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, USA
{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 [27] 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 [27,28,56], e-voting [39], and credentials [10,24]. They have also come into use in a number of recent industry applications, such as privacy-preserving ad-click measurement [2], Apple's iCloud Private Relay [1], Google One's VPN Service [4], and various forms of anonymous tokens [3,44].

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 [61], EdDSA [19], and ECDSA [7] 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 full version of this work can be found at [26].

L. Reyzin and D. Stebila (Eds.): CRYPTO 2024, LNCS 14920, pp. 174–209, 2024.
https://doi.org/10.1007/978-3-031-68376-3_6

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* [46], even when a malicious user can run several concurrent signing interactions with the signer. Pointcheval and Stern [60] were the first to prove security of blind Okamoto-Schnorr signatures [55] under bounded concurrency, in the random oracle model (ROM) [16], assuming the hardness of the discrete logarithm (DL) problem. Their approach was later abstracted in [43]. Blind Schnorr signatures [29] have also only been proved secure under bounded concurrency [37,47], in this case additionally assuming the Algebraic Group Model (AGM) [36], along with the stronger one-more discrete logarithm (OMDL) assumption [13]. These results are also in some sense best possible, as recent ROS attacks [18] 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 [25,50,59] to increase the number of concurrent sessions where a scheme such as Okamoto-Schnorr remains secure. The current state of the art [25] 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 [5], but its proof (in the ROM, assuming the hardness of DL) later turned out to be incorrect, and was only recently re-established in the AGM [47]. Similarly, all other provably secure solutions [33,37,63] 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 [35] using generic NIZKs with online extractability in the ROM, such as those from the MPC-in-the-head paradigm [45]. The signer uses a hash-based signature scheme (which exists under the hardness of the DL problem) [54] 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 [38] 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 [48] 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 [20] to prove security (in the pairing setting) of Blind BLS [22], 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 [40] and the Chevallier-Mames [30,51] signature schemes, respectively, with a number of modifications based on witness indistinguishable OR-proofs [31] 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 contrast, our CDH-based scheme BS_3 achieves the standard notion [46] 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 [41]). 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 [3,44], 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 [32].

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

[1] We avoid the naming "one-more CDH" to avoid ambiguity, as an alternative interpretation is used e.g. in [8].

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 (full version)	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. 4)‡	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 [5,47]	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 [37]	perf. blind & OMSUF-2	3	$1\ \mathbb{G} + 1\ \mathbb{Z}_p$	$2\ \mathbb{G} + 4\ \mathbb{Z}_p$	-	DL + AGM + mROS
Snowblind [33]	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 $K = \lambda$.

assumption. For the first two schemes, our random oracle proofs only require the Fiat-Shamir heuristic [34] 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 [21], following e.g. [64].

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 [33,47,63] 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 [52,53]), 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 [12] (expanded in [11]) 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 [15] achieves IMP-CA but does not yield secure blind signatures.

PAPER OUTLINE. Section 2 introduces the basic preliminaries. We then discuss the scheme BS_1 achieving OMUF-1 based on the CT-CDH assumption in Sect. 3. Lastly, we discuss the scheme BS_3 achieving OMUF-2 based on the CDH assumption in Sect. 4. Note that the scheme BS_2, achieving one-more strong unforgeability from the CT-CDH assumption, is presented in the full version.

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 [40], which can also be thought of as a "pairing-free" variant of BLS signatures [23]. 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 [62]. 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 [20], 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 [31] 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 [60]. 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 [6], 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 [6], and a variant of the recent ROS attacks [18] 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 [30,51] 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 [9,57,58] 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 [42], 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).

UPGRADING TO OMUF-2. 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 Sect. 4.1) derived from the commitment scheme in [9]. 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

Notation. 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 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 [17] 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) [20] problems. To capture these, for any adversary $\mathcal{A}$, we define the advantage of $\mathcal{A}$ playing the games $\{\mathrm{DLOG}, \mathrm{CDH}, \text{CT-CDH}\}$ (these games are defined in Fig. 1) as

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

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

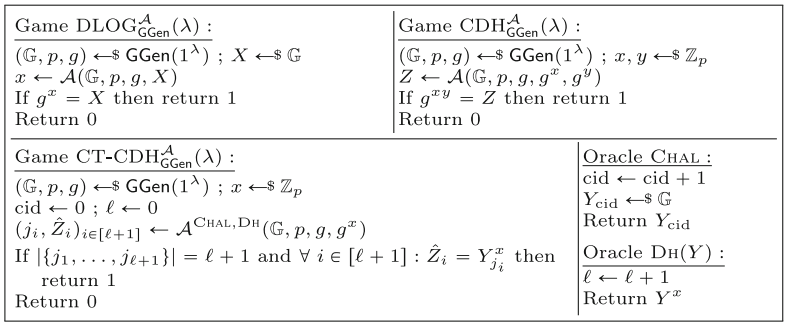

Fig. 1. The DLOG, CDH and CT-CDH games.

