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. We present the first concurrently-secure blind signatures making black-box use of a pairing-free group for which unforgeability, in the random oracle model, can be proved *without* relying on the algebraic group model (AGM), thus resolving a long-standing open question. Prior pairing-free blind signatures without AGM proofs have only been proved secure for bounded concurrency, relied on computationally expensive non-black-box use of NIZKs, or had complexity growing with the number of signing sessions due to the use of boosting techniques.

Our most efficient constructions rely on the chosen-target CDH assumption 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. The underlying signing protocols consist of four (in order to achieve regular unforgeability) or five moves (for strong unforgeability). All schemes are proved statistically blind in the random oracle model.

1 Introduction

Blind signatures [Cha82] are interactive protocols that allow a user to obtain a signature on a message 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 into 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].

Pairing-free blind signatures. There are at least two reasons that make it desirable 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$^+$22] to increase the number of concurrent sessions where a scheme such as Okamoto-Schnorr remains secure. The current state of the art [CAHL$^+$22] requires a 7-move protocol of which the communication and computational complexity grow logarithmically and linearly, respectively, 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

[⋆] A preliminary version of this paper appears in the proceedings of CRYPTO 2024. This is the full version.

incorrect, and was only recently re-stablished in the AGM [KLX22]. Similarly, all other provably secure solutions [FPS20, TZ22, CKM+23b] fundamentally rely on the AGM. Therefore, this paper aims to address the following central question.

> *Can we give blind signatures in pairing-free groups whose concurrent security, in the ROM, can be proved without the AGM?*

Non-black-box baselines. It is however often overlooked that, in principle, we can provide an affirmative answer to this question by relying on expensive non-black-box techniques. For example, we can instantiate Fischlin's transform [Fis06] using generic NIZKs with online extractability in the ROM, such as those from the MPC-in-the-head paradigm [IKOS07]. 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. The recent work by Fuchsbauer and Wolf [FW24] also relies on generic NIZKs, and assumes Schnorr signatures to be secure for a given fixed (non random oracle) hash function. The resulting protocol has four moves, and is non-black-box as well.

We point out here that concurrent work [KNR24] made progress in instantiating a variant of Fischlin's transform without generic NIZKs while relying on the Strong RSA assumption and the DDH assumption in pairing-free groups. However, their construction does not fundamentally leverage pairing-free elliptic curves, as it is built on top of an RSA-based signature while using DDH to instantiate components which have no RSA-based instantiation. Here, we aim for a solution purely based on black-box use of groups, without additional external assumptions.

Our contribution. We propose the first blind signatures making black-box use of a pairing-free group whose concurrent security is proved *without relying* on the AGM. We assume the ROM as well as variants of the Computational Diffie-Hellman (CDH) assumption. In particular, unlike the aforementioned pairing-free instantiations, we do not rely on implementing group operations as part of a relation verified by a NIZK proof.

Our results are summarized in Table 1. Our most efficient constructions are based on the *chosen target CDH (CT-CDH) assumption*, a falsifiable assumption introduced by Boldyreva [Bol03] to prove security (in the pairing setting) of Blind BLS [BLS01], which is a one-more version of CDH.[1] The signing protocols take four and five moves, respectively, with 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, and the signing protocol requires four moves.

One-more unforgeability. Our CT-CDH schemes BS_1 and BS_2 achieve a weaker than usual notion of one-more (strong) unforgeability (which we refer to as OM(S)UF-1) where a malicious user cannot come up with more signatures than the number of sessions it engages in, regardless of whether these terminate or not. In constrast, our CDH-based scheme BS_3 achieves the standard notion [JLO97] that only counts terminating sessions (we refer to this as OMUF-2).

Some applications inherently require OMUF-2 (e.g., the atomic swap construction from [HLTW24]). Nonetheless, we consider both BS_1 and BS_2 to be valuable, despite the weaker security they achieve. First of all, they are simpler and serve as stepping stones towards BS_3. Moreover, while this calls for a more careful analysis, OM(S)UF-1 appears sufficient for many applications. For example, in constructions of anonymous tokens [HIP+21, Tru], the weaker OMUF-1 notion means that the server needs to regard a token as issued as long as the first-round message to the user is sent. The advantage of OMUF-2 is that it guarantees that if the signing protocol aborts, the user will not come up with a valid token, but this does not appear to be important in this context, as the decision to issue a token has been made prior to starting the protocol.

This weaker form of accounting for sessions is also common in the definition of unforgeability used to prove security of many prominent threshold signatures, such as e.g., SPARKLE [CKM23a].

[1] We avoid the naming "one-more CDH" to avoid ambiguity, as an alternative interpretation is used e.g. in [BLT+24].

Table 1. Overview of our results and comparison with existing schemes in the AGM with provable security notions, number of moves, signature size, communication cost (note: $p = |\mathbb{G}|$), and assumptions required for each security notion.

Scheme	Security*	Mvs.	Sig. size	Comm.	Blind Asmp.	OMUF Asmp.**
BS_1 (Sec. 3)†	comp./stat. blindness & OMUF-1	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-1	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-2	4	$(\lambda+1)\ \mathbb{G} + (\lambda+7)\ \mathbb{Z}_p + \lambda^2$ bits	$(3\lambda+6)\ \mathbb{G} + (2\lambda+9)\ \mathbb{Z}_p + (\lambda+3\lambda^2)$ bits	ROM	CDH
Abe [Abe01, KLX22]	comp. blind & OMSUF-2	3	$2\ \mathbb{G} + 6\ \mathbb{Z}_p$	$3\ \mathbb{G} + 6\ \mathbb{Z}_p + \lambda$ bits	DDH	DL + AGM
Clause Blind Schnorr [FPS20]	perf. blind & OMSUF-2	3	$1\ \mathbb{G} + 1\ \mathbb{Z}_p$	$2\ \mathbb{G} + 4\ \mathbb{Z}_p$	-	DL + AGM + mROS
Snowblind [CKM+23b]	perf. blind & OMSUF-2	3	$1\ \mathbb{G} + 2\ \mathbb{Z}_p$	$2\ \mathbb{G} + 4\ \mathbb{Z}_p$	-	DL + AGM

(*): OMUF-X security (for $\mathsf{X} = 1, 2$) guarantees that no adversary can output $\ell + 1$ message-signature pairs with distinct messages (with distinct pairs for OMSUF-X), where ℓ denotes the number of started (for $\mathsf{X} = 1$) or completed (for $\mathsf{X} = 2$) signing sessions. (**): All OMUF guarantees assume the ROM. (†): For BS_1 and BS_2, we give a computationally blind version and a less efficient statistically blind one. (‡): The efficiencies of BS_3 depend on two parameters set to $N = 2$ and $K = \lambda$.

BLINDNESS. For all 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, our random oracle proofs only require 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].

OPEN PROBLEMS: DLOG & ROUND REDUCTION. An elusive open problem is to give blind signatures based solely on the hardness of the DL problem (or the stronger OMDL assumption), without resorting to NIZKs. Indeed, techniques from recent works in the AGM [KLX22, TZ22, CKM+23b] are not robust to rewinding in several subtle ways. One may argue the qualitative improvement is not significant (for several curves, indeed, DL and CDH are somewhat equivalent [Mau94, MS23]), but even in the non-blind setting, signatures with security based on DL tend to be actually more efficient. For example, it seems unlikely that we can obtain a three-move scheme without considering DL-based schemes. It should also be noted that we do not expect two-move schemes to be possible even in the AGM.

Recent work by Barreto and Zanon [BZ23] (expanded in [BRSJZ23]) claims a solution with 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 blind signature scheme. The proof appears to have some gaps, and we note that in general IMP-CA security does not yield concurrently secure blind signatures. For instance, Schnorr identification [BP02] achieves IMP-CA but 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 OMUF-1) and BS_2 (achieving OMSUF-1), in Sections 3 and 4 respectively. Lastly, we discuss the scheme BS_3 achieving OMUF-2 based on the CDH assumption in Section 5.

1.1 Technical Overview

CT-CDH based schemes. 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 $R_h = h^z Z^{-c}$, 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 $c = \mathsf{H}'(m, h, Z, R_g, R_h)$.

ONE-MORE UNFORGEABILITY. Our first goal is to prove that the above scheme achieves the weaker variant of one-more unforgeability (OMUF-1), i.e., the adversary cannot produce signatures for $\ell + 1$ *distinct messages* after initiating at most ℓ signing sessions. To do so, we rely on the hardness of the *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-1, 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) of a message m_i, by the soundness property 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.

The challenge here is that the DLEQ proof is merely honest-verifier zero-knowledge, and the adversary $\mathcal{A}$ sends an *arbitrary* challenge c to the signer, 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 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. Namely, the soundness of the OR proof π does not guarantee that $Z = h^{\mathsf{sk}}$, as it is possible, in principle, to use the witness w to generate a valid signature (Z, π) for m where $Z \neq \mathsf{H}(m)^{\mathsf{sk}}$. Our key observation here is that any adversary 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 [PS00]. 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: by fixing (R_g, R_h) and since $Z \neq \mathsf{H}(m)^{\mathsf{sk}}$, there exists *at most one* d that can generate an accepting (R_g, R_h, d, z). Therefore, $d = d'$ must hold, and hence $e \neq e'$.

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 $\log_g W$.

If we do not want blindness to rely on the discrete logarithm assumption, we can alternatively let the signer send 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.

STRONG UNFORGEABILITY. BS_1 is not strongly unforgeable, 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 $\lceil \log p \rceil$ signing sessions and outputs $\lceil \log p \rceil + 1$ distinct signatures for the message m. To transform BS_1 into a strongly unforgeable scheme, referred to as BS_2, the idea is to let H also take (R_g, A) as input, i.e., the group elements from the OR proof which are 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.

Achieving OMUF-2 from CDH. Our security proof of BS_1 fails to show the usual one-more unforgeability notion, i.e. OMUF-2, which guarantees that the adversary cannot output more message-signature pairs than the number of *completed* signing sessions. Indeed, the reduction queries its DH oracle to obtain $Z = h^{\mathsf{sk}}$ in order to answer the first-round query for each signing session, and thus, needs to output more solutions than the number of *started* sessions.

One possible fix is that instead of sending Z and R_h in the clear, we let the signer send *commitments* of Z and R_h, denoted by com_Z and com_{R_h} respectively, in the first round. Later in the second round, the signer opens these commitments accordingly. If the commitment scheme is *homomorphic* (with respect to the group operation) and *equivocable*, then, we can adapt the security reduction to simulate the signing protocol given $w = \log_g W$ as follows: (1) in the first signing round, generate hcom_Z as a random commitment and compute hcom_{R_h} from hcom_Z using the homomorphic property of the commitment scheme (the original reduction computed R_h from Z), (2) in the second signing round, query the DH oracle for $Z = h^{\mathsf{sk}}$ and use equivocation to open com_Z to Z. The interactive proof can still be simulated using w as in the proof of BS_1. Notice that the number of DH oracle queries is now the number of completed signing sessions, as we only query the oracle when completing the last round.

Unfortunately, all existing homomorphic equivocal commitments based on pairing-free groups [BCJ08, PW23, PW24] can only equivocate a random commitment to a group element of which the discrete logarithm to some pre-established base is known. This is not the case for Z obtained from the DH oracle, as h is adversarially chosen and the reduction does not know sk. To address this, we instead realize that a better starting point is to rely on a scheme which is secure under the CDH assumption directly. In particular, to obtain our third scheme BS_3, we go through the following two steps, which we explain below:

1. We apply ideas similar to those used for BS_1 above to a recently proposed pairing-based blind signature scheme, called Rai-Choo [HLW23], which only relies on the plain CDH assumption. Doing so, we obtain a pairing-free OMUF-1-secure blind signature scheme based on CDH.
2. We then realize that the structure of the resulting scheme and its security proof will allow us to upgrade its security to OMUF-2 using pairing-free homomorphic equivocal commitments.

PAIRING-FREE RAI-CHOO. Abstractly, one can interpret the CT-CDH assumption as stating the unforgeability of an interactive version of BLS signatures implemented in a pairing-free setting where efficient verification is not possible (the DH oracle is the signing oracle, and the challenge oracle corresponds to the random oracle). Similarly, as an intermediate abstraction, we can think of a game that captures the unforgeability of (non-blind) Rai-Choo in a pairing-free setting (where, again, efficient verifiability is lost). Its signing protocol proceeds as follows (where $\mathsf{pk} = g^{\mathsf{sk}}$):

- On an input message m, the user computes, for $(i, j) \in [K] \times [N]$, a commitment $\mu_{i,j} \leftarrow \mathsf{H}_\mu(m, \varphi_{i,j})$ to m and a random value $\varphi_{i,j}$, a commitment $\mathsf{com}_{i,j} \leftarrow \mathsf{H}_{\mathsf{com}}(\mu_{i,j})$, and a group element $h_{i,j} \leftarrow \mathsf{H}(\mu_{i,j})$. Then, it computes $\vec{J} \leftarrow \mathsf{H}_{cc}((\mathsf{com}_{i,j}, h_{i,j})_{i\in[K],j\in[N]}) \in [N]^K$, describing cut-and-choose indices for which the user has to reveal $\mu_{i,j}$ for all $i \in [K]$ and $j \neq \vec{J}_i$. Finally, the message sent to the signer is $(\vec{J}, ((\mu_{i,j})_{j\neq\vec{J}_i}, h_{i,\vec{J}_i}, \mathsf{com}_{i,\vec{J}_i})_{i\in[K]})$.
- The signer then recomputes $(\mathsf{com}_{i,j}, h_{i,j})_{i\in[K],j\neq\vec{J}_i}$ and checks that $\vec{J} = \mathsf{H}_{cc}((\mathsf{com}_{i,j}, h_{i,j})_{i,j})$. If the check passes, it uniformly samples $(\mathsf{sk}_i)_{i\in[K]}$ conditioning on $\sum_{i=1}^K \mathsf{sk}_i = \mathsf{sk}$ and sends $((\mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i})_{i\in[K]}, \bar{S} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_i})$.
- The final signature is $\sigma = ((\mathsf{pk}_i, \varphi_{i,\vec{J}_i})_{i\in[K]}, \bar{S})$ and inefficient verification checks whether $\mathsf{pk} = \prod_{i=1}^K \mathsf{pk}_i$ and $\bar{S} = \prod_{i=1}^K \mathsf{H}(\mathsf{H}_\mu(m, \varphi_{i,\vec{J}_i}))^{\log_g \mathsf{pk}_i}$.

Similar to BS_1, to translate this signing protocol into a blind signature scheme with efficient verification, we extend it to have the signer interact with the user to generate a non-interactive proof π that shows the knowledge of either the witness $\log_g W$ or the witness $\{\mathsf{sk}_i\}_{i\in[K]}$ such that $\mathsf{pk}_i = g^{\mathsf{sk}_i}$ and $\bar{S} = \prod_{i=1}^K \mathsf{H}(\mathsf{H}_\mu(m, \varphi_{i,\vec{J}_i}))^{\mathsf{sk}_i}$. The final signature consists of σ and π.

One can show that if there exists an adversary that breaks OMUF-1 for this scheme, then either (1) the adversary outputs one more valid Rai-Choo signatures than the number of signing sessions, which breaks the OMUF of Rai-Choo (and in turns this can be reduced to breaking the CDH assumption), or (2) the adversary outputs an invalid Rai-Choo signature but with a valid OR proof (and this can be reduced to finding the discrete logarithm of W).

<u>Upgrading to OMUF-2.</u> Still, this approach can only show OMUF-1 security for the scheme. The rather technical reason is due to how the random-oracle programming of $\mathsf{H}(\mathsf{H}_\mu(m, \varphi))$ is carried out in the reduction to CDH behind Step 1. Essentially, if the user signs honestly, the first-round message sent in the k-th session uniquely links this session with a message $m^{(k)}$, which can be extracted from the prior random-oracle queries. To properly simulate the signer's response to the first message in the k-th session, the reduction needs to ensure that, with sufficiently high probability, the random oracles are set up so that the discrete logarithm of $\mathsf{H}(\mathsf{H}_\mu(m^{(k)}, \varphi^{(k)}_{i,\vec{J}_i}))$ is known for some $i \in [K]$. For this reason, no CDH solution can be extracted from a signature on any of the messages associated with such a session. Therefore, for the reduction to succeed, a forgery needs to contain a signature for a message which was not associated with one of the sessions, regardless of whether these sessions were actually concluded.

To upgrade to OMUF-2 security, we instead use a homomorphic commitment scheme HECom with *special equivocation* (formally defined in Section 5.1) derived from the commitment scheme in [BCJ08]. More precisely, the scheme can embed a base $X \neq 1_{\mathbb{G}}$ into the commitment key, which then allows opening a commitment of a group element S to another element $S' = SX^c$ for any c thanks to a trapdoor generated along with the key. Then, instead of sending $\bar{S}$ in clear, we let the signer send the commitment $\mathsf{hcom}_{\bar{S}}$ of $\bar{S}$. Then, in the second round, the signer sends the opening of the commitment along with the same OR proof response.

While we defer the rather involved details to the body of the paper, the crucial point is that this will enable a new reduction which only needs to know the discrete logarithm of the $\mathsf{H}(\mathsf{H}_\mu(m^{(k)}, \varphi^{(k)}_{i,\vec{J}_i}))$'s if the k-th session indeed reaches the final message and terminates.

2 Preliminaries

<u>Notation.</u> For a positive integer n, we write $[n]$ for $\{1, \ldots, n\}$. We use λ to denote the security parameter. A *group parameter generator* is a probabilistic polynomial time algorithm GGen that takes an input 1^λ and

```
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 $\mathrm{CT\text{-}CDH}^{\mathcal{A}}_{\mathsf{GGen}}(\lambda)$ :
    $(\mathbb{G}, p, g) \leftarrow\$ \mathsf{GGen}(1^\lambda)$ ; $x \leftarrow\$ \mathbb{Z}_p$
    $\mathrm{cid} \leftarrow 0$ ; $\ell \leftarrow 0$
    $(j_i, \hat{Z}_i)_{i \in [\ell+1]} \leftarrow \mathcal{A}^{\textsc{Chal},\textsc{Dh}}(\mathbb{G}, p, g, g^x)$
    If $|\{j_1, \ldots, j_{\ell+1}\}| = \ell + 1$ and $\forall\, i \in [\ell+1] : \hat{Z}_i = Y^x_{j_i}$ then
        return 1
    Return 0

Oracle $\textsc{Chal}$ :
    $\mathrm{cid} \leftarrow \mathrm{cid} + 1$
    $Y_{\mathrm{cid}} \leftarrow\$ \mathbb{G}$
    Return $Y_{\mathrm{cid}}$

Oracle $\textsc{Dh}(Y)$ :
    $\ell \leftarrow \ell + 1$
    Return $Y^x$
```

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

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) and do not write modular reduction explicitly when it is clear from the context. Also, we write $a = \log_g A \in \mathbb{Z}_p$ for a group element $A \in \mathbb{G}$ where $A = g^a$.

Throughout this paper, we adopt a variant of the "Game-Playing Framework" by Bellare and Rogaway [BR06] for both definitions and proofs.

CRYPTOGRAPHIC ASSUMPTIONS. In this paper, we rely on the assumed hardness of the *discrete logarithm* (DL), the computational Diffie-Hellman (CDH), and the *chosen-target computational Diffie-Hellman* (CT-CDH) [Bol03] problems. To capture these, for any adversary $\mathcal{A}$, we define the advantage of $\mathcal{A}$ playing the games {DLOG, CDH, CT-CDH} (these games are defined in Figure 1) as

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

We note that the hardness of the CT-CDH problem implies the hardness of the CDH problem, which in turns implies the hardness of the DL problem.

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 public 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.[2] *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 a key-pair $(\mathsf{pk}, \mathsf{sk})$ is defined by the following experiments (1) for four-move and (2) for five-move blind signatures:

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

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

[2] We note that all of our schemes also admits an alternative definition without the setup algorithm (see some recent works with this definition [KRS23, KNR24]), by hashing a constant to generate the public parameters.

```
Game $\boxed{\mathrm{OMUF\text{-}X}_{\mathsf{BS}}^{\mathcal{A}}(\lambda)}$, $\boxed{\mathrm{OMSUF\text{-}X}_{\mathsf{BS}}^{\mathcal{A}}(\lambda)}$:
$\mathsf{par} \leftarrow \mathsf{BS.Setup}(1^\lambda)$
$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{BS.KG}(\mathsf{par})$
$\ell \leftarrow 0$ ; $\mathcal{I}_1, \ldots, \mathcal{I}_r \leftarrow \varnothing$
$\{(m_k^*, \sigma_k^*)\}_{k \in [\ell+1]} \leftarrow\$ \mathcal{A}^{\mathrm{S}_1, \ldots, \mathrm{S}_r}(\mathsf{par}, \mathsf{pk})$
$\boxed{\text{If } \exists\ k_1 \neq k_2,\ m_{k_1}^* = m_{k_2}^* \text{ then}}$
$\boxed{\text{If } \exists\ k_1 \neq k_2, (m_{k_1}^*, \sigma_{k_1}^*) = (m_{k_2}^*, \sigma_{k_2}^*) \text{ then}}$
    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 $\mathsf{BS}$ is 5-move and $j = 1$,
                         // the input $\mathsf{umsg}$ 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
    $\boxed{\ell \leftarrow \ell + 1}$                                    // For $\mathsf{X} = 1$
    $(\mathsf{st}_{\mathrm{sid}}^s, \mathsf{smsg}) \leftarrow \mathsf{BS.S}_1(\mathsf{sk}, \mathsf{umsg})$
If $j > 1$ then
    $\boxed{\text{If } j = r \text{ then } \ell \leftarrow \ell + 1}$                 // For $\mathsf{X} = 2$
    $(\mathsf{st}_{\mathrm{sid}}^s, \mathsf{smsg}) \leftarrow \mathsf{BS.S}_j(\mathsf{st}_{\mathrm{sid}}^s, \mathsf{umsg})$
                                              // for $j = r, \mathsf{st}_{\mathrm{sid}}^s = \perp$
Return $\mathsf{smsg}$
```

Fig. 2. The OMUF-X and OMSUF-X 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. The input umsg of S_1 is set as an empty string if BS is 5-move. The highlighted boxes along with the commented X value indicating how ℓ is counted in OMUF-X and OMSUF-X. The OMUF-X game contains everything but the solid boxes, and the OMSUF-X game contains everything but the dashed boxes.

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 corresponding experiment (either (1) or (2)) returns σ such that $\mathsf{BS.Ver}(\mathsf{pk}, m, \sigma) = 1$. All of our schemes are perfectly correct.

Note that since this work exclusively consider four-move and five-move blind signatures, we only give the syntax and security definitions for these objects for the sake of simplicity. However, the definition for k-move blind signatures can easily be obtained by generalizing the given definitions.

<u>One-more unforgeability.</u> We consider variants of *one-more (strong) unforgeability*, denoted OMUF-X and OMSUF-X for $\mathsf{X} \in \{1, 2\}$. OMUF-1 ensures 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. For OMSUF-1, we instead only require the adversary to output $\ell + 1$ distinct message-signature pairs. For the OMUF-2 and OMSUF-2 notions, ℓ is defined as the number of *completed* signing interactions instead, which is the more standard notion of one-more unforgeability used in the literature. The $\mathrm{OMUF\text{-}X}_{\mathsf{BS}}^{\mathcal{A}}$ and $\mathrm{OMSUF\text{-}X}_{\mathsf{BS}}^{\mathcal{A}}$ games for a blind signature scheme BS are defined in Figure 2. The corresponding advantage of $\mathcal{A}$ is defined as $\mathsf{Adv}_{\mathsf{BS}}^{\mathsf{omuf\text{-}X/omsuf\text{-}X}}(\mathcal{A}, \lambda) := \Pr[(\mathrm{OMUF\text{-}X/OMSUF\text{-}X})_{\mathsf{BS}}^{\mathcal{A}}(\lambda) = 1]$.

<u>Blindness.</u> 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 $\mathrm{BLIND}_{\mathsf{BS}}^{\mathcal{A}}$ is defined in Figure 3, and for any adversary $\mathcal{A}$, we define its advantage as $\mathsf{Adv}_{\mathsf{BS}}^{\mathsf{blind}}(\mathcal{A}, \lambda) := \left|\Pr[\mathrm{BLIND}_{\mathsf{BS}}^{\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right|$.

<u>Random oracles.</u> We note that most of our analyses further assume one or more random oracles, and we will clearly indicate so in the theorem statements. The random oracles are modeled as additional oracles to which the adversary $\mathcal{A}$ is given access.

<u>Forking lemma.</u> 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 following experiment*

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

```
Game $\mathrm{BLIND}^{\mathcal{A}}_{\mathsf{BS}}(\lambda)$ :
$\mathsf{par} \leftarrow \mathsf{BS.Setup}(1^\lambda)$
$b \leftarrow\!\!\$ \{0,1\}$
$b_0 \leftarrow b$ ; $b_1 \leftarrow 1 - b$
$b' \leftarrow\!\!\$ \mathcal{A}^{\mathrm{INIT},\mathrm{U}_1,\mathrm{U}_2,\mathrm{U}_3}(\mathsf{par})$
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 $\mathrm{U}_j(i, \mathsf{smsg}^{(i)})$ :                // $j = 1, \dots, 3$
                                   // 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
    $(\mathsf{st}^u_i, \mathsf{umsg}^{(i)}) \leftarrow \mathsf{BS.U}_1(\mathsf{pk}, m_{b_i}, \mathsf{smsg}^{(i)})$
    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 $\mathsf{sess}_0 = \mathsf{sess}_1 = 4$ then
    If $\sigma_0 \neq \perp$ and $\sigma_1 \neq \perp$ then return $(\sigma_0, \sigma_1)$
    Return $(\perp, \perp)$
Return $(i, \mathtt{closed})$
```

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 smsg of U_1 is set as an empty string.

