mathrm{sid}} \leftarrow \mathsf{sk} - \sum_{i=1}^{K-1} \mathsf{sk}_{i,\mathrm{sid}}$ // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_3^{\mathcal{A}}$
For $i \in [K] \backslash \{i^*\}$ : $\mathsf{sk}_{i,\mathrm{sid}} \leftarrow\!\$\ \mathbb{Z}_p$
$\mathsf{sk}_{i^*,\mathrm{sid}} \leftarrow \mathsf{sk} - \sum_{i \neq i^*} \mathsf{sk}_{i,\mathrm{sid}}$ // $\mathbf{G}_4^{\mathcal{A}}$
For $i \in [K]$ : $\mathsf{pk}_i \leftarrow g^{\mathsf{sk}_{i,\mathrm{sid}}}$
$\bar{S} \leftarrow \prod_{i=1}^{K} h_{i,\vec{J}_i}^{\mathsf{sk}_{i,\mathrm{sid}}}$
Return $((\mathsf{pk}_i)_{i \in [K]}, \bar{S})$

Algorithm $\mathsf{DetectCheat}(\mathsf{open})$:
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{open}$
For $i \in [K]$:
    If $\hat{r}[\mathsf{com}_{i,\vec{J}_i}] = (\mu, \varepsilon) \neq \perp$ and
        $h_{i,\vec{J}_i} = \mathsf{H}(\mu) g^{\mathsf{H}_\beta(\varepsilon)}$
        then return $i$
Return $\perp$                          // $\mathbf{G}_3^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$

Oracle $\mathsf{H}_{cc}(\mathsf{com}, h)$:
If $\mathsf{H}_{cc}(\mathsf{com}, h) \neq \perp$ then return $\mathsf{H}_{cc}(\mathsf{com}, h)$
For $(i, j) \in [K] \times [N]$:
    If $\exists \mathsf{r}' = (\mu, \varepsilon), \mathsf{H}_{\mathsf{r}}(\mathsf{r}') = \mathsf{com}_{i,j}$ then
        $\hat{\mathsf{r}}[\mathsf{com}_{i,j}] = \mathsf{r}'$
    Else, $\hat{\mathsf{r}}[\mathsf{com}_{i,j}] = \perp$    // $\mathbf{G}_2^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$\mathsf{H}_{cc}(\mathsf{com}, h) \leftarrow\!\$\ [N]^K$
Return $\mathsf{H}_{cc}(\mathsf{com}, h)$
```

Fig. 14. The OMUF = $\mathbf{G}_0^{\mathcal{A}}$ security game and the subsequent games $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$. We assume that the adversary $\mathcal{A}$ makes ℓ queries to the signing oracle S. We remark that $\mathsf{H}, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{r}}, \mathsf{H}_{cc}$ are modeled as random oracles and $\mathcal{A}$ has access to them. We omitted the description of the oracle H which is implemented by lazy sampling over $\mathbb{G}$. Each box type indicates the changes made in the game contained in the box and to make things clearer, for each box, we indicate which game contains the box by a comment by the side of it. Note: the subroutine $\mathsf{DetectCheat}$ is introduced to S in game $\mathbf{G}_3^{\mathcal{A}}$.

The view of $\mathcal{A}$ is identical except when the abort occurs. Since queries to $\mathsf{H}_{\mathsf{com}}$ is answered with uniformly random value from $\{0,1\}^\lambda$, the probability that the abort occurs is at most $Q_{\mathsf{H}_{\mathsf{com}}} Q_{\mathsf{H}_{cc}} / 2^\lambda$ by the union bound over all pairs of H_{cc} and H_{r} queries. Thus,

$$\Pr[\mathbf{G}_2^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_1^{\mathcal{A}} = 1] - \frac{Q_{\mathsf{H}_{\mathsf{com}}} Q_{\mathsf{H}_{cc}}}{2^\lambda} .$$

Before proceeding to the next game, we consider an event when $\mathcal{A}$ makes a call to S with the input open where

$$\mathsf{open} = \left(\vec{J}, \left((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i} \right)_{i \in [K]} \right) .$$

We consider the case where $\mathsf{Check}(\mathsf{open}) = 1$ which would define the values $\mathsf{com} = (\mathsf{com}_{i,j})_{(i,j) \in [K] \times [N]}$ and $h = (h_{i,j})_{(i,j) \in [K] \times [N]}$ such that $\mathsf{H}_{cc}(\mathsf{com}, h) = \vec{J}$. Also, consider the values $\hat{\mathsf{r}}_{i,j}$ related to the query $\mathsf{H}_{cc}(\mathsf{com}, h)$ defined in $\mathbf{G}_2^{\mathcal{A}}$. For each instance $i \in [K]$, we have the following observations:

```
Game $\mathbf{G}_4^{\mathcal{A}}, \boxed{\mathbf{G}_5^{\mathcal{A}}}, \boxed{\mathbf{G}_6^{\mathcal{A}}}, \boxed{\mathbf{G}_7^{\mathcal{A}}}, \boxed{\mathbf{G}_8^{\mathcal{A}}}, \boxed{\mathbf{G}_9^{\mathcal{A}}}$:
$(\mathbb{G}, p, g) \leftarrow\$ \mathsf{GGen}(1^\lambda)$; $\mathsf{par} \leftarrow (\mathbb{G}, p, g)$
$x \leftarrow\$ \mathbb{Z}_p$; $X \leftarrow g^x$; $\mathsf{sk} \leftarrow x$; $\mathsf{pk} \leftarrow X$
$\ell' \leftarrow 0$ ; $\mathcal{I} \leftarrow \varnothing$
Map $\hat{\mathsf{r}}[\cdot] : \{0,1\}^\lambda \rightarrow \{0,1\}^{2\lambda}$
Map $\hat{b}[\cdot] : \{0,1\}^* \rightarrow \{0,1\}$,
$b[\cdot] : \{0,1\}^\lambda \rightarrow \{0,1\}$,
$t[\cdot] : \{0,1\}^\lambda \rightarrow \mathbb{Z}_p$ // $\mathbf{G}_6^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
$\mathcal{L} \leftarrow \varnothing$                                   // $\mathbf{G}_7^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
$Y \leftarrow\$ \mathbb{G}$                                   // $\mathbf{G}_8^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
$\{(m_k^*, \sigma_k^*)\}_{k \in [\ell+1]} \leftarrow\$ \mathcal{A}^{\mathrm{S}}(\mathsf{par}, \mathsf{pk})$
If $\exists\ k_1 \neq k_2, m_{k_1}^* = m_{k_2}^*$ then
    return 0
If $\exists\ k \in [\ell+1]$ such that