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

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:

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

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

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

We say that BS is (perfectly) *correct* if for every message $m \in \{0,1\}^*$, with probability one over the sampling of parameters and the key pair $(\mathsf{pk}, \mathsf{sk})$, the 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.

[2] We note that all of our schemes also admits an alternative definition without the setup algorithm (see some recent works with this definition [48,49]), by hashing a constant to generate the public parameters.

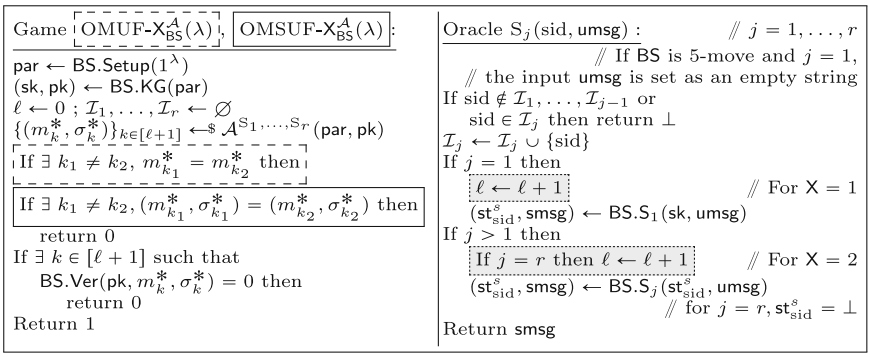

Fig. 2. The OMUF-X and OMSUF-X security games for a 4-move or 5-move blind signature scheme BS, where $r = 2$ if BS is 4-move and $r = 3$ if BS is 5-move. The input umsg of S_1 is set as an empty string if BS is 5-move. The highlighted boxes along with the commented X value indicating how ℓ is counted 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.

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.

ONE-MORE UNFORGEABILITY. 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 OMUF-X$^{\mathcal{A}}_{\mathsf{BS}}$ and OMSUF-X$^{\mathcal{A}}_{\mathsf{BS}}$ games for a blind signature scheme BS are defined in Fig. 2. The corresponding advantage of $\mathcal{A}$ is defined as $\mathsf{Adv}^{\text{omuf-X/omsuf-X}}_{\mathsf{BS}}(\mathcal{A}, \lambda) := \mathsf{Pr}[(\text{OMUF-X/OMSUF-X})^{\mathcal{A}}_{\mathsf{BS}}(\lambda) = 1]$.

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

RANDOM ORACLES. 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.

FORKING LEMMA. In our proof, we utilize the general forking lemma in the version introduced by Bellare and Neven [14] stated below:

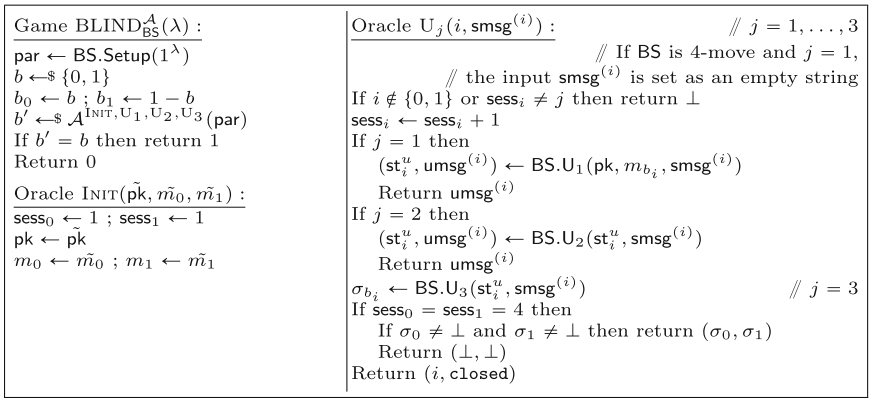

Fig. 3. The BLIND security game for a 4-move or 5-move blind signature scheme BS. The only difference between the game defined for 4-move schemes and the game defined for 5-move schemes is that if BS is a 4-move scheme, the input smsg of U_1 is set as an empty string.

Lemma 1 (General Forking Lemma [14]). *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*

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

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

Let $\mathsf{frk} = \mathsf{Pr}[b = 1 : x \leftarrow\$ \mathsf{IG}; b \leftarrow\$ F_{\mathcal{A}}(x)]$. 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 Fig. 4. The scheme can be viewed as a blind version of the signature scheme by Goh and

