Blind Signatures from Proofs of Inequality

Michael Klooß and Michael Reichle

Department of Computer Science
ETH Zurich, Zurich, Switzerland
{michael.klooss, michael.reichle}@inf.ethz.ch

Abstract. Blind signatures are an important primitive for privacy-preserving technologies. To date, highly efficient *pairing-free* constructions rely on the random oracle model, and additionally, a strong assumption, such as interactive assumptions or the algebraic group model. In contrast, for signatures we know many efficient constructions that rely on the random oracle model and standard assumptions. In this work, we develop techniques to close this gap. Compared to the most efficient pairing-free AGM-based blind signature by Crites et. al. (Crypto 2023), our construction has a relative overhead of only a factor 3× and 2× in terms of communication and signature size, and it is provable in the random oracle model under the DDH assumption. With one additional move and $\mathbb{Z}_p$ element, we also achieve one-more strong unforgeability.
Our construction is inspired by the recent works by Chairattana-Apirom, Tessaro, and Zhu (Crypto 2024) and Klooß, Reichle, and Wagner (Asiacrypt 2024), and we develop a tailored technique to circumvent the sources of inefficiency in their constructions. Concretely, we achieve signature and communication size of 192 B and 608 B, respectively.

1 Introduction

Since their introduction in [Cha82], blind signatures have found application in many privacy-preserving applications such as anonymous credentials [Bra94; CL01], e-voting [Cha88; FOO92], direct anonymous attestation [BCC04]. More recently, new applications in blockchains [Bus+22; YL19] and privacy-preserving authentication tokens [Hen+22] are emerging. A digital signature scheme allows a signer to issue a publicly verifiable signature σ on a message M. We say a signature is unforgeable if even after seeing Q-many valid signature-message pairs (σ_j, M_j), it remains hard to forge a signature σ for a fresh message $\overline{M}$. We say that the signature scheme is *strongly* unforgeable if the adversary only has to come up with a fresh pair $(\overline{\sigma}, \overline{M})$. A blind signature [Cha82] allows a signer to issue a signature σ on some (user-specified) message M in an interactive protocol to a user. The term "blind" refers to the anonymity property we demand from blind signatures: A malicious signer cannot link the pair (σ, M), where σ was obtained from some protocol execution for M, to the specific execution. As for digital signatures, we also demand that it is hard to forge a signature. As the signer does not learn the signed message, this is often formalized as one-more unforgeability (OMUF) [JLO97; PS00]: After engaging in at most Q (concurrent

and finalized) signing sessions, a malicious user cannot produce $Q + 1$ valid signature-message pairs $(\overline{\sigma}, \overline{M})$ with pairwise distinct messages $\overline{M}_j$. As with digital signatures, this notion can be strengthened by demanding that the pairs are pairwise distinct, coined one-more *strong* unforgeability (OMSUF).

Blind signatures in pairing-free groups. In this work, we are interested in blind signatures in pairing-free groups of prime order. Unlike pairing groups, these have been standardized, the arithmetic is generally faster, and there are several off-the-shelf libraries providing highly optimized implementations. The first construction in such groups, coined Blind Schnorr [PS00], issues a signature based on a Fiat-Shamir compiled Σ-protocol for a linear language [Sch90]. Then, the linear structure of the Σ-protocol is leveraged to blind the signing session. Other early constructions [AO00] and [Abe01] follow this template to construct both elegant and efficient blind signatures. Yet, these constructions were only proven secure in the random oracle model [BR93] under limited concurrency, namely, assuming the adversary does not engage in more than polylog-many signing sessions. In fact, this is not a limitation of the security proof, but rather an inherent problem: Benhamouda et. al. [Ben+21; Ben+22] demonstrate that the proof is tight by providing an efficient attack on [AO00; PS00]. Until recently, it was unclear whether this template can even be instantiated without restricting the adversary's concurrency. Tessaro and Zhu [TZ22] and Kastner, Loss and Xu [KLX22] answer this affirmatively: [KLX22] proves that the scheme by Abe [Abe01] is secure and [TZ22] make an elegant modification to the Schnorr blind signature and prove unforgeability. Later, the approach of [TZ22] is further optimized in [Cri+23]. Unfortunately, both proofs rely on the *algebraic group model (AGM)*, that is, the proof assumes that the adversary behaves in an "algebraic manner".[1] For (non-blind) signatures, the gap between provable security with the AGM or without it is quite small: the most efficient construction, namely Schnorr signatures, can be proven purely in the ROM (with rewinding-induced loss) and very efficient tightly secure constructions from DDH are known.