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

- *Pick a random tape ρ for $\mathcal{A}$ and sample $h_1, \dots, h_q \leftarrow\!\!\$ H$.*
- *Run $(I, \mathsf{aux}) \leftarrow \mathcal{A}(x, h_1, \dots, h_q; \rho)$.*
- *If $I = \perp$, return 0.*
- *Sample $h'_I, \dots, h'_q \leftarrow\!\!\$ H$, run $(I', \mathsf{aux}') \leftarrow \mathcal{A}(x, h_1, \dots, h_{I-1}, h'_I, \dots, h'_q; \rho)$.*
- *If $I = I'$ and $h_I \neq h'_I$, return 1. Otherwise, return 0.*

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

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

3 Four-Move Blind Signatures from CT-CDH

We present a four-move blind signature scheme BS_1, described in Figure 4. (A protocol diagram is also presented in Figure 13). The scheme can be viewed as a blind version of the signature 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 (pk, Z) are equal with respect to the base $(g, \mathsf{H}(m))$. 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, and 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 modeled as a random oracle. This property relies on the NIZK proof highlighted in Figure 4 to show equality of discrete logarithms of (pk, Z) to the base (g, h). In Section 3.3, we also show that if we omit this NIZK proof, we still achieve *computational* blindness 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 $\mathcal{A}$ for the game* BLIND *making at most $Q_{\mathsf{H}''} = Q_{\mathsf{H}''}(\lambda)$ queries to H'', modeled as a random oracle, we have*

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

```
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 $\mathsf{H}'$, $\boxed{\mathsf{H}''}$ : $\{0,1\}^* \to \mathbb{Z}_p$
Return $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \boxed{\mathsf{H}''})$

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

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

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

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

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

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

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

Fig. 4. The blind signature scheme $\mathsf{BS}_1 = \mathsf{BS}_1[\mathsf{GGen}]$. The public parameters par, as stated before, are implicit input to every algorithms except $\mathsf{BS}_1.\mathsf{KG}$. The highlighted boxes denote the NIZK proof used to show the equality of discrete logarithm of (pk, Z) to the base (g, h). We also give a protocol diagram of BS_1 in Figure 13.

<span style="font-variant:small-caps">One-more unforgeability.</span> The following theorem establishes the OMUF-1 security 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-1 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 the game OMUF-1 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 $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}', \mathsf{H}''\}$, modeled as random oracles, there exist adversaries $\mathcal{B}$ and $\mathcal{B}'$ for the games DLOG and CT-CDH, respectively, such that*

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

where $\widehat{Q}_{\mathsf{H}'} = Q_{\mathsf{H}'} + \ell + 1$. 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 $Q_{\mathsf{H}} + \ell + 1$ challenge queries to Chal *and ℓ queries to* Dh.

3.1 Correctness of BS_1

Theorem 3.3. BS_1 *satisfies correctness.*

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 $s' = s + \delta \cdot \mathsf{sk}$ and $\mathsf{pk} = g^{\mathsf{sk}}, Z = h^{\mathsf{sk}}$, we have $g^{s'}\mathsf{pk}^{-\delta} = g^s$ and $h^{s'}Z^{-\delta} = h^s$. Thus, $\mathsf{H}''(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^{s'}Z^{-\delta}) = \mathsf{H}''(h, g^{\mathsf{sk}}, Z, g^s, h^s) = \delta$.

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

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

1. $c' = d' + e'$. This follows from $c = d + e$ as $c' = c + \gamma_0 + \gamma_1 = d + e + \gamma_0 + \gamma_1 = d' + e'$.
2. $g^{z_1'}W^{-e'} = A'$. This follows from $z_1' = z_1 + \alpha_1$ and $e' = e + \gamma_1$, as
$$g^{z_1'}W^{-e'} = (g^{z_1}W^{-e})(g^{\alpha_1}W^{-\gamma_1}) = AW^{-\gamma_1}g^{\alpha_1} = A' \ .$$
3. $g^{z_0'}\mathsf{pk}^{-d'} = R_g'$. This follows from $z_0' = z_0 + \alpha_0$ and $d' = d + \gamma_0$, as
$$g^{z_0'}\mathsf{pk}^{-d'} = (g^{z_0}\mathsf{pk}^{-d})(g^{\alpha_0}\mathsf{pk}^{-\gamma_0}) = R_g\mathsf{pk}^{-\gamma_0}g^{\alpha_0} = R_g' \ ,$$
where the second equality follows from the check $R_g\mathsf{pk}^d = g^{z_0}$ in $\mathsf{BS}_1.\mathsf{U}_3$.
4. $h'^{z_0'}Z'^{-e'} = R_h'$. This follows from $z_0' = z_0 + \alpha_0, d' = d + \gamma_0, h' = hg^{-\beta}$, and $Z' = Z\mathsf{pk}^{-\beta}$ as
$$\begin{aligned} h'^{z_0'}Z'^{-d'} &= (h'^{z_0}Z'^{-d})(h'^{\alpha_0}Z'^{-\gamma_0}) \\ &= (h^{z_0}Z^{-d})(g^{z_0}\mathsf{pk}^{-d})^{-\beta}(h'^{\alpha_0}Z'^{-\gamma_0}) \\ &= R_hR_g^{-\beta}Z'^{-\gamma_0}h'^{\alpha_0} = R_h' \ , \end{aligned}$$
where the second to last equality follows from the checks $R_hZ^d = g^{z_0}$ and $R_g\mathsf{pk}^d = g^{z_0}$ in $\mathsf{BS}_1.\mathsf{U}_3$.

By all of the above, we have
$$\begin{aligned} \mathsf{H}'(m, h', Z', g^{z_0'}\mathsf{pk}^{-d'}, h'^{z_0'}Z'^{-d'}, g^{z_1'}W^{-e'}) &= \mathsf{H}'(m, h', Z', R_g', R_h', A') \\ &= c' = d' + e' \ , \end{aligned}$$
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''.
Game $\mathbf{G}_1^{\mathcal{A}}$: This game made the following changes:

- The oracle $\textsc{Init}(\mathsf{pk}, m_0, m_1)$ additionally computes $\mathsf{sk} \leftarrow \log_g \mathsf{pk}$ by exhaustive search.
- For each signing session $i \in \{0, 1\}$, when the oracle $\mathrm{U}_2(i, \mathsf{smsg}_1^{(i)})$ receives $\mathsf{smsg}_1^{(i)}$ from $\mathcal{A}$, it parses $(Z_i, R_{g,i}, R_{h,i}, A_i, \pi_i = (\delta_i, s_i)) \leftarrow \mathsf{smsg}_1^{(i)}$. Then, it computes $S_{g,i} = g^{s_i}\mathsf{pk}^{-\delta_i}$, $S_{h,i} = h_i^{s_i}Z_i^{-\delta_i}$, where h_i is the message returned by $\mathrm{U}_1(i)$, and checks whether $\delta_i = \mathsf{H}''(h_i, \mathsf{pk}, Z_i, S_{g,i}, S_{h,i})$. If this check passes, the game now aborts if $Z_i \neq h_i^{\mathsf{sk}}$.

The success probability of $\mathcal{A}$ only changes when the new abort occurs in either signing sessions, which corresponds to the following event:
$$Z_i \neq h_i^{\mathsf{sk}} \ \wedge \ \delta_i = \mathsf{H}''(h_i, \mathsf{pk}, Z_i, S_{g,i}, S_{h,i}) \ .$$
We will argue that this event occurs with negligible probability. Specifically, with how $S_{g,i}, S_{h,i}$ is defined, 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 $\delta_i \in \mathbb{Z}_p$ that satisfies such equation. Note that $\mathcal{A}$ makes at most $Q_{\mathsf{H}''}$ queries to H'', and if the query $\mathsf{H}''(h_i, \mathsf{pk}, Z_i, S_{g,i}, S_{h,i})$ was not made beforehand, the game makes this query when checking validity of π. Since δ_i is sampled uniformly at random after fixing the query, the probability of the abort occurring in a session is bounded by $(Q_{\mathsf{H}''} + 1)/p$. By the union bound over the two signing sessions,

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

For the last step, we show that the transcript and returned signatures are distributed identically between both cases of $b = 0$ and $b = 1$, which implies $\Pr[\mathbf{G}_1^{\mathcal{A}} = 1] = \frac{1}{2}$ 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 $\mathrm{View}_{\mathcal{A}}$ denote the set of all possible views of $\mathcal{A}$ in the game $\mathbf{G}_1^{\mathcal{A}}$. A view $\Delta \in \mathrm{View}_{\mathcal{A}}$ is of the form $\Delta = (W, \mathsf{pk}, m_0, m_1, T_0, T_1, \sigma_0, \sigma_1)$ 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 oracles in signing session i (we omitted π_i as it is distributed independently of (m_0, m_1) given (h_i, Z_i)), and $\sigma_i = (Z'_i, d'_i, e'_i, z'_{0,i}, z'_{1,i})$ denotes the valid signature for the message m_i. We need to show that the actual adversarial view, denoted as $v_{\mathcal{A}}$, is distributed identically between $b = 0$ and $b = 1$. Since the randomness of $\mathcal{A}$ is fixed, $v_{\mathcal{A}}$ 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 $v_{\mathcal{A}}(\eta)$ 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 view $\Delta \in \mathrm{View}_{\mathcal{A}}$, 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 $Z_i = h_i^{\mathsf{sk}}$ leading to $Z'_{b_i} = Z_i \mathsf{pk}^{-\beta_i} = (h_i g^{-\beta_i})^{\mathsf{sk}} = {h'_{b_i}}^{\mathsf{sk}}$.

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

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

For the "only if" direction, i.e., if $v_{\mathcal{A}}(\eta) = \Delta$, 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 $v_{\mathcal{A}}(\eta) = \Delta$. 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 view Δ where neither the game nor the oracle aborts, we have the following guarantees for $i \in \{0,1\}$:

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

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

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

where Equation (4) follows from the discussion above, Equation (5) follows from the checks 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 $\mathrm{U}_1(i)$ for $i \in \{0,1\}$: recall that the user oracle outputs $\mathsf{H}(m_{b_i})g^{\beta_i}$ and by the value of β_i from Equation (3), $\mathsf{H}(m_{b_i})g^{\beta_i} = h'_{b_i} g^{\beta_i} = h_i$, so the user's first 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 view Δ.

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

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

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

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

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 view Δ. Lastly, the signatures from the oracle U_3, for $i \in \{0,1\}$, are as follows

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

which are exactly the signatures in Δ. □

3.3 Computational Blindness of BS_1 without NIZK

As mentioned earlier, we can remove the NIZK proof from our scheme BS_1 (resulting in a scheme which we will call BS'_1 in this subsection to distinguish from the scheme with NIZK) 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 the game* BLIND *running in time $t_{\mathcal{A}} = t_{\mathcal{A}}(\lambda)$, there exists an adversary $\mathcal{B}$ for the game* DLOG *running in time $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$ such that*

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

Proof. The proof for this theorem mainly follows the proof of Theorem 3.1 with the only difference being the game $\mathbf{G}_1^{\mathcal{A}}$ 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:

- The oracle $\textsc{Init}(\mathsf{pk}, m_0, m_1)$ additionally computes $\mathsf{sk} \leftarrow \log_g \mathsf{pk}$ by exhaustive search.
- For each signing session $i \in \{0,1\}$, when the oracle $\mathrm{U}_3(i, \mathsf{smsg}_2^{(i)})$ is queried, it parses the signer's first and second messages as $(Z_i, R_{g,i}, R_{h,i}, A_i) \leftarrow \mathsf{smsg}_1^{(i)}$ and $(d_i, e_i, z_{0,i}, z_{1,i}) \leftarrow \mathsf{smsg}_2^{(i)}$. 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 message returned by $\mathrm{U}_1(i)$, 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 user messages in the signing protocol, i.e., (h_i, c_i) (since the event occurs before the signatures are returned).

To bound the probability of event Bad_i occurring, we will construct 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 describing $\mathcal{B}$, we make the following observation that h_i and c_i are uniformly random in $\mathbb{G}$ and $\mathbb{Z}_p$ respectively. First, denote $(\beta_i, \alpha_{0,i}, \alpha_{1,i}, \gamma_{0,i}, \gamma_{1,i})$ as the user randomness for signing session $i \in \{0,1\}$. To see this, consider that, as computed in the user algorithm, $h_i = h_i' g^{\beta_i}$ 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 of $\gamma_{1,i}$, 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 randomness $\beta_i, \alpha_{0,i}, \gamma_{0,i}, \gamma_{1,i}$ and a uniformly random 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 user's first message h_i of signing session i in advance.
2. The oracles $\textsc{Init}$, $\mathrm{U}_1(1-i), \mathrm{U}_2(1-i,\cdot)$, and $\mathrm{U}_3(1-i,\cdot)$ are simulated as in the game $\mathbf{G}_0^{\mathcal{A}}$. The oracle $\mathrm{U}_1(i)$ instead of computing the values as usual answers with h_i instead. While for $\mathrm{U}_2(i,\cdot)$, $\mathcal{B}$ returns $c_i \leftarrow\!\!\$\ \mathbb{Z}_p$.
3. For the call to $\mathrm{U}_3(i, \mathsf{smsg}_2^{(i)})$, if the user algorithm does not abort $\mathcal{B}$ rewinds the adversary $\mathcal{A}$ to when it queries $\mathrm{U}_2(i, \mathsf{smsg}_1^{(i)})$ and returns $c_i' \leftarrow\!\!\$\ \mathbb{Z}_p$. 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 $\mathrm{U}_3(i, {\mathsf{smsg}'_2}^{(i)})$, if the user algorithm does not abort, we can parse $(d_i, e_i, z_{0,i}, z_{1,i}) \leftarrow \mathsf{smsg}_2^{(i)}$ and $(d_i', e_i', z_{0,i}', z_{1,i}') \leftarrow {\mathsf{smsg}'_2}^{(i)}$. If $e_i \neq e_i'$, the reduction returns $(z_{1,i} - z_{1,i}')(e_i - e_i')^{-1}$. Otherwise, abort.

It is clear that the running time of $\mathcal{B}$ is about twice of $\mathcal{A}$'s. 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 where 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) $Z_i \neq h_i^{\mathsf{sk}}$.
(ii) $d_i + e_i = c_i \neq c_i' = d_i' + e_i'$.
(iii) $(R_{g,i}, R_{h,i}) = (g^{z_{0,i}} \mathsf{pk}^{-d_i}, h_i^{z_{0,i}} Z_i^{-d_i}) = (g^{z_{0,i}'} \mathsf{pk}^{-d_i'}, h_i^{z_{0,i}'} Z_i^{-d_i'})$.
(iv) $A_i = g^{z_{1,i}} W^{-e_i} = g^{z_{1,i}'} W^{-e_i'}$.

By considering (iii),

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

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

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

Oracle $\mathrm{S}_1(\mathrm{sid}, h)$:
If $\mathrm{sid} \in \mathcal{I}_1$ then return $\perp$
$\ell \leftarrow \ell + 1$ ; $\mathcal{I}_1 \leftarrow \mathcal{I}_1 \cup \{\mathrm{sid}\}$
$Z \leftarrow h^{\mathsf{sk}}$
$z_1, e, r_0 \leftarrow\$ \mathbb{Z}_p$
$R_g \leftarrow g^{r_0}$ ; $R_h \leftarrow h^{r_0}$
$A \leftarrow g^{z_1} W^{-e}$                  // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_3^{\mathcal{A}}$
$z_0, d, r_1 \leftarrow\$ \mathbb{Z}_p$
$R_g \leftarrow g^{z_0} \mathsf{pk}^{-d}$
$R_h \leftarrow h^{z_0} Z^{-d}$
$A \leftarrow g^{r_1}$                         // $\mathbf{G}_4^{\mathcal{A}}$
$s \leftarrow\$ \mathbb{Z}_p$ ; $\delta \leftarrow \mathsf{H}''(h, \mathsf{pk}, Z, g^s, h^s)$
$\pi \leftarrow (\delta, \delta \cdot \mathsf{sk} + s)$          // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_2^{\mathcal{A}}$
$\delta, s' \leftarrow\$ \mathbb{Z}_p$ ; $\pi \leftarrow (\delta, s')$          // $\mathbf{G}_3^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
If $\mathsf{H}''(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^s Z^{-\delta}) \neq \perp$ then
    abort game
$\mathsf{H}''(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^s Z^{-\delta}) \leftarrow \delta$
Return $(Z, R_g, R_h, A, \pi)$

Oracle $\mathrm{S}_2(\mathrm{sid}, c)$ :
If $\mathrm{sid} \notin \mathcal{I}_1$ or $\mathrm{sid} \in \mathcal{I}_2$ then return $\perp$
$\mathcal{I}_2 \leftarrow \mathcal{I}_2 \cup \{\mathrm{sid}\}$
$d \leftarrow c - e$
$z_0 \leftarrow r_0 + d \cdot \mathsf{sk}$                        // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_3^{\mathcal{A}}$
$e \leftarrow c - d$
$z_1 \leftarrow r_1 + e \cdot w$                         // $\mathbf{G}_4^{\mathcal{A}}$
Return $(d, e, z_0, z_1)$
```

Fig. 5. The OMUF-1 $= \mathbf{G}_0^{\mathcal{A}}$ security game for BS_1 and the subsequent games $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$. We remark that H, H' and H'' are modeled as random oracles to which $\mathcal{A}$ has access. Each box type indicates the changes made in the game name contained in the box. Also, to make things clearer, for each box, the comments indicate which game the changes in the boxes correspond to. Moreover, the signer state is omitted and we assume that *each variable initialized in* S_1 *of the same* sid *can be accessed in* S_2.

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, then $\mathcal{B}$ succeeds in the DLOG game. Thus, $\Pr[\mathsf{frk}] \leqslant \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{dlog}}(\mathcal{B}, \lambda)$.

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

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

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

3.4 Proof of Theorem 3.2 (OMUF-1 of BS_1)

To prove one-more unforgeability of 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 and the secret and public keys as $\mathsf{par} \leftarrow\!\!\$\ \mathsf{BS}_1.\mathsf{Setup}(1^\lambda)$ and $(\mathsf{sk}, \mathsf{pk}) \leftarrow\!\!\$\ \mathsf{BS}_1.\mathsf{KG}(\mathsf{par})$. 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 $\mathcal{A}$ queries the signing oracle S_1 for ℓ times and the random oracles H, H' and H'' for $Q_{\mathsf{H}}, Q_{\mathsf{H}'}$ and $Q_{\mathsf{H}''}$ times respectively. At the end of the game, $\mathcal{A}$ outputs $\ell+1$ message-signature pairs $(m_k^*, \sigma_k^*)_{k \in [\ell+1]}$. The adversary $\mathcal{A}$ succeeds if for all $k_1 \neq k_2, m_{k_1}^* \neq m_{k_2}^*$ and for all $k \in [\ell+1]$, $\mathsf{BS}_1.\mathsf{Ver}(\mathsf{pk}, m_k^*, \sigma_k^*) = 1$. We w.l.o.g. assume that $\mathcal{A}$ does not make the same random oracle query twice. Also, we assume that $\mathcal{A}$ makes the random oracle queries that would be made in $\mathsf{BS}_1.\mathsf{Ver}$ when verifying the forgeries. This adds at most $\ell+1$ queries to H and H', making the total query count $\widehat{Q}_{\mathsf{H}} = Q_{\mathsf{H}} + \ell + 1$ and $\widehat{Q}_{\mathsf{H}'} = Q_{\mathsf{H}'} + \ell + 1$, respectively. The success probability of $\mathcal{A}$ in game $\mathbf{G}_0^{\mathcal{A}}$ is exactly its advantage in the game OMUF-1, i.e.,

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

Game $\mathbf{G}_1^{\mathcal{A}}$: This game is identical to $\mathbf{G}_0^{\mathcal{A}}$ except that for the message-signature pairs $(m_k^*, \sigma_k^*)_{k \in [\ell+1]}$ output by the adversary $\mathcal{A}$, for $k \in [\ell+1]$, after parsing $(Z_k^*, d_k^*, e_k^*, z_{0,k}^*, z_{1,k}^*) \leftarrow \sigma_k^*$, the game additionally requires that $Z_k^* = \mathsf{H}(m_k^*)^{\mathsf{sk}}$.

Then, by Lemma 3.5, there exists an adversary $\mathcal{B}$ for the game DLOG, running in time $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$, such that

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

Game $\mathbf{G}_2^{\mathcal{A}}$: This game is identical to $\mathbf{G}_1^{\mathcal{A}}$ except that when generating the group element 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 π by sampling $s', \delta \leftarrow\!\!\$\ \mathbb{Z}_p$ and programming $\mathsf{H}''(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^{s'}Z^{-\delta})$ as δ. The game aborts if H'' is already defined at $(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^{s'}Z^{-\delta})$.

The view of $\mathcal{A}$ is identical to its view in $\mathbf{G}_2^{\mathcal{A}}$ if the game does not abort. Moreover, the game only aborts if $(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^{s'}Z^{-\delta})$ has been queried or programmed beforehand, but $g^{s'}\mathsf{pk}^{-\delta}$ is 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 possible collision events, i.e., 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 simulated by using w instead of sk. More specifically, $(A, R_g, R_h, d, e, z_0, z_1)$ are now generated as follows:

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

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 $(\mathbb{G}, p, g, X)$, samples $w \leftarrow\!\!\$\ \mathbb{Z}_p$, and sets $W \leftarrow g^w$. It then sends $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W), \mathsf{pk} \leftarrow X$ to $\mathcal{A}$.

2. The simulations of H' and H'' are done as in $\mathbf{G}_4^{\mathcal{A}}$. However, for queries to H (labeling each with $j \in [\widehat{Q}_{\mathsf{H}}]$), 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 $\widehat{Q}_{\mathsf{H}} = Q_{\mathsf{H}} + \ell + 1$ queries to $\textsc{Chal}$.)
3. The signing oracles are also simulated as in $\mathbf{G}_4^{\mathcal{A}}$ except for the computation of $Z = h^{\mathsf{sk}}$ in S_1 which is done by querying its $\textsc{Dh}$ oracle instead, i.e., $Z \leftarrow \textsc{Dh}(h)$.
4. After receiving the message-signature pairs $(m_k^*, \sigma_k^*)_{k \in [\ell+1]}$ 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 each $k \in [\ell+1]$ 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 $\mathcal{A}$ succeeds in the game $\mathbf{G}_4^{\mathcal{A}}$, then $\mathcal{A}$ returns $Z_k^* = \mathsf{H}(m_k^*)^{\mathsf{sk}} = Y_{j_k}^{\log_g X}$ for all $k \in [\ell+1]$ where $\mathsf{sk} = \log_g \mathsf{pk} = \log_g X$. Thus, $\mathcal{B}'$ succeeds in the game CT-CDH, as it returns $\ell + 1$ correct CT-CDH solutions while only querying $\textsc{Dh}$ for ℓ times. Therefore, $\Pr[\mathbf{G}_4^{\mathcal{A}} = 1] \leqslant \mathsf{Adv}_{\mathsf{GGen}}^{\text{ct-cdh}}(\mathcal{B}', \lambda)$. Then, by combining all the advantage changes,

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

Lemma 3.5. *There exists an adversary $\mathcal{B}$ for the game* DLOG*, running in time $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$, such that*

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

Proof. Let Bad be the event where $\mathbf{G}_0^{\mathcal{A}}$ outputs 1 but $\mathbf{G}_1^{\mathcal{A}}$ outputs 0. This 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 $k \in [\ell+1]$, $\mathsf{BS}_1.\mathsf{Ver}(\mathsf{pk}, m_k^*, \sigma_k^*) = 1$, and (3) *there exists some $k \in [\ell+1]$ where parsing the signature $(Z_k^*, d_k^*, e_k^*, z_{0,k}^*, z_{1,k}^*) \leftarrow \sigma_k^*$, we have that $Z_k^* \neq \mathsf{H}(m_k^*)^{\mathsf{sk}}$*. 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) 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 wrapper $\mathcal{A}_k$ over the adversary $\mathcal{A}$ where $\mathcal{A}_k$ receives the following inputs: an instance $(\mathbb{G}, p, g, W)$, the output tape $(c_1, \ldots, c_{\widehat{Q}_{\mathsf{H}'}})$ of H', and a random tape ρ.

1. Extract $(\mathsf{sk} \in \mathbb{Z}_p, (s_i \in \mathbb{Z}_p, r_{0,i} \in \mathbb{Z}_p, e_i \in \mathbb{Z}_p, z_{1,i} \in \mathbb{Z}_p)_{i \in [\ell]}, (h_i \in \mathbb{G})_{i \in [\widehat{Q}_{\mathsf{H}}]}, (\delta_i \in \mathbb{Z}_p)_{i \in [Q_{\mathsf{H}''} + \ell]}, \rho')$ from the random tape ρ.
2. Set $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W), \mathsf{pk} \leftarrow g^{\mathsf{sk}}$.
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 simulated as follows:
   - For the signing query with session ID j ($j \in [\ell]$) to S_1 and S_2, use $(\mathsf{sk}, s_i, r_{0,i}, e_i, z_{1,i})$ to answer the query as in $\mathsf{BS}_1.\mathsf{S}_1$ and $\mathsf{BS}_1.\mathsf{S}_2$ respectively.
   - For the i-th query ($i \in [\widehat{Q}_{\mathsf{H}}]$) to H, return h_i.
   - For the i-th query ($i \in [\widehat{Q}_{\mathsf{H}'}]$) to H', return c_i.
   - For the i-th query ($i \in [Q_{\mathsf{H}''} + \ell]$) to H'', return δ_i. (Note: In these queries, we accounted for the queries that the wrapper made to generate π in each query to S_1.)
4. If the event Bad_k does not occur, return $(\bot, \bot)$. Otherwise, return $(I, (m_k^*, \sigma_k^*))$ where I is the index of the query to H' that corresponds to the verification of (m_k^*, σ_k^*). More specifically, after parsing $(Z_k^*, d_k^*, e_k^*, z_{0,k}^*, z_{1,k}^*) \leftarrow \sigma_k^*$, I is the index corresponding to the query (m, h, Z, R_g, R_h, A) to H' where $m = m_k^*$, $h = \mathsf{H}(m), Z = Z_k^*, R_g = g^{z_{0,k}^*}\mathsf{pk}^{-d_k^*}, R_h = h^{z_{0,k}^*}Z^{-d_k^*}, A = g^{z_{1,k}^*}W^{-e_k^*}$. 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 $c_1, \ldots, c_{\widehat{Q}_{\mathsf{H}'}} \leftarrow\!\!\$\ \mathbb{Z}_p$ along with the random tape ρ of $\mathcal{A}_k$.
2. Run $(I, (m, \sigma)) \leftarrow\!\!\$\ \mathcal{A}_k((\mathbb{G}, p, g, W), (c_1, \ldots, c_{\widehat{Q}_{\mathsf{H}'}}); \rho)$.
3. If $I = \perp$, abort. If not, sample $c'_I, \ldots, c'_{\widehat{Q}_{\mathsf{H}'}} \leftarrow\!\!\$\ \mathbb{Z}_p$ and run $(I', (m', \sigma')) \leftarrow\!\!\$\ \mathcal{A}_k((\mathbb{G}, p, g, W), (c_1, \ldots, c_{I-1}, c'_I, \ldots, c'_{\widehat{Q}_{\mathsf{H}'}}); \rho)$.
4. If $I = I'$ and $c'_I \neq c_I$, parse $(Z, d, e, z_0, z_1) \leftarrow \sigma, (Z', d', e', z'_0, z'_1) \leftarrow \sigma'$, 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 $\mathcal{A}$, $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$. Next, 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 message-signature pairs (m, σ) and (m', σ'): (a) are valid signatures corresponding to the I-th query from $\mathcal{A}$ to H' of the form (m, h, Z, R_g, R_h, A) and (b) satisfy $Z \neq \mathsf{H}(m)^{\mathsf{sk}}$ and $Z' \neq \mathsf{H}(m')^{\mathsf{sk}}$. By (a), we know the following

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

We will argue that $d = d'$. First, the equations in (iii) give $Z^{d-d'} = h^{z_0 - z'_0} = g^{(z_0 - z'_0)\log_g h} = \mathsf{pk}^{(d-d')\log_g h} = h^{\mathsf{sk}(d-d')}$. Since $Z \neq h^{\mathsf{sk}}$, only $d = d'$ satisfies the equation. Since $d + e = c_I \neq c'_I = d' + e'$, we have $e \neq e'$. Thus, by (iv), $\mathcal{B}$ returns $(z_1 - z'_1)(e - e')^{-1} = \log_g W$. Hence,

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

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),

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

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 Section 3 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 presented in Figure 6 (a protocol diagram is also presented in Figure 14), and we will show it indeed satisfies OMSUF-1 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).

<u>Blindness.</u> 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. Below, we state a theorem for the latter property and prove it in Section 4.2. While for the version of the scheme without the NIZK, we give the proof for computational blindness in Section 4.3.