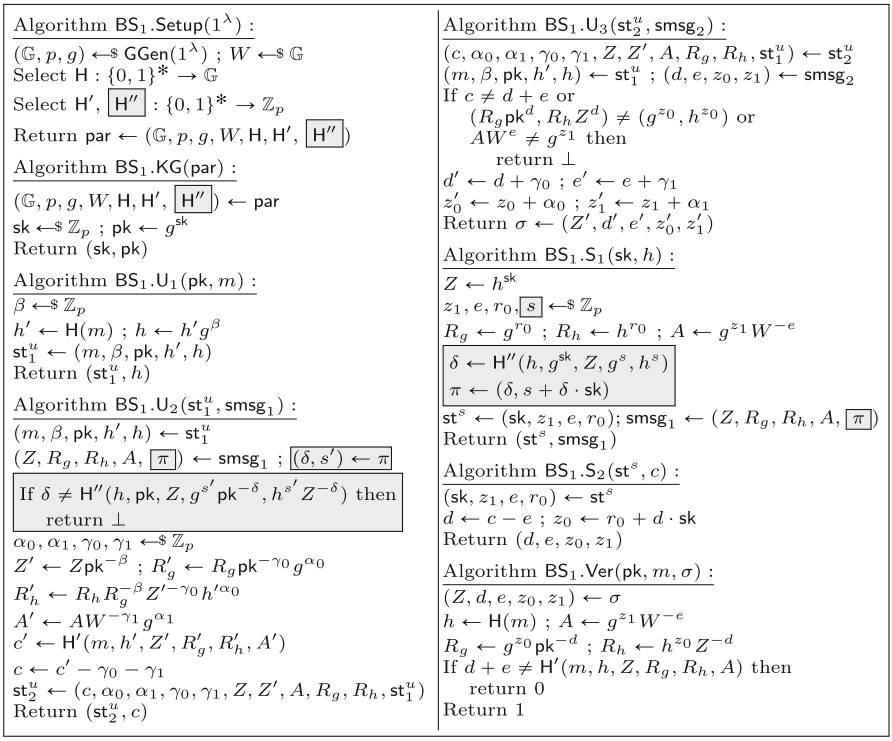

Fig. 4. The blind signature scheme $\mathsf{BS}_1 = \mathsf{BS}_1[\mathsf{GGen}]$. The public parameters par, as stated before, are implicit input to every algorithms except $\mathsf{BS}_1.\mathsf{KG}$. The highlighted boxes denote the NIZK proof used to show the equality of discrete logarithm of (pk,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

Jarecki [40], 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 see that the scheme satisfies correctness.

In the full version, we present a related five-move blind signature scheme BS_2 achieving one-more strong unforgeability (OMSUF-1) security from CT-CDH.

Blindness. The following theorem, proved in the full version, shows that BS_1 is *statistically* blind when H'' is modeled as a random oracle. This property relies on the NIZK proof highlighted in Fig. 4 to show equality of discrete logarithms of (pk, Z) to the base (g, h). In the full version, we also show that if we omit this NIZK proof, we still achieve *computational* blindness under the discrete logarithm assumption, without random oracles.

Theorem 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*

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

ONE-MORE UNFORGEABILITY. The following theorem establishes the OMUF-1 security of BS_1 in the random oracle model under the CT-CDH assumption. We refer to Sect. 1.1 for a proof sketch, whereas the full proof is in Sect. 3.1.

Theorem 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*

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

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 Proof of Theorem 2 (OMUF-1 of BS_1)

To prove one-more unforgeability of BS_1, we consider the following sequence of games.

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.,

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

Game $\mathbf{G}_1^{\mathcal{A}}$: This game is identical to $\mathbf{G}_0^{\mathcal{A}}$ except that 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 2, there exists an adversary $\mathcal{B}$ for the game DLOG, running in time $t_{\mathcal{B}} \approx 2t_{\mathcal{A}}$, such that

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

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 $\mathbf{G}_1^{\mathcal{A}}$.

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

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''),

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

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,

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

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 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 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 DH oracle instead, i.e., $Z \leftarrow \mathrm{DH}(h)$.
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 DH for ℓ times. Therefore, $\mathsf{Pr}[\mathbf{G}_4^{\mathcal{A}} = 1] \leqslant \mathsf{Adv}^{\text{ct-cdh}}_{\mathsf{GGen}}(\mathcal{B}', \lambda)$. Then, by combining all the advantage changes,

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

□

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

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

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

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

Now, define a 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 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' 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,

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

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 1),

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

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

4 Achieving OMUF-2 Security from CDH

In this section, we present a four-move blind signature scheme BS_3, described in Sect. 4.2, achieving the OMUF-2 security based on the CDH assumption. A key ingredient used in this construction is the homomorphic equivocal commitment HECom given in Sect. 4.1.

4.1 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 Fig. 5. 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 [9]. 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, proved in the full version, summarizes the properties of our commitment scheme.