Removing the AGM requirement. In their recent work, Chairattana-Apirom, Tessaro and Zhu [CATZ24] manage to remove the reliance on the AGM in an elegant manner. Their work can be seen as instantiating the pairing-based scheme Blind BLS [BLS04] in pairing-free groups as follows. [CATZ24] observes that only verification relies on the pairing in Blind BLS. To add verification in pairing-free groups, [CATZ24] includes a proof π that ensures that σ is valid (*i.e.*, fulfils the equations previously verified by a pairing). The proof is instantiated via a simple Fiat-Shamir compiled Σ-protocol. Then, to issue a signature blindly, [CATZ24] proceeds in two steps:

[1] Another recent scheme by Fuchsbauer and Wolf [FW24] issues regular Schnorr signatures blindly, but still under strong assumptions: Either one assumes the security of Schnorr signatures [Sch91], which is an interactive but falsifiable assumption. Or one relies on the hardness of DLog, but must generate SNARK proofs of random oracle evaluations.

(1) Blindly issue σ (following the template in [Bol03]).
(2) Blindly and interactively issue a proof π that σ verifies (following Blind Schnorr techniques [PS00; Sch90]).

The resulting protocol is simple and elegant, and from a practical viewpoint even competitive with AGM-based constructions [Abe01; Cri+23; FPS20; KLX22; TZ22] in terms of communication and signature size, cf. Table 1. Yet it is far from trivial to prove security. For the proof to go through, some minor modifications to π are made and the security model is weakened (*i.e.*, the adversary is asked to provide a valid signature for every *opened* sessions instead of only for every *finished* session). Based on the above template, [CATZ24] provide a scheme BS_1 that achieves OMUF in 4 moves and a scheme BS_2 that achieves OMSUF in 5 moves, albeit in the weakened security model.

While BS_1 and BS_2 are proven secure under a one-more assumptions ($\mathsf{CT\text{-}OMCDH}$), [CATZ24] also provides a construction BS_3 based on [HLW23]. The scheme BS_3 achieves the standard notion of OMUF under CDH, but also inherits the efficiency limitations from [HLW23].

Subsequently, Klooß, Reichle and Wagner [KRW24] apply the above template to the pairing-based construction in [KRS23] and prove security under the DDH assumption. In the process, the authors also equip the scheme with *partial blindness* (*i.e.*, the parties can agree upon a common message I that is signed in addition to message M). While the signature size is compact and competitive with AGM-based constructions [Abe01; CATZ24; Cri+23], the communication remains linear (compared to constant in the AGM). This gap in efficiency in both [CATZ24] and [KRW24] comes from cut-and-choose techniques. In [KRW24], this is due to a *straightline extractable (SLE)* proof system, but unfortunately, this reliance seems inherent for their proof technique, since known techniques for SLE of $\mathbb{Z}_p$ witnesses are limited to bit (or digit) decomposition or cut-and-choose techniques [Fis06; Ks22; Pas03] or not efficiently applicable [Kat21]. We therefore ask the following question.

Is it possible to close the efficiency gap between AGM-based and AGM-free constructions in pairing-free groups proven under non-interactive assumptions?

1.1 Our Contributions

In this paper, we present a novel technique to construct pairing-free blind signatures. While it is inspired by the template in [CATZ24; KRW24], our construction is not based on translating a pairing-based blind signature. Rather, we develop techniques tailored to the pairing-free setting and obtain the following:

- $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$: We obtain a 4-move blind signature $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ with partial blindness and one-more unforgeability under the DDH assumption in the ROM. Notably, $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ has communication and signature size of $10\mathbb{G} + 9\mathbb{Z}_p$ and $1\mathbb{G} + 5\mathbb{Z}_p$, respectively.[2]

[2]This is an asymptotic improvement in communication over [CATZ24; KRW24]. This is also achieved by the concurrent work [Bra+24], see Table 1 and Section 1.3.