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 the game* BLIND *making at most* $Q_{\mathsf{H}''} = Q_{\mathsf{H}''}(\lambda)$ *queries to* H''*, modeled as a random oracle, we have*

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

```
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 $\mathsf{H}'$, $\boxed{\mathsf{H}''}$ : $\{0,1\}^* \to \mathbb{Z}_p$
Return $\mathsf{par} = (\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \boxed{\mathsf{H}''})$

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

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

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

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

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

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

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

Algorithm $\mathsf{BS}_2.\mathsf{U}_3(\mathsf{st}_2^u, \mathsf{smsg}_3)$ :
$(c, Z, Z', R_h, \mathsf{st}_1^u) \leftarrow \mathsf{st}_2^u$
$(m, \beta, \alpha_0, \alpha_1, \gamma_0, \gamma_1, \mathsf{pk}, h', h, R_g, R'_g, A, A') \leftarrow \mathsf{st}_1^u$
$(d, e, z_0, z_1) \leftarrow \mathsf{smsg}_3$
If $c \neq d + e$ or
    $(R_g \mathsf{pk}^d, R_h Z^d) \neq (g^{z_0}, h^{z_0})$ 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 public parameters par, as stated before, are implicit input to every algorithms except $\mathsf{BS}_2.\mathsf{KG}$. The highlighted boxes denote the NIZK proof used to show the equality of discrete logarithm of (pk, Z) to the base (g, h). We also give a protocol diagram of BS_2 in Figure 14.

ONE-MORE UNFORGEABILITY. The following theorem establishes the OMSUF-1 security 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 (OMSUF-1 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 the game* OMSUF-1 *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* $\mathcal{B}_1$ *for the game* DLOG, *and adversaries* $\mathcal{B}'$ *and* $\mathcal{B}_2$ *for the game* CT-CDH, *such that*

$$\mathsf{Adv}^{\mathrm{omsuf}\text{-}1}_{\mathsf{BS}_2}(\mathcal{A}, \lambda) \leqslant \frac{\ell(\ell + Q_{\mathsf{H}''})}{p} + (\ell + 1)\left(\sqrt{\widehat{Q}_{\mathsf{H}'} \mathsf{Adv}^{\mathrm{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)} + \frac{\widehat{Q}_{\mathsf{H}'}}{p}\right) + \mathsf{Adv}^{\mathrm{dlog}}_{\mathsf{GGen}}(\mathcal{B}_1, \lambda) + \mathsf{Adv}^{\mathrm{ct\text{-}cdh}}_{\mathsf{GGen}}(\mathcal{B}_2, \lambda) + \mathsf{Adv}^{\mathrm{ct\text{-}cdh}}_{\mathsf{GGen}}(\mathcal{B}', \lambda) ,$$

where $\widehat{Q}_{\mathsf{H}'} = Q_{\mathsf{H}'} + \ell + 1$. *Furthermore,* $\mathcal{B}$ *and* $\mathcal{B}_1$ *run in time* $t_\mathcal{B} \approx 2t_\mathcal{A}$ *and* $t_{\mathcal{B}_1} \approx t_\mathcal{A}$ *respectively, whereas* $\mathcal{B}'$ *runs in time* $t_{\mathcal{B}'} \approx t_\mathcal{A}$, *makes* $Q_\mathsf{H} + \ell + 1$ *queries to* CHAL, *and* ℓ *queries to* DH, *and lastly,* $\mathcal{B}_2$ *runs in time* $t_{\mathcal{B}_2} \approx t_\mathcal{A}$, *and makes* $\ell + 1$ *queries to* CHAL, *and* ℓ *queries to* DH.

The proof builds on top of the approach for proving OMUF-1 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 that this implies the OMSUF-1 security of BS_2, 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 $(m_j, (Z_j, d_j, e_j, z_{0,j}, z_{1,j}))$ 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.

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 $s' = s + \delta \cdot \mathsf{sk}$ and $\mathsf{pk} = g^{\mathsf{sk}}, Z = h^{\mathsf{sk}}$, we have $g^{s'}\mathsf{pk}^{-\delta} = g^s$ and $h^{s'}Z^{-\delta} = h^s$. Thus, $\mathsf{H}''(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^{s'}Z^{-\delta}) = \mathsf{H}''(h, g^{\mathsf{sk}}, Z, g^s, h^s) = \delta$.

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

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

1. $c' = d' + e'$. This follows from $c = d + e$ as $c' = c + \gamma_0 + \gamma_1 = d + e + \gamma_0 + \gamma_1 = d' + e'$.
2. $g^{z_1'}W^{-e'} = A'$. This follows from $z_1' = z_1 + \alpha_1$ and $e' = e + \gamma_1$, as
$$g^{z_1'}W^{-e'} = (g^{z_1}W^{-e})(W^{-\gamma_1}g^{\alpha_1}) = A(W^{-\gamma_1}g^{\alpha_1}) = A' \ .$$
3. $g^{z_0'}\mathsf{pk}^{-d'} = R_g'$. This follows from $z_0' = z_0 + \alpha_0$ and $d' = d + \gamma_0$, as
$$g^{z_0'}\mathsf{pk}^{-d'} = (g^{z_0}\mathsf{pk}^{-d})(\mathsf{pk}^{-\gamma_0}g^{\alpha_0}) = R_g(\mathsf{pk}^{-\gamma_0}g^{\alpha_0}) = R_g' \ ,$$
where the second equality follows from the check $R_g\mathsf{pk}^d = g^{z_0}$ in $\mathsf{BS}_2.\mathsf{U}_3$.
4. $h'^{z_0'}Z'^{-e'} = R_h'$. This follows from $z_0' = z_0 + \alpha_0, d' = d + \gamma_0, h' = hg^{-\beta}$, and $Z' = Z\mathsf{pk}^{-\beta}$
$$\begin{aligned} h'^{z_0'}Z'^{-d'} &= h'^{z_0}Z'^{-d}(h'^{\alpha_0}Z'^{-\gamma_0}) \\ &= h^{z_0}Z^{-d}(g^{z_0}\mathsf{pk}^{-d})^{-\beta}(h'^{\alpha_0}Z'^{-\gamma_0}) \\ &= R_hR_g^{-\beta}(h'^{\alpha_0}Z'^{-\gamma_0}) = R_h' \ , \end{aligned}$$
where the second to last equality follows from the check $R_hZ^d = g^{z_0}$ and $R_g\mathsf{pk}^d = g^{z_0}$ in $\mathsf{BS}_2.\mathsf{U}_3$.

By the points above, we have $\mathsf{H}(m, g^{z_0'}\mathsf{pk}^{-d'}, g^{z_1'}W^{-e'}) = \mathsf{H}(m, R_g', A') = h'$ and
$$\mathsf{H}'(m, h', Z', g^{z_0'}\mathsf{pk}^{-d'}, h'^{z_0'}Z'^{-d'}, g^{z_1'}W^{-e'}) = \mathsf{H}'(m, h', Z', R_g', R_h', A') = c' = d' + e' \ ,$$
proving the scheme's correctness. □

4.2 Proof of Theorem 4.1 (Blindness of BS_2)

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

- The oracle $\textsc{Init}(\mathsf{pk}, m_0, m_1)$ additionally computes $\mathsf{sk} \leftarrow \log_g \mathsf{pk}$ by exhaustive search.
- For each signing session $i \in \{0,1\}$, when the oracle $\mathrm{U}_2(i, \mathsf{smsg}_2^{(i)})$ receives $\mathsf{smsg}_2^{(i)}$ from $\mathcal{A}$, it parses $(Z_i, R_{h,i}, \pi_i = (\delta_i, s_i)) \leftarrow \mathsf{smsg}_2^{(i)}$. Then, it computes $S_{g,i} = g^{s_i}\mathsf{pk}^{-\delta_i}$, $S_{h,i} = h_i^{s_i} Z_i^{-\delta_i}$ where h_i is the message returned by $\mathrm{U}_1(i, \cdot)$, and checks whether $\delta_i = \mathsf{H}''(h_i, \mathsf{pk}, Z_i, S_{g,i}, S_{h,i})$. If this check passes, the game now aborts if $Z_i \neq h_i^{\mathsf{sk}}$.

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

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

We will argue that this event occurs with negligible probability. Specifically, with how $S_{g,i}$ and $S_{h,i}$ are defined and that $Z_i \neq h_i^{\mathsf{sk}} = \mathsf{pk}^{\log_g h_i}$, 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 $\delta_i \in \mathbb{Z}_p$ that satisfies such equation. Note that $\mathcal{A}$ makes at most $Q_{\mathsf{H}''}$ queries to H'', and if the query $\mathsf{H}''(h_i, \mathsf{pk}, Z_i, S_{g,i}, S_{h,i})$ was not made beforehand, the game makes this query when checking validity of π. Since δ_i is sampled uniformly at random after fixing the query, the probability of the abort occurring in a session is bounded by $(Q_{\mathsf{H}''} + 1)/p$. By the union bound over the two signing sessions,

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

For the last step, we show that the transcript and returned signatures are distributed identically between both cases of $b = 0$ and $b = 1$, 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 $\mathrm{View}_{\mathcal{A}}$ denote the set of all possible views of $\mathcal{A}$ that can occur in the game $\mathbf{G}_1^{\mathcal{A}}$. A view $\Delta \in \mathrm{View}_{\mathcal{A}}$ is of the form $\Delta = (W, \mathsf{pk}, m_0, m_1, T_0, T_1, \sigma_0, \sigma_1)$ 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 oracles in signing session i (we omitted π_i as it is distributed independently of (m_0, m_1) given (h_i, Z_i)), and $\sigma_i = (Z_i', d_i', e_i', z_{0,i}', z_{1,i}')$ denotes the valid signature for the message m_i. We need to show that the distribution of the actual adversarial view, denoted as $v_{\mathcal{A}}$, is distributed identically between $b = 0$ and $b = 1$. Since the randomness of $\mathcal{A}$ is fixed, $v_{\mathcal{A}}$ 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 $v_{\mathcal{A}}(\eta)$ 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 view $\Delta \in \mathrm{View}_{\mathcal{A}}$, we have that $Z_{b_i}' = {h_{b_i}'}^{\mathsf{sk}}$ where $h_{b_i}' = \mathsf{H}(m_{b_i}, \mathsf{pk}^{-d_{b_i}'} g^{z_{0,b_i}'}, W^{-e_{b_i}'} g^{z_{1,b_i}'})$. This is because of the abort introduced in $\mathbf{G}_1^{\mathcal{A}}$ that induces $Z_i = h_i^{\mathsf{sk}}$ leading to $Z_{b_i}' = Z_i \mathsf{pk}^{-\beta_i} = (h_i g^{-\beta_i})^{\mathsf{sk}} = {h_{b_i}'}^{\mathsf{sk}}$.

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

$$\left.\begin{aligned} \beta_i \quad &= \log_g h_i - \log_g h_{b_i}' \\ \alpha_{0,i} \quad &= z_{0,b_i}' - z_{0,i}, \ \ \alpha_{1,i} = z_{1,b_i}' - z_{1,i} \\ \gamma_{0,i} \quad &= d_{b_i}' - d_i, \ \ \gamma_{1,i} = e_{b_i}' - e_i \ . \end{aligned}\right\} \tag{7}$$

For the "only if" direction, i.e., if $v_{\mathcal{A}}(\eta) = \Delta$, 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 $v_{\mathcal{A}}(\eta) = \mathit{\Delta}$. 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 $(h_0, h_1), (c_0, c_1)$, and (σ_0, σ_1) respectively.

Again, since we only consider a view $\mathit{\Delta}$ where neither the game nor the oracle aborts, we have the following guarantees for $i \in \{0, 1\}$:

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

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

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

where Equation (8) follows from the discussion above, Equation (9) follows from the checks 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 the randomness of $\mathcal{A}$ is fixed, $\mathcal{A}$'s first message will be $(R_{g,i}, A_i)$ from the view $\mathit{\Delta}$. Consider the blinded values of $R_{g,i}$ and A_i

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

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

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

which is consistent with $\mathit{\Delta}$. Thus, the next message from $\mathcal{A}$ will be $(Z_i, R_{h,i})$ from the view $\mathit{\Delta}$.

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

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

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

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

so the user's second message is consistent with $\mathit{\Delta}$. Thus, the next message from $\mathcal{A}$ will be $(d_i, e_i, z_{0,i}, z_{1,i})$ from the view $\mathit{\Delta}$. Lastly, the signatures from the oracle U_3, for $i \in \{0, 1\}$, are as follows

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

which are exactly the signatures in $\mathit{\Delta}$. □

4.3 Computational Blindness of BS_2 without NIZK

As mentioned earlier, we can remove the NIZK proof from our scheme BS_2 (resulting in a scheme which we will call BS_2' in this subsection to distinguish from the scheme with NIZK) 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 the game* 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*

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

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:

- The oracle $\textsc{Init}(\mathsf{pk}, m_0, m_1)$ additionally computes $\mathsf{sk} \leftarrow \log_g \mathsf{pk}$ by exhaustive search.
- For each signing session $i \in \{0, 1\}$, when the oracle $\mathrm{U}_3(i, \mathsf{smsg}_3^{(i)})$ is queried, it parses all the signer messages as $(R_{g,i}, A_i) \leftarrow \mathsf{smsg}_1^{(i)}, (Z_i, R_{h,i}) \leftarrow \mathsf{smsg}_2^{(i)}$ 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 returned by $\mathrm{U}_1(i, \cdot)$, 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 user messages in the signing protocol, i.e., (h_i, c_i) (since the event occurs before the signatures are returned).

To bound the probability of event Bad_i occurring, we will construct 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 describing $\mathcal{B}$, we make the following observation that h_i and c_i are uniformly random in $\mathbb{G}$ and $\mathbb{Z}_p$ respectively. First, denote $(\beta_i, \alpha_{0,i}, \alpha_{1,i}, \gamma_{0,i}, \gamma_{1,i})$ as the user randomness for signing session $i \in \{0, 1\}$. To see this, consider that, as computed in the user algorithm, $h_i = h_i' g^{\beta_i}$ 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 $Z_i', R_{g,i}', R_{h,i}', A_i'$ are the blinded values of $Z_i, R_{g,i}, R_{h,i}, A_i$ respectively, and $h_i' = \mathsf{H}(m_{b_i}, R_{g,i}', A_i')$. 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 of $\gamma_{1,i}$, 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 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 $\mathcal{A}$ twice.

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 user's first message h_i of signing session i in advance.
2. The oracles $\textsc{Init}$, $\mathrm{U}_1(1 - i, \cdot), \mathrm{U}_2(1 - i, \cdot)$, and $\mathrm{U}_3(1 - i, \cdot)$ are simulated 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 $\mathrm{U}_2(i, \cdot)$, $\mathcal{B}$ returns $c_i \leftarrow\$\ \mathbb{Z}_p$.

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 returns $c_i' \leftarrow_\$ \mathbb{Z}_p$. 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 can parse $(d_i, e_i, z_{0,i}, z_{1,i}) \leftarrow \mathsf{smsg}_3^{(i)}$ and $(d_i', e_i', z_{0,i}', z_{1,i}') \leftarrow \mathsf{smsg}_3'^{(i)}$. If $e_i \neq e_i'$, the reduction returns $(z_{1,i} - z_{1,i}')(e_i - e_i')^{-1}$. Otherwise, abort.

It is clear that the running time of $\mathcal{B}$ is about twice of $\mathcal{A}$'s. 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) $Z_i \neq h_i^{\mathsf{sk}}$.
(ii) $d_i + e_i = c_i \neq c_i' = d_i' + e_i'$.
(iii) $(R_{g,i}, R_{h,i}) = (g^{z_{0,i}} \mathsf{pk}^{-d_i}, h_i^{z_{0,i}} Z_i^{-d_i}) = (g^{z_{0,i}'} \mathsf{pk}^{-d_i'}, h_i^{z_{0,i}'} Z_i^{-d_i'})$
(iv) $A_i = g^{z_{1,i}} W^{-e_i} = g^{z_{1,i}'} W^{-e_i'}$

By considering (iii),

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

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, $\mathcal{B}$ succeeds in the DLOG game, i.e., $\Pr[\mathsf{frk}] \leqslant \mathsf{Adv}^{\mathrm{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda)$.

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

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

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

4.4 Proof of Theorem 4.2 (OMSUF-1 of BS_2)

To prove one-more strong unforgeability (OMSUF-1) 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 secret and public keys $(\mathsf{sk}, \mathsf{pk}) \leftarrow_\$ \mathsf{BS}_2.\mathsf{KG}(\mathsf{par})$. 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 $\mathcal{A}$ queries the signing oracle S_1 for ℓ times and the random oracles H, H' and H'' for $Q_{\mathsf{H}}, Q_{\mathsf{H}'}$ and $Q_{\mathsf{H}''}$ times respectively. At the end of the game, $\mathcal{A}$ outputs $\ell + 1$ message-signature pairs $(m_k^*, \sigma_k^*)_{k \in [\ell+1]}$. 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 for all $k \in [\ell+1]$, $\mathsf{BS}_2.\mathsf{Ver}(\mathsf{pk}, m_k^*, \sigma_k^*) = 1$. We w.l.o.g. assume that $\mathcal{A}$ does not make the same random oracle query twice. Also, we assume that $\mathcal{A}$ makes the random oracle queries that would be made in $\mathsf{BS}_2.\mathsf{Ver}$ when verifying the forgeries. This adds at most

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

```
Oracle $\mathrm{S}_1(\mathrm{sid})$:
If $\mathrm{sid} \in \mathcal{I}_1$ then return $\perp$
$\ell \leftarrow \ell + 1$ ; $\mathcal{I}_1 \leftarrow \mathcal{I}_1 \cup \{\mathrm{sid}\}$
$z_1, e, r_0 \leftarrow\$ \mathbb{Z}_p$
$R_g \leftarrow g^{r_0}$
$A \leftarrow g^{z_1} W^{-e}$      // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$z_0, d, r_1 \leftarrow\$ \mathbb{Z}_p$
$R_g \leftarrow g^{z_0}\mathsf{pk}^{-d}$
$A \leftarrow g^{r_1}$      // $\mathbf{G}_5^{\mathcal{A}}$
Return $(R_g, A)$

Oracle $\mathrm{S}_2(\mathrm{sid}, h)$ :
If $\mathrm{sid} \notin \mathcal{I}_1$ or $\mathrm{sid} \in \mathcal{I}_2$ then return $\perp$
$\mathcal{I}_2 \leftarrow \mathcal{I}_2 \cup \{\mathrm{sid}\}$
$Z \leftarrow h^{\mathsf{sk}}$
$R_h \leftarrow h^{r_0}$                                   // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$R_h \leftarrow h^{z_0} Z^{-d}$                             // $\mathbf{G}_5^{\mathcal{A}}$
$s \leftarrow\$ \mathbb{Z}_p$ ; $\delta \leftarrow \mathsf{H}''(h, \mathsf{pk}, Z, g^s, h^s)$
$\pi \leftarrow (\delta, s + \delta \cdot \mathsf{sk})$      // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_1^{\mathcal{A}}$
$\delta, s' \leftarrow\$ \mathbb{Z}_p$ ; $\pi \leftarrow (\delta, s')$ // $\mathbf{G}_2^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$
If $\mathsf{H}''(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^s Z^{-\delta}) \neq \perp$
    then **abort game**
$\mathsf{H}''(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^s Z^{-\delta}) \leftarrow \delta$
Return $(Z, R_h, \pi)$

Oracle $\mathrm{S}_3(\mathrm{sid}, c)$ :
If $\mathrm{sid} \notin \mathcal{I}_1$ or $\mathrm{sid} \notin \mathcal{I}_2$ or $\mathrm{sid} \in \mathcal{I}_3$ then return $\perp$
$\mathcal{I}_3 \leftarrow \mathcal{I}_3 \cup \{\mathrm{sid}\}$
$d \leftarrow c - e$
$z_0 \leftarrow r_0 + d \cdot \mathsf{sk}$                        // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$e \leftarrow c - d$
$z_1 \leftarrow r_1 + e \cdot w$                                // $\mathbf{G}_5^{\mathcal{A}}$
Return $(d, e, z_0, z_1)$
```

Fig. 7. The OMSUF-1 $= \mathbf{G}_0^{\mathcal{A}}$ security game for BS_2 and the subsequent games $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_5^{\mathcal{A}}$. We remark that H, H' and H'' are modeled as random oracles to which $\mathcal{A}$ has access. Each box type indicates the changes made in the game name contained in the box. Also, to make things clearer, for each box, the comments indicate which game the changes in the boxes correspond to. Moreover, the signer state is omitted and we assume that *each variable initialized in signing oracles of earlier round can be accessed by the signing oracles of the same* sid *in later rounds.*

$\ell+1$ queries to H and H', making the total query count $\widehat{Q}_{\mathsf{H}} = Q_{\mathsf{H}} + \ell + 1$ and $\widehat{Q}_{\mathsf{H}'} = Q_{\mathsf{H}'} + \ell + 1$, respectively. The success probability of $\mathcal{A}$ in game $\mathbf{G}_0^{\mathcal{A}}$ is exactly its advantage in game OMSUF-1, i.e.,

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

Game $\mathbf{G}_1^{\mathcal{A}}$: This game is identical to $\mathbf{G}_0^{\mathcal{A}}$ except that for the message-signature pairs $(m_k^*, \sigma_k^*)_{k\in[\ell+1]}$ output by the adversary $\mathcal{A}$, *for* $k \in [\ell+1]$, *after parsing the signature* $(Z_k^*, d_k^*, e_k^*, z_{0,k}^*, z_{1,k}^*) \leftarrow \sigma_k^*$ *and setting* $R_{g,k}^* \leftarrow g^{z_{0,k}^*}\mathsf{pk}^{-d_k^*}$, $A_k^* \leftarrow g^{z_{1,k}^*}W^{-e_k^*}$, *the game additionally 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

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

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 $(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^{s'}Z^{-\delta})$ as δ. The game aborts if H'' is already defined at $(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^{s'}Z^{-\delta})$.

The view of $\mathcal{A}$ is identical to its view in $\mathbf{G}_1^{\mathcal{A}}$ if the game does not abort. Moreover, the game only aborts if $(h, \mathsf{pk}, Z, g^{s'}\mathsf{pk}^{-\delta}, h^{s'}Z^{-\delta})$ has been queried or programmed beforehand, but $g^{s'}\mathsf{pk}^{-\delta}$ is 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 possible collision events, i.e., all pairs of queries to 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 $(Z_k^*, d_k^*, e_k^*, z_{0,k}^*, z_{1,k}^*) \leftarrow \sigma_k^*$ and setting $R_{g,k}^* \leftarrow g^{z_{0,k}^*}\mathsf{pk}^{-d_k^*}, A_k^* \leftarrow g^{z_{1,k}^*}W^{-e_k^*}$ 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 and 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: $e_{k_1}^* \neq e_{k_2}^*$. 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_1, by a direct reduction $\mathcal{B}_1$ 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_1 occurs. Thus, the probability of E_1 occurring is bounded by $\mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{dlog}}(\mathcal{B}_1, \lambda)$. We can also see that the running time of $\mathcal{B}_1$ is about that of $\mathcal{A}$.
2. Case E_2: $d_{k_1}^* \neq d_{k_2}^*$. With the same argument and $R_{g,k_1}^* = R_{g,k_2}^*$, this allows us to extract the discrete logarithm of pk. However, the reduction here would need to send $Z = h^{\mathsf{sk}}$ to the adversary without knowing sk. To achieve this, we give the following reduction $\mathcal{B}_2$ to CT-CDH assumption instead.
   - At the beginning, $\mathcal{B}_2$ receives the CT-CDH instance $(\mathbb{G}, p, g, X)$ and queries CHAL for $\ell+1$ challenges $Y_1, \ldots, Y_{\ell+1}$. It then computes $W \leftarrow g^w$ where $w \leftarrow\!\!\$\ \mathbb{Z}_p$ and sends $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W), \mathsf{pk} \leftarrow X$ to $\mathcal{A}$. The random oracles are simulated with lazy sampling as in $\mathbf{G}_3^{\mathcal{A}}$.
   - For each S_1 query, $\mathcal{B}_2$ samples $z_0, d, r_1 \leftarrow\!\!\$\ \mathbb{Z}_p$ and returns $(R_g \leftarrow g^{z_0}\mathsf{pk}^{-d}, A \leftarrow g^{r_1})$.
   - For each S_2 query, $\mathcal{B}_2$ forwards its query h to its own DH oracle and receives $Z = h^{\log_g X}$, instead of using the secret key $\mathsf{sk} = \log_g \mathsf{pk} = \log_g X$ to compute Z, and simulates the protocol on by setting $R_h \leftarrow h^{z_0}Z^{-d}$ (using z_0, d initialized in S_1 query of the same session ID). It then returns (Z, R_h).
   - Lastly, for each S_3 query, $\mathcal{B}_2$ returns $(d, e \leftarrow c - d, z_0, z_1 \leftarrow r_1 + e \cdot w)$ (using z_0, d initialized in S_1 query of the same session ID). Note here that the simulations of the oracle $\mathrm{S}_1, \mathrm{S}_2, \mathrm{S}_3$ do not require the reduction to know $\mathsf{sk} = \log_g \mathsf{pk}$.
   - At the end when E_2 occurs, $\mathcal{B}_2$ extracts the discrete logarithm 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 $(k, Y_k^x)_{k \in [\ell+1]}$.

   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_2 occurring in the game simulated by $\mathcal{B}_2$ is exactly the same as in $\mathbf{G}_3^{\mathcal{A}}$. With $x = \log_g X$, $\mathcal{B}_2$ succeeds in the CT-CDH game if E_2 occurs. Thus, the probability of the event E_2 is bounded by $\mathsf{Adv}_{\mathsf{GGen}}^{\mathsf{ct\text{-}cdh}}(\mathcal{B}_2, \lambda)$. We can also see that the running time of $\mathcal{B}_2$ is about that of $\mathcal{A}$.
3. Case E_3: $(d_{k_1}^*, e_{k_1}^*) = (d_{k_2}^*, e_{k_2}^*)$. 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.

Hence, applying the union bound on the three cases,

$$\mathsf{Pr}[\mathbf{G}_3^{\mathcal{A}} = 1] \geqslant \mathsf{Pr}[\mathbf{G}_2^{\mathcal{A}} = 1] - \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{dlog}}(\mathcal{B}_1, \lambda) - \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{ct\text{-}cdh}}(\mathcal{B}_2, \lambda) \ .$$

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 sets $W \leftarrow g^w$.

Since the game runs the oracles in the same way and W still has the same distribution as in $\mathbf{G}_3^{\mathcal{A}}$, we have

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

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

1. Sample $r_1, d, z_0 \leftarrow\!\!\$\ \mathbb{Z}_p$ and set $A \leftarrow g^{r_1}$, $R_g \leftarrow g^{z_0}\mathsf{pk}^{-d}$. Later, after receiving h, set $R_h \leftarrow h^{z_0} Z^{-d}$.
2. After receiving c, set $e \leftarrow c - d$ and $z_1 \leftarrow r_1 + e \cdot w$.

Since the joint distributions of $(A, R_g, R_h, d, e, z_0, z_1)$ 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,

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

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

1. The reduction $\mathcal{B}'$ takes as input a CT-CDH instance $(\mathbb{G}, p, g, X)$, samples $w \leftarrow\!\!\$\ \mathbb{Z}_p$, and sets $W \leftarrow g^w$. It then sends $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W)$, $\mathsf{pk} \leftarrow X$ to $\mathcal{A}$.
2. The simulations of H' and H'' are done as in $\mathbf{G}_5^{\mathcal{A}}$. However, for queries to H (labeling each with $j \in [\widehat{Q}_{\mathsf{H}}]$), 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 $\widehat{Q}_{\mathsf{H}} = Q_{\mathsf{H}} + \ell + 1$ queries to $\textsc{Chal}$.)
3. The signing oracles are also simulated as in $\mathbf{G}_5^{\mathcal{A}}$ except for the computation of $Z = h^{\mathsf{sk}}$ in S_2 which is done by querying its $\textsc{Dh}$ oracle instead, i.e., $Z \leftarrow \textsc{Dh}(h)$.
4. After receiving the message-signature pairs $(m_k^*, \sigma_k^*)_{k\in[\ell+1]}$ from $\mathcal{A}$, $\mathcal{B}'$ parses $(Z_k^*, e_k^*, d_k^*, z_{0,k}^*, z_{1,k}^*) \leftarrow \sigma_k^*$, sets $R_{g,k}^* \leftarrow g^{z_{0,k}^*}\mathsf{pk}^{-d_k^*}$, $A_k^* \leftarrow g^{z_{1,k}^*}W^{-e_k^*}$, 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, j_k are all distinct, meaning there are 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 identically to the game $\mathbf{G}_5^{\mathcal{A}}$. Then, if $\mathcal{A}$ succeeds in game $\mathbf{G}_5^{\mathcal{A}}$, then $\mathcal{A}$ returns $Z_k^* = \mathsf{H}(m_k^*, R_{g,k}^*, A_k^*)^{\mathsf{sk}} = Y_{j_k}^{\log_g X}$ for all $k \in [\ell+1]$ where $\mathsf{sk} = \log_g \mathsf{pk} = \log_g X$. Thus, $\mathcal{B}'$ returns $\ell+1$ correct CT-CDH solutions while only querying the oracle $\textsc{Dh}$ for at most ℓ times. Hence, if $\mathcal{A}$ succeeds in the game $\mathbf{G}_5^{\mathcal{A}}$, $\mathcal{B}'$ succeeds in the game CT-CDH. Thus,

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

By combining all the advantage changes,

$$\begin{aligned}\mathsf{Adv}_{\mathsf{BS}_2}^{\mathrm{omsuf\text{-}1}}(\mathcal{A}, \lambda) \leqslant{}& \frac{\ell(\ell + Q_{\mathsf{H}''})}{p} + (\ell+1)\left(\sqrt{\widehat{Q}_{\mathsf{H}'}\mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{dlog}}(\mathcal{B}, \lambda)} + \frac{\widehat{Q}_{\mathsf{H}'}}{p}\right) \\ &+ \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{dlog}}(\mathcal{B}_1, \lambda) + \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{ct\text{-}cdh}}(\mathcal{B}_2, \lambda) + \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{ct\text{-}cdh}}(\mathcal{B}', \lambda) \ .\end{aligned}$$

□

Lemma 4.5. *There exists an adversary $\mathcal{B}$ for the game* DLOG*, running in time $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$, such that*

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

Proof. Let event Bad be the event where $\mathbf{G}_0^{\mathcal{A}}$ outputs 1 but $\mathbf{G}_1^{\mathcal{A}}$ outputs 0. This corresponds to the following event: $\mathcal{A}$ outputs $\ell + 1$ message-signature pairs $(m_k^*, \sigma_k^*)_{k \in [\ell+1]}$ which we parse for each $k \in [\ell + 1]$, $(Z_k^*, d_k^*, e_k^*, z_{0,k}^*, z_{1,k}^*) \leftarrow \sigma_k^*$ and set $R_{g,k}^* \leftarrow g^{z_{0,k}^*}\mathsf{pk}^{-d_k^*}, A_k^* \leftarrow g^{z_{1,k}^*}W^{-e_k^*}$; 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 $k \in [\ell + 1]$, $\mathsf{BS}_2.\mathsf{Ver}(\mathsf{pk}, m_k^*, \sigma_k^*) = 1$, 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 wrapper $\mathcal{A}_k$ over the adversary $\mathcal{A}$ where $\mathcal{A}_k$ receives the following inputs: instance $(\mathbb{G}, p, g, W)$, outputs $(c_1, \ldots, c_{\widehat{Q}_{\mathsf{H}'}})$ of H', and a random tape ρ. $\mathcal{A}_k$ is defined as follows:

1. Extract $(\mathsf{sk} \in \mathbb{Z}_p, (s_i \in \mathbb{Z}_p, r_{0,i} \in \mathbb{Z}_p, e_i \in \mathbb{Z}_p, z_{1,i} \in \mathbb{Z}_p)_{i \in [\ell]}, (h_i \in \mathbb{G})_{i \in [\widehat{Q}_{\mathsf{H}}]}, (\delta_i \in \mathbb{Z}_p)_{i \in [Q_{\mathsf{H}''} + \ell]}, \rho')$ from the random tape ρ.
2. Set $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W), \mathsf{pk} \leftarrow g^{\mathsf{sk}}$.
3. Run $(m_k^*, \sigma_k^*)_{k \in [\ell+1]} \leftarrow \mathcal{A}^{\mathrm{S}_1, \mathrm{S}_2, \mathrm{S}_3, \mathsf{H}, \mathsf{H}', \mathsf{H}''}(\mathsf{par}, \mathsf{pk}; \rho')$ where each oracle is answered as follows:
   - For the signing query with session ID j ($j \in [\ell]$) to $\mathrm{S}_1, \mathrm{S}_2$, and S_3, use $(\mathsf{sk}, r_{0,i}, e_i, z_{1,i}, s_i)$ to answer the query as in $\mathsf{BS}_2.\mathsf{S}_1, \mathsf{BS}_2.\mathsf{S}_2$ and $\mathsf{BS}_2.\mathsf{S}_3$ respectively.
   - For the i-th query ($i \in [\widehat{Q}_{\mathsf{H}}]$) to H, return h_i.
   - For the i-th query ($i \in [\widehat{Q}_{\mathsf{H}'}]$) to H', return c_i.
   - For the i-th query ($i \in [Q_{\mathsf{H}''} + \ell]$) to H'', return δ_i. (Note: In these queries, we accounted for the queries that the wrapper made to generate π in each query to S_1.)
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 $\mathcal{A}$ that corresponds to the verification of (m_k^*, σ_k^*). More specifically, after parsing $(Z_k^*, d_k^*, e_k^*, z_{0,k}^*, z_{1,k}^*) \leftarrow \sigma_k^*$, I is the index that corresponds to the query (m, h, Z, R_g, R_h, A) to H' where $m = m_k^*, R_g = g^{z_{0,k}^*}\mathsf{pk}^{-d_k^*}, A = g^{z_{1,k}^*}W^{-e_k^*}, h = \mathsf{H}(m, R_g, A), Z = Z_k^*, R_h = h^{z_{0,k}^*}Z^{-d_k^*}$. Note that I is well-defined as we assume that all random oracle queries made during 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 $c_1, \ldots, c_{\widehat{Q}_{\mathsf{H}'}} \leftarrow\$ \mathbb{Z}_p$ along with the random tape ρ for $\mathcal{A}_k$.
2. Run $(I, (m, \sigma)) \leftarrow\$ \mathcal{A}_k((\mathbb{G}, p, g, W), (c_1, \ldots, c_{\widehat{Q}_{\mathsf{H}'}}); \rho)$.
3. If $I = \perp$, abort. If not, sample $c'_I, \ldots, c'_{\widehat{Q}_{\mathsf{H}'}} \leftarrow\$ \mathbb{Z}_p$ and
   run $(I', (m', \sigma')) \leftarrow\$ \mathcal{A}_k((\mathbb{G}, p, g, W), (c_1, \ldots, c_{I-1}, c'_I, \ldots, c'_{\widehat{Q}_{\mathsf{H}'}}); \rho)$.
4. If $I = I'$ and $c'_I \neq c_I$, parse $(Z, d, e, z_0, z_1) \leftarrow \sigma, (Z', d', e', z'_0, z'_1) \leftarrow \sigma'$, 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 $\mathcal{A}$, $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$. Next, 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 message-signature pairs (m, σ) and (m', σ'): (a) are valid signatures corresponding to the I-th query from $\mathcal{A}$ to H' of the form (m, h, Z, R_g, R_h, A) and (b) satisfy $Z \neq \mathsf{H}(m, R_g, A)^{\mathsf{sk}}$ and $Z' \neq \mathsf{H}(m', R'_g, A')^{\mathsf{sk}}$ where $R_g = g^{z_0}\mathsf{pk}^{-d}, A = g^{z_1}W^{-e}, R'_g = g^{z'_0}\mathsf{pk}^{-d'}$, and $A' = g^{z'_1}W^{-e'}$. By (a), we know the following

(i) $m = m', h = \mathsf{H}(m, R_g, A) = \mathsf{H}(m', R'_g, A'), Z = Z'$.

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

We will argue that $d = d'$. First, the equations in (iii) give $Z^{d-d'} = h^{z_0 - z'_0} = g^{(z_0 - z'_0)\log_g h} = \mathsf{pk}^{(d-d')\log_g h} = h^{\mathsf{sk}(d-d')}$. Since $Z \neq \mathsf{H}(m, R_g, A)^{\mathsf{sk}} = h^{\mathsf{sk}}$, only $d = d'$ satisfies the equation. Since $d + e = c_I \neq c'_I = d' + e'$, we have $e \neq e'$. Thus, by (iv), $\mathcal{B}$ returns $(z_1 - z'_1)(e - e')^{-1} = \log_g W$. Hence,

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

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

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

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

5 Achieving OMUF-2 security from CDH

In this section, we present a four-move blind signature scheme BS_3, described in Section 5.3, achieving the OMUF-2 security based on the CDH assumption. The key ingredients used in this construction are the homomorphic equivocal commitment HECom, given in Section 5.1, and a non-interactive proof system Π (for guaranteeing blindness), given in Section 5.2.

5.1 Homomorphic Equivocal Commitment Scheme

In this section, we present the commitment scheme HECom which is a tuple of algorithms $(\mathsf{Gen}, \mathsf{TGen}, \mathsf{Com}, \mathsf{TCom}, \mathsf{TOpen})$, described in Figure 8. The algorithm Gen generates a uniform commitment key $\mathsf{ck} \leftarrow\!\!\$\ \mathbb{G}^{2\times 2}$, which can be done transparently. For the rest of the scheme, one can view our commitment as a variant of the commitment scheme of [BCJ08]. Both commitments commit to a group element, and are additively homomorphic and computationally binding based on the DLOG assumption. For equivocation, we can generate the commitment key with a base $X \in \mathbb{G}$ embedded, allowing us to open a commitment of S' to $S = S'X^c$ for any $c \in \mathbb{Z}_p$. On the other hand, their equivocation allows opening a commitment to $g^a X^c$ for a uniformly random $a \in \mathbb{Z}_p$ and any $c \in \mathbb{Z}_p$. The following theorem summarizes the properties of our commitment scheme.

Theorem 5.1. *Assume that* GGen *outputs the description of a group* $\mathbb{G}$ *of prime order* $p = p(\lambda)$. *The commitment* $\mathsf{HECom} = \mathsf{HECom}[\mathsf{GGen}]$ *satisfies the following properties:*

- ***Additive Homomorphism.*** *For* $\mathsf{hcom}_0, \mathsf{hcom}_1 \in \mathbb{G}^2$, *denote* $\mathsf{hcom}_0 \cdot \mathsf{hcom}_1$ *as element-wise application of group operation. For all* $(\mathbb{G}, p, g) \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda)$, $\mathsf{ck} \in \mathbb{G}^{2\times 2}, S_0, S_1 \in \mathbb{G}$, *and* $\mathsf{crnd}_0, \mathsf{crnd}_1 \in \mathbb{Z}_p^2$,

$$\mathsf{Com}(\mathsf{ck}, S_0; \mathsf{crnd}_0) \cdot \mathsf{Com}(\mathsf{ck}, S_1; \mathsf{crnd}_1) = \mathsf{Com}(\mathsf{ck}, S_0 S_1; \mathsf{crnd}_0 + \mathsf{crnd}_1) \ .$$

- ***Special Equivocation.*** *For all* $\mathsf{par} \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda), X \neq 1_{\mathbb{G}}$ *and* $(\mathsf{ck}, \mathsf{td}) \leftarrow\!\!\$\ \mathsf{TGen}(\mathsf{par}, X)$ *such that* $\mathbf{D}$ *contained in* $\mathsf{td} = (\mathbf{D}, X)$ *is invertible, and for any group element* $S = X^c S'$, *the following distributions* D_0 *and* D_1 *are identical:*

$$D_0 := \{(\mathsf{hcom}, S, \mathsf{crnd}) : (\mathsf{hcom}, \mathsf{st}) \leftarrow\!\!\$\ \mathsf{TCom}(\mathsf{td}, S') \ ; \ (S, \mathsf{crnd}) \leftarrow\!\!\$\ \mathsf{TOpen}(\mathsf{st}, c)\} \ ,$$
$$D_1 := \left\{(\mathsf{hcom}, S, \mathsf{crnd}) : \mathsf{crnd} \leftarrow\!\!\$\ \mathbb{Z}_p^2 \ ; \ \mathsf{hcom} \leftarrow \mathsf{Com}(\mathsf{ck}, S; \mathsf{crnd})\right\} \ .$$

- ***Uniform Keys.*** *For all* $\mathsf{par} \leftarrow\!\!\$\ \mathsf{GGen}(1^\lambda)$ *and* $X \neq 1_{\mathbb{G}}$, ck *generated by* $(\mathsf{ck}, \mathsf{td}) \leftarrow\!\!\$\ \mathsf{TGen}(\mathsf{par}, X)$ *is uniformly distributed in* $\mathbb{G}^{2\times 2}$ *(i.e., distributed identically to* $\mathsf{ck} \leftarrow\!\!\$\ \mathsf{Gen}(\mathsf{par})$*).*