    $\mathsf{BS}_R.\mathsf{Ver}(\mathsf{pk}, m_k^*, \sigma_k^*) = 0$ then
        return 0
$m^\star \leftarrow m^*_{\arg\min\{k \in [\ell+1] : (\cdot, m_k^*) \notin \mathcal{L}\}}$
If $\hat{b}[m^\star] = 0$ or $\exists(\mu, m) \in \mathcal{L}, b[\mu] = 1$
    then return 0            // $\mathbf{G}_7^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
Return 1

Oracle $\mathsf{H}(\mu)$:
If $\mathsf{H}(\mu) \neq \perp$ then return $\mathsf{H}(\mu)$
If $\exists(m, \cdot), \mathsf{H}(m, \cdot) = \mu$
    then $b[\mu] \leftarrow \hat{b}[m]$
    Else, $b[\mu] \leftarrow 0$ // $\mathbf{G}_6^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
$\mathsf{H}(\mu) \leftarrow\$ \mathbb{G}$                                   // $\mathbf{G}_4^{\mathcal{A}} - \mathbf{G}_7^{\mathcal{A}}$
$t[\mu] \leftarrow\$ \mathbb{Z}_p$
$\mathsf{H}(\mu) \leftarrow Y^{b[\mu]} g^{t[\mu]}$                                   // $\mathbf{G}_8^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
Return $\mathsf{H}(m)$
```

```
Oracle $\mathrm{S}(\mathrm{sid}, \mathsf{open})$:
If $\mathrm{sid} \in \mathcal{I}$ then return $\perp$
$\ell' \leftarrow \ell' + 1$ ; $\mathcal{I} \leftarrow \mathcal{I} \cup \{\mathrm{sid}\}$
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{open}$
If $\mathsf{Check}(\mathsf{open}) = 0$ then return $\perp$
$i^* \leftarrow\$ \mathsf{DetectCheat}(\mathsf{open})$
If $i^* = \perp$ then **abort game**
$(\mu, \varepsilon) \leftarrow \hat{\mathsf{r}}[\mathsf{com}_{i^*, \vec{J}_{i^*}}]$
If $b[\mu] = 1$ then **abort game**
If $\exists(m, \cdot), \mathsf{H}(m, \cdot) = \mu$
    then $\mathcal{L} \leftarrow \mathcal{L} \cup \{(\mu, m)\}$ // $\mathbf{G}_7^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
For $i \in [K] \backslash \{i^*\}$ : $\mathsf{sk}_{i,\mathrm{sid}} \leftarrow\$ \mathbb{Z}_p$
$\mathsf{sk}_{i^*,\mathrm{sid}} \leftarrow \mathsf{sk} - \sum_{i \neq i^*} \mathsf{sk}_{i,\mathrm{sid}}$
For $i \in [K]$ : $\mathsf{pk}_i \leftarrow g^{\mathsf{sk}_{i,\mathrm{sid}}}$
$\bar{S} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_{i,\mathrm{sid}}}$ // $\mathbf{G}_4^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$
For $i \in [K] \backslash \{i^*\}$:
    $\mathsf{sk}_{i,\mathrm{sid}} \leftarrow\$ \mathbb{Z}_p$; $\mathsf{pk}_{i,\mathrm{sid}} \leftarrow g^{\mathsf{sk}_{i,\mathrm{sid}}}$
$\mathsf{pk}_{i^*,\mathrm{sid}} \leftarrow \mathsf{pk} \prod_{i \neq i^*} \mathsf{pk}_{i,\mathrm{sid}}^{-1}$
$\bar{S} \leftarrow \mathsf{pk}_{i^*,\mathrm{sid}}^{t[\mu]} \prod_{i \neq i^*} h_{i,\vec{J}_i}^{\mathsf{sk}_{i,\mathrm{sid}}}$ // $\mathbf{G}_9^{\mathcal{A}}$
Return $((\mathsf{pk}_i)_{i \in [K]}, \bar{S})$

Oracle $\mathsf{H}_\mu(\mathrm{str})$:
If $\mathsf{H}_\mu(\mathrm{str}) \neq \perp$ then return $\mathsf{H}_\mu(\mathrm{str})$
If $\mathrm{str} = (m, \varphi)$ and      // $\mathbf{G}_6^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
        $\nexists(m, \cdot), \mathsf{H}_\mu(m, \cdot) \neq \perp$
    then $\hat{b}[m] \leftarrow \begin{cases} 1 & \text{w.p. } 1/(\ell+1) \\ 0 & \text{otherwise} \end{cases}$
$\mathsf{H}_\mu(\mathrm{str}) \leftarrow\$ \{0,1\}^\lambda$
If $\exists \mathrm{str}' \neq \mathrm{str}, \mathsf{H}_\mu(\mathrm{str}) = \mathsf{H}_\mu(\mathrm{str}')$
    or $\mathsf{H}(\mathsf{H}_\mu(\mathrm{str})) \neq \perp$                         // $\mathbf{G}_5^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
    then **abort game**.
Return $\mathsf{H}_\mu(\mathrm{str})$
```

Fig. 15. The games $\mathbf{G}_4^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$ for the proof of Lemma 5.6 continued from Figure 14. We omitted the description of the oracles $\mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}$ and the algorithm $\mathsf{DetectCheat}$ as they are identical to their description in $\mathbf{G}_4^{\mathcal{A}}$ throughout this sequence of games. Each box type indicates the changes made in the game contained in the box and to make things clearer, for each box, we indicate which game contains the box by a comment by the side of it.

1. If for some $j \in [N]$, $\hat{\mathsf{r}}_{i,j} = \perp$, then $j = \vec{J}_i$. For other $j' \neq \vec{J}_i$, since $\mathsf{r}_{i,j'}$ is revealed in open and $\mathsf{Check}(\mathsf{open}) = 1$, $\mathsf{com}_{i,j'} = \mathsf{H}_{\mathsf{r}}(\mathsf{r}_{i,j'})$, by the change in $\mathbf{G}_2^{\mathcal{A}}$, $\hat{\mathsf{r}}_{i,j'} \neq \perp$.
2. If for some $j \in [N]$, $\hat{\mathsf{r}}_{i,j} = (\mu, \varepsilon) \neq \perp$, but $h_{i,j} \neq \mathsf{H}(\mu) \cdot g^\beta$ where $\beta \leftarrow \mathsf{H}_\beta(\varepsilon_{i,j})$, then $j = \vec{J}_i$. This is because of the no collision condition in $\mathsf{H}_{\mathsf{com}}$ we introduced in $\mathbf{G}_1^{\mathcal{A}}$, so for $j' \neq \vec{J}_i$, $\hat{\mathsf{r}}_{i,j'} = \mathsf{r}_{i,j'} = (\mu_{i,j'}, \varepsilon_{i,j'})$. Also, with $\mathsf{Check}(\mathsf{open}) = 1$, $h_{i,j} = \mathsf{H}(\mu_{i,j'}) \cdot g^{\mathsf{H}_\beta(\varepsilon_{i,j'})}$.