Assuming $\lambda = 128$ and that group and field elements are represented using 256 bit, we obtain 192 B signatures and 608 B communication. In comparison, [KRW24] achieves 224 B signatures and 2.5 KB communication. Our construction is a factor 4× and 1.16× improvement over [KRW24] in communication and signature size, respectively,

– Based on $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$, we obtain a 5-move blind signature $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ with partial blindness and one-more *strong* unforgeability under the DDH assumption in the ROM. Communication is identical to $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ and the signature contains only one additional $\mathbb{Z}_p$ element.

The blindness of our schemes is computational, but with minor overhead in signature size and communication, we can upgrade both $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ and $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ to *statistical* blindness. We believe that this is a significant step towards answering our research question affirmatively. Indeed, as visible in Table 1, we improve both communication, signature size, and security (to OMSUF) compared to [CATZ24; KRW24]. For completeness, we provide a complementary overview over other related works in Appendix A.

Scheme	Assumption	Unforgeability		Moves	Communication	Signature
		OMUF	*OMSUF*			
Cl-Schnorr [FPS20]	OMDL, mROS	✓	✓	3	$2\mathbb{G} + 3\mathbb{Z}_p$	$1\mathbb{G} + 1\mathbb{Z}_p$
Abe [Abe01; KLX22]	DLOG	✓	✓	3	$\lambda + 3\mathbb{G} + 6\mathbb{Z}_p$	$2\mathbb{G} + 6\mathbb{Z}_p$
TZ [TZ22]	DLOG	✓	✓	3	$2\mathbb{G} + 4\mathbb{Z}_p$	$4\mathbb{Z}_p$
Snowblind [Cri+23]	DLOG	✓	✓	3	$2\mathbb{G} + 4\mathbb{Z}_p$	$1\mathbb{G} + 2\mathbb{Z}_p$
BS_1 [CATZ24]	CT-OMCDH	(✓)	✗	4	$5\mathbb{G} + 5\mathbb{Z}_p$	$1\mathbb{G} + 4\mathbb{Z}_p$
BS_2 [CATZ24]	CT-OMCDH	(✓)	(✓)	5	$5\mathbb{G} + 5\mathbb{Z}_p$	$1\mathbb{G} + 4\mathbb{Z}_p$
BS_3 [CATZ24]	CDH	✓	✗	4	$\Theta(\lambda)(\lambda + \mathbb{G} + \mathbb{Z}_p)$	$\Theta(\lambda)(\lambda + \mathbb{G} + \mathbb{Z}_p)$
BS [KRW24]	DDH	✓	✗	4	$\Omega(\lambda)(\lambda + \mathbb{G} + \mathbb{Z}_p)$	$2\mathbb{G} + 5\mathbb{Z}_p$
BS [Bra+24]	DDH	✓	✗	4	$37\mathbb{G} + 40\mathbb{Z}_p$	$10\mathbb{G} + 29\mathbb{Z}_p$
$\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ (see Section 4)	DDH	✓	✗	4	$10\mathbb{G} + 9\mathbb{Z}_p$	$1\mathbb{G} + 5\mathbb{Z}_p$
$\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ (see Section 5)	DDH	✓	✓	5	$10\mathbb{G} + 9\mathbb{Z}_p$	$1\mathbb{G} + 6\mathbb{Z}_p$

Table 1: Comparison of blind signature schemes in pairing-free prime order groups with concurrent security. All constructions rely on the random oracle model. The schemes above the line (highlighted in red) additionally require the algebraic group model. We compare the assumptions and security, and the communication complexity and signature size in terms of number of group elements and number of field elements. The schemes BS_1 and BS_2 [CATZ24] only satisfy a weaker variant of one-more (strong) unforgeability, denoted by (✓).