```
Algorithm Gen(par = (𝔾, p, g)):
    Return ck ←$ $\mathbb{G}^{2\times 2}$
Algorithm TGen(par = (𝔾, p, g), X):
    $d_{11}, d_{12}, d_{21}, d_{22} \leftarrow_\$ \mathbb{Z}_p$
    $\mathbf{D} \leftarrow \begin{pmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{pmatrix}$
    $\mathsf{ck} \leftarrow \begin{pmatrix} g^{d_{11}} & g^{d_{12}} \\ X^{d_{21}} & X^{d_{22}} \end{pmatrix}$
    Return $(\mathsf{ck}, \mathsf{td} \leftarrow (\mathbf{D}, X))$

Algorithm Com(ck = $\mathbf{A} \in \mathbb{G}^{2\times 2}$, $S \in \mathbb{G}$; crnd $\in \mathbb{Z}_p^2$):
    Return hcom ← $(A_{11}^{\mathsf{crnd}_1} A_{12}^{\mathsf{crnd}_2}, S \cdot A_{21}^{\mathsf{crnd}_1} A_{22}^{\mathsf{crnd}_2})$
Algorithm TCom(td = $(\mathbf{D}, X)$, $S' \in \mathbb{G}$):
    $\tau, \rho \leftarrow_\$ \mathbb{Z}_p$ ; hcom ← $(g^\tau, S' \cdot X^\rho)$ ; st ← $(\mathbf{D}, X, S', \tau, \rho)$
    Return (hcom, st)
Algorithm TOpen(st = $(\mathbf{D}, X, S', \tau, \rho)$, $c \in \mathbb{Z}_p$):
    If $\mathbf{D}$ is not invertible, then return ⊥
    Return $(S \leftarrow S' \cdot X^c, \mathsf{crnd} \leftarrow \mathbf{D}^{-1}(\tau, \rho - c)^T)$
```

```
Game $\mathrm{Binding}_{\mathsf{HECom}}^{\mathcal{A}}(\lambda)$:
    par ←$ GGen($1^\lambda$)
    ck ←$ Gen(par) ; $(S, S', \mathsf{crnd}, \mathsf{crnd}') \leftarrow_\$ \mathcal{A}(\mathsf{par}, \mathsf{ck})$
    If Com(ck, S; crnd) ≠ Com(ck, S'; crnd') or S = S'
        then return 0
    Return 1
```

Fig. 8. Description of the special commitment scheme $\mathsf{HECom} = \mathsf{HECom}[\mathsf{GGen}]$ and its binding game. For the algorithms $\mathsf{Com}, \mathsf{TCom}$, and TOpen, $\mathsf{par} = (\mathbb{G}, p, g)$ is taken as an implicit input.

- ***Computationally Binding.*** *For any adversary $\mathcal{A}$ for the game* Binding *(described in Figure 8) with running time $t_{\mathcal{A}} = t_{\mathcal{A}}(\lambda)$, there exists an adversary $\mathcal{B}$ for the game* DLOG *with running time $t_{\mathcal{B}} \approx t_{\mathcal{A}}$ such that the advantage of $\mathcal{A}$ in the game is bounded by*

$$\mathsf{Adv}_{\mathsf{HECom}}^{\mathrm{binding}}(\mathcal{A}, \lambda) = \Pr[\mathrm{Binding}_{\mathsf{HECom}}^{\mathcal{A}}(\lambda) = 1] \leqslant \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{dlog}}(\mathcal{B}, \lambda) + \frac{1}{p} .$$

Proof. We consider each property as follows:

- **Additive Homomorphism.** Consider $\mathsf{ck} = \mathbf{A} \in \mathbb{G}^{2\times 2}$, $S_0, S_1 \in \mathbb{G}$ and $\mathsf{crnd}_0, \mathsf{crnd}_1 \in \mathbb{Z}_p^2$.

$$\begin{aligned} &\mathsf{Com}(\mathsf{ck}, S_0; \mathsf{crnd}_0) \cdot \mathsf{Com}(\mathsf{ck}, S_1; \mathsf{crnd}_1) \\ &= (A_{11}^{\mathsf{crnd}_{0,1}} A_{12}^{\mathsf{crnd}_{0,2}}, S_0 A_{21}^{\mathsf{crnd}_{0,1}} A_{22}^{\mathsf{crnd}_{0,2}}) \cdot (A_{11}^{\mathsf{crnd}_{1,1}} A_{12}^{\mathsf{crnd}_{1,2}}, S_1 A_{21}^{\mathsf{crnd}_{1,1}} A_{22}^{\mathsf{crnd}_{1,2}}) \\ &= (A_{11}^{\mathsf{crnd}_{0,1}+\mathsf{crnd}_{1,1}} A_{12}^{\mathsf{crnd}_{0,2}+\mathsf{crnd}_{1,2}}, S_0 S_1 \cdot A_{21}^{\mathsf{crnd}_{0,1}+\mathsf{crnd}_{1,1}} A_{22}^{\mathsf{crnd}_{0,2}+\mathsf{crnd}_{1,2}}) \\ &= \mathsf{Com}(\mathsf{ck}, S_0 S_1; \mathsf{crnd}_0 + \mathsf{crnd}_1) . \end{aligned}$$

- **Special Equivocation.** To show this, suppose $X \neq 1_{\mathbb{G}}$ and the trapdoor $\mathbf{D}$ is invertible. Let $S = S'X^c$ for $S' \in \mathbb{G}$ and $c \in \mathbb{Z}_p$. Then, we will argue that the crnd generated using the trapdoor is uniformly random and the commitment hcom is exactly $\mathsf{Com}(\mathsf{ck}, S; \mathsf{crnd})$. By the algorithms TCom and TOpen, we have that $\mathsf{crnd} = \mathbf{D}^{-1}(\tau, \rho - c)^T$ for uniformly random $\tau, \rho \leftarrow_\$ \mathbb{Z}_p$. Because $\mathbf{D}$ is invertible, crnd is uniformly random in $\mathbb{Z}_p^2$. Moreover,

$$\mathsf{hcom} = (g^\tau, S' \cdot X^\rho) = (g^\tau, S'X^c \cdot X^{\rho - c}) = (g^\tau, S \cdot X^{\rho - c})$$

Then, by how crnd is defined, we have that $\mathsf{Com}(\mathsf{ck}, S; \mathsf{crnd}) = (g^\tau, S \cdot X^{\rho - c})$.

- **Uniform Keys.** Consider when $X \neq 1_{\mathbb{G}}$, meaning X is a generator of $\mathbb{G}$. Then, for uniformly random $d_{11}, d_{12}, d_{21}, d_{22} \leftarrow_\$ \mathbb{Z}_p$, we have that $\mathsf{ck} = \begin{pmatrix} g^{d_{11}} & g^{d_{12}} \\ X^{d_{21}} & X^{d_{22}} \end{pmatrix}$ is uniformly distributed in $\mathbb{G}^{2\times 2}$.

- **Computational Binding.**

  Consider a reduction $\mathcal{B}$ which on input $(\mathbb{G}, p, g, X)$ generates the commitment key $\mathsf{ck} \leftarrow \begin{pmatrix} g^r & X \\ A & B \end{pmatrix}$ where $r \leftarrow_\$ \mathbb{Z}_p, A, B \leftarrow_\$ \mathbb{G}$, it aborts if $r = 0$. Otherwise, it runs $\mathcal{A}$ on the input $((\mathbb{G}, p, g), \mathsf{ck})$. After $\mathcal{A}$ returns

```
$\underline{\Pi.\mathsf{Prove}^{\mathsf{H}_\Pi}((g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}), (\mathsf{sk}_i)_{i\in[K]}):}$
$\vec{r} \leftarrow_\$ \mathbb{Z}_p^K$
For $i \in [K]$: $R_i \leftarrow g^{\vec{r}_i}$
$\bar{R} \leftarrow \prod_{i=1}^K h_i^{\vec{r}_i}$
$c \leftarrow \mathsf{H}_\Pi(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}, (R_i)_{i\in[K]}, \bar{R})$
For $i \in [K]$ : $\vec{s}_i \leftarrow \vec{r}_i + c \cdot \mathsf{sk}_i$
Return $\pi \leftarrow (c, \vec{s})$
```

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

Fig. 9. Description of the proof system Π with access to the hash function $\mathsf{H}_\Pi : \{0,1\}^* \to \mathbb{Z}_p$

$(S, S', \mathsf{crnd}, \mathsf{crnd}')$, it checks if $\mathcal{A}$ succeeds in the game, i.e., $\mathsf{Com}(\mathsf{ck}, S; \mathsf{crnd}) = \mathsf{Com}(\mathsf{ck}, S'; \mathsf{crnd}')$ and $S \neq S'$. Finally, $\mathcal{B}$ returns $r(\mathsf{crnd}'_1 - \mathsf{crnd}_1)(\mathsf{crnd}_2 - \mathsf{crnd}'_2)^{-1}$.
First, the distribution of ck that $\mathcal{B}$ generates is exactly uniform in $\mathbb{G}^{2\times 2}$ except when the reduction aborts (which occurs with probability at most $1/p$). Consider the output of $\mathcal{A}$. Since $\mathsf{Com}(\mathsf{ck}, S; \mathsf{crnd}) = \mathsf{Com}(\mathsf{ck}, S'; \mathsf{crnd}')$, we have that

$$g^{r(\mathsf{crnd}'_1 - \mathsf{crnd}_1)} X^{\mathsf{crnd}'_2 - \mathsf{crnd}_2} = S' S^{-1} A^{\mathsf{crnd}'_1 - \mathsf{crnd}_1} B^{\mathsf{crnd}'_2 - \mathsf{crnd}_2} = 1_{\mathbb{G}} \; .$$

Then, with $r \neq 0$ and g being a generator, if $\mathsf{crnd}'_2 = \mathsf{crnd}_2$, we have $\mathsf{crnd}'_1 = \mathsf{crnd}_1$ and $S = S'$. Hence, $\mathsf{crnd}'_2 \neq \mathsf{crnd}_2$ and $r(\mathsf{crnd}'_1 - \mathsf{crnd}_1)(\mathsf{crnd}_2 - \mathsf{crnd}'_2)^{-1}$ is well-defined as $\log_g X$. Thus, $\mathsf{Adv}^{\mathrm{binding}}_{\mathsf{HECom}}(\mathcal{A}, \lambda) \leqslant \mathsf{Adv}^{\mathrm{dlog}}_{\mathsf{GGen}}(\mathcal{B}, \lambda) + \frac{1}{p}$. □

5.2 Proof System Π

In this section, we present a non-interactive proof system Π, described in Figure 9, with access to a hash function $\mathsf{H}_\Pi : \{0,1\}^* \to \mathbb{Z}_p$. The proof system Π attests membership of the language $\mathcal{L}_{\mathbb{G},K}$, defined for a group $\mathbb{G}$ of prime order p with a generator g, and a positive integer K as follows:

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

We require that Π satisfies completeness, soundness, and zero-knowledge as established by the following lemma with the hash function $\mathsf{H}_\Pi : \{0,1\}^* \to \mathbb{Z}_p$ modeled as a random oracle.

Lemma 5.2. *Let $\mathbb{G}$ be a group of prime order $p = p(\lambda)$ with generator g and $K = K(\lambda)$ be a positive integer. The proof system Π (defined in Figure 9) satisfies the following properties with respect to $\mathcal{L}_{\mathbb{G},K}$ where the corresponding security games are defined in Figure 10:*

- ***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 $i \in [K]$,*

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

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

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

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

$$\left| \Pr[\mathrm{ZK}^{\mathcal{A}}_{\Pi}(\lambda) = 1] - \frac{1}{2} \right| \leqslant \frac{Q_{\mathrm{CHAL}}(Q_{\mathrm{CHAL}} + Q_{\mathsf{H}_\Pi})}{p} \; .$$

```
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

Game $\mathrm{ZK}_{\Pi}^{\mathcal{A}}(\lambda)$
$(\mathbb{G}, p, g) \leftarrow\$ \mathsf{GGen}(1^{\lambda})$ ; $b \leftarrow\$ \{0,1\}$
$b' \leftarrow\$ \mathcal{A}^{\textsc{Chal}, \mathsf{H}_{\Pi}}(\mathbb{G}, p, g)$
If $b = b'$ 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 $\mathsf{H}_{\Pi}(\mathrm{str})$
```

```
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 $\perp$
If $b = 0$ then
    $\pi \leftarrow\$ \Pi.\mathsf{Prove}^{\mathsf{H}_{\Pi}}((g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}), (\mathsf{sk}_i)_{i\in[K]})$
If $b = 1$ then
    $\pi \leftarrow\$ \mathsf{Sim}(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S})$
Return $\pi$

Algorithm $\mathsf{Sim}(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S})$ :
$c \leftarrow\$ \mathbb{Z}_p$, $\vec{s} \leftarrow\$ \mathbb{Z}_p^K$
For $i \in [K]$: $R_i \leftarrow g^{\vec{s}_i} \mathsf{pk}_i^{-c}$
$\bar{R} \leftarrow \bar{S}^{-c} \prod_{i=1}^{K} h_i^{\vec{s}_i}$
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 $\pi \leftarrow (c, \vec{s})$
```

Fig. 10. The security games $\mathrm{Sound}_{\Pi}^{\mathcal{A}}$ and $\mathrm{ZK}_{\Pi,b}^{\mathcal{A}}$ for the proof system Π.

Proof. We consider each of the listed properties.

- **Completeness.** Completeness follows by inspection.
- **Soundness.** Let $\mathcal{A}$ be an adversary playing the soundness game and outputting $(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}) \notin \mathcal{L}_{\mathbb{G},K}$ and a proof $\pi = (c, \vec{s})$ where $\vec{s} \in \mathbb{Z}_p^K$. 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 $(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S})$,

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

  Here, assume that $\mathcal{A}$ already made this query, adding the query count by 1 (i.e., the game can make the query when checking the validity of π). 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 $\vec{s} \in \mathbb{Z}_p^K$ such that

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

  We consider such c, which gives us the above equations. Then, by raising $\mathsf{pk}_i^c R_i = g^{\vec{s}_i}$ to $\log_g h_i$, we have $h_i^{c \log_g \mathsf{pk}_i} R_i^{\log_g h_i} = h_i^{\vec{s}_i}$ for all $i \in [K]$. Thus, we have that $\prod_{i=1}^{K} h_i^{c\log_g \mathsf{pk}_i} R_i^{\log_g h_i} = \prod_{i=1}^{K} h_i^{\vec{s}_i} = \bar{S}^c\bar{R}$, 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 equation. 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 $1/p$. Since at most $Q_{\mathsf{H}_{\Pi}} + 1$ queries are made to H_{Π},

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

- **Zero-knowledge.** Consider the simulator Sim as described in Figure 10 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 bound 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 a tuple which includes $R_1 = g^{\vec{s}_1}\mathsf{pk}_1^{-c}$ for $\vec{s}_1 \leftarrow\$ \mathbb{Z}_p$ 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 $(Q_{\mathsf{H}_\Pi} + Q_{\text{CHAL}})/p$ (counting the random oracle queries and the programming attempts). Thus, bounding this over Q_{CHAL} queries to CHAL,

$$\left|\Pr[\mathrm{ZK}_\Pi^{\mathcal{A}}(\lambda) = 1] - \frac{1}{2}\right| \leqslant \frac{Q_{\text{CHAL}}(Q_{\text{CHAL}} + Q_{\mathsf{H}_\Pi})}{p} .$$

□

5.3 Four-Move Blind Signatures from CDH

The scheme BS_3 is described across Figures 11 and 12. (A protocol diagram is also presented in Figure 15.) 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 Rai-Choo producing signatures of the form $((\mathsf{pk}_i, \varphi_i)_{i\in[K]}, \bar{S})$ with *inefficient verification* checking

$$\mathsf{pk} = \prod_{i=1}^{K} \mathsf{pk}_i \text{ and } \bar{S} = \prod_{i=1}^{K} \mathsf{H}(\mathsf{H}_\mu(m, \varphi_i))^{\log_g \mathsf{pk}_i} .$$

To make the scheme efficiently verifiable, we apply a witness-indistinguishable OR proof showing that the signature is valid, i.e., $((\mathsf{pk}_i)_{i\in[K]}, \bar{S})$ satisfies the verification equation with regard to $(\mathsf{H}(\mathsf{H}_\mu(m, \varphi_i)))_{i\in[K]}$, or that we know the discrete logarithm of a public parameter W. Finally, using the homomorphic equivocal commitment HECom from Section 5.1, the signer commits to the group element $\bar{S}$ from the Rai-Choo protocol and the nonce $\bar{R}$ in the OR proof as $\mathsf{hcom}_{\bar{S}}$ and $\mathsf{hcom}_{\bar{R}}$ respectively. These commitments are sent in the second move instead of $\bar{S}$ and $\bar{R}$ and opened later in the last move. The final signature consists of a Rai-Choo signature $((\mathsf{pk}_i, \varphi_i)_{i\in[K]}, \bar{S})$, the OR proof response $(d, e, \vec{z}_0, z_1)$, and the commitment randomness used to compute $\mathsf{hcom}_{\bar{S}}$ and $\mathsf{hcom}_{\bar{R}}$. It is easy to show that this scheme satisfies correctness, and we prove this in Section 5.4.

As mentioned in the prior section, the commitment key of HECom can be generated transparently; thus, so are the public parameters of BS_3. We also remark that the complexity of the scheme depends on two parameters N and K of which N^{-K} needs to be negligible for the OMUF proof. To achieve the signature size and communication in Table 1, we set $N = 2$ and $K = \lambda$.

BLINDNESS. The blindness of BS_3 can be guaranteed by the following steps:

- We apply the blinding procedure from Rai-Choo (as described in $\mathsf{U}_1, \mathsf{U}_2$ and ReRa) to make the distribution of $((\mathsf{pk}_i')_{i\in[K]}, \bar{S}')$ in the signature independent of the transcript.
- We then blind the OR proof (as described in U_2 and U_3) to make the the distribution of $(d', e', \vec{z}_0', z_1')$ in the signature independent of the transcript.
- To blind $\bar{S}$ and $\bar{R}$ according to the above points, we use the homomorphic property of HECom and blind $\mathsf{hcom}_{\bar{S}}$ and $\mathsf{hcom}_{\bar{R}}$ instead. We also rerandomize the commitments as the commitment randomness is included in the final signature.
- Finally, we need to ensure that the signer cannot send $((\mathsf{pk}_i)_{i\in[K]}, \bar{S})$ such that $\bar{S} \neq \prod_{i=1}^{K} h_{i,\vec{J}_i}^{\log_g \mathsf{pk}_i}$ where $h_{i,\vec{J}_i}$ for $i \in [K]$ are group elements contained in the user's first message. 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 signer's second response attesting that $((\mathsf{pk}_i)_{i\in[K]}, \bar{S})$ is honestly generated. For this, we use the non-interactive proof system $\Pi = (\Pi.\mathsf{Prove}^{\mathsf{H}_\Pi}, \Pi.\mathsf{Ver}^{\mathsf{H}_\Pi})$, described in Figure 9, with access to the hash function $\mathsf{H}_\Pi : \{0,1\}^* \to \mathbb{Z}_p$ modeled as a random oracle in the security proofs. As established in Section 5.2, Π satisfies completeness, soundness, and zero-knowledge in the random oracle model.

Similar to BS_1 and BS_2, one could also not include Π in the protocol, and show computational blindness based on the DL assumption. Still, this proof would 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 Π included, and prove the following theorem in Section 5.5.

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

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

Algorithm $\mathsf{BS}_3.\mathsf{S}_1(\mathsf{sk}, \mathsf{umsg}_1)$ :
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]})$
                                                        $\leftarrow \mathsf{umsg}_1$
If $\mathsf{Check}(\mathsf{umsg}_1) = 0$
    then return $\perp$
For $i \in [K-1]$:
    $\mathsf{sk}_i \leftarrow\!\$\ \mathbb{Z}_p$ ; $\mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i}$
$\mathsf{sk}_K \leftarrow \mathsf{sk} - \sum_{i=1}^{K-1} \mathsf{sk}_i$
$\mathsf{pk}_K \leftarrow g^{\mathsf{sk}_K}$
$z_1, e \leftarrow\!\$\ \mathbb{Z}_p, \vec{r}_0 \leftarrow\!\$\ \mathbb{Z}_p^K$
$\bar{S} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_i}$
$A \leftarrow g^{z_1} W^{-e}$
$\vec{R} \leftarrow (g^{\vec{r}_{0,1}}, \ldots, g^{\vec{r}_{0,K}})$
$\bar{R} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{r}_{0,i}}$
$\mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}} \leftarrow\!\$\ \mathbb{Z}_p^2$
$\mathsf{hcom}_{\bar{S}} \leftarrow \mathsf{Com}(\bar{S}; \mathsf{crnd}_{\bar{S}})$
$\mathsf{hcom}_{\bar{R}} \leftarrow \mathsf{Com}(\bar{R}; \mathsf{crnd}_{\bar{R}})$
Return $((\mathsf{pk}_i)_{i \in [K-1]}, \mathsf{hcom}_{\bar{S}}, \vec{R}, \mathsf{hcom}_{\bar{R}}, A)$

Algorithm $\mathsf{BS}_3.\mathsf{S}_2(c)$ :
$d \leftarrow c - e$
For $i \in [K]$:
    $\vec{z}_{0,i} \leftarrow \vec{r}_{0,i} + d \cdot \mathsf{sk}_i$
$\mathsf{input} \leftarrow (g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S})$
$\pi \leftarrow \mathsf{Prove}^{\mathsf{H}_\Pi}(\mathsf{input}, (\mathsf{sk}_i)_{i \in [K]})$
Return $(d, e, \vec{z}_0, z_1, \bar{S}, \bar{R}, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \pi)$
```