We say *the adversary $\mathcal{A}$ successfully cheats in instance* $i \in [K]$ if one of the two cases above occurs (as a reminder $\mathsf{Check}(\mathsf{open}) = 1$). Since the values $\hat{\mathsf{r}}_{i,j}$ are fixed when $\vec{J} := \mathsf{H}_{cc}(\mathsf{com}, h)$ is queried and $\vec{J}$ is uniformly random, the probability which $\mathcal{A}$ successfully cheats in instance $i \in [K]$ is at most $1/N$. By $\vec{J}_i$ sampled independently, the probability in which $\mathcal{A}$ successfully cheats in all instance is at most $1/N^K$.

Game $\mathbf{G}_3^{\mathcal{A}}$: This game is identical to $\mathbf{G}_2^{\mathcal{A}}$ except that the game aborts if $\mathcal{A}$ successfully cheats in all instance $i \in [K]$ in some signing interaction. By the previous discussion, and applying the union bound over all signing sessions

$$\Pr[\mathbf{G}_3^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_2^{\mathcal{A}} = 1] - \frac{\ell}{N^K} .$$

Game $\mathbf{G}_4^{\mathcal{A}}$: This game is identical to $\mathbf{G}_3^{\mathcal{A}}$ except that in the signing oracle S, we change how sk_i's are generated. Recall that in $\mathbf{G}_3^{\mathcal{A}}$, the game aborts if in any signing session, $\mathcal{A}$ successfully cheats in all instances. Hence, there exists an instance $i* \in [K]$ such that $\mathcal{A}$ did not successfully cheat. Fixing $i*$ as the first such instance, the game generates sk_i's as follows:

$$\mathsf{sk}_i \leftarrow_{\$} \mathbb{Z}_p, \text{ for } i \in [K] \backslash \{i*\}, \mathsf{sk}_{i*} \leftarrow \mathsf{sk} - \sum_{i \in [K] \backslash i*} \mathsf{sk}_i \; .$$

Here, pk_i's are defined in the same manner as before. Since sk_i's are still random additive share of sk, the view of $\mathcal{A}$ remains the same. Thus,

$$\mathsf{Pr}[\mathbf{G}_4^{\mathcal{A}} = 1] = \mathsf{Pr}[\mathbf{G}_3^{\mathcal{A}} = 1] \; .$$

Game $\mathbf{G}_5^{\mathcal{A}}$: This game is identical to $\mathbf{G}_4^{\mathcal{A}}$ except that there is an introduced abort regarding H and H_μ. More specifically, if $\mathcal{A}$ queries $\mathsf{H}(\mu)$ and later there is a query $\mathsf{H}_\mu(x) = \mu$, the game will abort. When μ is queried and there is no x such that $\mathsf{H}_\mu(x) = \mu$, then as $\mathsf{H}_\mu(\cdot)$ outputs a uniformly random string in $\{0,1\}^\lambda$, the probability that the abort occurs is at most $Q_{\mathsf{H}} Q_{\mathsf{H}_\mu} / 2^\lambda$ by applying the union bound over all pairs of H and H_μ queries. Thus,

$$\mathsf{Pr}[\mathbf{G}_5^{\mathcal{A}} = 1] \geqslant \mathsf{Pr}[\mathbf{G}_4^{\mathcal{A}} = 1] - \frac{Q_{\mathsf{H}} Q_{\mathsf{H}_\mu}}{2^\lambda} \; .$$

Game $\mathbf{G}_6^{\mathcal{A}}$: This game is identical to $\mathbf{G}_5^{\mathcal{A}}$ except that the game introduces two maps $\hat{b}[\cdot], b[\cdot]$ such that when $\mathcal{A}$ queries $\mathsf{H}_\mu(m, \varphi)$ and no query in the form of $(m, *)$ has occurred before, $\hat{b}[m]$ is set to 1 with probability $1/(\ell+1)$ and 0 otherwise. Moreover, when there is a query $\mathsf{H}(\mu)$ of which the value is not yet defined, the game searches for a previous query (m, φ) such that $\mathsf{H}_\mu(m, \varphi) = \mu$ and sets $b[\mu] = \hat{b}[m]$. If such query does not exist, set $b[\mu] = 0$. Note that with the no collision condition introduced in $\mathbf{G}_6^{\mathcal{A}}$, there is at most one (m, φ) for each μ such that $\mathsf{H}_\mu(m, \varphi) = \mu$.

Since $\hat{b}[\cdot], b[\cdot]$ does not affect the view of $\mathcal{A}$, we have

$$\mathsf{Pr}[\mathbf{G}_6^{\mathcal{A}} = 1] = \mathsf{Pr}[\mathbf{G}_5^{\mathcal{A}} = 1] \; .$$

Game $\mathbf{G}_7^{\mathcal{A}}$: This game changes from $\mathbf{G}_6^{\mathcal{A}}$ as follows:

- Introduce a list $\mathcal{L} \leftarrow \varnothing$.
- In a call to the signing oracle S where there exists an instance $i* \in [K]$ such that the adversary did not successfully cheat, fix $i*$ as the first of such instance and let $\bar{\mathsf{r}}_{i*, \vec{J}_{i*}} = (\mu, \varepsilon)$ be the extracted value such that $\mathsf{H}_{\mathsf{com}}(\bar{\mathsf{r}}_{i*, \vec{J}_{i*}}) = \mathsf{com}_{i*, \vec{J}_{i*}}$. The game then tries to extract m from μ through H_μ. If it succeeds, (μ, m) is added to $\mathcal{L}$. Also, if $b[\mu] = 1$, the game aborts.
- At the end of the game where $\mathcal{A}$ successfully outputs forgeries $(m_k^*, \sigma_k^*)_{k \in [\ell+1]}$, pick the first $m^\star \in \{m_1^*, \ldots, m_{\ell+1}^*\}$ such that no $(\cdot, m^\star) \in \mathcal{L}$. If $\hat{b}[m^\star] = 0$, abort. We know $|\mathcal{L}| \leqslant \ell < \ell + 1$ as there is at most ℓ calls to S which is the only place $\mathcal{L}$ increases in size. Hence, by pigeonhole principle, there exists $m^\star$ as described.