1.2 Technical Overview

We follow the strategies and approach of prior Schnorr-based works, that is, interactively issuing the Fiat–Shamir proof. In particular, we build on the techniques in [CATZ24; KRW24]. In these previous approaches, a pairing-based verification

was made pairing-free by replacing the pairing check with a NIZK proof. The underlying pairing-free schemes were

- BLS signatures [BLS04] in [CATZ24], which either required a type of "one-more CDH" assumption (CT-CDH) or the cut-and-choose technique from Rai-Choo [HLW23]; or
- signatures derived from the Boneh–Boyen IBE [BB04] in [KRW24], which additionally required a straightline extractable (SLE) NIZK for a scalar, i.e., $\mathbb{Z}_p$ elements.

While it is unclear how to remove the interactive assumptions or cut-and-choose from [CATZ24], the approach of [KRW24] seems more promising towards achieving better concrete efficiency. In particular, the signature size is already quite compact and it only remains to improve communication size to obtain a competitive scheme. However, we are faced with a crucial obstruction: as mentioned, [KRW24] relies on an SLE NIZK for proving knowledge of scalars (i.e., DLogs) from prime-order group-based assumptions, and this represents a major bottle neck in terms of efficiency. Indeed, improving this component would require a major (and unexpected) improvement in SLE techniques in itself. Our natural solution to this problem is to switch from hashing the messages into $\mathbb{Z}_p$ to hashing into the group $\mathbb{G}$. Straightline extraction of group elements is efficient through a well-known folklore technique (sometimes dubbed encryption-to-the-sky), which is, *e.g.*, used in [KLN23; KRS23] to instantiate SLE NIZKs.

However, while we now have efficient choices for SLE NIZK, we are not aware of suitable and concretely efficient pairing-based candidate schemes that implement the translation used in [CATZ24; KRW24].[3] Here we must deviate from the prior blueprints. We realize that the approach of [KRW24] is based on an *all-but-one trapdoor* (adapted from [PK22]). The *all-but-one* trapdoor allows to generate signatures for all messages M except M^* efficiently. A valid signature for M^* would solve a hard problem (in the case of [KRW24] it solves CDH and relies on the all-but-one trapdoor for selective security of the Boneh–Boyen IBE [BB04; BB08]). To prove security, it is additionally necessary to first extract the to-be-signed hashed message M using the SLE NIZK, since knowledge of M is required to use the all-but-one trapdoor.

A tailored singing trapdoor: Our approach is based on a surprisingly simple all-but-one trapdoor construction which does not have an obvious pairing-based equivalent. It is easiest to explain by sketching our basic signature scheme:

- The public parameters contain a public key pk and ciphertext $\mathbf{C}$.
- The verification key vk is a DDH tuple and its witness is the secret key.
- A signature for (hashed) message $M = \mathsf{H}_M(\mu)$ is a Fiat–Shamir-compiled OR-proof that:
  Either: vk is a DDH tuple (and one knows the secret key).

[3] The concurrent work [Bra+24] observed that the tightly secure structure-preserving pairing-based signature scheme [Abe+17; Abe+23] does, in fact, provide a candidate, though lacking practical efficiency. See Section 1.3 for detailed discussions.

Or: The ciphertext $\mathbf{C}$ *does not* encrypt M.

The all-but-one structure of the OR-branch trapdoor is obvious: After encrypting a challenge message M^* in $\mathbf{C}$, we can sign every message $M \neq M^*$ by using the OR-branch. Importantly, the OR-branch can be realized from ElGamal encryption, and thus, we can hash into group elements. Now, we can rely on the very efficient SLE NIZK in this case (and indeed, our NIZK just adds $3\mathbb{G} + 3\mathbb{Z}_p$ in communication).

Blind signature: To make the above idea a blind signature, we can follow the footsteps of prior works [CATZ24; KRW24]. We implement the signature as a Fiat–Shamir transformation of Σ-protocols for the two OR-statements, and use the OR-composition of Σ-protocols [CDS94]. Then we make issuance of the signature interactive by letting the user send the challenge which the signer would derive from the random oracle. Moreover, the user will blind the signing transcript by rerandomization techniques. As in [KRW24], we manage to circumvent rewinding in our security analysis which gives tighter bounds compared to [CATZ24].