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

Algorithm $\mathsf{BS}_3.\mathsf{U}_2(\mathsf{smsg}_1)$ :
$((\mathsf{pk}_i)_{i \in [K-1]}, \mathsf{hcom}_{\bar{S}}, \vec{R}, \mathsf{hcom}_{\bar{R}}, A) \leftarrow \mathsf{smsg}_1$
$\mathsf{pk}_K \leftarrow \mathsf{pk} \prod_{i \in [K]} \mathsf{pk}_i^{-1}$
$\alpha_1, \gamma_0, \gamma_1 \leftarrow\!\$\ \mathbb{Z}_p$ ; $\vec{\alpha}_0 \leftarrow\!\$\ \mathbb{Z}_p^K$ ; $\delta_S, \delta_R \leftarrow\!\$\ \mathbb{Z}_p^2$
$\widehat{\mathsf{hcom}}_{\bar{S}} \leftarrow \mathsf{hcom}_{\bar{S}} \cdot \mathsf{Com}(\mathsf{ck}, \prod_{i=1}^K \mathsf{pk}_i^{-\beta_{i,\vec{J}_i}}; \delta_S)$
$((\mathsf{pk}'_i)_{i \in [K]}, \mathsf{hcom}'_{\bar{S}}, \vec{\tau}) \leftarrow\!\$\ \mathsf{ReRa}((\mathsf{pk}_i, h'_{i,\vec{J}_i})_{i \in [K]}, \widehat{\mathsf{hcom}}_{\bar{S}})$
For $i \in [K]$ : $\vec{R}'_i \leftarrow \vec{R}_i \mathsf{pk}_i'^{-\gamma_0} g^{\vec{\alpha}_{0,i}}$
$\widehat{\mathsf{hcom}}_{\bar{R}} \leftarrow \mathsf{Com}(\prod_{i=1}^K \vec{R}_i^{-\beta_{i,\vec{J}_i}} h'^{\vec{\alpha}_{0,i}}_{i,\vec{J}_i}; \delta_R)$
$\mathsf{hcom}'_{\bar{R}} \leftarrow \mathsf{hcom}_{\bar{R}} \cdot \mathsf{hcom}'^{-\gamma_0}_{\bar{S}} \cdot \widehat{\mathsf{hcom}}_{\bar{R}}$
$A' \leftarrow A W^{-\gamma_1} g^{\alpha_1}$
$c' \leftarrow \mathsf{H}'(m, (h'_{i,\vec{J}_i}, \mathsf{pk}'_i)_{i \in [K]}, \mathsf{hcom}'_{\bar{S}}, \vec{R}', \mathsf{hcom}'_{\bar{R}}, A')$
$c \leftarrow c' - \gamma_0 - \gamma_1$
Return $c$

Algorithm $\mathsf{BS}_3.\mathsf{U}_3(\mathsf{smsg}_2)$ :
$(d, e, \vec{z}_0, z_1, \bar{S}, \bar{R}, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \pi) \leftarrow \mathsf{smsg}_2$
If $c \neq d + e$ or $AW^e \neq g^{z_1}$ 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
    $\mathsf{hcom}_{\bar{S}} \neq \mathsf{Com}(\bar{S}; \mathsf{crnd}_{\bar{S}})$ or
    $\mathsf{hcom}_{\bar{R}} \neq \mathsf{Com}(\bar{R}; \mathsf{crnd}_{\bar{R}})$ or
    $\mathsf{Ver}^{\mathsf{H}_\Pi}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}), \pi) = 0$ then
        return $\perp$
$\bar{S}' \leftarrow \bar{S} \prod_{i=1}^K \mathsf{pk}_i^{-\beta_{i,\vec{J}_i}} h'^{\vec{\tau}_i}_{i,\vec{J}_i}$
$d' \leftarrow d + \gamma_0$ ;            $e' \leftarrow e + \gamma_1$
$\vec{z}'_0 \leftarrow \vec{z}_0 + \vec{\alpha}_0 + d \cdot \vec{\tau}$ ; $z'_1 \leftarrow z_1 + \alpha_1$
$\mathsf{crnd}'_{\bar{S}} \leftarrow \mathsf{crnd}_{\bar{S}} + \delta_S$
$\mathsf{crnd}'_{\bar{R}} \leftarrow \mathsf{crnd}_{\bar{R}} - \gamma_0 \cdot \mathsf{crnd}'_{\bar{S}} + \delta_R$
$\sigma \leftarrow ((\mathsf{pk}'_i, \varphi_{i,\vec{J}_i})_{i \in [K]}, \bar{S}', d', e', \vec{z}'_0, z'_1, \mathsf{crnd}'_{\bar{S}}, \mathsf{crnd}'_{\bar{R}})$
Return $\sigma$
```

Fig. 11. The setup and key generation algorithms along with the signing protocol of the blind signature scheme $\mathsf{BS}_3 = \mathsf{BS}_3[\mathsf{GGen}]$. The verification algorithm $\mathsf{BS}_3.\mathsf{Ver}$, the algorithms Check and ReRa are given separately in Figure 12, while the proof system $\Pi = (\Pi.\mathsf{Prove}^{\mathsf{H}_\Pi}, \Pi.\mathsf{Ver}^{\mathsf{H}_\Pi})$ is given in Figure 9. For the ease of understanding, we omitted the states of both the user and signer algorithms and assume that any values initialized in the prior rounds are accessible to the later rounds. The public parameters par, as stated before, are implicit input to every algorithms except $\mathsf{BS}_3.\mathsf{KG}$. The notation $\mathsf{Com}(\cdot\,;\,\cdot)$ denotes $\mathsf{HECom.Com}(\mathsf{ck}, \cdot\,;\,\cdot)$ for the commitment scheme HECom from Section 5.1. Similarly, we write $(\mathsf{Prove}^{\mathsf{H}_\Pi}, \mathsf{Ver}^{\mathsf{H}_\Pi})$ instead of $(\Pi.\mathsf{Prove}^{\mathsf{H}_\Pi}, \Pi.\mathsf{Ver}^{\mathsf{H}_\Pi})$. We also give a protocol diagram of BS_3 in Figure 15.

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

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