In the case that the game does not abort, the view of $\mathcal{A}$ remains the same as in $\mathbf{G}_6^{\mathcal{A}}$. Also, the game only aborts during the simulation of oracle S when (μ, m) is added to $\mathcal{L}$. If (μ, m) is not added to $\mathcal{L}$, it means the game could not extract μ, so, by the change in $\mathbf{G}_6^{\mathcal{A}}$, $b[\mu] = 0$. Thus,

$$\begin{aligned} \mathsf{Pr}[\mathbf{G}_7^{\mathcal{A}} = 1] &= \mathsf{Pr}[\mathcal{A} \text{ succeeds} | \mathbf{G}_7^{\mathcal{A}} \text{ does not abort}] \mathsf{Pr}[\mathbf{G}_7^{\mathcal{A}} \text{ does not abort}] \\ &= \mathsf{Pr}[\mathbf{G}_6^{\mathcal{A}} = 1] \mathsf{Pr}[\hat{b}[m^\star] = 1 \wedge (\forall (\mu, m) \in \mathcal{L}, b[\mu] = 0)] \; . \end{aligned}$$

If $\mathsf{H}_\mu(m,\cdot)=\mu$, then $b[\mu]=\hat{b}[m]$ as $b[\mu]$ is defined when $\mathsf{H}(\mu)$ is called and by the change in $\mathbf{G}_5^{\mathcal{A}}$, the query $\mathsf{H}_\mu(m,\cdot)=\mu$ cannot come later than $\mathsf{H}(\mu)$. Thus, we can lower bound the following

$$\begin{aligned}\Pr[\hat{b}[m^\star]=1\wedge(\forall(\mu,m)\in\mathcal{L},b[\mu]=0)]&=\Pr[\hat{b}[m^\star]=1\wedge\forall(\mu,m)\in\mathcal{L},b[m]=0]\\&=\Pr[\hat{b}[m^\star]=1]\Pr[\forall(\mu,m)\in\mathcal{L},b[m]=0]\\&\geqslant\frac{1}{\ell+1}\cdot\left(1-\frac{1}{\ell+1}\right)^{\ell}\\&=\frac{1}{\ell}\cdot\left(1-\frac{1}{\ell+1}\right)^{\ell+1}\geqslant\frac{1}{4\ell}\ .\end{aligned}$$

The equality in the second line is because $\hat{b}[m^\star]$ is defined independently from $\hat{b}[m]$ for $(\mu,m)\in\mathcal{L}$. The inequality in the third line follows from $\Pr[\hat{b}[m]=1]=1/(\ell+1)$ for any m and the number distinct messages m in $\mathcal{L}$ is at most $|\mathcal{L}|\leqslant\ell$. The last inequality comes from $(1-1/x)^x\geqslant 1/4$ for all $x\geqslant 2$. Hence,

$$\Pr[\mathbf{G}_7^{\mathcal{A}}=1]\geqslant\frac{1}{4\ell}\Pr[\mathbf{G}_6^{\mathcal{A}}=1]\ .$$

Game $\mathbf{G}_8^{\mathcal{A}}$: In this game, we change how the random oracle query $\mathsf{H}(\mu)$ is answered. First, at the beginning of the game, the game samples $Y\leftarrow_\$\mathbb{G}$. Then, for each new query $\mathsf{H}(\mu)$, define $b[\mu]$ as in $\mathbf{G}_6^{\mathcal{A}}$, and set $\mathsf{H}(\mu)\leftarrow Y^{b[\mu]}g^{t[\mu]}$ for $t[\mu]\leftarrow_\$\mathbb{Z}_p$. Since the view of $\mathcal{A}$ does not change as the distribution of $\mathsf{H}(\mu)$ is still uniformly random over $\mathbb{G}$ and the winning event stays the same,

$$\Pr[\mathbf{G}_8^{\mathcal{A}}=1]=\Pr[\mathbf{G}_7^{\mathcal{A}}=1]\ .$$

Game $\mathbf{G}_9^{\mathcal{A}}$: This game changes how $(\mathsf{pk}_i)_{i\in[K]}$ and $\bar{S}$ is computed in the signing oracle S. The game now generates $(\mathsf{pk}_i)_{i\in[K]},\bar{S}$ as follows:

- Recall from the change in $\mathbf{G}_3^{\mathcal{A}}$ that if the game does not abort, then there exists an instance i^* which is the smallest instance out of $[K]$ where $\mathcal{A}$ does not successfully cheat in. The game then generates
$$\mathsf{sk}_i\leftarrow_\$\mathbb{Z}_p,\mathsf{pk}_i\leftarrow g^{\mathsf{sk}_i}\text{ for }i\neq i^*,\mathsf{pk}_{i^*}\leftarrow\mathsf{pk}\prod_{i\neq i^*}\mathsf{pk}_i^{-1}\ .$$
- Also, since i^*-th instance is a not a successful cheat instance, the game can extract $\mathsf{r}_{i^*,\vec{J}_{i^*}}=(\mu,\varepsilon)\neq\perp$ and $b[\mu]=0$. Otherwise, the game will abort because of the change in $\mathbf{G}_7^{\mathcal{A}}$. Let $\beta=\mathsf{H}_\beta(\varepsilon)$, so that
$$h_{i^*,\vec{J}_{i^*}}=\mathsf{H}(\mu)g^\beta=Y^{b[\mu]}g^{t[\mu]}g^\beta=g^{\beta+t[\mu]}\ .$$
- Lastly, set
$$\bar{S}=\mathsf{pk}_{i^*}^{\beta+t[\mu]}\cdot\prod_{i\neq i^*}h_{i,\vec{J}_i}^{\mathsf{sk}_i}\ .$$

Since the pk_i's have the same distribution and $\mathsf{pk}_{i^*}^{\beta+t[\mu]}=\left(g^{\beta+t[\mu]}\right)^{\mathsf{sk}_{i^*}}=h_{i,\vec{J}_i}^{\mathsf{sk}_{i^*}}$, the view of $\mathcal{A}$ remains the same. Hence,

$$\Pr[\mathbf{G}_9^{\mathcal{A}}=1]=\Pr[\mathbf{G}_8^{\mathcal{A}}=1]\ .$$

Lastly, by Lemma 5.8 below stating that there exists an adversary $\mathcal{B}'$ playing CDH game with running time $t_{\mathcal{B}'}\approx t_{\mathcal{A}}$ such that $\Pr[\mathbf{G}_9^{\mathcal{A}}=1]\leqslant\mathsf{Adv}^{\mathrm{cdh}}_{\mathsf{GGen}}(\mathcal{B}',\lambda)$, we have

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathsf{BS}_R}(\mathcal{A},\lambda)\leqslant\frac{Q^2_{\mathsf{H}_{\mathsf{com}}}+Q^2_{\mathsf{H}_\mu}+Q_{\mathsf{H}_{\mathsf{com}}}Q_{\mathsf{H}_{cc}}+Q_{\mathsf{H}}Q_{\mathsf{H}_\mu}}{2^\lambda}+\frac{\ell}{N^K}+4\ell\cdot\mathsf{Adv}^{\mathrm{cdh}}_{\mathsf{GGen}}(\mathcal{B}',\lambda)\ .$$

Lemma 5.8. *There exists an adversary $\mathcal{B}'$ playing the* CDH *game with running time $t_{\mathcal{B}'}\approx t_{\mathcal{A}}$ such that* $\Pr[\mathbf{G}_9^{\mathcal{A}}=1]\leqslant\mathsf{Adv}^{\mathrm{cdh}}_{\mathsf{GGen}}(\mathcal{B}',\lambda)$.