Theorem 3. *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:*

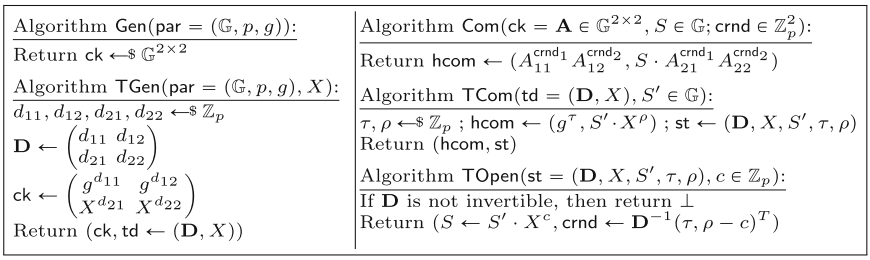

Fig. 5. Description of the special commitment scheme $\mathsf{HECom} = \mathsf{HECom}[\mathsf{GGen}]$ and its binding game. For the algorithms Com, TCom, and TOpen, $\mathsf{par} = (\mathbb{G}, p, g)$ is taken as an implicit input.

- ***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:*

$$D_0 := \left\{ (\mathsf{hcom}, S, \mathsf{crnd}) : \begin{array}{l} (\mathsf{hcom}, \mathsf{st}) \leftarrow\!\!\$\ \mathsf{TCom}(\mathsf{td}, S'); \\ (S, \mathsf{crnd}) \leftarrow\!\!\$\ \mathsf{TOpen}(\mathsf{st}, c) \end{array} \right\}\ ,$$

$$D_1 := \left\{ (\mathsf{hcom}, S, \mathsf{crnd}) : \mathsf{crnd} \leftarrow\!\!\$\ \mathbb{Z}_p^2\ ;\ \mathsf{hcom} \leftarrow \mathsf{Com}(\mathsf{ck}, S; \mathsf{crnd}) \right\}\ .$$

- ***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* $\mathsf{ck} \leftarrow\!\!\$\ \mathsf{Gen}(\mathsf{par})$*).*
- ***Computationally Binding.*** *For any adversary* $\mathcal{A}$ *for the game* Binding *(described in Fig. 5) 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*

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

4.2 Four-Move Blind Signatures from CDH

The scheme BS_3 is described across Figs. 6 and 7. Our starting point is Rai-Choo [42], 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 HEComm from Sect. 4.1, 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.

As mentioned in the prior section, the commitment key of HEComm 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 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 HEComm 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 Fig. 7, with access to the hash function $\mathsf{H}_\Pi : \{0,1\}^* \to \mathbb{Z}_p$ modeled as a random oracle in the security proofs. We require that Π satisfies completeness, soundness, and zero-knowledge in the random oracle model. The formal definitions and proofs are given in the full version.

$\underline{\text{Algorithm } \mathsf{BS}_3.\mathsf{Setup}(1^\lambda, K, N):}$
$(\mathbb{G}, p, g) \leftarrow\$ \mathsf{GGen}(1^\lambda)$
$W \leftarrow\$ \mathbb{G}$; $\mathsf{ck} \leftarrow\$ \mathsf{HECom.Gen}((\mathbb{G}, p, g))$
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 par

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

$\underline{\text{Algorithm } \mathsf{BS}_3.\mathsf{S}_1(\mathsf{sk}, \mathsf{umsg}_1):}$
$(\vec{J}, ((\mathsf{r}_{i,j})_{j \neq \vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i\in[K]}) \leftarrow \mathsf{umsg}_1$
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 $((\mathsf{pk}_i)_{i\in[K-1]}, \mathsf{hcom}_{\bar{S}}, \vec{R}, \mathsf{hcom}_{\bar{R}}, A)$

$\underline{\text{Algorithm } \mathsf{BS}_3.\mathsf{S}_2(c):}$
$d \leftarrow c - e$
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 $(d, e, \vec{z}_0, z_1, \bar{S}, \bar{R}, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \pi)$

$\underline{\text{Algorithm } \mathsf{BS}_3.\mathsf{U}_1(\mathsf{pk}, m):}$
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 $(\vec{J}, ((\mathsf{r}_{i,j})_{j\neq\vec{J}_i}, \mathsf{com}_{i,\vec{J}_i}, h_{i,\vec{J}_i})_{i\in[K]})$

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

$\underline{\text{Algorithm } \mathsf{BS}_3.\mathsf{U}_3(\mathsf{smsg}_2):}$
$(d, e, \vec{z}_0, z_1, \bar{S}, \bar{R}, \mathsf{crnd}_{\bar{S}}, \mathsf{crnd}_{\bar{R}}, \pi) \leftarrow \mathsf{smsg}_2$
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 σ

Fig. 6. 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, ReRa and the proof system $\Pi = (\Pi.\mathsf{Prove}^{\mathsf{H}_\Pi}, \Pi.\mathsf{Ver}^{\mathsf{H}_\Pi})$ are given separately in Fig. 7. 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 Sect. 4.1. 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 the full version.