```
Algorithm $\mathsf{ReRa}((\mathsf{pk}_i, h_i)_{i\in[K]}, \mathsf{hcom}_{\bar{S}})$:
Let $\vec{\tau} \in \mathbb{Z}_p^K$
$\vec{\tau}_1, \ldots, \vec{\tau}_{K-1} \leftarrow\!\!\$\ \mathbb{Z}_p$; $\vec{\tau}_K \leftarrow -\sum_{i=1}^{K-1} \vec{\tau}_i$
For $i \in [K]$: $\mathsf{pk}'_i \leftarrow \mathsf{pk}_i g^{\vec{\tau}_i}$
$\mathsf{hcom}'_{\bar{S}} \leftarrow \mathsf{hcom}_{\bar{S}} \cdot \mathsf{Com}(\prod_{i=1}^{K} h_i^{\vec{\tau}_i}; 0)$
Return $((\mathsf{pk}_i)_{i\in[K]}, \mathsf{hcom}'_{\bar{S}}, \vec{\tau})$
```

Fig. 12. The verification algorithm $\mathsf{BS}_3.\mathsf{Ver}$ and the algorithms Check and ReRa used in the signing protocol of BS_3. The public parameters par are implicit input to $\mathsf{BS}_3.\mathsf{Ver}$.

Theorem 5.3 (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* $\mathcal{A}$ *for the game* BLIND *making at most* $Q_{\mathsf{H}_\star} = Q_{\mathsf{H}_\star}(\lambda)$ *queries to* $\mathsf{H}_\star \in \{\mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_\Pi\}$, *modeled as random oracles, we have*

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

<u>ONE-MORE UNFORGEABILITY.</u> The following theorem, proved in Section 5.6, establishes the OMUF-2 security of BS_3 in the random oracle model under the CDH assumption.

Theorem 5.4 (OMUF-2 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 the game* OMUF-2 *with running time* $t_\mathcal{A} = t_\mathcal{A}(\lambda)$, *making at most* $Q_{\mathrm{S}_1} = Q_{\mathrm{S}_1}(\lambda)$ *and* $\ell = \ell(\lambda)$ *queries to* S_1 *and* S_2, *respectively, and* $Q_{\mathsf{H}_\star} = Q_{\mathsf{H}_\star}(\lambda)$ *queries to* $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}', \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}, \mathsf{H}_\Pi\}$, *modeled as random oracles, there exist adversaries* $\mathcal{B}$ *for the game* Binding *of* HECom, $\mathcal{B}'$ *for the game* DLOG, *and* $\mathcal{B}''$ *for the game* CDH, *such that*

$$\mathsf{Adv}^{\mathrm{omuf\text{-}2}}_{\mathsf{BS}_3}(\mathcal{A}, \lambda) \leqslant (\ell+1)\left(\sqrt{\widehat{Q}_{\mathsf{H}'}\left(\mathsf{Adv}^{\mathrm{binding}}_{\mathsf{HECom}}(\mathcal{B}, \lambda) + \mathsf{Adv}^{\mathrm{dlog}}_{\mathsf{GGen}}(\mathcal{B}', \lambda)\right)} + \frac{\widehat{Q}_{\mathsf{H}'}}{p}\right) + \frac{\ell(\ell + Q_{\mathsf{H}_\Pi} + 12)}{p}$$
$$+ \frac{Q_{\mathrm{S}_1}}{N^K} + \frac{Q^2_{\mathsf{H}_{\mathsf{com}}} + \widehat{Q}^2_{\mathsf{H}_\mu} + Q_{\mathsf{H}_{\mathsf{com}}} Q_{\mathsf{H}_{cc}} + \widehat{Q}_{\mathsf{H}} \widehat{Q}_{\mathsf{H}_\mu}}{2^\lambda} + 4\ell \cdot \mathsf{Adv}^{\mathrm{cdh}}_{\mathsf{GGen}}(\mathcal{B}'', \lambda) .$$

where $\widehat{Q}_{\mathsf{H}} = Q_{\mathsf{H}} + (\ell+1)K, \widehat{Q}_{\mathsf{H}'} = Q_{\mathsf{H}'} + \ell + 1$, *and* $\widehat{Q}_{\mathsf{H}_\mu} = Q_{\mathsf{H}_\mu} + (\ell+1)K$. *Furthermore,* $\mathcal{B}$, $\mathcal{B}'$ *and* $\mathcal{B}''$ *run in time* $t_\mathcal{B}, t_{\mathcal{B}'} \approx 2t_\mathcal{A}$, *and* $t_{\mathcal{B}''} \approx t_\mathcal{A}$ *respectively.*

The proof in Section 5.6 consists of the game sequence $\mathbf{G}_0 - \mathbf{G}_{13}$ which is split into the following parts, with $\mathbf{G}_0$ corresponding to the OMUF-2 game:

- Game $\mathbf{G}_1$ forbids the adversary from returning a message-signature pair that contains $((\mathsf{pk}_i, \varphi_i)_{i\in[K]}, \bar{S})$ with $\bar{S} \neq \prod_{i=1}^{K} \mathsf{H}(\mathsf{H}_\mu(m, \varphi_i))^{\log_g \mathsf{pk}_i}$. If such event occurs, we rewind $\mathcal{A}$ to either break the binding of HECom or extract the discrete logarithm of W in the public parameters.
- Games $\mathbf{G}_2 - \mathbf{G}_4$ change the simulation of the *interactive proof* in the protocol to now use $w = \log_g W$ instead of $\{\mathsf{sk}_i\}_{i\in[K]}$.

- Games $\mathbf{G}_5 - \mathbf{G}_{10}$ follow the security proof of Rai-Choo [HLW23] and program the random oracles such that, in any signing session where the signer's second response is requested, $\log_g h_{i*,\vec{J}_{i*}}$ for some $i* \in [K]$ is known, and that there is still a message-signature pair output by the adversary from which one can extract a CDH solution. Essentially, the proof does the following:
  1. First, the proof argues that for each of the user's first message, there exists some $i* \in [K]$ where $h_{i*,\vec{J}_{i*}}$ is computed honestly, i.e. $h_{i*,\vec{J}_{i*}} = \mathsf{H}(\mathsf{H}_\mu(m,\varphi))$ for some (m,φ) (extractable from the random oracle transcript). This then binds each signing session with some message.
  2. Then, it programs the random oracles such that, still with non-negligible probability, the discrete logarithm of $\mathsf{H}(\mathsf{H}_\mu(m,\varphi))$ is known for the sessions where the adversary requested the signer's *second* response. Since there is at most ℓ such sessions, it is still possible to program the oracles to extract CDH solution from one of the $\ell+1$ forgeries. Note that for the sessions where only the user's first message is received, it does not matter whether such discrete logarithm is known.
- Games $\mathbf{G}_{11} - \mathbf{G}_{13}$ generate the commitment key ck with the base $X = \mathsf{pk}$ embedded and simulate the rest of each signing session (i.e., $(\mathsf{pk}_i)_{i\in[K]}, \mathsf{hcom}_{\bar{S}}$, and $\bar{S}$) without the secret key. More specifically, one can sample $\mathsf{sk}_i \leftarrow_\$ \mathbb{Z}_p$ for $i \neq i*$, set $\mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i}$ and compute pk_{i*} such that $\mathsf{pk} = \prod_{i=1}^K \mathsf{pk}_i$. Then, observe that $\bar{S}$ as computed in the protocol can be written as

$$\bar{S} = \prod_{i=1}^{K} h_{i,\vec{J}_i}^{\mathsf{sk}_i} = h_{i*,\vec{J}_{i*}}^{\mathsf{sk}-\sum_{i\neq i*}\mathsf{sk}_i} \prod_{i\neq i*} h_{i,\vec{J}_i}^{\mathsf{sk}_i} = \mathsf{pk}^{\log_g h_{i*,\vec{J}_{i*}}} \prod_{i\neq i*} h_{i,\vec{J}_i}^{\mathsf{sk}_i} h_{i*,\vec{J}_{i*}}^{-\mathsf{sk}_i} .$$

  Since we know $\log_g h_{i*,\vec{J}_{i*}}$ only for sessions where the signer's second response is requested, we cannot compute $\bar{S}$ without sk for every first signer's response. However, using the special equivocation property, we can send $\mathsf{hcom}_{\bar{S}}$ as a commitment to $S' = \prod_{i\neq i*} h_{i,\vec{J}_i}^{\mathsf{sk}_i} h_{i*,\vec{J}_{i*}}^{-\mathsf{sk}_i}$ and open it later to $\bar{S} = \mathsf{pk}^{\log_g h_{i*,\vec{J}_{i*}}} S'$.
- Finally, we construct a reduction to CDH using an adversary playing the game $\mathbf{G}_{13}$.

5.4 Correctness of BS_3

Theorem 5.5. 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 $\mathsf{com} = (\mathsf{com}_{i,j})_{i\in[K],j\in[N]}, h = (h_{i,j})_{i\in[K],j\in[N]}$.
- 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 $j \in [N]\backslash\{\vec{J}_i\}$, 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 $((\mathsf{pk}_i)_{i\in[K-1]}, \mathsf{hcom}_{\bar{S}}, \vec{R}, \mathsf{hcom}_{\bar{R}}, A, B)$ each defined as follows:
  - $\mathsf{pk}_i = g^{\mathsf{sk}_i}$ for $i \in [K]$ and $\mathsf{sk}_K = \mathsf{sk} - \sum_{i=1}^{K-1} \mathsf{sk}_i$.
  - $\mathsf{hcom}_{\bar{S}} = \mathsf{Com}(\bar{S}; \mathsf{crnd}_{\bar{S}})$ with $\bar{S} = \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_i}$.
  - $\vec{R} = (g^{\vec{r}_{0,1}}, \ldots, g^{\vec{r}_{0,K}})$, $\mathsf{hcom}_{\bar{R}} = \mathsf{Com}(\bar{R}; \mathsf{crnd}_{\bar{R}})$ with $\bar{R} = \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{r}_{0,i}}$ for $\vec{r}_0 \leftarrow_\$ \mathbb{Z}_p^K$.
  - $A = g^{z_1} W^{-e}$ for $z_1, e \leftarrow_\$ \mathbb{Z}_p$.
- The second user algorithm $\mathsf{BS}_3.\mathsf{U}_2$: Then, the blinded values of $\mathsf{pk}_i, \mathsf{hcom}_{\bar{S}}, R_i, \mathsf{hcom}_{\bar{R}}, A$ are as follows:
  - By the definition of ReRa, $\mathsf{pk}'_i = \mathsf{pk}_i g^{\vec{\tau}_i}$ for $i \in [K]$, $\prod_{i=1}^K \mathsf{pk}'_i = \mathsf{pk}$ and $\mathsf{hcom}'_{\bar{S}} = \mathsf{Com}(\bar{S} \prod_{i=1}^K \mathsf{pk}_i^{-\beta_{i,\vec{J}_i}} {h'}_{i,\vec{J}_i}^{\vec{\tau}_i}; \mathsf{crnd}_{\bar{S}} + \delta_S)$.

  - $\vec{R}'_i = \vec{R}_i \mathsf{pk}'^{-\gamma_0}_i g^{\vec{\alpha}_{0,i}}$ for $i \in [K]$ and
    $\mathsf{hcom}'_{\bar{R}} = \mathsf{Com}(\bar{R}\bar{S}'^{-\gamma_0} \prod_{i=1}^K \vec{R}_i^{-\beta_{i,\vec{J}_i}} h'^{\vec{\alpha}_{0,i}}_{i,\vec{J}_i}; \mathsf{crnd}_{\bar{R}} - \gamma_0 \cdot \mathsf{crnd}'_{\bar{S}} + \delta_R)$
  - $A' = A g^{\alpha_1} W^{-\gamma_1}$
- The third user algorithm $\mathsf{BS}_3.\mathsf{U}_3$: On the signer message $(\bar{S}, \bar{R}, d, e, \vec{z}_0, z_1, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \pi)$, the following checks pass:
  - $c = d + e$ because d is defined as $c - e$ 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, $\bar{R}\bar{S}^d = \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{r}_{0,i} + d\cdot\mathsf{sk}_i} = \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{z}_{0,i}}$.
  - The checks on A and $\mathsf{hcom}_{\bar{S}}, \mathsf{hcom}_{\bar{R}}$ trivially pass because of how the values are defined.
  - The algorithm checks that the $\Pi.\mathsf{Ver}^{\mathsf{H}_\Pi}$ on π returns 1 which is always true by the completeness of Π.
- Signature verification: The final signature is $\sigma = ((\mathsf{pk}'_i, \varphi_{i,\vec{J}_i})_{i\in[K]}, \bar{S}', d', e', \vec{z}'_0, z'_1, \mathsf{crnd}'_{\bar{S}}, \mathsf{crnd}'_{\bar{R}})$, 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 $i \in [K]$,
$$g^{\vec{z}'_{0,i}} \mathsf{pk}'^{-d'}_i = g^{\vec{z}_{0,i} + \vec{\alpha}_{0,i} + d\cdot\vec{\tau}_i} (\mathsf{pk}_i g^{\vec{\tau}_i})^{-d} \mathsf{pk}'^{-\gamma_0}_i = \vec{R}_i g^{\vec{\alpha}_{0,i}} \mathsf{pk}'^{-\gamma_0}_i = \vec{R}'_i\ .$$
  - Also,
$$\begin{aligned}
\bar{S}'^{-d'} \prod_{i=1}^K h'^{\vec{z}'_{0,i}}_{i,\vec{J}_i} &= (\bar{S} \prod_{i=1}^K \mathsf{pk}_i^{-\beta_{i,\vec{J}_i}} h'^{\vec{\tau}_i}_{i,\vec{J}_i})^{-d} \bar{S}'^{-\gamma_0} \prod_{i=1}^K h'^{\vec{z}_{0,i} + \vec{\alpha}_{0,i} + d\cdot\vec{\tau}_i}_{i,\vec{J}_i} \\
&= \bar{S}^{-d} \prod_{i=1}^K \mathsf{pk}_i^{d\beta_{i,\vec{J}_i}} h'^{\vec{z}_{0,i}}_{i,\vec{J}_i} \bar{S}'^{-\gamma_0} \prod_{i=1}^K h'^{\vec{\alpha}_{0,i}}_{i,\vec{J}_i} \\
&= \bar{S}^{-d} \prod_{i=1}^K \mathsf{pk}_i^{d\beta_{i,\vec{J}_i}} (h_{i,\vec{J}_i} g^{-\beta_{i,\vec{J}_i}})^{\vec{z}_{0,i}} \bar{S}'^{-\gamma_0} \prod_{i=1}^K h'^{\vec{\alpha}_{0,i}}_{i,\vec{J}_i} \\
&= \bar{S}^{-d} \prod_{i=1}^K h^{\vec{z}_{0,i}}_{i,\vec{J}_i} (\mathsf{pk}_i^{-d} g^{\vec{z}_{0,i}})^{-\beta_{i,\vec{J}_i}} \bar{S}'^{-\gamma_0} \prod_{i=1}^K h'^{\vec{\alpha}_{0,i}}_{i,\vec{J}_i} \\
&= \bar{R}\bar{S}'^{-\gamma_0} \prod_{i=1}^K \vec{R}_i^{-\beta_{i,\vec{J}_i}} h'^{\vec{\alpha}_{0,i}}_{i,\vec{J}_i}
\end{aligned}$$
  - $A' = A g^{\alpha_1} W^{-\gamma_1} = g^{z_1 + \alpha_1} W^{-e-\gamma_1} = g^{z'_1} W^{-e'}$.
  - $\mathsf{crnd}'_{\bar{S}} = \mathsf{crnd}_{\bar{S}} + \delta_S$, $\mathsf{crnd}'_{\bar{R}} = \mathsf{crnd}_{\bar{R}} - \gamma_0 \cdot \mathsf{crnd}'_{\bar{S}} + \delta_R$, which gives
    $\mathsf{hcom}'_{\bar{S}} = \mathsf{Com}(\bar{S}'; \mathsf{crnd}'_{\bar{S}})$ and $\mathsf{hcom}'_{\bar{R}} = \mathsf{Com}(\bar{R}'; \mathsf{crnd}'_{\bar{R}})$ with
    $\bar{R}' = \bar{R}\bar{S}'^{-\gamma_0} \prod_{i=1}^K \vec{R}_i^{-\beta_{i,\vec{J}_i}} h'^{\vec{\alpha}_{0,i}}_{i,\vec{J}_i}$.

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

□

5.5 Proof of Theorem 5.3 (Blindness of BS_3)

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 $\mathsf{H}_\star \in \{\mathsf{H}_\mu, \mathsf{H}_\beta, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_\Pi\}$. 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, in contrast to BS_1 and BS_2.)

Game $\mathbf{G}_1^{\mathcal{A}}$: In this game, we introduce an abort in the oracle $\mathrm{U}_3(k,\cdot)$ (for both $k = 0,1$) such that on input $(d, e, \vec{z}_0, z_1, \bar{S}, \bar{R}, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \pi) \leftarrow \mathsf{smsg}_2$: the oracle aborts if the proof π verifies, but $\bar{S} \neq \prod_{i=1}^{K} h_{i,\vec{J}_i}^{\log_g \mathsf{pk}_i}$ where $h_{i,\vec{J}_i}, \mathsf{pk}_i$ for $i \in [K]$ are the corresponding values from the user's and signer's first messages in that particular signing session $k \in \{0,1\}$ (omitting the superscripts).

Notice that the view of $\mathcal{A}$ only changes when the abort occurs, i.e., the event where $\mathcal{A}$ queries U_3 for $k \in \{0,1\}$ with a valid proof π for a statement $(g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i\in[K]}, \bar{S})$ with $\bar{S} \neq \prod_{i=1}^{K}(h_{i,\vec{J}_i})^{\log_g \mathsf{pk}_i}$. This corresponds to breaking the soundness property of Π. By Lemma 5.2, any adversary with Q_{H_Π}-query access to H_Π breaks the soundness of Π only with probability $(Q_{\mathsf{H}_\Pi} + 1)/p$. Thus, bounding over both signing sessions $k \in \{0,1\}$, we have

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

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 in $\{0,1\}^\lambda$ and hidden from the view of $\mathcal{A}$ throughout the game,

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

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

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

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

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

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

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

Game $\mathbf{G}_6^{\mathcal{A}}$: This game changes how $\beta_{i,\vec{J}_i^{(k)}}^{(k)}$ and $\mathsf{com}_{i,\vec{J}_i^{(k)}}^{(k)}$ are computed for $k \in \{0,1\}, i \in [K]$. Previously, it was defined as $\mathsf{H}_\beta(\varepsilon_{i,\vec{J}_i^{(k)}}^{(k)})$ and $\mathsf{H}_{\mathsf{com}}(\mathsf{r}_{i,\vec{J}_i^{(k)}}^{(k)})$; however, now it is only sampled uniformly at random 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 $\mathsf{H}_\beta(\varepsilon_{i,\vec{J}_i^{(k)}}^{(k)})$ nor $\mathsf{H}_{\mathsf{com}}(\cdot, \varepsilon_{i,\vec{J}_i^{(k)}}^{(k)})$ has been queried by $\mathcal{A}$. Since $\beta_{i,\vec{J}_i^{(k)}}^{(k)}$ and $\mathsf{com}_{i,\vec{J}_i^{(k)}}^{(k)}$ are distributed identically as before,

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

Lastly, we claim (in the following lemma) 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

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

concluding the proof.

Lemma 5.6. *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 $\mathrm{View}_{\mathcal{A}}$ denote the set of all possible views of $\mathcal{A}$ that can occur in the game $\mathbf{G}_6^{\mathcal{A}}$. A view $\Delta \in \mathrm{View}_{\mathcal{A}}$ is of the form

$$\Delta = (W, \mathsf{pk}, m_0, m_1, T_0, T_1, \sigma_0, \sigma_1) ,$$

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:

$$T_k = ((h^{(k)}_{i,\vec{J}_i^{(k)}}, \mathsf{pk}_i^{(k)})_{i\in[K]}, \mathsf{hcom}_{\bar{S}}^{(k)}, \mathsf{hcom}_{\bar{R}}^{(k)}, \bar{S}^{(k)}, \vec{R}^{(k)}, \bar{R}^{(k)}, A^{(k)}, c^{(k)}, d^{(k)}, e^{(k)}, \vec{z}_0^{(k)}, z_1^{(k)}, \mathsf{crnd}_{\bar{S}}^{(k)}, \mathsf{crnd}_{\bar{R}}^{(k)}) ,$$
$$\sigma_k = ((\mathsf{pk}'^{(k)}_i, \varphi'^{(k)}_i)_{i\in[K]}, \bar{S}'^{(k)}, d'^{(k)}, e'^{(k)}, \vec{z}'^{(k)}_0, z'^{(k)}_1, \mathsf{crnd}'^{(k)}_{\bar{S}}, \mathsf{crnd}'^{(k)}_{\bar{R}}) .$$

Note that we omitted the $(\vec{J}^{(k)}, ((\mathsf{r}^{(k)}_{i,j})_{j\neq\vec{J}_i^{(k)}}, \mathsf{com}^{(k)}_{i,\vec{J}_i^{(k)}})_{i\in[K]})$ portion of $\mathsf{umsg}_1^{(k)}$ because they are now independent of the messages (m_0, m_1) by the changes introduced to the games $\mathbf{G}_2^{\mathcal{A}} - \mathbf{G}_6^{\mathcal{A}}$. Also, we rename some variables from the signing protocol as follows,

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

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

$$\eta = ((\beta_i^{(k)}, \varphi_i^{(k)})_{i\in[K]}, \vec{\tau}^{(k)}, \vec{\alpha}_0^{(k)}, \alpha_1^{(k)}, \gamma_0^{(k)}, \gamma_1^{(k)}, \delta_{\bar{S}}^{(k)}, \delta_{\bar{R}}^{(k)})_{k\in\{0,1\}} ,$$

and we write $v_{\mathcal{A}}(\eta)$ to make this explicit.

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

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

for $\mathsf{sk}_i^{(k)} = \log_g \mathsf{pk}_i^{(k)}$.

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

$$\begin{aligned}
&\left.\begin{aligned}
\varphi_i^{(k)} &= \varphi_i'^{(b_k)} \\
\beta_i^{(k)} &= \log_g h_i^{(k)} - \log_g h_i'^{(k)} \\
\vec{\tau}_i^{(k)} &= \log_g \mathsf{pk}_i'^{(b_k)} - \log_g \mathsf{pk}_i^{(k)}
\end{aligned}\right\} \text{ for } i \in [K], \\
&\vec{\alpha}_0^{(k)} = \vec{z'}_0^{(b_k)} - \vec{z}_0^{(k)} - d^{(k)} \cdot \vec{\tau}^{(k)},\ \alpha_1^{(k)} = z'^{(b_k)}_1 - z_1^{(k)}, \\
&\gamma_0^{(k)} = d'^{(b_k)} - d^{(k)},\ \gamma_1^{(k)} = e'^{(b_k)} - e^{(k)} \\
&\delta_{\bar{S}}^{(k)} = \mathsf{crnd}'^{(b_k)}_{\bar{S}} - \mathsf{crnd}_{\bar{S}}^{(k)},\ \delta_{\bar{R}}^{(k)} = \mathsf{crnd}'^{(b_k)}_{\bar{R}} - \mathsf{crnd}_{\bar{R}}^{(k)} + \gamma_0^{(k)} \cdot \mathsf{crnd}'^{(b_k)}_{\bar{S}},
\end{aligned} \tag{13}$$

For the "only if" direction, i.e., if $v_{\mathcal{A}}(\eta) = \Delta$, 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 $v_{\mathcal{A}}(\eta) = \Delta$. 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 $((h_i^{(0)})_{i\in[K]}, (h_i^{(1)})_{i\in[K]})$, $(c^{(0)}, c^{(1)})$, and (σ_0, σ_1) respectively.

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

$$\left.\begin{aligned}
&\vec{R}_i^{(k)} = (\mathsf{pk}_i^{(k)})^{-d^{(k)}} g^{\vec{z}_{0,i}^{(k)}} \text{ for } i \in [K], \\
&\bar{R}^{(k)} = (\bar{S}^{(k)})^{-d^{(k)}} \textstyle\prod_{i=1}^{K} h_i^{(k)\,\vec{z}_{0,i}^{(k)}}, \\
&A^{(k)} = W^{-e^{(k)}} g^{z_1^{(k)}}, \quad c^{(k)} = d^{(k)} + e^{(k)}, \\
&\mathsf{hcom}_{\bar{S}}^{(k)} = \mathsf{Com}(\mathsf{ck}, \bar{S}^{(k)}; \mathsf{crnd}_{\bar{S}}^{(k)}), \mathsf{hcom}_{\bar{R}}^{(k)} = \mathsf{Com}(\mathsf{ck}, \bar{R}^{(k)}; \mathsf{crnd}_{\bar{R}}^{(k)}),
\end{aligned}\right\} \tag{14}$$

Then, by defining the intermediate values $\mathsf{hcom}'^{(b_k)}_{\bar{S}}$ and $\mathsf{hcom}'^{(b_k)}_{\bar{R}}$ used in the verification of σ_{b_k} as

$$\begin{aligned}
\mathsf{hcom}'^{(b_k)}_{\bar{S}} &= \mathsf{Com}(\mathsf{ck}, \bar{S}'^{(b_k)}; \mathsf{crnd}'^{(b_k)}_{\bar{S}}), \\
\mathsf{hcom}'^{(b_k)}_{\bar{R}} &= \mathsf{Com}(\mathsf{ck}, (\bar{S}'^{(k)})^{-d'^{(k)}} \prod_{i=1}^{K} h_i'^{(k)\,\vec{z'}^{(k)}_{0,i}}; \mathsf{crnd}'^{(b_k)}_{\bar{R}}),
\end{aligned}$$

we have

$$\begin{aligned}
d'^{(b_k)} + e'^{(b_k)} = \mathsf{H}'(m_{b_k}, (h_i'^{(k)}, \mathsf{pk}_i'^{(b_k)})_{i\in[K]}, \mathsf{hcom}'^{(b_k)}_{\bar{S}}, (g^{\vec{z'}^{(b_k)}_{0,i}} \mathsf{pk}_i'^{(b_k)\,-d'^{(b_k)}})_{i\in[K]}, \quad &(15) \\
\mathsf{hcom}'^{(b_k)}_{\bar{R}}, W^{-e'^{(b_k)}} g^{z_1'^{(b_k)}}), \quad &(16)
\end{aligned}$$

where Equation (14) follows from the checks in $\mathsf{BS}_3.\mathrm{U}_3$, and Equation (16) follows from the validity of the signatures.

First, we argue that the user's first 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 $((\mathsf{pk}_i^{(k)})_{i\in[K]}, \mathsf{hcom}_{\bar{S}}^{(k)}, \vec{R}^{(k)}, \mathsf{hcom}_{\bar{R}}^{(k)}, A^{(k)}, B^{(k)})$ from the view Δ (we included $\mathsf{pk}_K^{(k)}$ as well just for simplicity, as it can be recomputed from pk and $(\mathsf{pk}_i^{(k)})_{i\in[K-1]}$.).

Next, we argue that the user's second message from $\mathrm{U}_2(k, \cdot)$ will be $c^{(k)}$. To do this, we consider the blinded values of $(\mathsf{pk}_i^{(k)}, R_i^{(k)})_{i\in[K]}, A^{(k)}, \mathsf{hcom}_{\bar{S}}^{(k)}, \mathsf{hcom}_{\bar{R}}^{(k)}$ which will be the inputs to H' when computing $c^{(k)}$. Note that below we also consider the blinded values of $\bar{S}^{(k)}, \bar{R}^{(k)}$ which are the values committed by

$\mathsf{hcom}_{\bar{S}}^{(k)}, \mathsf{hcom}_{\bar{R}}^{(k)}$ respectively.

$$
\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}'^{(b_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 $\gamma_0^{(k)}$ and $\vec{\alpha}_0^{(k)}$ in Equation (13); and the last equality follows from the value of $\vec{R}_i^{(k)}$ in Equation (14). Then, we argue that the blinded commitments ${\mathsf{hcom}'_{\bar{S}}}^{(k)}$ and ${\mathsf{hcom}'_{\bar{R}}}^{(k)}$ are exactly ${\mathsf{hcom}'_{\bar{S}}}^{(b_k)}$ and ${\mathsf{hcom}'_{\bar{R}}}^{(b_k)}$ respectively.

$$
\begin{aligned}
{\mathsf{hcom}'_{\bar{S}}}^{(k)} &= {\mathsf{hcom}_{\bar{S}}}^{(k)} \cdot \mathsf{Com}(\mathsf{ck}, \prod_{i=1}^{K} {\mathsf{pk}_i^{(k)}}^{-\beta_i^{(k)}}; \delta_{\bar{S}}^{(k)}) \cdot \mathsf{Com}(\mathsf{ck}, \prod_{i=1}^{K} ({h'_i}^{(k)})^{\vec{\tau}_i^{(k)}}; 0) \\
&= \mathsf{Com}(\mathsf{ck}, \bar{S}'^{(b_k)}; \mathsf{crnd}_{\bar{S}}^{(k)} + \delta_{\bar{S}}^{(k)}) = \mathsf{Com}(\mathsf{ck}, \bar{S}'^{(b_k)}; {\mathsf{crnd}'_{\bar{S}}}^{(b_k)}) = {\mathsf{hcom}'_{\bar{S}}}^{(b_k)}
\end{aligned}
$$

$$\begin{aligned}\mathsf{hcom}'^{(k)}_{\bar{R}} &= \mathsf{hcom}^{(k)}_{\bar{R}} \cdot \mathsf{hcom}'^{(k)\gamma_0}_{\bar{S}} \cdot \mathsf{Com}(\mathsf{ck}, \prod_{i=1}^{K} (\vec{R}^{(k)}_i)^{-\beta^{(k)}_i} (h'^{(k)}_i)^{\vec{\alpha}_{0,i}}; \delta^{(k)}_{\bar{R}}) \\ &= \mathsf{Com}(\mathsf{ck}, \bar{R}'^{(k)}; \mathsf{crnd}^{(k)}_{\bar{R}} - \gamma^{(k)}_0 \cdot \mathsf{crnd}'^{(b_k)}_{\bar{S}} + \delta^{(k)}_{\bar{R}}) = \mathsf{hcom}'^{(b_k)}_{\bar{R}}\end{aligned}$$

With these equalities, we have

$$\begin{aligned}&\mathsf{H}'(m_{b_k}, (h'^{b_k}_i, \mathsf{pk}'^{(b_k)}_i)_{i\in[K]}, \mathsf{hcom}'^{(k)}_{\bar{S}}, \vec{R}'^{(k)}, \mathsf{hcom}'^{(k)}_{\bar{R}}, A'^{(k)}) - \gamma^{(k)}_0 - \gamma^{(k)}_1 \\ &= d'^{(b_k)} + e'^{(b_k)} - \gamma^{(k)}_0 - \gamma^{(k)}_1 = d^{(k)} + e^{(k)} = c^{(k)},\end{aligned}$$

where the first equality follows from Equation (16), the second to last equality follows from the values of $\gamma^{(k)}_0, \gamma^{(k)}_1$ in Equation (13), and the last equality follows from Equation (14). Thus, the next message from $\mathcal{A}$ will be $\bar{S}^{(k)}, \bar{R}^{(k)}, d^{(k)}, e^{(k)}, \vec{z}^{(k)}_0, z^{(k)}_1, \mathsf{crnd}^{(k)}_{\bar{S}}, \mathsf{crnd}^{(k)}_{\bar{R}}$, 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). □

5.6 Proof of Theorem 5.4 (OMUF-2 of BS_3)

Let $\mathcal{A}$ be an adversary playing the OMUF-2 game of BS_3. We consider the following sequence of games (with the pseudocode description given in Figures 16 to 18).

Game $\mathbf{G}^{\mathcal{A}}_0$: The game first generates the public parameters $\mathsf{par} \leftarrow\!\!\$\, \mathsf{BS}_3.\mathsf{Setup}(1^\lambda, N, K)$ and the secret and public keys $(\mathsf{sk}, \mathsf{pk}) \leftarrow\!\!\$\, \mathsf{BS}_3.\mathsf{KG}(\mathsf{par})$. 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 hash functions $\mathsf{H}, \mathsf{H}', \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}, \mathsf{H}_\varPi$, modeled as random oracles and simulated via lazy sampling. The adversary $\mathcal{A}$ queries the signing oracles S_1 and S_2 for Q_{S_1} and ℓ times respectively, and the random oracles $\mathsf{H}_\star$ for $Q_{\mathsf{H}_\star}$ times for $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}', \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}, \mathsf{H}_\varPi\}$. At the end of the game, $\mathcal{A}$ outputs $\ell + 1$ message-signature pairs $(m^*_k, \sigma^*_k)_{k\in[\ell+1]}$. The adversary $\mathcal{A}$ succeeds if for all $k_1 \neq k_2, m^*_{k_1} \neq m^*_{k_2}$ and for all $k \in [\ell+1], \mathsf{BS}_3.\mathsf{Ver}(\mathsf{pk}, m^*_k, \sigma^*_k) = 1$. We w.l.o.g. assume that $\mathcal{A}$ does not make the same random oracle query twice. Also, we assume that $\mathcal{A}$ makes the random oracle queries that would be made in $\mathsf{BS}_3.\mathsf{Ver}$ when verifying the forgeries. Thus, the total query counts become $\widehat{Q}_{\mathsf{H}} = Q_{\mathsf{H}} + (\ell+1)K, \widehat{Q}_{\mathsf{H}'} = Q_{\mathsf{H}'} + \ell + 1$, and $\widehat{Q}_{\mathsf{H}_\mu} = Q_{\mathsf{H}_\mu} + (\ell+1)K$ for H, H', and H_μ, respectively. The success probability of $\mathcal{A}$ in the game $\mathbf{G}^{\mathcal{A}}_0$ is exactly its advantage in OMUF-2 i.e.

$$\mathsf{Adv}^{\mathsf{omuf}\text{-}2}_{\mathsf{BS}_3}(\mathcal{A}, \lambda) = \Pr[\mathbf{G}^{\mathcal{A}}_0 = 1]\,.$$

Game $\mathbf{G}^{\mathcal{A}}_1$: In this game, in addition to the adversary $\mathcal{A}$ outputting $\ell + 1$ valid message-signature pairs (m^*_k, σ^*_k), the game requires that for each $k \in [\ell+1]$, after parsing $((\mathsf{pk}^*_{i,k}, \varphi^*_{i,k})_{i\in[K]}, \bar{S}^*_k, d^*_k, e^*_k, \vec{z}^*_{0,k}, z^*_{1,k}, \mathsf{crnd}^*_{\bar{S},k}, \mathsf{crnd}^*_{\bar{R},k}) \leftarrow \sigma^*_k$, the game checks that

$$\bar{S}^*_k = \prod_{i=1}^{K} \mathsf{H}(\mu^*_{i,k})^{\mathsf{sk}^*_{i,k}}.$$

where $\mu^*_{i,k} = \mathsf{H}_\mu(m^*_k, \varphi^*_{i,k}), \mathsf{sk}^*_{i,k} = \log_g \mathsf{pk}^*_{i,k}$. If this check fails, the game aborts. We note that if the game knows $\log_g \mathsf{H}(\mu^*_{i,k})$, the game can efficiently check if $\bar{S}^*_k = \prod_{i=1}^{K} \mathsf{pk}^*_{i,k}{}^{\log_g \mathsf{H}(\mu^*_{i,k})}$ instead.

Let Bad denote the event that $\mathcal{A}$ succeeds in game $\mathbf{G}^{\mathcal{A}}_0$ but not $\mathbf{G}^{\mathcal{A}}_1$, which gives $\Pr[\mathbf{G}^{\mathcal{A}}_1 = 1] \geqslant \Pr[\mathbf{G}^{\mathcal{A}}_0 = 1] - \Pr[\mathsf{Bad}]$. Then, by Lemma 5.7, there exist adversaries $\mathcal{B}$ and $\mathcal{B}'$ for the games Binding of HECom and DLOG, respectively, both running in time $t_{\mathcal{B}}, t_{\mathcal{B}'} \approx 2t_{\mathcal{A}}$, such that

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

Game $\mathbf{G}^{\mathcal{A}}_2$: In this game, the game generates W in par as $W \leftarrow g^w$ for $w \leftarrow\!\!\$\, \mathbb{Z}_p$. Then, the signing oracles S_1 and S_2 now generate $(\vec{R}, \bar{R}, A, d, e, \vec{z}_0, z_1)$ as follows:

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

Since the joint distributions of $(\vec{R}, \bar{R}, A, d, e, \vec{z}_0, z_1)$ in this game and the game $\mathbf{G}_1^{\mathcal{A}}$ are identical, we have

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

Game $\mathbf{G}_3^{\mathcal{A}}$: In this game, $\mathsf{hcom}_{\bar{R}}$ is generated as $\mathsf{hcom}_{\bar{S}}^{-d} \cdot \mathsf{Com}(\mathsf{ck}, \prod_{i=1}^{K} h_{i,\vec{J}_i}^{\vec{z}_{0,i}}; \delta_{\bar{R}})$ with $\delta_{\bar{R}} \leftarrow\$ \mathbb{Z}_p^2$, and the game now sets $\mathsf{crnd}_{\bar{R}} \leftarrow \delta_{\bar{R}} - d \cdot \mathsf{crnd}_{\bar{S}}$. Here, $\mathsf{crnd}_{\bar{R}}$ is still uniformly random over $\mathbb{Z}_p^2$ and $\mathsf{hcom}_{\bar{R}}$ still commits to the same $\bar{R}$. Thus,

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

Note that in $\mathbf{G}_3^{\mathcal{A}}$, we only need $\bar{S}$ and $\mathsf{crnd}_{\bar{S}}$ when opening $\mathsf{hcom}_{\bar{R}}$ in S_2, while computing $\mathsf{hcom}_{\bar{R}}$ in S_1 only requires $\mathsf{hcom}_{\bar{S}}$.

Game $\mathbf{G}_4^{\mathcal{A}}$: In this game, the signing oracle S_2 now generates the proof π by using a simulator Sim (of which existence is implied by Lemma 5.2) on the input $(g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i\in[K]}, \bar{S})$. Following Lemma 5.2, by the zero-knowledge property of Π, and the fact that $\mathcal{A}$ makes ℓ and Q_{H_Π} queries to S_2 and H_Π respectively, 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 $\mathbf{G}_5^{\mathcal{A}}$: In this game, the game aborts if one of the following occurs.

(a) For each $\mathsf{H}_\star \in \{\mathsf{H}_{\mathsf{com}}, \mathsf{H}_\mu\}$, there exist two queries $x \neq x'$ to $\mathsf{H}_\star$ such that $\mathsf{H}_\star(x) = \mathsf{H}_\star(x')$.

(b) The game additionally keeps track of a mapping $\hat{\mathsf{r}}[\cdot] : \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$. Then, for each query (com, h) to H_{cc} where $\mathsf{com} = (\mathsf{com}_{i,j})_{i\in[K],j\in[N]}$ and $h = (h_{i,j})_{i\in[K],j\in[N]}$ the game does the following: For each $i \in [K]$ and $j \in [N]$, check if there exists a query r' to $\mathsf{H}_{\mathsf{com}}$ such that $\mathsf{H}_{\mathsf{com}}(\mathsf{r}') = \mathsf{com}_{i,j}$, then if there is one, set $\hat{\mathsf{r}}[\mathsf{com}_{i,j}] \leftarrow \mathsf{r}'$; otherwise, set $\hat{\mathsf{r}}[\mathsf{com}_{i,j}] \leftarrow \perp$ and abort if later there is a query r' to $\mathsf{H}_{\mathsf{com}}$ where $\mathsf{H}_{\mathsf{com}}(\mathsf{r}') = \mathsf{com}_{i,j}$.

The view of $\mathcal{A}$ in this game only differs from its view in $\mathbf{G}_5^{\mathcal{A}}$ if the game aborts. The abort probability for (a) corresponds to the probability of collisions in the outputs of $\mathsf{H}_{\mathsf{com}}$ and H_μ which is bounded by $(Q_{\mathsf{H}_{\mathsf{com}}}^2 + \widehat{Q}_{\mathsf{H}_\mu}^2)/2^\lambda$. Also, since the output of $\mathsf{H}_{\mathsf{com}}$ is uniformly random in $\{0,1\}^\lambda$, the abort probability for (b) is bounded by $Q_{\mathsf{H}_{\mathsf{com}}} Q_{\mathsf{H}_{cc}}/2^\lambda$, considering all pairs of queries to $\mathsf{H}_{\mathsf{com}}$ and H_{cc}. Thus,

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