Proof. To show the claim, we give a reduction $\mathcal{B}'$ playing the CDH game using the adversary $\mathcal{A}$ as a subroutine. $\mathcal{B}'$ is defined as follows:

1. The reduction $\mathcal{B}'$ takes as input a CDH instance $(\mathbb{G}, p, g, X, Y)$ and runs $\mathcal{A}$ with $\mathsf{par} = (\mathbb{G}, p, g), \mathsf{pk} = X$.
2. The simulations of $\mathsf{H}, \mathsf{H}_\mu, \mathsf{H}_{cc}, \mathsf{H}_{\mathsf{com}}$ are as in $\mathbf{G}_9^{\mathcal{A}}$ including the aborts defined in the game. Specifically for the random oracle H, $\mathcal{B}$ uses the element Y from the CDH instance to program the oracle queries.
3. The signing oracle S is also simulated as in $\mathbf{G}_9^{\mathcal{A}}$ including the aborts defined in the game. We note that this simulation can be done efficiently as the signing oracle in $\mathbf{G}_9^{\mathcal{A}}$ does not require the secret key sk.
4. At last after receiving the forgery $(m_k, \sigma_k)_{k \in [\ell+1]}$ from $\mathcal{A}$, $\mathcal{B}'$ checks if all the messages are distinct. If not, it aborts. Also, it finds $m^\star$ and abort as in $\mathbf{G}_7^{\mathcal{A}}$ if $\hat{b}[m^\star] = 0$.
5. Lastly, let $(m^\star, \sigma^\star)$ be the message-signature pair corresponding to $m^\star$ defined above. Parse $\sigma^\star = ((\mathsf{pk}_i^\star, \varphi_i^\star)_{i \in [K]}, \bar{S}^\star)$, and let $\mu_i^\star = \mathsf{H}_\mu(m^\star, \varphi_i^\star)$ for $i \in [K]$. Then, since $\mathsf{H}(\mu_i^\star)$ is called in the verification algorithm, $t[\mu_i^\star]$ is defined and is known to $\mathcal{B}'$. $\mathcal{B}'$ returns

$$Z = \bar{S}^\star \cdot \prod_{i=1}^{K} \mathsf{pk}_i^{\star\, -t[\mu_i^\star]} .$$

It is clear that the running time of $\mathcal{B}'$ is about that of $\mathcal{A}$. Next, we analyze the success probability of $\mathcal{B}'$. Let $\mathsf{sk} = x = \log_g X$ and $y = \log_g Y$. Consider when $\mathcal{A}$ wins $\mathbf{G}_9^{\mathcal{A}}$, i.e., the game does not abort and for all $k \in [\ell + 1]$, the component $\bar{S}_k^*$ in (m_k^*, σ_k^*) satisfies

$$\bar{S}_k^* = \prod_{i=1}^{K} \mathsf{H}(\mu_{k,i}^*)^{\mathsf{sk}_{k,i}} = \prod_{i=1}^{K} \mathsf{pk}_{k,i}^{*\ \ y \cdot b[\mu_{k,i}^*] + t[\mu_{k,i}^*]}$$

where $\mathsf{sk}_{k,i} = \log_g \mathsf{pk}_{k,i}$.

Since $\hat{b}[m^\star] = 1$ or else $\mathcal{B}'$ aborts, $b[\mu_i^\star] = 1$ for all $i \in [K]$ as $\mathsf{H}_\mu(m^\star, \varphi_i^\star) = \mu_i^\star$ is always queried before $\mathsf{H}(\mu_i^\star)$ by the change in $\mathbf{G}_5^{\mathcal{A}}$. Thus, if $\mathcal{A}$ wins in game $\mathbf{G}_9^{\mathcal{A}}$

$$\bar{S}^\star = \prod_{i=1}^{K} \mathsf{pk}_i^{\star\, y \cdot b[\mu_i^\star] + t[\mu_i^\star]} = \left(\prod_{i=1}^{K} \mathsf{pk}_i^\star \right)^y \prod_{i=1}^{K} \mathsf{pk}_i^{\star\, t[\mu_i^\star]} = \mathsf{pk}^y \prod_{i=1}^{K} \mathsf{pk}_i^{\star\, t[\mu_i^\star]} .$$

Hence, $Z = \bar{S}^\star \cdot \prod_{i=1}^{K} \mathsf{pk}_i^{\star\, -t[\mu_i^\star]} = \mathsf{pk}^y = X^y$, which is the CDH solution to the instance (X, Y). Therefore, we conclude that

$$\Pr[\mathbf{G}_9^{\mathcal{A}} = 1] \leqslant \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{cdh}}(\mathcal{B}', \lambda) . \qquad \square$$

Acknowledgments

The authors wish to thank Renas Bacho, Julian Loss, and Benedikt Wagner for discussions regarding our weaker security notion. This research was partially supported by NSF grants CNS-2026774, CNS-2154174, a JP Morgan Faculty Award, a CISCO Faculty Award, and a gift from Microsoft.

References

Abe01. Masayuki Abe. A secure three-move blind signature scheme for polynomially many signatures. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 136–151. Springer, Heidelberg, May 2001.

Ame05. American National Standards Institute, Inc. ANSI X9.62 public key cryptography for the financial services industry: the elliptic curve digital signature algorithm (ECDSA), November 16, 2005.

AO00. Masayuki Abe and Tatsuaki Okamoto. Provably secure partially blind signatures. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *LNCS*, pages 271–286. Springer, Heidelberg, August 2000.

App. icloud private relay overview. https://www.apple.com/privacy/docs/iCloud_Private_Relay_Overview_Dec2021.PDF.

BDF$^+$11. Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, Heidelberg, December 2011.

BDL$^+$12. Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang. High-speed high-security signatures. *Journal of Cryptographic Engineering*, 2(2):77–89, September 2012.