Summary and Extensions: With the above techniques, we obtain a concretely efficient blind signature scheme with small communication and signature size, whose security can be based on DDH. In the main body, we generate the public parameters $(\mathsf{pk}, \mathbf{C})$ from as a random oracle output $(\mathsf{pk}_I, \mathbf{C}_I) = \mathsf{H}_{\mathsf{par}}(I)$ and thus obtain a partially blind signature with common input I. Moreover, we use the approach of [CATZ24] for achieving one-more *strong* unforgeability (OMSUF) to construct our signature scheme $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ (at the cost of increasing the number of rounds to 5). Compared to [CATZ24], we manage to avoid rewinding, the weaker OMSUF model, and the interactive assumption in our OMSUF proof. We refer to Section 5 for details. Finally, we also show how to achieve *statistical* blindness by replacing the public parameter pk with a lossy/dual-mode encryption scheme which is in lossy/hiding mode for random keys.[4]

1.3 Concurrent work

The concurrent work [Bra+24] constructs a tightly secure signature scheme with similar characteristics to ours: it is in the pairing-free setting and has constant communication. Moreover, their blind signature supports efficient zero-knowledge proofs over the to-be-signed message (and thus predicate blindness [FW24]), and it can be shown secure in the *non-programmable* ROM, whereas our approach cannot support these properties as it hashes the message through a programmable RO (which we need for the all-but-one trapdoor). To achieve these features, Brandt et al. [Bra+24] rely on fundamentally different techniques: while [Bra+24] builds on [CATZ24; KRW24], as we do, they translate the tightly secure structure preserving pairing-based signature scheme [Abe+23] to the pairing-free setting.[5]

[4] To achieve this, we use dual-mode commitments from [GS12], similar to the pairing-based blind signature scheme [Bla+13].

[5] The scheme of [Abe+23] gives a semi-generic recipe from suitable proof systems and instantiates it with Groth–Sahai proofs [GS12]. The recipe is amenable to Σ-protocols.

As a consequence, they can also use encryption-to-the-sky and get constant-sized proofs. To be able to transfer the tight security proof from the underlying signature to their scheme, their protocol is necessarily more complex, and, in particular, must rely on different techniques to achieve blindness. While their approach for blindness significantly increases communication (cf. Table 1) and is not practically competitive, the overhead is a small constant over the non-blind protocol. Thus, they also achieve constant communication (in $\mathbb{G}$ and $\mathbb{Z}_p$ elements) which is an asymptotic improvement over prior works (cf. Table 1). In terms of concrete efficiency for 256-bit groups, the communication and signature size of [Bra+24] are roughly 2.5 KB and 1.3 KB, respectively. This is roughly 1× (resp. 6×) the size of [KRW24], and roughly 4× (resp. 6.5×) the size of our work. Moreover, [Bra+24] does not consider OMSUF security nor statistical blindness. Indeed, it is unclear how [Bra+24] could achieve these properties due to their different techniques for blinding.

2 Preliminaries

General Conventions

Throughout, λ denotes the security parameter and $\mathbb{G}$ denotes a group of prime order p with generator G (which is implicitly provided through public parameters depending on λ). We use additive group notation and denote group elements $H \in \mathbb{G}$ by capital letters and $\mathbb{Z}_p$ elements by lowercase letters. We write $\mathbb{Z}_p^\times \coloneqq \mathbb{Z}_p \setminus \{0\}$ and $\mathbb{G}^\times \coloneqq \mathbb{G} \setminus \{0\}$. We write $y \leftarrow A(x)$ to run (probabilistic) algorithm A with fresh randomness on input x; we write $A \rightleftarrows B$ for interactive protocols; and we write $y \leftarrow S$ to sample y uniformly from a set S. Finally, write $y \coloneqq x$ for algorithmic assignment and $\mathsf{H}(x) \coloneqq y$ to program a random oracle at query x to output y. Throughout the paper, we assume that algorithms check their inputs are in the right space (e.g. encode a group element), and return $\perp$ otherwise.

ElGamal Encryption The ElGamal encryption scheme [ElG85] with message space $\mathbb{G}$ is defined as follows.