Before proceeding to the next game, we consider an event where $\mathcal{A}$ queries S_1 with the input $\mathsf{umsg}_1 = (\vec{J}, ((\mathsf{r}_{i,j})_{j\neq\vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i\in[K]})$. We consider the case where $\mathsf{Check}(\mathsf{umsg}_1) = 1$ which would define values $\mathsf{com} = (\mathsf{com}_{i,j})_{i\in[K],j\in[N]}$ and $h = (h_{i,j})_{i\in[K],j\in[N]}$ such that $\mathsf{H}_{cc}(\mathsf{com}, h) = \vec{J}$. Also, consider the values $\hat{\mathsf{r}}[\mathsf{com}_{i,j}]$ related to the query $\mathsf{H}_{cc}(\mathsf{com}, h)$ defined in $\mathbf{G}_5^{\mathcal{A}}$. For each instance $i \in [K]$, we have the following observations:

- If for some $j \in [N]$, $\hat{\mathsf{r}}[\mathsf{com}_{i,j}] = \perp$, then $j = \vec{J}_i$. For other $j' \neq \vec{J}_i$, since $\mathsf{r}_{i,j'}$ is revealed in umsg_1 and $\mathsf{Check}(\mathsf{umsg}_1) = 1$, $\mathsf{com}_{i,j'} = \mathsf{H}_{\mathsf{com}}(\mathsf{r}_{i,j'})$, by the abort (b) introduced in $\mathbf{G}_5^{\mathcal{A}}$, $\hat{\mathsf{r}}[\mathsf{com}_{i,j'}] \neq \perp$.
- If for some $j \in [N]$, $\hat{\mathsf{r}}[\mathsf{com}_{i,j}] = (\mu, \varepsilon) \neq \perp$, but $h_{i,j} \neq \mathsf{H}(\mu)g^\beta$ where $\beta \leftarrow \mathsf{H}_\beta(\varepsilon_{i,j})$, then $j = \vec{J}_i$. This is because of the no collision condition (abort (a)) in $\mathsf{H}_{\mathsf{com}}$ introduced in $\mathbf{G}_5^{\mathcal{A}}$, meaning for $j' \neq \vec{J}_i$, $\hat{\mathsf{r}}[\mathsf{com}_{i,j'}] = \mathsf{r}_{i,j'} = (\mu_{i,j'}, \varepsilon_{i,j'})$. Then, with $\mathsf{Check}(\mathsf{umsg}_1) = 1$, we have $h_{i,j} = \mathsf{H}(\mu_{i,j'})g^{\mathsf{H}_\beta(\varepsilon_{i,j'})}$.

We say *the adversary $\mathcal{A}$ successfully cheats in instance $i \in [K]$* if one of the two cases above occurs while $\mathsf{Check}(\mathsf{umsg}_1) = 1$. Since the values $\hat{\mathsf{r}}[\mathsf{com}_{i,j}]$ 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 $1/N$. Then, the probability in which $\mathcal{A}$ successfully cheats in all instance is at most $1/N^K$.

Game $\mathbf{G}_6^{\mathcal{A}}$: In this game, if $\mathcal{A}$ successfully cheats in all instance $i \in [K]$ in some signing query to S_1, the game aborts. By the above discussion and applying the union-bound over all queries to S_1,

$$\Pr[\mathbf{G}_6^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_5^{\mathcal{A}} = 1] - \frac{Q_{\mathrm{S}_1}}{N^K} .$$

Game $\mathbf{G}_7^{\mathcal{A}}$: In this game, the game aborts if $\mathcal{A}$ queries H with μ such that there is no x where $\mathsf{H}_\mu(x) = \mu$ at the time, but later on there is a query x to H_μ where $\mathsf{H}_\mu(x) = \mu$. The view of $\mathcal{A}$ only changes if the game aborts. Then, since the outputs to $\mathsf{H}_\mu(\cdot)$ is uniformly random, we can bound the probability of the abort by considering all pairs of queries to H and H_μ. Thus,

$$\Pr[\mathbf{G}_7^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_6^{\mathcal{A}} = 1] - \frac{\widehat{Q}_{\mathsf{H}}\widehat{Q}_{\mathsf{H}_\mu}}{2^\lambda} .$$

Game $\mathbf{G}_8^{\mathcal{A}}$: In this game, the game introduces two mappings $\hat{b}[\cdot], b[\cdot]$ such that when $\mathcal{A}$ queries $\mathsf{H}_\mu(m, \varphi)$ and no query of the form $(m, \cdot)$ has been made 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 defined, the game searches for a previous query (m, φ) such that $\mathsf{H}_\mu(m, \varphi) = \mu$ and set $b[\mu] \leftarrow \hat{b}[m]$. If such query does not exist, set $b[\mu] \leftarrow 0$. Since both b and $\hat{b}$ are hidden from the view of $\mathcal{A}$, the view of $\mathcal{A}$ remains the same. Thus,

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

Note that by the change in $\mathbf{G}_7^{\mathcal{A}}$, it cannot be the case that $\hat{b}[m] = 1$ but $b[\mu] = 0$ for some m and $\mu = \mathsf{H}_\mu(m, \cdot)$, since this means that the query $\mathsf{H}(\mu)$ is made before $\mathsf{H}_\mu(m, \cdot)$.
Game $\mathbf{G}_9^{\mathcal{A}}$: In this game, we made the following changes to $\mathbf{G}_8^{\mathcal{A}}$ as follows:

- The game introduce a list $\mathcal{L}$.
- Recall that by the change in $\mathbf{G}_6^{\mathcal{A}}$, for each signing session, there exists an instance $i^* \in [K]$ where $\mathcal{A}$ does not successfully cheat. Thus, the game can extract $\mathsf{r} = (\mu, \varepsilon)$ such that $\mathsf{H}_{\mathsf{com}}(\mathsf{r}) = \mathsf{com}_{i*, \vec{J}_{i*}}$ and $\mathsf{H}(\mu) g^{\mathsf{H}_\beta(\varepsilon)} = h_{i*, \vec{J}_{i*}}$. Then, **for each query to $\mathbf{S}_2$**, the game aborts if $b[\mu] = 1$. Otherwise, the game tries to find a previous query $(m, \cdot)$ such that $\mu = \mathsf{H}_\mu(m, \cdot)$ and sets $\mathcal{L} \leftarrow \mathcal{L} \cup \{(\mu, m)\}$, if such m exists.
- When $\mathcal{A}$ returns $\ell + 1$ forgeries for distinct messages, since $\mathcal{A}$ queries S_2 for ℓ times, there exists $m^\star$ from one of the message-signature pairs such that $(\cdot, m^\star) \notin \mathcal{L}$. The game aborts if $\hat{b}[m^\star] = 0$.

Consider the success probability of $\mathcal{A}$.

$$\Pr[\mathbf{G}_9^{\mathcal{A}} = 1] = \Pr[\mathcal{A} \text{ succeeds} | \mathbf{G}_9^{\mathcal{A}} \text{ does not abort}] \Pr[\mathbf{G}_9^{\mathcal{A}} \text{ does not abort}] .$$

Notice that the view of $\mathcal{A}$, if the game does not abort, is exactly as in $\mathbf{G}_8^{\mathcal{A}}$. Thus, we consider the probability that $\mathbf{G}_9^{\mathcal{A}}$ does not abort, which corresponds to the event that for all $(\mu, m) \in \mathcal{L}, b[\mu] = 0$ and $\hat{b}[m^\star] = 1$. Hence, we can bound

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

The first equality follows from the independence of sampling each $\hat{b}$ and that $b[\mu] = \hat{b}[m]$. The next inequality follows from $|\mathcal{L}| \leqslant \ell$ (since the game appends to $\mathcal{L}$ only in S_2) and $\hat{b}[m]$ for distinct m being independently sampled. The last inequality follows from $(1 - 1/x)^x \geqslant 1/4$ for $x \geqslant 2$. Therefore, we have

$$\Pr[\mathbf{G}_9^{\mathcal{A}} = 1] \geqslant \frac{1}{4\ell} \Pr[\mathbf{G}_8^{\mathcal{A}} = 1] .$$

Game $\mathbf{G}_{10}^{\mathcal{A}}$: In this game, the game keeps track of a mapping $t[\cdot] : \{0,1\}^\lambda \to \mathbb{Z}_p$ and initialize a $Y \leftarrow_\$ \mathbb{G}$ at the start of the game. Then, for each new query $\mathsf{H}(\mu)$, the game returns $\mathsf{H}(\mu) \leftarrow Y^{b[\mu]} g^{t[\mu]}$ where $t[\mu] \leftarrow_\$ \mathbb{Z}_p$ and $b[\mu]$ is as defined in $\mathbf{G}_8^{\mathcal{A}}$. The view of $\mathcal{A}$ is the same as in $\mathbf{G}_9^{\mathcal{A}}$ since $\mathsf{H}(\mu)$ is still uniformly random over $\mathbb{G}$. Thus,

$$\Pr[\mathbf{G}_{10}^{\mathcal{A}} = 1] = \Pr[\mathbf{G}_9^{\mathcal{A}} = 1] \ .$$

Game $\mathbf{G}_{11}^{\mathcal{A}}$: In this game, the game generates $\{\mathsf{sk}_i\}_{i \in [K]}$ in each signing session as follows: recall the non-cheating instance i^* from $\mathbf{G}_6^{\mathcal{A}}$, the game now generates $\mathsf{sk}_i \leftarrow_\$ \mathbb{Z}_p$ for $i \neq i^*$ and sets $\mathsf{sk}_{i^*} \leftarrow \mathsf{sk} - \sum_{i \neq i^*} \mathsf{sk}_i$, along with $\mathsf{pk}_{i^*} \leftarrow \mathsf{pk} \prod_{i \neq i^*} \mathsf{pk}_i^{-1}$. This is only a syntactical change and the view of $\mathcal{A}$ stays the same.

$$\Pr[\mathbf{G}_{11}^{\mathcal{A}} = 1] = \Pr[\mathbf{G}_{10}^{\mathcal{A}} = 1] \ .$$

Game $\mathbf{G}_{12}^{\mathcal{A}}$: In this game, the game now aborts if $\mathsf{sk} = 0$, and if this abort does not occur, the commitment key ck is now generated along with a trapdoor td with a base pk embedded i.e., $(\mathsf{ck}, \mathsf{td}) \leftarrow_\$ \mathsf{HECom.TGen}((\mathbb{G}, p, g), \mathsf{pk})$. The probability of the abort occurring is at most $1/p$. Also, by the uniform key property of HECom, ck generated with $\mathsf{pk} \neq 1_{\mathbb{G}}$ is distributed identically to $\mathsf{ck} \leftarrow_\$ \mathsf{HECom.Gen}((\mathbb{G}, p, g))$. Thus,

$$\Pr[\mathbf{G}_{12}^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_{11}^{\mathcal{A}} = 1] - \frac{1}{p} \ .$$

Game $\mathbf{G}_{13}^{\mathcal{A}}$: In this game, the game does not compute sk_{i^*} in each signing session anymore and changes the way $\mathsf{hcom}_{\bar{S}}$ is computed and opened as follows:

- First, observe that we can write $\bar{S}$ as

$$\bar{S} = h_{i^*, \vec{J}_{i^*}}^{\mathsf{sk}_{i^*}} \prod_{i \neq i^*} h_{i, \vec{J}_i}^{\mathsf{sk}_i} = h_{i^*, \vec{J}_{i^*}}^{\mathsf{sk} - \sum_{i \neq i^*} \mathsf{sk}_i} \prod_{i \neq i^*} h_{i, \vec{J}_i}^{\mathsf{sk}_i} = \mathsf{pk}^{\log_g h_{i^*, \vec{J}_{i^*}}} \prod_{i \neq i^*} h_{i, \vec{J}_i}^{\mathsf{sk}_i} h_{i^*, \vec{J}_{i^*}}^{-\mathsf{sk}_i} \ .$$

  Then, in S_1, the game now computes $(\mathsf{hcom}_{\bar{S}}, \mathsf{st}_{\mathsf{com}}) \leftarrow_\$ \mathsf{HECom.TCom}(\mathsf{td}, S')$ for $S' = \prod_{i \neq i^*} h_{i, \vec{J}_i}^{\mathsf{sk}_i} h_{i^*, \vec{J}_{i^*}}^{-\mathsf{sk}_i}$.
- When S_2 of the same session is queried, by the change in $\mathbf{G}_9^{\mathcal{A}}$, we know that $h_{i^*, \vec{J}_{i^*}} = \mathsf{H}(\mu) g^{\beta}$ for some (μ, ε) with $\beta = \mathsf{H}_\beta(\varepsilon)$ and that $b[\mu] = 0$ (otherwise, the game aborts). Then, by the change in $\mathbf{G}_{10}^{\mathcal{A}}$, the game knows $\log_g h_{i^*, \vec{J}_{i^*}} = \beta + t[\mu]$. Thus, the game opens $\mathsf{hcom}_{\bar{S}}$ as $(\bar{S}, \mathsf{crnd}_{\bar{S}}) \leftarrow_\$ \mathsf{HECom.TOpen}(\mathsf{st}_{\mathsf{com}}, \beta + t[\mu])$.

By the special equivocation property of HECom, the view of $\mathcal{A}$ stays the same, unless the matrix $\mathbf{D} \in \mathbb{Z}_p^{2 \times 2}$ contained in td is not invertible, which occurs with probability at most $2/p$ by the Schwartz-Zippel lemma. Thus,

$$\Pr[\mathbf{G}_{13}^{\mathcal{A}} = 1] \geqslant \Pr[\mathbf{G}_{12}^{\mathcal{A}} = 1] - \frac{2}{p} \ .$$

Lastly, we give a reduction $\mathcal{B}''$ playing the CDH game as follows:

- The reduction $\mathcal{B}''$ takes input $(\mathbb{G}, p, g, X, Y)$. If $X = 1_{\mathbb{G}}$, $\mathcal{B}''$ returns $1_{\mathbb{G}}$. Otherwise, the game sets $\mathsf{pk} \leftarrow X$, $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W, \mathsf{ck}, K, N)$, with W and ck generated as in $\mathbf{G}_{13}^{\mathcal{A}}$, and runs $\mathcal{A}(\mathsf{par}, \mathsf{pk})$.
- The random oracles $\mathsf{H}_\mu, \mathsf{H}_{cc}, \mathsf{H}_{\mathsf{com}}, \mathsf{H}', \mathsf{H}_\Pi$ are simulated as in $\mathbf{G}_{13}^{\mathcal{A}}$; however, for H, the game uses the CDH input Y in place of the Y used in $\mathbf{G}_{10}^{\mathcal{A}}$.
- The signing oracles are simulated without sk as in $\mathbf{G}_{13}^{\mathcal{A}}$.
- When the adversary returns $\ell+1$ message-signature pairs, the reduction checks if all the pairs are valid and the messages are distinct. If not, $\mathcal{B}''$ aborts. Then, the reduction identifies $m^\star$ as in $\mathbf{G}_9^{\mathcal{A}}$ and let $\sigma^\star$ be the corresponding signature for $m^\star$. The reduction parses $((\mathsf{pk}_i^\star, \varphi_i^\star)_{i \in [K]}, \bar{S}^\star, d^\star, e^\star, \vec{z}_0^\star, z_1^\star, \mathsf{crnd}_{\bar{S}}^\star, \mathsf{crnd}_{\bar{R}}^\star) \leftarrow \sigma^\star$, computes $\mu_i^\star = \mathsf{H}_\mu(m^\star, \varphi_i^\star)$, and returns

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

First, we can see that the running time of $\mathcal{B}''$ is about that of $\mathcal{A}$. Next, we will show the correctness of the reduction. We can see that if $X = 1_{\mathbb{G}}$, the game is trivial for $\mathcal{B}''$; otherwise, $\mathcal{B}''$ simulates the game $\mathbf{G}_{13}^{\mathcal{A}}$ perfectly. Then, suppose $\mathcal{A}$ succeeds in $\mathbf{G}_{13}^{\mathcal{A}}$. By the change in $\mathbf{G}_{1}^{\mathcal{A}}$, this means that for $(m^\star, \sigma^\star)$, we have $\bar{S}^\star = \prod_{i=1}^{K} \mathsf{pk}_i^{\star \log_g \mathsf{H}(\mu_i^\star)}$. Thus,

$$\bar{S}^\star = \prod_{i=1}^{K} \mathsf{pk}_i^{\star \log_g \mathsf{H}(\mu_i^\star)} = \prod_{i=1}^{K} \mathsf{pk}_i^{\star b[\mu_i^\star]\cdot \log_g Y + t[\mu_i^\star]} = \mathsf{pk}^{\log_g Y} \prod_{i=1}^{K} \mathsf{pk}_i^{\star t[\mu_i^\star]} ,$$

where the third equality follows from $b[\mu_i^\star] = \hat{b}[m^\star] = 1$ for any $i \in [K]$ (due to the changes in games $\mathbf{G}_{7}^{\mathcal{A}} - \mathbf{G}_{9}^{\mathcal{A}}$ and that $\mathsf{H}_\mu(m^\star, \varphi_i^\star) = \mu_i^\star$). Hence, $\mathcal{B}''$ succeeds in the CDH game as $Z = \mathsf{pk}^{\log_g Y} = X^{\log_g Y}$, implying $\Pr[\mathbf{G}_{13}^{\mathcal{A}} = 1] \leqslant \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{cdh}}(\mathcal{B}'', \lambda)$. Finally, combining all the advantage changes,

$$\begin{aligned}\mathsf{Adv}_{\mathsf{BS}_3}^{\mathrm{omuf}\text{-}2}(\mathcal{A}, \lambda) \leqslant & (\ell+1)\left(\sqrt{\widehat{Q}_{\mathsf{H}'}\left(\mathsf{Adv}_{\mathsf{HECom}}^{\mathrm{binding}}(\mathcal{B}, \lambda) + \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{dlog}}(\mathcal{B}', \lambda)\right)} + \frac{\widehat{Q}_{\mathsf{H}'}}{p}\right) + \frac{\ell(\ell + Q_{\mathsf{H}_\Pi} + 12)}{p} \\ & + \frac{Q_{\mathrm{S}_1}}{N^K} + \frac{Q_{\mathsf{H}_{\mathsf{com}}}^2 + \widehat{Q}_{\mathsf{H}_\mu}^2 + Q_{\mathsf{H}_{\mathsf{com}}} Q_{\mathsf{H}_{cc}} + \widehat{Q}_{\mathsf{H}} \widehat{Q}_{\mathsf{H}_\mu}}{2^\lambda} + 4\ell \cdot \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{cdh}}(\mathcal{B}'', \lambda) . \square\end{aligned}$$

Lemma 5.7. *Let* Bad *be the event where $\mathcal{A}$ succeeds in game $\mathbf{G}_0^{\mathcal{A}}$ but not $\mathbf{G}_1^{\mathcal{A}}$. Then, there exist adversaries $\mathcal{B}$ for the game* Binding *of* HECom *and $\mathcal{B}'$ for the game* DLOG *both with running time $t_{\mathcal{B}}, t_{\mathcal{B}'} \approx 2t_{\mathcal{A}}$ such that*

$$\Pr[\mathsf{Bad}] \leqslant (\ell+1)\left(\sqrt{\widehat{Q}_{\mathsf{H}'}\left(\mathsf{Adv}_{\mathsf{HECom}}^{\mathrm{binding}}(\mathcal{B}, \lambda) + \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{dlog}}(\mathcal{B}', \lambda)\right)} + \frac{\widehat{Q}_{\mathsf{H}'}}{p}\right) .$$

Proof. First, observe that Bad 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 $k \in [\ell+1]$, $\mathsf{BS}_3.\mathsf{Ver}(\mathsf{pk}, \sigma_k^*, m_k^*) = 1$, and (3) there exists $k \in [\ell+1]$ such that after parsing the signature $((\mathsf{pk}_{i,k}^*, \varphi_{i,k}^*)_{i\in[K]}, \bar{S}_k^*, d_k^*, e_k^*, \vec{z}_{0,k}^*, z_{1,k}^*, \mathsf{crnd}_{\bar{S},k}^*, \mathsf{crnd}_{\bar{R},k}^*) \leftarrow \sigma_k^*$, and setting $\mu_{i,k}^* \leftarrow \mathsf{H}_\mu(m_k^*, \varphi_{i,k}^*)$, we have $\bar{S}_k^* \neq \prod_{i=1}^{K} \mathsf{H}(\mu_{i,k}^*)^{\log_g \mathsf{pk}_{i,k}^*}$. 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 message-signature pair (m_k^*, σ_k^*). We can see that $\mathsf{Bad} = \bigcup_{k=1}^{\ell+1} \mathsf{Bad}_k$.

To bound Bad_k, define the following wrapper $\mathcal{A}_k$ over $\mathcal{A}$, which takes inputs: the instance $(\mathbb{G}, p, g, W, \mathsf{ck}, K, N)$, the outputs $(c_1, \ldots, c_{\widehat{Q}_{\mathsf{H}'}})$ of H', and a random tape ρ.

1. Extract from the random tape ρ, the following

$$(\mathsf{sk}, ((\mathsf{sk}_{j,i})_{i\in[K-1]}, \vec{r}_{0,j}, e_j, z_{1,j}, \rho_{\Pi,j}, \mathsf{crnd}_{\bar{S},j}, \mathsf{crnd}_{\bar{R},j})_{j\in[Q_{\mathrm{S}_1}]}, (t_i)_{i\in[\widehat{Q}_{\mathsf{H}}]}, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_\Pi, \mathsf{H}_{cc}, \rho')$$

   where $\mathsf{sk} \in \mathbb{Z}_p$ and for $i \in [K], j \in [Q_{\mathrm{S}_1}]$, $\mathsf{sk}_{i,j}, e_j, z_{1,j} \in \mathbb{Z}_p, \vec{r}_{0,j} \in \mathbb{Z}_p^K$, while $\rho_{\Pi,j}$ denotes the randomness used to generate π in the j-th signing session, $\mathsf{crnd}_{\bar{S},j}, \mathsf{crnd}_{\bar{R},j} \in \mathbb{Z}_p^2$ denote the randomness for the commitments in the j-th signing session, $(t_i)_{i\in[\widehat{Q}_{\mathsf{H}}]}$ denotes a list of values from $\mathbb{Z}_p$ which will be used to program H, $\mathsf{H}_\star \in \{\mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}\}$ denote a lists of $Q_{\mathsf{H}_\star}$ values ($\widehat{Q}_{\mathsf{H}_\mu}$ values for $\mathsf{H}_\star = \mathsf{H}_\mu$) in the codomain of $\mathsf{H}_\star$, and H_Π denotes a list of $Q_{\mathsf{H}_\Pi} + \ell$ values in $\mathbb{Z}_p$. Additionally, we denote $\mathsf{H}_\star[i]$ as the i-th entry in the list for $\mathsf{H}_\star \in \{\mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}, \mathsf{H}_\Pi\}$.
2. Set $\mathsf{par} \leftarrow (\mathbb{G}, p, g, W, \mathsf{ck})$ and $\mathsf{pk} \leftarrow g^{\mathsf{sk}}$.
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 signing query with session ID j ($j \in [Q_{\mathrm{S}_1}]$) to S_1 and S_2, use $(\mathsf{sk}, (\mathsf{sk}_{j,i})_{i\in[K-1]}, \vec{r}_{0,j}, e_j, z_{1,j}, \rho_{\Pi,j}, \mathsf{crnd}_{\bar{S},j}, \mathsf{crnd}_{\bar{R},j})$ to answer the query as in $\mathsf{BS}_3.\mathrm{S}_1$ and $\mathsf{BS}_3.\mathrm{S}_2$ respectively.
   - For the i-th query to H ($i \in [\widehat{Q}_{\mathsf{H}}]$), return g^{t_i} and set $t[\cdot] \leftarrow t_i$ accordingly.
   - For the i-th query to H' ($i \in [\widehat{Q}_{\mathsf{H}'}]$), return c_i.
   - For the i-th query to $\mathsf{H}_\star \in \{\mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}\}$ ($i \in [Q_{\mathsf{H}_\star}]$ and $i \in [\widehat{Q}_{\mathsf{H}_\mu}]$ for $\mathsf{H}_\star = \mathsf{H}_\mu$), return $\mathsf{H}_\star[i]$.

- For the i-th query to H_Π ($i \in [Q_{\mathsf{H}_\Pi} + \ell]$), return $\mathsf{H}_\Pi[i]$. (In these queries, we accounted for the queries that the wrapper made to generate π in each query to S_2.)

4. If the event Bad_k does not occur, return $(\bot, \bot)$.
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, I is the index of a query of the form $(m, (h_i, \mathsf{pk}_i)_{i\in[K]}, \mathsf{hcom}_{\bar{S}}, \vec{R}, \mathsf{hcom}_{\bar{R}}, A)$, where each value is defined as:
   - $m = m_k^*$.
   - For $i \in [K]$, $\mathsf{pk}_i = \mathsf{pk}_{i,k}^*, h_i = \mathsf{H}(\mathsf{H}_\mu(m_k^*, \varphi_{i,k}^*))$, and $\vec{R}_i = g^{\vec{z}^*_{0,k,i}} \mathsf{pk}_i^{-d_k^*}$.
   - $\mathsf{hcom}_{\bar{S}} = \mathsf{Com}(\mathsf{ck}, \bar{S}_k^*; \mathsf{crnd}_{\bar{S},k}^*)$
   - $\mathsf{hcom}_{\bar{R}} = \mathsf{Com}(\mathsf{ck}, \bar{R}; \mathsf{crnd}_{\bar{S},k}^*)$ where $\bar{R} = (\bar{S}_k^*)^{-d_k^*} \prod_{i=1}^K h_i^{\vec{z}^*_{0,k,i}}$.
   - $A = W^{-e_k^*} g^{z^*_{1,k}}$.

   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 in $\mathcal{A}_k$ allows us to check for event Bad_k efficiently, i.e., by checking $\bar{S}_k^* \neq \prod_{i=1}^K \mathsf{pk}_{i,k}^{*\,t[\mu^*_{i,k}]}$, which means that the running time of $\mathcal{A}_k$ is roughly that of $\mathcal{A}$.

Now, consider another wrapper $\mathsf{Fork}^{\mathcal{A}_k}$ taking the input $(\mathbb{G}, p, g, W, \mathsf{ck})$ defined as follows:

1. First, $\mathsf{Fork}^{\mathcal{A}_k}$ samples $c_1, \ldots, c_{\hat{Q}_{\mathsf{H}'}} \leftarrow\!\!\$\, \mathbb{Z}_p$ along with the random tape ρ.
2. Run $(I, (m, \sigma)) \leftarrow\!\!\$\, \mathcal{A}_k((\mathbb{G}, p, g, W, \mathsf{ck}, K, N), (c_1, \ldots, c_{\hat{Q}_{\mathsf{H}'}}); \rho)$.
3. If $I = 0$, abort. If not, sample $c'_I, \ldots, c'_{\hat{Q}_{\mathsf{H}'}} \leftarrow\!\!\$\, \mathbb{Z}_p$ and
run $(I', (m', \sigma')) \leftarrow\!\!\$\, \mathcal{A}_k((\mathbb{G}, p, g, W, \mathsf{ck}, K, N), (c_1, \ldots, c_{I-1}, c'_I, \ldots, c'_{\hat{Q}_{\mathsf{H}'}}); \rho)$.
4. If $I \neq I'$ or $c'_I = c_I$, abort. Otherwise, parse
$$((\mathsf{pk}_i, \varphi_i)_{i\in[K]}, \bar{S}, d, e, \vec{z}_0, z_1, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}) \leftarrow \sigma\ ,$$
$$((\mathsf{pk}'_i, \varphi'_i)_{i\in[K]}, \bar{S}', d', e', \vec{z}'_0, z'_1, \mathsf{crnd}'_{\bar{S}}, \mathsf{crnd}'_{\bar{R}}) \leftarrow \sigma'\ .$$
Then, compute $\bar{R} = \bar{S}^{-d} \prod_{i=1}^K h_i^{\vec{z}_{0,i}}$ and $\bar{R}' = \bar{S}'^{-d'} \prod_{i=1}^K h_i'^{\vec{z}'_{0,i}}$ and return
$$(\bar{S}, \bar{S}', \bar{R}, \bar{R}', \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \mathsf{crnd}'_{\bar{S}}, \mathsf{crnd}'_{\bar{R}}, z_1 - z'_1, e - e')\ .$$

Since $\mathsf{Fork}^{\mathcal{A}_k}$ runs $\mathcal{A}_k$ twice and the running time of $\mathcal{A}_k$ is about that of $\mathcal{A}$, we have $t_{\mathsf{Fork}^{\mathcal{A}_k}} \approx 2t_{\mathcal{A}}$. Next, we consider the event where $\mathsf{Fork}^{\mathcal{A}_k}$ does not abort (i.e., $I = I' \neq \bot$ and $c_I \neq c'_I$). Notice that $I = I' \neq \bot$, so the message-signature pairs (m, σ) and (m', σ'): (a) are valid signatures corresponding to the I-th query of $\mathcal{A}$ to H', and (b) for $i \in [K]$, let $\mu_i \leftarrow \mathsf{H}_\mu(m, \varphi_i), \mu'_i \leftarrow \mathsf{H}_\mu(m', \varphi'_i)$, we have $\bar{S} \neq \prod_{i=1}^K \mathsf{H}(\mu_i)^{\log_g \mathsf{pk}_i}$ and $\bar{S}' \neq \prod_{i=1}^K \mathsf{H}(\mu'_i)^{\log_g \mathsf{pk}'_i}$. Consider two events: (E_1) $\bar{S} \neq \bar{S}'$ or $\bar{R} \neq \bar{R}'$, and (E_2) $\bar{S} = \bar{S}'$ and $\bar{R} = \bar{R}'$. We can see that
$$\Pr[I = I' \neq \bot \wedge c_I \neq c'_I] = \Pr[\mathsf{Fork}^{\mathcal{A}_k} \text{ does not abort}] \leqslant \Pr[E_1] + \Pr[E_2]\ .$$

For the event E_1, by the observation (a), we have that $\mathsf{Com}(\mathsf{ck}, \bar{S}; \mathsf{crnd}_{\bar{S}}) = \mathsf{Com}(\mathsf{ck}, \bar{S}'; \mathsf{crnd}'_{\bar{S}})$ and $\mathsf{Com}(\mathsf{ck}, \bar{R}; \mathsf{crnd}_{\bar{R}}) = \mathsf{Com}(\mathsf{ck}, \bar{R}'; \mathsf{crnd}'_{\bar{R}})$. Thus, we can construct a reduction $\mathcal{B}$ playing the binding game of HECom and using $\mathsf{Fork}^{\mathcal{A}_k}$, with running time $t_{\mathcal{B}} \approx t_{\mathsf{Fork}^{\mathcal{A}_k}}$, such that $\Pr[E_1] \leqslant \mathsf{Adv}^{\mathrm{binding}}_{\mathsf{HECom}}(\mathcal{B}, \lambda)$.

For the event E_2 ($\bar{S} = \bar{S}'$ and $\bar{R} = \bar{R}'$), we have that

(i) $\bar{S}^{-d} \prod_{i=1}^K h_i^{\vec{z}_{0,i}} = \bar{R} = \bar{R}' = \bar{S}'^{-d'} \prod_{i=1}^K h_i'^{\vec{z}'_{0,i}}$
(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 $i \in [K], \mathsf{pk}_i^{-d} g^{\vec{z}_{0,i}} = \mathsf{pk}_i'^{-d'} g^{\vec{z}'_{0,i}}$.
(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 $i \in [K]$, we have $\mathsf{pk}_i^{(d-d')\log h_i} = (\mathsf{pk}_i^d \mathsf{pk}_i'^{-d'})^{\log_g h_i} = g^{(\vec{z}_{0,i} - \vec{z}'_{0,i})\log_g h_i} = h_i^{\vec{z}_{0,i} - \vec{z}'_{0,i}}$. Then,

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

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 $e \neq e'$. Therefore, we have that $(z_1 - z'_1)(e - e')^{-1} = \log_g W$. Hence, we can construct a reduction $\mathcal{B}'$ playing the DLOG game and using $\mathsf{Fork}^{\mathcal{A}_k}$, with running time $t_{\mathcal{B}'} \approx t_{\mathsf{Fork}^{\mathcal{A}_k}}$, such that $\Pr[E_2] \leqslant \mathsf{Adv}^{\mathrm{dlog}}_{\mathsf{GGen}}(\mathcal{B}', \lambda)$.

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

$$\begin{aligned} \Pr[\mathsf{Bad}_k] &\leqslant \sqrt{\widehat{Q}_{\mathsf{H}'} \Pr[I = I' \neq \bot \wedge c_I \neq c'_I]} + \frac{\widehat{Q}_{\mathsf{H}'}}{p} \\ &\leqslant \sqrt{\widehat{Q}_{\mathsf{H}'} \left(\mathsf{Adv}^{\mathrm{binding}}_{\mathsf{HECom}}(\mathcal{B}, \lambda) + \mathsf{Adv}^{\mathrm{dlog}}_{\mathsf{GGen}}(\mathcal{B}', \lambda)\right)} + \frac{\widehat{Q}_{\mathsf{H}'}}{p} . \end{aligned}$$

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

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.

BCJ08. Ali Bagherzandi, Jung Hee Cheon, and Stanislaw Jarecki. Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM CCS 2008*, pages 449–458. ACM Press, October 2008.

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.

BLT$^+$24. Renas Bacho, Julian Loss, Stefano Tessaro, Benedikt Wagner, and Chenzhi Zhu. Twinkle: Threshold signatures from ddh with full adaptive security. In *EUROCRYPT 2024: 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26–30, 2024, Proceedings, Part I*, page 429–459, Berlin, Heidelberg, 2024. Springer-Verlag.

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.

BRSJZ23. Paulo L Barreto, Devin D Reich, Marcos A Simplicio Jr, and Gustavo HM Zanon. Blind signatures from zero knowledge in the kummer variety. In *Anais do XXIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 139–152. SBC, 2023.

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.

CKM23a. Elizabeth C. Crites, Chelsea Komlo, and Mary Maller. Fully adaptive Schnorr threshold signatures. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part I*, volume 14081 of *LNCS*, pages 678–709. Springer, Heidelberg, August 2023.

CKM$^+$23b. 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, *CRYPTO 2023, Part I*, volume 14081 of *LNCS*, pages 710–742. Springer, Heidelberg, August 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.

FW24. Georg Fuchsbauer and Mathias Wolf. Concurrently secure blind schnorr signatures. In *Advances in Cryptology – EUROCRYPT 2024: 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26–30, 2024, Proceedings, Part II*, page 124–160, Berlin, Heidelberg, 2024. Springer-Verlag.

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.

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.

HLTW24. L. Hanzlik, J. Loss, S. Thyagarajan, and B. Wagner. Sweep-uc: Swapping coins privately. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 84–84, Los Alamitos, CA, USA, may 2024. IEEE Computer Society.

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.

KNR24. Julia Kastner, Ky Nguyen, and Michael Reichle. Pairing-free blind signatures from standard assumptions in the rom. CRYPTO 2024, 2024. "to appear".

KRS23. Shuichi Katsumata, Michael Reichle, and Yusuke Sakai. Practical round-optimal blind signatures in the ROM from standard assumptions. In Jian Guo and Ron Steinfeld, editors, *ASIACRYPT 2023, Part II*, volume 14439 of *LNCS*, pages 383–417. Springer, Heidelberg, December 2023.

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.

PW23. Jiaxin Pan and Benedikt Wagner. Chopsticks: Fork-free two-round multi-signatures from non-interactive assumptions. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 597–627. Springer, Heidelberg, April 2023.

PW24. Jiaxin Pan and Benedikt Wagner. Toothpicks: More efficient fork-free two-round multi-signatures. In *Advances in Cryptology – EUROCRYPT 2024: 43rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zurich, Switzerland, May 26–30, 2024, Proceedings, Part I*, page 460–489, Berlin, Heidelberg, 2024. Springer-Verlag.

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 Figures

$\underline{\mathsf{BS}_1.\mathsf{S}(\mathsf{par}, \mathsf{sk})}$		$\underline{\mathsf{BS}_1.\mathsf{U}(\mathsf{par} = (\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \mathsf{H}''), \mathsf{pk}, m)}$
		$\beta \leftarrow\$ \mathbb{Z}_p$
$Z \leftarrow h^{\mathsf{sk}}$	$\xleftarrow{\quad h \quad}$	$h' \leftarrow \mathsf{H}(m); h \leftarrow h' g^{\beta}$
$z_1, e, r_0, s \leftarrow\$ \mathbb{Z}_p$		
$R_g \leftarrow g^{r_0}; R_h \leftarrow h^{r_0}$		
$A \leftarrow g^{z_1} W^{-e}$		
$\delta \leftarrow \mathsf{H}''(h, g^{\mathsf{sk}}, Z, g^{s}, h^{s})$		**if** $\delta \neq \mathsf{H}''(h, \mathsf{pk}, Z, g^{s'} \mathsf{pk}^{-\delta}, h^{s'} Z^{-\delta})$:
$\pi \leftarrow (\delta, s' = \delta \cdot \mathsf{sk} + s)$		**return** $\perp$
	$\xrightarrow{(Z, R_g, R_h, A, \pi)}$	$\alpha_0, \alpha_1, \gamma_0, \gamma_1 \leftarrow\$ \mathbb{Z}_p$
		$Z' \leftarrow Z \mathsf{pk}^{-\beta}; R_g' \leftarrow R_g \mathsf{pk}^{-\gamma_0} g^{\alpha_0}$
		$R_h' \leftarrow R_h R_g^{-\beta} Z'^{-\gamma_0} h'^{\alpha_0}$
		$A' \leftarrow A W^{-\gamma_1} g^{\alpha_1}$
		$c' \leftarrow \mathsf{H}'(m, h', Z', R_g', R_h', A')$
$d \leftarrow c - e$	$\xleftarrow{\quad c \quad}$	$c \leftarrow c' - \gamma_0 - \gamma_1$
$z_0 \leftarrow r_0 + d \cdot \mathsf{sk}$	$\xrightarrow{(d, e, z_0, z_1)}$	**if** $c \neq e + d$ **or**
		$(R_g \mathsf{pk}^{d}, R_h Z^{d}) \neq (g^{z_0}, h^{z_0})$ **or**
		$A W^{e} \neq g^{z_1}$: **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. 13. Protocol diagram for the signing protocol of BS_1

$\underline{\mathsf{BS}_2.\mathsf{S}(\mathsf{par}, \mathsf{sk})}$		$\underline{\mathsf{BS}_2.\mathsf{U}(\mathsf{par} = (\mathbb{G}, p, g, W, \mathsf{H}, \mathsf{H}', \mathsf{H}''), \mathsf{pk}, m)}$
$z_1, e, r_0, s \leftarrow\$ \mathbb{Z}_p$		
$R_g \leftarrow g^{r_0}; A \leftarrow g^{z_1} W^{-e}$		$\alpha_0, \alpha_1, \gamma_0, \gamma_1, \beta \leftarrow\$ \mathbb{Z}_p$
	$\xrightarrow{(R_g, A)}$	$R'_g \leftarrow R_g \mathsf{pk}^{-\gamma_0} g^{\alpha_0}; A' \leftarrow AW^{-\gamma_1} g^{\alpha_1}$
$Z \leftarrow h^{\mathsf{sk}}; R_h \leftarrow h^{r_0}$	$\xleftarrow{h}$	$h' \leftarrow \mathsf{H}(m, R'_g, A'); h \leftarrow h' g^{\beta}$
$\delta \leftarrow \mathsf{H}''(h, g^{\mathsf{sk}}, Z, g^s, h^s)$		
$\pi \leftarrow (\delta, s' = s + \delta \cdot \mathsf{sk})$	$\xrightarrow{(Z, R_h, \pi)}$	**if** $\delta \neq \mathsf{H}''(h, \mathsf{pk}, Z, g^{s'} \mathsf{pk}^{-\delta}, h^{s'} Z^{-\delta})$:
		return $\perp$
		$Z' \leftarrow Z \mathsf{pk}^{-\beta}$
		$R'_h \leftarrow R_h R_g^{-\beta} Z'^{-\gamma_0} h'^{\alpha_0}$
		$c' \leftarrow \mathsf{H}'(m, h', Z', R'_g, R'_h, A')$
$d \leftarrow c - e$	$\xleftarrow{c}$	$c \leftarrow c' - \gamma_0 - \gamma_1$
$z_0 \leftarrow r_0 + d \cdot \mathsf{sk}$	$\xrightarrow{(d, e, z_0, z_1)}$	**if** $c \neq d + e$ **or**
		$(R_g \mathsf{pk}^d, R_h Z^d) \neq (g^{z_0}, h^{z_0})$ **or**
		$AW^e \neq g^{z_1}$: **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. 14. Protocol diagram for the signing protocol of BS_2