BL13. Foteini Baldimtsi and Anna Lysyanskaya. Anonymous credentials light. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 1087–1098. ACM Press, November 2013.

BLL$^+$21. Fabrice Benhamouda, Tancrède Lepoint, Julian Loss, Michele Orrù, and Mariana Raykova. On the (in)security of ROS. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part I*, volume 12696 of *LNCS*, pages 33–53. Springer, Heidelberg, October 2021.

BLS01. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, Heidelberg, December 2001.

BLS04. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. *Journal of Cryptology*, 17(4):297–319, September 2004.

BN06. Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 390–399. ACM Press, October / November 2006.

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. *Journal of Cryptology*, 16(3):185–215, June 2003.

Bol03. Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 31–46. Springer, Heidelberg, January 2003.

BP02. Mihir Bellare and Adriana Palacio. GQ and Schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 162–177. Springer, Heidelberg, August 2002.

BR93. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93*, pages 62–73. ACM Press, November 1993.

BR06. Mihir Bellare and Phillip Rogaway. The security of triple encryption and a framework for code-based game-playing proofs. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 409–426. Springer, Heidelberg, May / June 2006.

Bra94. Stefan Brands. Untraceable off-line cash in wallets with observers (extended abstract). In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 302–318. Springer, Heidelberg, August 1994.

BRJZ23. Paulo L. Barreto, Devin D. Reich, Marcos A. Simplicio Jr., and Gustavo H. M. Zanon. Blind signatures from zero knowledge in the kummer variety. Cryptology ePrint Archive, Paper 2023/1484, 2023. https://eprint.iacr.org/2023/1484.

BZ23. Paulo L. Barreto and Gustavo H. M. Zanon. Blind signatures from zero-knowledge arguments. Cryptology ePrint Archive, Report 2023/067, 2023. https://eprint.iacr.org/2023/067.

CAHL$^+$22. Rutchathon Chairattana-Apirom, Lucjan Hanzlik, Julian Loss, Anna Lysyanskaya, and Benedikt Wagner. PI-cut-choo and friends: Compact blind signatures via parallel instance cut-and-choose and more. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part III*, volume 13509 of *LNCS*, pages 3–31. Springer, Heidelberg, August 2022.

CDS94. Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 174–187. Springer, Heidelberg, August 1994.

CFN90. David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 319–327. Springer, Heidelberg, August 1990.

Cha82. David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO'82*, pages 199–203. Plenum Press, New York, USA, 1982.

Che05. Benoît Chevallier-Mames. An efficient CDH-based signature scheme with a tight security reduction. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 511–526. Springer, Heidelberg, August 2005.

CKM$^+$23. Elizabeth C. Crites, Chelsea Komlo, Mary Maller, Stefano Tessaro, and Chenzhi Zhu. Snowblind: A threshold blind signature in pairing-free groups. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology - CRYPTO 2023 - 43rd Annual International Cryptology Conference, CRYPTO 2023, Santa Barbara, CA, USA, August 20-24, 2023, Proceedings, Part I*, volume 14081 of *Lecture Notes in Computer Science*, pages 710–742. Springer, 2023.

CP93. David Chaum and Torben P. Pedersen. Wallet databases with observers. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 89–105. Springer, Heidelberg, August 1993.

Fis06. Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 60–77. Springer, Heidelberg, August 2006.

FKL18. Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 33–62. Springer, Heidelberg, August 2018.

FOO93. Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A practical secret voting scheme for large scale elections. In Jennifer Seberry and Yuliang Zheng, editors, *AUSCRYPT'92*, volume 718 of *LNCS*, pages 244–251. Springer, Heidelberg, December 1993.

FPS20. Georg Fuchsbauer, Antoine Plouviez, and Yannick Seurin. Blind schnorr signatures and signed ElGamal encryption in the algebraic group model. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 63–95. Springer, Heidelberg, May 2020.

FS87. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987.

FW22. Georg Fuchsbauer and Mathias Wolf. (Concurrently secure) blind schnorr from schnorr. Cryptology ePrint Archive, Report 2022/1676, 2022. https://eprint.iacr.org/2022/1676.

GJ03. Eu-Jin Goh and Stanislaw Jarecki. A signature scheme as secure as the Diffie-Hellman problem. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 401–415. Springer, Heidelberg, May 2003.

Goo. Vpn by google one, explained. https://one.google.com/about/vpn/howitworks.

HIP$^+$21. Scott Hendrickson, Jana Iyengar, Tommy Pauly, Steven Valdez, and Christopher A. Wood. Private Access Tokens. Internet-Draft draft-private-access-tokens-01, Internet Engineering Task Force, October 2021. Work in Progress.

HKKL07. Carmit Hazay, Jonathan Katz, Chiu-Yuen Koo, and Yehuda Lindell. Concurrently-secure blind signatures without random oracles or setup assumptions. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 323–341. Springer, Heidelberg, February 2007.

HKL19. Eduard Hauck, Eike Kiltz, and Julian Loss. A modular treatment of blind signatures from identification schemes. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 345–375. Springer, Heidelberg, May 2019.

HLTW22. Lucjan Hanzlik, Julian Loss, Sri AravindaKrishnan Thyagarajan, and Benedikt Wagner. Sweep-UC: Swapping coins privately. Cryptology ePrint Archive, Report 2022/1605, 2022. https://eprint.iacr.org/2022/1605.

HLW23. Lucjan Hanzlik, Julian Loss, and Benedikt Wagner. Rai-choo! Evolving blind signatures to the next level. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 753–783. Springer, Heidelberg, April 2023.

IKOS07. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Zero-knowledge from secure multiparty computation. In David S. Johnson and Uriel Feige, editors, *39th ACM STOC*, pages 21–30. ACM Press, June 2007.

JLO97. Ari Juels, Michael Luby, and Rafail Ostrovsky. Security of blind digital signatures (extended abstract). In Burton S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 150–164. Springer, Heidelberg, August 1997.

KLP17. Eike Kiltz, Julian Loss, and Jiaxin Pan. Tightly-secure signatures from five-move identification protocols. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 68–94. Springer, Heidelberg, December 2017.

KLR21. Jonathan Katz, Julian Loss, and Michael Rosenberg. Boosting the security of blind signature schemes. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part IV*, volume 13093 of *LNCS*, pages 468–492. Springer, Heidelberg, December 2021.

KLX22. Julia Kastner, Julian Loss, and Jiayu Xu. On pairing-free blind signature schemes in the algebraic group model. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022, Part II*, volume 13178 of *LNCS*, pages 468–497. Springer, Heidelberg, March 2022.