- $\mathsf{KeyGen}(1^\lambda)$: Samples $x \leftarrow \mathbb{Z}_p$ and outputs $(\mathsf{pk}, \mathsf{sk})$, where $H \coloneqq xG$, $\mathsf{pk} \coloneqq (G, H)$ and $\mathsf{sk} \coloneqq x$.
- $\mathsf{Enc}(\mathsf{pk}, M)$: Samples $t \leftarrow \mathbb{Z}_p$ and outputs $\mathbf{C} \coloneqq (0, M) + t \cdot \mathsf{pk}$.
- $\mathsf{Dec}(\mathsf{sk}, \mathbf{C})$: Parses $\mathsf{sk} = x$ and outputs $M \coloneqq C_1 - xC_0 = \mathbf{C} \cdot (-x, 1)$.

The IND-CPA security of ElGamal reduces tightly to the DDH assumption.

Definition 2.1 (QDDH Assumption). *The Q-fold decisional Diffie–Hellman assumption holds in group $\mathbb{G}$ with generator G if for any PPT adversary $\mathcal{A}$, the advantage*

$$\begin{aligned}\mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{A}}(\lambda, Q) \coloneqq \big|&\Pr[\mathcal{A}(G, aG, (b_iG, (ab_i)G)_{i\in[Q]}) = 1 \mid a \leftarrow \mathbb{Z}_p, \boldsymbol{b} \leftarrow \mathbb{Z}_p^Q] \\ &- \Pr[\mathcal{A}(G, aG, (b_iG, c_iG)_{i\in[Q]}) = 1 \mid a \leftarrow \mathbb{Z}_p, \boldsymbol{b}, \boldsymbol{c} \leftarrow \mathbb{Z}_p^Q]\big|\end{aligned}$$

is negligible.

The DDH assumption (1-DDH) is tightly equivalent to QDDH. Concretely, for any $\mathcal{A}$ there is an $\mathcal{B}$ with roughly the same running time such that $\mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{A}}(\lambda, Q) \leq \mathsf{AdvDDH}^{\mathbb{G}}_{\mathcal{B}}(\lambda) + 1/(p-1)$ (see, e.g., [Esc+13]).

2.1 Relations and Σ-Protocols

Definitions of NP relations and Σ protocols for linear languages are standard. We re-use the definition and notation from [KRW24] often verbatim, but introduce the notion of randomizable transcripts (Definition 2.6) as a convenient abstraction for the blindness proof.

Next, we define Σ-protocols for NP-relations. We start by defining NP-relations.

Definition 2.2 (NP-Relation and Language). *Let $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ be a binary relation. We say that $\mathcal{R}$ is an* NP-relation*, if $\mathcal{R}$ is efficiently decidable and there is a polynomial p such that for every $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$, we have $|\mathbb{w}| \leq |\mathbb{x}|$. We denote by $\mathcal{L}_{\mathcal{R}} = \{\mathbb{x} \in \{0,1\}^* \mid \exists w \text{ s.t. } (\mathbb{x}, \mathbb{w}) \in \mathcal{R}\}$ the language induced by $\mathcal{R}$.*

Let $\mathcal{R}$ be an NP-relation with statements $\mathbb{x}$ and witnesses $\mathbb{w}$. A Σ-protocol for an NP-relation $\mathcal{R}$ with efficiently sampleable challenge space $\mathcal{C}$ is a tuple of PPT algorithms $\boldsymbol{\Sigma} = (\mathsf{Init}, \mathsf{Resp}, \mathsf{Verify})$ such that

- $\mathsf{Init}(\mathbb{x}, \mathbb{w})$: given a statement-witness pair $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$, outputs a first flow message A (a.k.a. commitment) and a state st, where we assume st includes $(\mathbb{x}, \mathbb{w})$;
- $\mathsf{Resp}(\mathsf{st}, \gamma)$: given a state st and a challenge $\gamma \in \mathcal{C}$, outputs a third flow message (*i.e.*, response) z,
- $\mathsf{Verify}(\mathbb{x}, A, \gamma, z)$: given a (purported) statement $\mathbb{x}$, a first flow message A, challenge $\gamma \in \mathcal{C}$, and a response z, outputs a bit $b \in \{0,1\}$.

We call the tuple (A, γ, z) the *transcript*. It is *valid for* $\mathbb{x}$ if $\mathsf{Verify}(\mathbb{x}, A, \gamma, z)$ outputs 1 . When the context is clear, we simply say it is valid and omit $\mathbb{x}$. Next, we define the standard notions of correctness, special honest-verifier zero-knowledge, and (2-)special soundness.