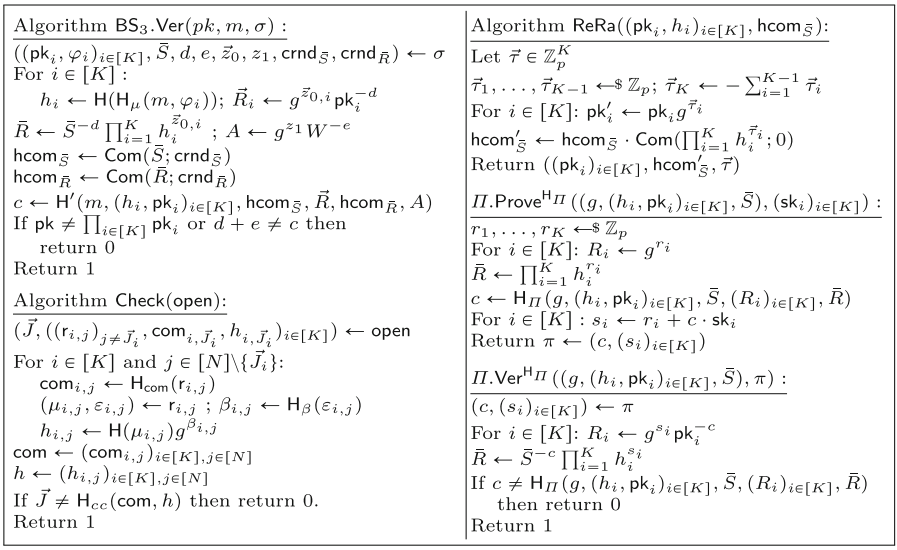

Fig. 7. The verification algorithm $\mathsf{BS}_3.\mathsf{Ver}$ and the algorithms Check and ReRa used in the signing protocol of BS_3 and the proof system Π. The public parameters par are implicit input to $\mathsf{BS}_3.\mathsf{Ver}$.

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 the full version.

Theorem 4. (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*

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

ONE-MORE UNFORGEABILITY. The following theorem, proved in Sect. 4.3, establishes the OMUF-2 security of BS_3 in the random oracle model under the CDH assumption.

Theorem 5. (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 $Q_{S_1} = Q_{S_1}(\lambda)$ 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* HEComm*,* $\mathcal{B}'$ *for the game* DLOG*, and* $\mathcal{B}''$ *for the game* CDH*, such that*

$$\begin{aligned}\mathsf{Adv}^{omuf\text{-}2}_{\mathsf{BS}_3}(\mathcal{A},\lambda) \leqslant &(\ell+1)\left(\sqrt{\widehat{Q}_{\mathsf{H}'}\left(\mathsf{Adv}^{\text{binding}}_{\mathsf{HEComm}}(\mathcal{B},\lambda)+\mathsf{Adv}^{\text{dlog}}_{\mathsf{GGen}}(\mathcal{B}',\lambda)\right)}+\frac{\widehat{Q}_{\mathsf{H}'}}{p}\right)\\ &+\frac{Q_{S_1}}{N^K}+\frac{Q^2_{\mathsf{H}_{\mathsf{com}}}+\widehat{Q}^2_{\mathsf{H}_\mu}+Q_{\mathsf{H}_{\mathsf{com}}}Q_{\mathsf{H}_{cc}}+\widehat{Q}_{\mathsf{H}}\widehat{Q}_{\mathsf{H}_\mu}}{2^\lambda}\\ &+\frac{\ell(\ell+Q_{\mathsf{H}_\Pi}+12)}{p}+4\ell\cdot\mathsf{Adv}^{\text{cdh}}_{\mathsf{GGen}}(\mathcal{B}'',\lambda)\ .\end{aligned}$$

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 below 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 HEComm 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 [42] 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

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

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 $\mathbf{G}_{13}$.

4.3 Proof of Theorem 5 (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.

Game $\mathbf{G}_0^{\mathcal{A}}$: 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 $\mathsf{H}, \mathsf{H}', \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}, \mathsf{H}_{\Pi}$, 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 $\mathsf{H}_\star \in \{\mathsf{H}, \mathsf{H}', \mathsf{H}_\mu, \mathsf{H}_{\mathsf{com}}, \mathsf{H}_{cc}, \mathsf{H}_{\Pi}\}$. 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 $\mathbf{G}_0^{\mathcal{A}}$ is exactly its advantage in OMUF-2 i.e.

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

Game $\mathbf{G}_1^{\mathcal{A}}$: 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 $\mathbf{G}_0^{\mathcal{A}}$ but not $\mathbf{G}_1^{\mathcal{A}}$, which gives $\mathsf{Pr}[\mathbf{G}_1^{\mathcal{A}} = 1] \geqslant \mathsf{Pr}[\mathbf{G}_0^{\mathcal{A}} = 1] - \mathsf{Pr}[\mathsf{Bad}]$. Then, by Lemma 3, 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

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

Game $\mathbf{G}_2^{\mathcal{A}}$: 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

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

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,

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

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 $\mathsf{Sim}(g, (h_{i,\vec{J}_i}, \mathsf{pk}_i)_{i\in[K]}, \bar{S})$ defined as follows:

- Sample $\delta \leftarrow\!\!\$\ \mathbb{Z}_p, \vec{s} \leftarrow\!\!\$\ \mathbb{Z}_p^K$.
- If $\mathsf{H}_{\Pi}(g, (h_i, \mathsf{pk}_i)_{i\in[K]}, \bar{S}, (g^{\vec{s}_i}\mathsf{pk}_i^{-\delta})_{i\in[K]}, \bar{S}^{-\delta}\prod_{i=1}^{K} h_i^{\vec{s}_i})$ is defined, abort.
- Otherwise, program H_{Π} of that input to c and return $\pi \leftarrow (c, \vec{s})$.