```
$\underline{\mathsf{BS}_3.\mathsf{S}(\mathsf{par}, \mathsf{sk})}$
$\textbf{for } i \in [K-1] : \mathsf{sk}_i \leftarrow\$ \mathbb{Z}_p$
$\mathsf{sk}_K \leftarrow \mathsf{sk} - \sum_{i=1}^{K-1} \mathsf{sk}_i$
$\textbf{for } i \in [K] : \mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i}$
$z_1, e \leftarrow\$ \mathbb{Z}_p; \vec{r}_0 \leftarrow\$ \mathbb{Z}_p^K$
$\mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}} \leftarrow\$ \mathbb{Z}_p^2$
$A \leftarrow g^{z_1} W^{-e}$

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

$\xleftarrow{\mathsf{umsg}_1}$

$\textbf{if } \mathsf{Check}(\mathsf{umsg}_1) = 0 : \textbf{abort}$
$\bar{S} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_i}$
$\vec{R} \leftarrow (g^{\vec{r}_{0,1}}, \ldots, g^{\vec{r}_{0,K}})$
$\bar{R} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{r}_{0,i}}$
$\mathsf{hcom}_{\bar{S}} \leftarrow \mathsf{Com}(\mathsf{ck}, \bar{S}; \mathsf{crnd}_{\bar{S}})$
$\mathsf{hcom}_{\bar{R}} \leftarrow \mathsf{Com}(\mathsf{ck}, \bar{R}; \mathsf{crnd}_{\bar{R}})$

$\xrightarrow{(\mathsf{pk}_i)_{i \in [K-1]}, \mathsf{hcom}_{\bar{S}}, \vec{R}, \mathsf{hcom}_{\bar{R}}, A}$

$\mathsf{pk}_K \leftarrow \mathsf{pk} \prod_{i=1}^{K-1} \mathsf{pk}_i^{-1}$
$\alpha_1, \gamma_0, \gamma_1 \leftarrow\$ \mathbb{Z}_p, \vec{\alpha}_0 \leftarrow\$ \mathbb{Z}_p^K; \delta_S, \delta_R \leftarrow\$ \mathbb{Z}_p^2$
$\widehat{\mathsf{hcom}} \leftarrow \mathsf{hcom}_{\bar{S}} \cdot \mathsf{Com}(\mathsf{ck}, \prod_{i=1}^K \mathsf{pk}_i^{-\beta_{i,\vec{J}_i}}; \delta_S)$
$((\mathsf{pk}'_i)_{i \in [K]}, \mathsf{hcom}_{\bar{S}'}, \vec{\tau}) \leftarrow\$$
        $\mathsf{ReRa}((\mathsf{pk}_i, h'_{i,\vec{J}_i})_{i \in [K]}, \widehat{\mathsf{hcom}})$
$\textbf{for } i \in [K] : \vec{R}'_i \leftarrow \vec{R}_i \mathsf{pk}_i'^{-\gamma_0} g^{\vec{\alpha}_{0,i}}$
$\widehat{\mathsf{hcom}}_{\bar{R}} \leftarrow \mathsf{Com}(\mathsf{ck}, \prod_{i=1}^K \vec{R}_i^{-\beta_{i,\vec{J}_i}} h'^{\vec{\alpha}_{0,i}}_{i,\vec{J}_i}; \delta_R)$
$\mathsf{hcom}'_{\bar{R}} \leftarrow \mathsf{hcom}_{\bar{R}} \cdot \mathsf{hcom}'^{-\gamma_0}_{\bar{S}} \cdot \widehat{\mathsf{hcom}}_{\bar{R}}$
$A' \leftarrow A W^{-\gamma_1} g^{\alpha_1}$
$c' \leftarrow \mathsf{H}'(m, (h'_{i,\vec{J}_i}, \mathsf{pk}'_i)_{i \in [K]}, \mathsf{hcom}'_{\bar{S}}, \vec{R}', \mathsf{hcom}'_{\bar{R}}, A')$
$c \leftarrow c' - \gamma_0 - \gamma_1$

$\xleftarrow{c}$

$d \leftarrow c - e$
$\textbf{for } i \in [K] : \vec{z}_{0,i} \leftarrow \vec{r}_{0,i} + d \cdot \mathsf{sk}_i$
$\mathsf{input} \leftarrow (g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S})$
$\pi \leftarrow \mathsf{Prove}^{\mathsf{H}_\Pi}(\mathsf{input}, (\mathsf{sk}_i)_{i \in [K]})$

$\xrightarrow{\bar{S}, \bar{R}, d, e, \vec{z}_0, z_1, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \pi}$

$\mathsf{smsg}_2 \leftarrow (\bar{S}, \bar{R}, d, e, \vec{z}_0, z_1, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \pi)$
$\textbf{return } \sigma \leftarrow \mathsf{BS}_3.\mathsf{U}_3(\mathsf{smsg}_2)$
```

Fig. 15. Protocol diagram for the signing protocol of BS_3. The algorithms $\mathsf{Check}, \mathsf{ReRa}$, and $\mathsf{Prove}^{\mathsf{H}_\Pi}$ are defined in Figure 12, while the third user algorithm $\mathsf{BS}_3.\mathsf{U}_3$ is as defined in Figure 11. For readability, we omitted the hash function descriptions from the public parameters par in this figure.

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

Oracle $\mathrm{S}_2(\mathrm{sid}, c)$ :
If $\mathrm{sid} \notin \mathcal{I}_1$ or $\mathrm{sid} \in \mathcal{I}_2$ then return $\perp$
$\ell \leftarrow \ell + 1$ ; $\mathcal{I}_2 \leftarrow \mathcal{I}_2 \cup \{\mathrm{sid}\}$
$d \leftarrow c - e$                     // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_1^{\mathcal{A}}$
For $i \in [K]$ : $\vec{z}_{0,i} \leftarrow \vec{r}_{0,i} + d \cdot \mathsf{sk}_i$
$e \leftarrow c - d$
$z_1 \leftarrow r_1 + e \cdot w$                               // $\mathbf{G}_2^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$\pi \leftarrow \mathsf{Prove}^{\mathsf{H}_\Pi}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}), (\mathsf{sk}_i)_{i \in [K]})$
                                                  // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_3^{\mathcal{A}}$
$\pi \leftarrow \mathsf{Sim}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}))$
If $\pi = \perp$ then **abort game**          // $\mathbf{G}_4^{\mathcal{A}}$
$\bar{R} \leftarrow \bar{S}^{-d} \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{z}_{0,i}}$
$\mathsf{hcom}_{\bar{R}} \leftarrow \delta_{\bar{R}} - d \cdot \mathsf{crnd}_{\bar{S}}$  // $\mathbf{G}_3^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
Return $(\bar{S}, \bar{R}, d, e, \vec{z}_0, z_1, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \pi)$
```

```
Oracle $\mathrm{S}_1(\mathrm{sid}, \mathsf{umsg}_1)$:
If $\mathrm{sid} \in \mathcal{I}_1$ then return $\perp$
$\mathcal{I}_1 \leftarrow \mathcal{I}_1 \cup \{\mathrm{sid}\}$
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{umsg}_1$
If $\mathsf{Check}(\mathsf{umsg}_1) = 0$ then return $\perp$
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 $i \in [K]$ : $\mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i}$
$\mathsf{crnd}_{\bar{S}} \leftarrow\$ \mathbb{Z}_p^2$
$\bar{S} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_i}$
$\mathsf{hcom}_{\bar{S}} \leftarrow \mathsf{Com}(\bar{S}; \mathsf{crnd}_{\bar{S}})$
$z_1, e \leftarrow\$ \mathbb{Z}_p, \vec{r}_0 \leftarrow\$ \mathbb{Z}_p^K$
For $i \in [K]$ : $\vec{R}_i \leftarrow g^{\vec{r}_{0,i}}$
$\bar{R} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{r}_{0,i}}$
$A \leftarrow g^{z_1} W^{-e}$  // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_1^{\mathcal{A}}$
$d, r_1 \leftarrow\$ \mathbb{Z}_p, \vec{z}_0 \leftarrow\$ \mathbb{Z}_p^K$
For $i \in [K]$ : $\vec{R}_i \leftarrow \mathsf{pk}_i^{-d} g^{\vec{z}_{0,i}}$
$\bar{R} \leftarrow \bar{S}^{-d} \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{z}_{0,i}}$  // $\mathbf{G}_2^{\mathcal{A}}$
$A \leftarrow g^{r_1}$              // $\mathbf{G}_2^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$\mathsf{crnd}_{\bar{R}} \leftarrow\$ \mathbb{Z}_p^2$  // $\mathbf{G}_0^{\mathcal{A}} - \mathbf{G}_2^{\mathcal{A}}$
$\mathsf{hcom}_{\bar{R}} \leftarrow \mathsf{Com}(\bar{R}; \mathsf{crnd}_{\bar{R}})$
$\delta_{\bar{R}} \leftarrow\$ \mathbb{Z}_p^2$                      // $\mathbf{G}_3^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$
$\mathsf{hcom}_{\bar{R}} \leftarrow \mathsf{hcom}_{\bar{S}}^{-d} \cdot \mathsf{Com}(\prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{z}_{0,i}}; \delta_{\bar{R}})$
Return $((\mathsf{pk}_i)_{i \in [K-1]}, \mathsf{hcom}_{\bar{S}}, \vec{R}, \mathsf{hcom}_{\bar{R}}, A)$

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

Fig. 16. The OMUF-2 = $\mathbf{G}_0^{\mathcal{A}}$ security game for BS_3 and the subsequent games $\mathbf{G}_1^{\mathcal{A}} - \mathbf{G}_4^{\mathcal{A}}$. The subsequent games $\mathbf{G}_5^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$ and $\mathbf{G}_9^{\mathcal{A}} - \mathbf{G}_{13}^{\mathcal{A}}$ can be found in Figures 17 and 18 respectively. We remark that $\mathsf{H}, \mathsf{H}', \mathsf{H}_\Pi, \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}$ are modeled as random oracles to which $\mathcal{A}$ has access. Each box type indicates the changes made in the game contained in the box. Also, to make things clearer, for each box, the comments indicate which game the changes in the boxes correspond to. The signer state is omitted and we assume that each variable initialized in S_1 of the same sid can be accessed in S_2.

```
Game $\mathbf{G}_4^{\mathcal{A}}, \mathbf{G}_5^{\mathcal{A}}, \mathbf{G}_6^{\mathcal{A}}, \mathbf{G}_7^{\mathcal{A}}, \mathbf{G}_8^{\mathcal{A}}$:
$(\mathbb{G}, p, g) \leftarrow\$ \mathsf{GGen}(1^\lambda)$
$\mathsf{sk} \leftarrow\$ \mathbb{Z}_p$ ; $\mathsf{pk} \leftarrow g^{\mathsf{sk}}$
$\mathsf{ck} \leftarrow\$ \mathsf{HECom.Gen}(\mathbb{G}, p, g)$
$w \leftarrow\$ \mathbb{Z}_p$ ; $W \leftarrow g^w$
$\mathsf{par} \leftarrow (\mathbb{G}, p, g, W, \mathsf{ck}, K, N)$
Map $\hat{\mathsf{r}}[\cdot] : \{0,1\}^\lambda \rightarrow \{0,1\}^{2\lambda}$ // $\mathbf{G}_5^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$
Map $\hat{b}[\cdot], b[\cdot] : \{0,1\}^* \rightarrow \{0,1\}$ // $\mathbf{G}_6^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
$\ell \leftarrow 0$ ; $\mathcal{I}_1, \mathcal{I}_2 \leftarrow \varnothing$
$\{(m_k^*, \sigma_k^*)\}_{k \in [\ell+1]} \leftarrow\$ \mathcal{A}^{\mathrm{S}_1, \mathrm{S}_2}(\mathsf{par}, \mathsf{pk})$
If $\exists\ k_1 \neq k_2, m_{k_1}^* = m_{k_2}^*$ then
    return 0
If $\exists\ k \in [\ell+1]$ such that
    $\mathsf{BS}_3.\mathsf{Ver}(\mathsf{pk}, m_k^*, \sigma_k^*) = 0$ then
        return 0
For $k \in [\ell+1]$:
    $((\mathsf{pk}_{i,k}^*, \varphi_{i,k}^*)_{i \in [K]}, \bar{S}_k^*, d_k^*, e_k^*,$
        $\vec{z}_{0,k}^*, z_{1,k}^*, \mathsf{crnd}_{\bar{S},k}^*, \mathsf{crnd}_{\bar{R},k}^*) \leftarrow \sigma_k^*$
    For $i \in [K] : \mu_{i,k}^* \leftarrow \mathsf{H}_\mu(m_k^*, \varphi_{i,k}^*)$
    If $\bar{S}_k^* \neq \prod_{i=1}^K \mathsf{H}(\mu_{i,k}^*)^{\log_g \mathsf{pk}_{i,k}^*}$
        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 $\exists \mathrm{str}' \neq \mathrm{str}, \mathsf{H}_\mu(\mathrm{str}) = \mathsf{H}_\mu(\mathrm{str}')$
    or $\mathsf{H}(\mathsf{H}_\mu(\mathrm{str})) \neq \perp$ // $\mathbf{G}_7^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$
    then abort game. // $\mathbf{G}_5^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$
If $\mathrm{str} = (m, \varphi)$ and // $\mathbf{G}_8^{\mathcal{A}}$
        $\nexists(m, \cdot), \mathsf{H}_\mu(m, \cdot) \neq \perp$
    then $\hat{b}[m] \leftarrow \begin{cases} 1 & \text{w.p. } 1/(\ell+1) \\ 0 & \text{otherwise} \end{cases}$
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.
If $\hat{\mathsf{r}}[\mathsf{com}] = \perp$ and // $\mathbf{G}_5^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$
    $\exists(\mathsf{com}', h'), (\mathsf{H}_{cc}(\mathsf{com}', h') \neq \perp$
      and $\exists(i,j), \mathsf{com}'_{i,j} = \mathsf{com})$
    then abort game.
Return $\mathsf{H}_{\mathsf{com}}(\mathrm{str}) \leftarrow \mathsf{com}$

Oracle $\mathrm{S}_1(\mathrm{sid}, \mathsf{umsg}_1)$:
If $\mathrm{sid} \in \mathcal{I}_1$ then return $\perp$
$\mathcal{I}_1 \leftarrow \mathcal{I}_1 \cup \{\mathrm{sid}\}$
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{umsg}_1$
If $\mathsf{Check}(\mathsf{umsg}_1) = 0$ then return $\perp$
$i^* \leftarrow\$ \mathsf{DetectCheat}(\mathsf{umsg}_1)$ // $\mathbf{G}_6^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$
If $i^* = \perp$ then abort game
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 $i \in [K] : \mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i}$
$\mathsf{crnd}_{\bar{S}} \leftarrow\$ \mathbb{Z}_p^2$
$\bar{S} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_i}$
$\mathsf{hcom}_{\bar{S}} \leftarrow \mathsf{Com}(\bar{S}; \mathsf{crnd}_{\bar{S}})$
$d, r_1 \leftarrow\$ \mathbb{Z}_p, \vec{z}_0 \leftarrow\$ \mathbb{Z}_p^K, \delta_{\bar{R}} \leftarrow\$ \mathbb{Z}_p^2$
For $i \in [K] : \vec{R}_i \leftarrow \mathsf{pk}_i^{-d} g^{\vec{z}_{0,i}}$
$A \leftarrow g^{r_1}$
$\mathsf{hcom}_{\bar{R}} \leftarrow \mathsf{hcom}_{\bar{S}}^{-d} \cdot \mathsf{Com}(\prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{z}_{0,i}}; \delta_{\bar{R}})$
Return $((\mathsf{pk}_i)_{i \in [K-1]}, \mathsf{hcom}_{\bar{S}}, \vec{R}, \mathsf{hcom}_{\bar{R}}, A)$

Algorithm $\mathsf{DetectCheat}(\mathsf{umsg}_1)$:
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{umsg}_1$
For $i \in [K]$:
    If $\hat{\mathsf{r}}[\mathsf{com}_{i,\vec{J}_i}] = (\mu, \varepsilon) \neq \perp$ and
        $h_{i,\vec{J}_i} = \mathsf{H}(\mu) g^{\mathsf{H}_\beta(\varepsilon)}$
        then return $i$
Return $\perp$ // $\mathbf{G}_6^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$

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 $\exists \mathsf{r}' = (\mu, \varepsilon), \mathsf{H}_{\mathsf{com}}(\mathsf{r}') = \mathsf{com}_{i,j}$ then
        $\hat{\mathsf{r}}[\mathsf{com}_{i,j}] = \mathsf{r}'$
    Else, $\hat{\mathsf{r}}[\mathsf{com}_{i,j}] = \perp$ // $\mathbf{G}_5^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$
$\mathsf{H}_{cc}(\mathsf{com}, h) \leftarrow\$ [N]^K$
Return $\mathsf{H}_{cc}(\mathsf{com}, h)$

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, $b[\mu] \leftarrow 0$ // $\mathbf{G}_8^{\mathcal{A}}$
Return $\mathsf{H}(\mu) \leftarrow\$ \mathbb{G}$
```

Fig. 17. The games $\mathbf{G}_4^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$ for the proof of Theorem 5.4 continued from Figure 16. We omitted the description of the oracles $\mathrm{S}_2, \mathsf{H}'$ and H_Π as they are unchanged in the games $\mathbf{G}_4^{\mathcal{A}} - \mathbf{G}_8^{\mathcal{A}}$. Each box type indicates the changes made in the game contained in the box. Also, to make things clearer, for each box, the comments indicate which game the changes in the boxes correspond to. Note: the subroutine DetectCheat is introduced to S_1 in game $\mathbf{G}_6^{\mathcal{A}}$.

```
Game $\mathbf{G}_8^{\mathcal{A}}$, $\boxed{\mathbf{G}_9^{\mathcal{A}}}$, $\boxed{\mathbf{G}_{10}^{\mathcal{A}}}$, $\boxed{\mathbf{G}_{11}^{\mathcal{A}}}$, $\boxed{\mathbf{G}_{12}^{\mathcal{A}}}$, $\boxed{\mathbf{G}_{13}^{\mathcal{A}}}$:
$(\mathbb{G}, p, g) \leftarrow\$ \mathsf{GGen}(1^\lambda)$
$\mathsf{sk} \leftarrow\$ \mathbb{Z}_p$ ; $\mathsf{pk} \leftarrow g^{\mathsf{sk}}$
$\mathsf{ck} \leftarrow\$ \mathsf{HECom.Gen}(\mathbb{G}, p, g)$ // $\mathbf{G}_8^{\mathcal{A}} - \mathbf{G}_{11}^{\mathcal{A}}$
$(\mathsf{ck}, \mathsf{td}) \leftarrow\$ \mathsf{HECom.TGen}((\mathbb{G}, p, g), \mathsf{pk})$ // $\mathbf{G}_{12}^{\mathcal{A}} - \mathbf{G}_{13}^{\mathcal{A}}$
$w \leftarrow\$ \mathbb{Z}_p$ ; $W \leftarrow g^w$
$\mathsf{par} \leftarrow (\mathbb{G}, p, g, W, \mathsf{ck}, K, N)$
Map $\hat{\mathsf{r}}[\cdot] : \{0,1\}^\lambda \rightarrow \{0,1\}^{2\lambda}$
Map $\hat{b}[\cdot], b[\cdot] : \{0,1\}^* \rightarrow \{0,1\}$
$\mathcal{L} \leftarrow \varnothing$ // $\mathbf{G}_9^{\mathcal{A}} - \mathbf{G}_{13}^{\mathcal{A}}$
$t[\cdot] : \{0,1\}^\lambda \rightarrow \mathbb{Z}_p$
$Y \leftarrow\$ \mathbb{G}$ // $\mathbf{G}_{10}^{\mathcal{A}} - \mathbf{G}_{13}^{\mathcal{A}}$
$\ell \leftarrow 0$ ; $\mathcal{I}_1, \mathcal{I}_2 \leftarrow \varnothing$
$\{(m_k^*, \sigma_k^*)\}_{k \in [\ell+1]} \leftarrow\$ \mathcal{A}^{\mathrm{S}_1, \mathrm{S}_2}(\mathsf{par}, \mathsf{pk})$
If $\exists\ k_1 \neq k_2, m_{k_1}^* = m_{k_2}^*$ then
    return 0
If $\exists\ k \in [\ell+1]$ such that
    $\mathsf{BS}_3.\mathsf{Ver}(\mathsf{pk}, m_k^*, \sigma_k^*) = 0$ then
        return 0
For $k \in [\ell+1]$:
    $((\mathsf{pk}_{i,k}^*, \varphi_{i,k}^*)_{i \in [K]}, \bar{S}_k^*, d_k^*, e_k^*,$
        $\vec{z}_{0,k}^*, z_{1,k}^*, \mathsf{crnd}_{\bar{S},k}^*, \mathsf{crnd}_{\bar{R},k}^*) \leftarrow \sigma_k^*$
    For $i \in [K] : \mu_{i,k}^* \leftarrow \mathsf{H}_\mu(m_k^*, \varphi_{i,k}^*)$
    If $\bar{S}_k^* \neq \prod_{i=1}^K \mathsf{H}(\mu_{i,k}^*)^{\log_g \mathsf{pk}_{i,k}^*}$
      then return 0
$m^\star \leftarrow m^*_{\arg\min\{k \in [\ell+1] : (\cdot, m_k^*) \notin \mathcal{L}\}}$
If $\hat{b}[m^\star] = 0$
   then return 0 // $\mathbf{G}_9^{\mathcal{A}} - \mathbf{G}_{13}^{\mathcal{A}}$
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, $b[\mu] \leftarrow 0$
$\mathsf{H}(\mu) \leftarrow\$ \mathbb{G}$ // $\mathbf{G}_8^{\mathcal{A}} - \mathbf{G}_9^{\mathcal{A}}$
$t[\mu] \leftarrow\$ \mathbb{Z}_p$
$\mathsf{H}(\mu) \leftarrow Y^{b[\mu]} g^{t[\mu]}$ // $\mathbf{G}_{10}^{\mathcal{A}} - \mathbf{G}_{13}^{\mathcal{A}}$
Return $\mathsf{H}(m)$
```

```
Oracle $\mathrm{S}_1(\mathrm{sid}, \mathsf{umsg}_1)$:
If $\mathrm{sid} \in \mathcal{I}_1$ then return $\perp$
$\mathcal{I}_1 \leftarrow \mathcal{I}_1 \cup \{\mathrm{sid}\}$
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i \in [K]}) \leftarrow \mathsf{umsg}_1$
If $\mathsf{Check}(\mathsf{umsg}_1) = 0$ then return $\perp$
$i^* \leftarrow\$ \mathsf{DetectCheat}(\mathsf{umsg}_1)$
If $i^* = \perp$ then **abort game**
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 $i \in [K] : \mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i}$ // $\mathbf{G}_8^{\mathcal{A}} - \mathbf{G}_{10}^{\mathcal{A}}$
For $i \in [K] \backslash \{i^*\}$: $\mathsf{sk}_i \leftarrow\$ \mathbb{Z}_p$; $\mathsf{pk}_i \leftarrow g^{\mathsf{sk}_i}$
$\mathsf{pk}_{i^*} \leftarrow \mathsf{pk} \prod_{i \neq i^*} \mathsf{pk}_i^{-1}$ // $\mathbf{G}_{11}^{\mathcal{A}} - \mathbf{G}_{13}^{\mathcal{A}}$
$\mathsf{crnd}_{\bar{S}} \leftarrow\$ \mathbb{Z}_p^2$ ; $\bar{S} \leftarrow \prod_{i=1}^K h_{i,\vec{J}_i}^{\mathsf{sk}_i}$
$\mathsf{hcom}_{\bar{S}} \leftarrow \mathsf{Com}(\bar{S}; \mathsf{crnd}_{\bar{S}})$ // $\mathbf{G}_8^{\mathcal{A}} - \mathbf{G}_{12}^{\mathcal{A}}$
$S' \leftarrow \prod_{i \neq i^*} h_{i,\vec{J}_i}^{\mathsf{sk}_i} h_{i^*,\vec{J}_{i^*}}^{-\mathsf{sk}_i}$ // $\mathbf{G}_{13}^{\mathcal{A}}$
$(\mathsf{hcom}_{\bar{S}}, \mathsf{st}_{\mathsf{com}}) \leftarrow \mathsf{HECom.TCom}(\mathsf{td}, \mathsf{ck}, S')$
$d, r_1 \leftarrow\$ \mathbb{Z}_p, \vec{z}_0 \leftarrow\$ \mathbb{Z}_p^K, \delta_{\bar{R}} \leftarrow\$ \mathbb{Z}_p^2$
For $i \in [K] : \vec{R}_i \leftarrow \mathsf{pk}_i^{-d} g^{\vec{z}_{0,i}}$
$A \leftarrow g^{r_1}$
$\mathsf{hcom}_{\bar{R}} \leftarrow \mathsf{hcom}_{\bar{S}}^{-d} \cdot \mathsf{Com}(\prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{z}_{0,i}}; \delta_{\bar{R}})$
Return $((\mathsf{pk}_i)_{i \in [K-1]}, \mathsf{hcom}_{\bar{S}}, \vec{R}, \mathsf{hcom}_{\bar{R}}, A)$

Oracle $\mathrm{S}_2(\mathrm{sid}, c)$ :
If $\mathrm{sid} \notin \mathcal{I}_1$ or $\mathrm{sid} \in \mathcal{I}_2$ then return $\perp$
$\ell \leftarrow \ell + 1$ ; $\mathcal{I}_2 \leftarrow \mathcal{I}_2 \cup \{\mathrm{sid}\}$
$(\mu, \varepsilon) \leftarrow \hat{\mathsf{r}}[\mathsf{com}_{i^*,\vec{J}_{i^*}}]$
If $b[\mu] = 1$ then **abort game**
If $\exists (m, \cdot), \mathsf{H}(m, \cdot) = \mu$
   then $\mathcal{L} \leftarrow \mathcal{L} \cup \{(\mu, m)\}$ // $\mathbf{G}_9^{\mathcal{A}} - \mathbf{G}_{13}^{\mathcal{A}}$
$e \leftarrow c - d$ ; $z_1 \leftarrow r_1 + e \cdot w$
$\pi \leftarrow \mathsf{Sim}((g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i \in [K]}, \bar{S}))$
If $\pi = \perp$ then **abort game**
$(\bar{S}, \mathsf{crnd}_{\bar{S}}) \leftarrow \mathsf{HECom.TOpen}(\mathsf{st}_{\mathsf{com}}, t[\mu] + \mathsf{H}_\beta(\varepsilon))$
                                                    // $\mathbf{G}_{13}^{\mathcal{A}}$
$\bar{R} \leftarrow \bar{S}^{-d} \prod_{i=1}^K h_{i,\vec{J}_i}^{\vec{z}_{0,i}}$
$\mathsf{hcom}_{\bar{R}} \leftarrow \delta_{\bar{R}} - d \cdot \mathsf{crnd}_{\bar{S}}$
Return $(\bar{S}, \bar{R}, d, e, \vec{z}_0, z_1, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \pi)$
```

Fig. 18. The games $\mathbf{G}_8^{\mathcal{A}} - \mathbf{G}_{13}^{\mathcal{A}}$ for the proof of Theorem 5.4 continued from Figure 17. We omitted the description of the oracles $\mathsf{H}', \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}$ and H_Π as they are unchanged in the games $\mathbf{G}_8^{\mathcal{A}} - \mathbf{G}_{13}^{\mathcal{A}}$. Each box type indicates the changes made in the game contained in the box. Also, to make things clearer, for each box, the comments indicate which game the changes in the boxes correspond to. Note: the subroutine $\mathsf{DetectCheat}$ is as described in $\mathbf{G}_6^{\mathcal{A}}$.