Definition 2.3 (Correctness). *Let $\mathcal{R}$ be an NP-relation and $\boldsymbol{\Sigma} = (\mathsf{Init}, \mathsf{Resp}, \mathsf{Verify})$ be a Σ-protocol for $\mathcal{R}$ with challenge space $\mathcal{C}$. We say $\boldsymbol{\Sigma}$ is* correct*, if for all $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$,*

$$\Pr[\mathsf{Verify}(\mathbb{x}, A, \gamma, z) = 1 \mid (A, \mathsf{st}) \leftarrow \mathsf{Init}(\mathbb{x}, \mathbb{w});\ \gamma \leftarrow \mathcal{C};\ z \leftarrow \mathsf{Resp}(\mathsf{st}, \gamma)] = 1$$

Definition 2.4 (Special Soundness). *Let $\mathcal{R}$ be an NP-relation and $\boldsymbol{\Sigma} = (\mathsf{Init}, \mathsf{Resp}, \mathsf{Verify})$ be an Σ-protocol for $\mathcal{R}$. We call $\boldsymbol{\Sigma}$* (2-)special sound*, if there exists a* deterministic *polynomial-time extractor* Ext *such that given statement $\mathbb{x}$ and two valid transcripts $\{(A, \gamma_b, z_b)\}_{b \in \{0,1\}}$ with $\gamma_0 \neq \gamma_1$, outputs a witness $\mathbb{w}$ such that $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$.*

Definition 2.5 ((Perfect) Special HVZK). *Let $\mathcal{R}$ be an NP-relation and $\boldsymbol{\Sigma} = (\mathsf{Init}, \mathsf{Resp}, \mathsf{Verify})$ be an Σ-protocol for $\mathcal{R}$ We say that $\boldsymbol{\Sigma}$ is* (perfect) special honest-verifier zero-knowledge *(SHVZK), if there exists a PPT zero-knowledge simulator Sim such that for any (potentially unbounded) adversary $\mathcal{A}$, it holds that for any $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$ and $\gamma \in \mathcal{C}$ that $\mathsf{D}_{\mathsf{Real}} = \mathsf{D}_{\mathsf{SVHZK}}$ for*

$$D_{\mathsf{Real}} \coloneqq \{(\mathbb{x}, \mathbb{w}, (A, \gamma, z)) \mid A \leftarrow \mathsf{Init}(\mathbb{x}, \mathbb{w});\ z \leftarrow \mathsf{Resp}(\mathsf{st}, \gamma)\},$$
$$D_{\mathsf{SHVZK}} \coloneqq \{(\mathbb{x}, \mathbb{w}, (A, \gamma, z)) \mid (A, z) \leftarrow \mathsf{Sim}(\mathbb{x}, \gamma)\}.$$

In this work, we write HVZK for short.

We define a notion of randomizable transcripts w.r.t. SHVZK Σ-protocols. This modularizes common steps in blindness proofs. For simplicity, we only define a perfect version of randomizability.

Definition 2.6 ((Perfect) Randomizable Transcripts). *Let $\boldsymbol{\Sigma}$ be a Σ-protocol for relation $\mathcal{R}$ with challenge space $\mathcal{C}$, and suppose $\boldsymbol{\Sigma}$ is SHVZK. Let Rand be an efficient randomization algorithm, such that $\mathsf{Rand}(\mathbb{x}, (A, \gamma, z))$, given a valid transcript (A, γ, z) for $\mathbb{x}$ outputs a new valid transcript for $\mathbb{x}$. We say $\boldsymbol{\Sigma}$ has* randomizable transcripts *(resp.* strongly randomizable transcripts*) if a Rand exists such that for all $\mathbb{x} \in \mathcal{L}_{\mathcal{R}}$ (resp. all $\mathbb{x}$) and all accepting $\pi^* = (A^*, \gamma^*, z^*)$, the distributions*

$$D_{\mathsf{SHVZK}} \coloneqq \{(\mathbb{x}, (A, \gamma, z)) \mid \gamma \leftarrow \mathcal{C};\ (A, z) \leftarrow \mathsf{Sim}(\mathbb{x}, \gamma)\}$$
$$D_{\mathsf{Rand}} \coloneqq \{(\mathbb{x}, (A, \gamma, z)) \mid (A, \gamma, z) \leftarrow \mathsf{Rand}(\mathbb{x}, \pi^*)\}$$

are identical.