If the simulator abort, the game aborts.

The view of $\mathcal{A}$ is identical to its view in $\mathbf{G}_3^{\mathcal{A}}$ except when the game aborts (i.e., the simulator fails to program H_{Π}). Also, notice that $g^{\vec{s}_1}\mathsf{pk}^{-\delta}$ is uniformly random and independent of the view of $\mathcal{A}$ and previous programming attempts of H_{Π}. Thus, by the union bound over possible collision events, i.e., all pairs of queries to oracle S_2 and queries to both H_{Π} and S_2 (accounting for attempts to program H_{Π}),

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

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 $(Q^2_{\mathsf{H}_{\mathsf{com}}} + \widehat{Q}^2_{\mathsf{H}_\mu})/2^\lambda$. 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,

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

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,

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

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,

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

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,

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

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** 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 $\mathcal{A}$.

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

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

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

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

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

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,

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

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.

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

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,

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

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

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

  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 HEComm, 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,

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

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 $\mathsf{Pr}[\mathbf{G}_{13}^{\mathcal{A}} = 1] \leqslant \mathsf{Adv}_{\mathsf{GGen}}^{\mathrm{cdh}}(\mathcal{B}'', \lambda)$. Finally, combining all the advantage changes,

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

□

Lemma 3. *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*

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

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

$$\mathsf{Pr}[I = I' \neq \bot \wedge c_I \neq c'_I] = \mathsf{Pr}[\mathsf{Fork}^{\mathcal{A}_k} \text{ does not abort}] \leqslant \mathsf{Pr}[E_1] + \mathsf{Pr}[E_2] \ .$$

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 $\mathsf{Pr}[E_1] \leqslant \mathsf{Adv}^{\text{binding}}_{\mathsf{HECom}}(\mathcal{B}, \lambda)$.

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 $\mathsf{Pr}[E_2] \leqslant \mathsf{Adv}^{\mathrm{dlog}}_{\mathsf{GGen}}(\mathcal{B}', \lambda)$.

Finally, by the forking lemma (Lemma 1) and that $\mathcal{A}_k$ only outputs $I \neq \perp$ when Bad_k occurs,

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

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

1. icloud private relay overview. https://www.apple.com/privacy/docs/iCloud_Private_Relay_Overview_Dec2021.PDF
2. PCM: Click fraud prevention and attribution sent to advertiser. https://webkit.org/blog/11940/pcm-click-fraud-prevention-and-attribution-sent-to-advertiser/. Accessed 3 Sept 2021
3. Trust tokens. https://developer.chrome.com/docs/privacy-sandbox/trust-tokens/
4. VPN by google one, explained. https://one.google.com/about/vpn/howitworks
5. Abe, M.: A secure three-move blind signature scheme for polynomially many signatures. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 136–151. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44987-6_9
6. Abe, M., Okamoto, T.: Provably secure partially blind signatures. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 271–286. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_17

7. American National Standards Institute, Inc.: ANSI X9.62 public key cryptography for the financial services industry: the elliptic curve digital signature algorithm (ECDSA), 16 November 2005. https://standards.globalspec.com/std/1955141/ANSI%20X9.62
8. Bacho, R., Loss, J., Tessaro, S., Wagner, B., Zhu, C.: Twinkle threshold signatures from DDH with full adaptive security. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024, Part I. LNCS, pp. 429–459. Springer, Heidelberg (2024). https://doi.org/10.1007/978-3-031-58716-0_15
9. Bagherzandi, A., Cheon, J.H., Jarecki, S.: Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM CCS 2008, pp. 449–458. ACM Press, October 2008. https://doi.org/10.1145/1455770.1455827
10. Baldimtsi, F., Lysyanskaya, A.: Anonymous credentials light. In: Sadeghi, A.R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013, pp. 1087–1098. ACM Press, November 2013. https://doi.org/10.1145/2508859.2516687
11. Barreto, P.L., Reich, D.D., Simplicio Jr, M.A., Zanon, G.H.: Blind signatures from zero knowledge in the Kummer variety. In: Anais do XXIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais, pp. 139–152. SBC (2023). https://doi.org/10.5753/sbseg.2023.233503
12. Barreto, P.L., Zanon, G.H.M.: Blind signatures from zero-knowledge arguments. Cryptology ePrint Archive, Report 2023/067 (2023). https://eprint.iacr.org/2023/067
13. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme. J. Cryptol. **16**(3), 185–215 (2003). https://doi.org/10.1007/s00145-002-0120-1
14. Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM CCS 2006, pp. 390–399. ACM Press, October/November 2006. https://doi.org/10.1145/1180405.1180453
15. Bellare, M., Palacio, A.: GQ and Schnorr identification schemes: proofs of security against impersonation under active and concurrent attacks. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 162–177. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45708-9_11
16. Bellare, M., Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: Denning, D.E., Pyle, R., Ganesan, R., Sandhu, R.S., Ashby, V. (eds.) ACM CCS 1993, pp. 62–73. ACM Press, November 1993. https://doi.org/10.1145/168588.168596
17. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_25
18. Benhamouda, F., Lepoint, T., Loss, J., Orrù, M., Raykova, M.: On the (in)security of ROS. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 33–53. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77870-5_2
19. Bernstein, D.J., Duif, N., Lange, T., Schwabe, P., Yang, B.Y.: High-speed high-security signatures. J. Cryptogr. Eng. **2**(2), 77–89 (2012). https://doi.org/10.1007/s13389-012-0027-1
20. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In: Desmedt, Y.G. (ed.) PKC