Mau94. Ueli M. Maurer. Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete algorithms. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 271–281. Springer, Heidelberg, August 1994.

MS23. Alexander May and Carl Richard Theodor Schneider. Dlog is practically as hard (or easy) as DH - solving dlogs via DH oracles on EC standards. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2023(4):146–166, 2023.

NY89. Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *21st ACM STOC*, pages 33–43. ACM Press, May 1989.

Oka94. Tatsuaki Okamoto. Designated confirmer signatures and public-key encryption are equivalent. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *LNCS*, pages 61–74. Springer, Heidelberg, August 1994.

OO92. Tatsuaki Okamoto and Kazuo Ohta. Universal electronic cash. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 324–337. Springer, Heidelberg, August 1992.

PCM. PCM: Click fraud prevention and attribution sent to advertiser. https://webkit.org/blog/11940/pcm-click-fraud-prevention-and-attribution-sent-to-advertiser/. Accessed: 2021-09-30.

Poi98. David Pointcheval. Strengthened security for blind signatures. In Kaisa Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 391–405. Springer, Heidelberg, May / June 1998.

PS00. David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, June 2000.

Sch90. Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252. Springer, Heidelberg, August 1990.

Sch91. Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, January 1991.

Tru. Trust tokens. https://developer.chrome.com/docs/privacy-sandbox/trust-tokens/.

TZ22. Stefano Tessaro and Chenzhi Zhu. Short pairing-free blind signatures with exponential security. In Orr Dunkelman and Stefan Dziembowski, editors, *EUROCRYPT 2022, Part II*, volume 13276 of *LNCS*, pages 782–811. Springer, Heidelberg, May / June 2022.

Unr17. Dominique Unruh. Post-quantum security of Fiat-Shamir. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 65–95. Springer, Heidelberg, December 2017.

A Deferred Protocol-Style Figures

$$\begin{array}{lcl}
\underline{\mathsf{BS}_1.\mathsf{S}(\mathsf{par}, \mathsf{sk} = x)} & & \underline{\mathsf{BS}_1.\mathsf{U}(\mathsf{par} = (\mathbb{G}, p, g, W), \mathsf{pk} = X, m)} \\
X \leftarrow g^x & & h' \leftarrow \mathsf{H}(m) \\
Z \leftarrow h^x & \xleftarrow{\qquad h \qquad} & \beta \overset{\$}{\leftarrow} \mathbb{Z}_p; h \leftarrow h' \cdot g^\beta \\
z_1, e, r_0, s \overset{\$}{\leftarrow} \mathbb{Z}_p & & \\
R_g \leftarrow g^{r_0}; R_h \leftarrow h^{r_0} & & \\
A \leftarrow g^{z_1} W^{-e} & & \\
\delta \leftarrow \mathsf{H}''(h, X, Z, g^s, h^s) & & \\
\pi \leftarrow (\delta, s' = \delta \cdot x + s) & \xrightarrow{(Z, \pi, R_g, R_h, A)} & \textbf{if } \delta \neq \mathsf{H}''(h, X, Z, g^{s'} X^{-\delta}, h^{s'} Z^{-\delta}) : \\
 & & \quad \textbf{return } \bot \\
 & & \alpha_0, \alpha_1, \gamma_0, \gamma_1 \overset{\$}{\leftarrow} \mathbb{Z}_p \\
 & & Z' \leftarrow Z \cdot X^{-\beta}; R'_g \leftarrow R_g X^{-\gamma_0} g^{\alpha_0} \\
 & & R'_h \leftarrow R_h R_g^{-\beta} Z'^{-\gamma_0} h'^{\alpha_0} \\
 & & A' \leftarrow A W^{-\gamma_1} g^{\alpha_1} \\
 & & c' \leftarrow \mathsf{H}'(m, h', Z', R'_g, R'_h, A') \\
d \leftarrow c - e & \xleftarrow{\qquad c \qquad} & c \leftarrow c' - \gamma_0 - \gamma_1 \\
z_0 \leftarrow r_0 + d \cdot x & \xrightarrow{(d, e, z_0, z_1)} & \textbf{if } c \neq e + d \textbf{ or} \\
 & & \quad (R_g \cdot X^d, R_h \cdot Z^d) \neq (g^{z_0}, h^{z_0}) \textbf{ or} \\
 & & \quad A \cdot W^e \neq g^{z_1} : \textbf{return } \bot \\
 & & d' \leftarrow d + \gamma_0; e' \leftarrow e + \gamma_1 \\
 & & z'_0 \leftarrow z_0 + \alpha_0; z'_1 \leftarrow z_1 + \alpha_1 \\
 & & \textbf{return } \sigma \leftarrow (Z', d', e', z'_0, z'_1)
\end{array}$$

Fig. 16. Protocol style figure for the signing protocol of BS_1

$\mathsf{BS}_2.\mathsf{S}(\mathsf{par}, \mathsf{sk} = x)$		$\mathsf{BS}_2.\mathsf{U}(\mathsf{par} = (\mathbb{G}, p, g, W), \mathsf{pk} = X, m)$
$X \leftarrow g^x$		
$z_1, e, r_0, s \leftarrow\$ \mathbb{Z}_p$		
$R_g \leftarrow g^{r_0}; A \leftarrow g^{z_1} W^{-e}$		$\alpha_0, \alpha_1, \gamma_0, \gamma_1 \leftarrow\$ \mathbb{Z}_p$
	$\xrightarrow{(R_g, A)}$	$R'_g \leftarrow R_g X^{-\gamma_0} g^{\alpha_0}; A' \leftarrow A W^{-\gamma_1} g^{\alpha_1}$
		$h' \leftarrow \mathsf{H}(m, R'_g, A')$
$Z \leftarrow h^x; R_h \leftarrow h^{r_0}$	$\xleftarrow{h}$	$\beta \leftarrow\$ \mathbb{Z}_p; h \leftarrow h' \cdot g^\beta$