2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36288-6_3
21. Boneh, D., Dagdelen, Ö., Fischlin, M., Lehmann, A., Schaffner, C., Zhandry, M.: Random oracles in a quantum world. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 41–69. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25385-0_3
22. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–532. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_30
23. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. J. Cryptol. **17**(4), 297–319 (2004). https://doi.org/10.1007/s00145-004-0314-9
24. Brands, S.: Untraceable off-line cash in wallet with observers. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 302–318. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48329-2_26
25. Chairattana-Apirom, R., Hanzlik, L., Loss, J., Lysyanskaya, A., Wagner, B.: PI-cut-choo and friends: Compact blind signatures via parallel instance cut-and-choose and more. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part III. LNCS, vol. 13509, pp. 3–31. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-15982-4_1
26. Chairattana-Apirom, R., Tessaro, S., Zhu, C.: Pairing-free blind signatures from CDH assumptions. Cryptology ePrint Archive, Report 2023/1780 (2023). https://eprint.iacr.org/2023/1780
27. Chaum, D.: Blind signatures for untraceable payments. In: Chaum, D., Rivest, R.L., Sherman, A.T. (eds.) CRYPTO 1982, pp. 199–203. Plenum Press, New York (1982). https://doi.org/10.1007/978-1-4757-0602-4_18
28. Chaum, D., Fiat, A., Naor, M.: Untraceable electronic cash. In: Goldwasser, S. (ed.) CRYPTO 1988. LNCS, vol. 403, pp. 319–327. Springer, New York (1990). https://doi.org/10.1007/0-387-34799-2_25
29. Chaum, D., Pedersen, T.P.: Wallet databases with observers. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 89–105. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-48071-4_7
30. Chevallier-Mames, B.: An efficient CDH-based signature scheme with a tight security reduction. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 511–526. Springer, Heidelberg (2005). https://doi.org/10.1007/11535218_31
31. Cramer, R., Damgård, I., Schoenmakers, B.: Proofs of partial knowledge and simplified design of witness hiding protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48658-5_19
32. Crites, E.C., Komlo, C., Maller, M.: Fully adaptive Schnorr threshold signatures. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part I. LNCS, vol. 14081, pp. 678–709. Springer, Heidelberg (2023). https://doi.org/10.1007/978-3-031-38557-5_22
33. Crites, E.C., Komlo, C., Maller, M., Tessaro, S., Zhu, C.: SnowBlind: a threshold blind signature in pairing-free groups. In: Handschuh, H., Lysyanskaya, A. (eds.) CRYPTO 2023, Part I. LNCS, vol. 14081, pp. 710–742. Springer, Heidelberg (2023). https://doi.org/10.1007/978-3-031-38557-5_23
34. Fiat, A., Shamir, A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987). https://doi.org/10.1007/3-540-47721-7_12