$\delta \leftarrow \mathsf{H}''(h, X, Z, g^s, h^s)$		
$\pi \leftarrow (\delta, s' = \delta \cdot x + s)$	$\xrightarrow{(Z, \pi, R_h)}$	**if** $\delta \neq \mathsf{H}''(h, X, Z, g^{s'} X^{-\delta}, h^{s'} Z^{-\delta})$:
		return $\bot$
		$Z' \leftarrow Z \cdot X^{-\beta};$
		$R'_h \leftarrow R_h R_g^{-\beta} Z'^{-\gamma_0} h'^{\alpha_0}$
		$c' \leftarrow \mathsf{H}'(m, h', Z', R'_g, R'_h, A')$
$d \leftarrow c - e$	$\xleftarrow{c}$	$c \leftarrow c' - \gamma_0 - \gamma_1$
$z_0 \leftarrow r_0 + d \cdot x$	$\xrightarrow{(d, e, z_0, z_1)}$	**if** $c \neq d + e$ **or**
		$(R_g \cdot X^d, R_h \cdot Z^d) \neq (g^{z_0}, h^{z_0})$ **or**
		$A \cdot W^e \neq g^{z_1}$: **return** $\bot$
		$d' \leftarrow d + \gamma_0; e' \leftarrow e + \gamma_1$
		$z'_0 \leftarrow z_0 + \alpha_0; z'_1 \leftarrow z_1 + \alpha_1$
		return $\sigma \leftarrow (Z', d', e', z'_0, z'_1)$

Fig. 17. Protocol style figure for the signing protocol of BS_2

$\underline{\mathsf{BS}_3.\mathsf{S}(\mathsf{par} = (\mathbb{G}, p, g, W, N, K), \mathsf{sk} = x)}$		$\underline{\mathsf{BS}_3.\mathsf{U}(\mathsf{par}, \mathsf{pk} = X, m)}$
for $i \in [K-1] : \mathsf{sk}_i \xleftarrow{\$} \mathbb{Z}_p$		**for** $(i,j) \in [K] \times [N]$:
$\mathsf{sk}_K \leftarrow \mathsf{sk} - \sum_{i=1}^{K-1} \mathsf{sk}_i$		$\quad \varphi_{i,j} \xleftarrow{\$} \{0,1\}^\lambda, \mu_{i,j} \leftarrow \mathsf{H}_\mu(m, \varphi_{i,j})$
for $i \in [K] : \mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i}$		$\quad \varepsilon_{i,j} \xleftarrow{\$} \{0,1\}^\lambda, \beta_{i,j} \leftarrow \mathsf{H}_\beta(\varepsilon_{i,j})$
$z_1, e \xleftarrow{\$} \mathbb{Z}_p; \vec{r}_0 \xleftarrow{\$} \mathbb{Z}_p^K$		$\quad \mathsf{r}_{i,j} \leftarrow (\mu_{i,j}, \varepsilon_{i,j}), \mathsf{com}_{i,j} \leftarrow \mathsf{H}_{\mathsf{com}}(\mathsf{r}_{i,j})$
$A \leftarrow g^{z_1} W^{-e}$		$\quad h'_{i,j} \leftarrow \mathsf{H}(\mu_{i,j}), h_{i,j} \leftarrow h'_{i,j} \cdot g^{\beta_{i,j}}$
		$\mathsf{com} \leftarrow (\mathsf{com}_{i,j})_{(i,j) \in [K] \times [N]}$
		$h \leftarrow (h_{i,j})_{(i,j) \in [K] \times [N]}; \vec{J} \leftarrow \mathsf{H}_{cc}(\mathsf{com}, h)$
if $\mathsf{Check}(\mathsf{open}) = 0$: **abort**		$\mathsf{open} \leftarrow (\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]})$
$\bar{S} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_i}$	$\xleftarrow{\quad \mathsf{open} \quad}$	
$\pi \leftarrow \Pi.\mathsf{Prove}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}), (\mathsf{sk}_i)_{i \in [K]})$		$\mathsf{pk}_K \leftarrow \mathsf{pk} \cdot \prod \mathsf{pk}_i^{-1}$
$\vec{R} \leftarrow (g^{\vec{r}_{0,1}}, \ldots, g^{\vec{r}_{0,K}})$		**if** $\Pi.\mathsf{Verify}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}), \pi) = 0$:
$\bar{R} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{r}_{0,i}}$	$\xrightarrow{(\mathsf{pk}_i)_{i \in [K-1]}, \bar{S}, \pi, \vec{R}, \bar{R}, A}$	$\qquad$ **abort**
		$((\mathsf{pk}'_i)_{i \in [K]}, \bar{S}', \vec{\tau}) \xleftarrow{\$}$
		$\quad \mathsf{ReRa}((\mathsf{pk}_i, h'_{i,\vec{J}_i})_{i \in [K]}, \bar{S} \cdot \prod_{i=1}^K \mathsf{pk}_i^{-\beta_{i,\vec{J}_i}})$
		$\alpha_1, \gamma_0, \gamma_1 \xleftarrow{\$} \mathbb{Z}_p, \vec{\alpha}_0 \xleftarrow{\$} \mathbb{Z}_p^K$
		for $i \in [K] : \vec{R}'_i \leftarrow \vec{R}_i \mathsf{pk}_i'^{-\gamma_0} g^{\vec{\alpha}_{0,i}}$
		$\bar{R}' \leftarrow \bar{R} \bar{S}'^{-\gamma_0} \prod_{i=1}^K \vec{R}_i^{-\beta_{i,\vec{J}_i}} h'^{\vec{\alpha}_{0,i}}_{i,\vec{J}_i}$
		$A' \leftarrow A W^{-\gamma_1} g^{\alpha_1}$
		$c' \leftarrow \mathsf{H}'(m, (h'_{i,\vec{J}_i}, \mathsf{pk}'_i)_{i \in [K]}, \bar{S}', \vec{R}', \bar{R}', A')$
		$c \leftarrow c' - \gamma_0 - \gamma_1$
$d \leftarrow c - e$	$\xleftarrow{\quad c \quad}$	
for $i \in [K]$:	$\xrightarrow{\quad d, e, \vec{z}_0, z_1 \quad}$	
$\quad \vec{z}_{0,i} \xleftarrow{\$} d \cdot \mathsf{sk}_i + \vec{r}_{0,i}$		**if** $d + e \neq c'$ or
		$\quad \exists i \in [K], \vec{R}_i \mathsf{pk}_i^d \neq g^{\vec{z}_{0,i}}$ or
		$\quad \bar{R} \bar{S}^d \neq \prod_{i=1}^{K} h_{i,\vec{J}_i}^{\vec{z}_{0,i}}$ or
		$\quad g^{z_1} \neq W^e \cdot A$: **abort**
		$d' \leftarrow d + \gamma_0; e' \leftarrow e + \gamma_1$
		$\vec{z}'_0 \leftarrow \vec{z}_0 + \vec{\alpha}_0 + d \cdot \vec{\tau}; z'_1 \leftarrow z_1 + \alpha_1$
		$\sigma \leftarrow ((\mathsf{pk}'_i, \varphi_{i,\vec{J}_i})_{i \in [K]}, \bar{S}', d', e', \vec{z}'_0, z'_1)$
		return σ

Fig. 18. Protocol style figure of BS_3. The proof system Π and the algorithms Check and ReRa are defined in Figures 8 and 10.