35. Fischlin, M.: Round-optimal composable blind signatures in the common reference string model. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 60–77. Springer, Heidelberg (2006). https://doi.org/10.1007/11818175_4
36. Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. LNCS, vol. 10992, pp. 33–62. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-96881-0_2
37. Fuchsbauer, G., Plouviez, A., Seurin, Y.: Blind Schnorr signatures and signed ElGamal encryption in the algebraic group model. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part II. LNCS, vol. 12106, pp. 63–95. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-45724-2_3
38. Fuchsbauer, G., Wolf, M.: Concurrently secure blind Schnorr signatures. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024, Part II. LNCS, vol. 14652, pp. 124–160. Springer, Heidelberg (2024). https://doi.org/10.1007/978-3-031-58723-8_5
39. Fujioka, A., Okamoto, T., Ohta, K.: A practical secret voting scheme for large scale elections. In: Seberry, J., Zheng, Y. (eds.) AUSCRYPT 1992. LNCS, vol. 718, pp. 244–251. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-57220-1_66
40. Goh, E.-J., Jarecki, S.: A signature scheme as secure as the Diffie-Hellman problem. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 401–415. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-39200-9_25
41. Hanzlik, L., Loss, J., Thyagarajan, S., Wagner, B.: Sweep-UC: swapping coins privately. In: 2024 IEEE Symposium on Security and Privacy (SP), Los Alamitos, CA, USA, p. 84. IEEE Computer Society, May 2024. https://doi.org/10.1109/SP54263.2024.00081
42. Hanzlik, L., Loss, J., Wagner, B.: Rai-choo! Evolving blind signatures to the next level. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part V. LNCS, vol. 14008, pp. 753–783. Springer, Heidelberg (2023). https://doi.org/10.1007/978-3-031-30589-4_26
43. Hauck, E., Kiltz, E., Loss, J.: A modular treatment of blind signatures from identification schemes. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part III. LNCS, vol. 11478, pp. 345–375. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_12
44. Hendrickson, S., Iyengar, J., Pauly, T., Valdez, S., Wood, C.A.: Private Access Tokens. Internet-Draft draft-private-access-tokens-01, Internet Engineering Task Force, October 2021. https://datatracker.ietf.org/doc/html/draft-private-access-tokens-01, work in Progress
45. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Zero-knowledge from secure multiparty computation. In: Johnson, D.S., Feige, U. (eds.) 39th ACM STOC, pp. 21–30. ACM Press, June 2007. https://doi.org/10.1145/1250790.1250794
46. Juels, A., Luby, M., Ostrovsky, R.: Security of blind digital signatures. In: Kaliski, B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 150–164. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0052233
47. Kastner, J., Loss, J., Xu, J.: On pairing-free blind signature schemes in the algebraic group model. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) PKC 2022, Part II. LNCS, vol. 13178, pp. 468–497. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-030-97131-1_16
48. Kastner, J., Nguyen, K., Reichle, M.: Pairing-free blind signatures from standard assumptions in the rom. In: Reyzin, L., Stebila D. (eds.) CRYPTO 2024. LNCS, vol. 14920, pp. 210–245. Springer, Cham (2024)
49. Katsumata, S., Reichle, M., Sakai, Y.: Practical round-optimal blind signatures in the ROM from standard assumptions. In: Guo, J., Steinfeld, R. (eds.) ASI-

ACRYPT 2023, Part II. LNCS, vol. 14439, pp. 383–417. Springer, Heidelberg (2023). https://doi.org/10.1007/978-981-99-8724-5_12
50. Katz, J., Loss, J., Rosenberg, M.: Boosting the security of blind signature schemes. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part IV. LNCS, vol. 13093, pp. 468–492. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-92068-5_16
51. Kiltz, E., Loss, J., Pan, J.: Tightly-secure signatures from five-move identification protocols. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part III. LNCS, vol. 10626, pp. 68–94. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70700-6_3
52. Maurer, U.M.: Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 271–281. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48658-5_26
53. May, A., Schneider, C.R.T.: 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). https://doi.org/10.46586/tches.v2023.i4.146-166
54. Naor, M., Yung, M.: Universal one-way hash functions and their cryptographic applications. In: 21st ACM STOC, pp. 33–43. ACM Press, May 1989. https://doi.org/10.1145/73007.73011
55. Okamoto, T.: Designated confirmer signatures and public-key encryption are equivalent. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 61–74. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-48658-5_8
56. Okamoto, T., Ohta, K.: Universal electronic cash. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 324–337. Springer, Heidelberg (1992). https://doi.org/10.1007/3-540-46766-1_27
57. Pan, J., Wagner, B.: Chopsticks: fork-free two-round multi-signatures from non-interactive assumptions. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part V. LNCS, vol. 14008, pp. 597–627. Springer, Heidelberg (2023). https://doi.org/10.1007/978-3-031-30589-4_21
58. Pan, J., Wagner, B.: Toothpicks: more efficient fork-free two-round multi-signatures. In: Joye, M., Leander, G. (eds.) EUROCRYPT 2024, Part I. LNCS, vol. 14651, pp. 460–489. Springer, Heidelberg (2024). https://doi.org/10.1007/978-3-031-58716-0_16
59. Pointcheval, D.: Strengthened security for blind signatures. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 391–405. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054141
60. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. J. Cryptol. **13**(3), 361–396 (2000). https://doi.org/10.1007/s001450010003
61. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, New York (1990). https://doi.org/10.1007/0-387-34805-0_22
62. Schnorr, C.P.: Efficient signature generation by smart cards. J. Cryptol. **4**(3), 161–174 (1991). https://doi.org/10.1007/BF00196725
63. Tessaro, S., Zhu, C.: Short pairing-free blind signatures with exponential security. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 782–811. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-07085-3_27
64. Unruh, D.: Post-quantum security of Fiat-Shamir. In: Takagi, T., Peyrin, T. (eds.) ASIACRYPT 2017, Part I. LNCS, vol. 10624, pp. 65–95. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70694-8_3