From the perfect identity of distributions in SVHZK and randomizable transcripts, we immediately obtain the following corollary.

Corollary 2.7. *Suppose $\boldsymbol{\Sigma}$ is a Σ-protocol for relation $\mathcal{R}$ which is SVHZK and has randomizable transcripts. Then for all $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$ and all accepting $\pi^* = (A^*, \gamma^*, z^*)$, the following distributions are identical:*

$$D_{\mathsf{Real}} \coloneqq \{(\mathbb{x}, (A, \gamma, z)) \mid A \leftarrow \mathsf{Init}(\mathbb{x}, \mathbb{w});\ \gamma \leftarrow \mathcal{C};\ z \leftarrow \mathsf{Resp}(\mathsf{st}, \gamma)\}$$
$$D_{\mathsf{SHVZK}} \coloneqq \{(\mathbb{x}, (A, \gamma, z)) \mid \gamma \leftarrow \mathcal{C};\ (A, z) \leftarrow \mathsf{Sim}(\mathbb{x}, \gamma)\}$$
$$D_{\mathsf{Rand}} \coloneqq \{(\mathbb{x}, (A, \gamma, z)) \mid (A, \gamma, z) \leftarrow \mathsf{Rand}(\mathbb{x}, \pi^*)\}$$

Definition 2.8 (Unique response). *A Σ-protocol $\boldsymbol{\Sigma}$ has* unique response *if for any tuple $(\mathbb{x}, A, \gamma)$ the exists at most one z such that $\boldsymbol{\Sigma}.\mathsf{Verify}(\mathbb{x}, (A, \gamma, z)) = 1$.*

$\boldsymbol{\Sigma}$-Protocols for Preimages of Linear Maps The generalization of Schnorr's protocol to proving knowledge of a preimage $\boldsymbol{w}$ for a $\mathbb{Z}_p$-linear map $\phi(\boldsymbol{w}) = \boldsymbol{x}$ is well-known [Mau15]. Namely, let $\phi\colon \mathcal{W} \to \mathcal{X}$ be a $\mathbb{Z}_p$-linear map. Define the *canonical Σ-protocol* $\boldsymbol{\Sigma}_\phi$ for the preimage relation $\mathcal{R}_\phi \coloneqq \{(\boldsymbol{x}, \boldsymbol{w}) \mid \phi(\boldsymbol{w}) = \boldsymbol{x}\}$ with challenge space $\mathcal{C} = \mathbb{Z}_p$ as follows:

- $\mathsf{Init}(\boldsymbol{x}, \boldsymbol{w})$: Sample $\boldsymbol{r} \leftarrow \mathcal{W}$. Output $(\mathsf{st}, \boldsymbol{A})$ where $\mathsf{st} = (\boldsymbol{w}, \boldsymbol{r})$ and $\boldsymbol{A} = \phi(\boldsymbol{r})$.
- $\mathsf{Resp}(\mathsf{st}, \gamma)$: Output $\boldsymbol{z} = \boldsymbol{r} + \gamma \boldsymbol{w}$
- $\mathsf{Verify}(\boldsymbol{x}, \boldsymbol{A}, \gamma, \boldsymbol{z})$: Return 1 if $\phi(\boldsymbol{z}) = \boldsymbol{A} + \gamma \boldsymbol{x}$. (It implicitly checks that all elements are in their respective spaces of definition, i.e., in $\mathcal{X}, \mathcal{C}, \mathcal{W}$ respectively).

We summarize following well-known facts about $\boldsymbol{\Sigma}_\phi$.

Lemma 2.9. *Let $\boldsymbol{\Sigma}_\phi$ the above canonical Σ-protocol for $\mathcal{R}_\phi$. Then $\boldsymbol{\Sigma}_\phi$ is 2-special sound,* SHVZK*, and has strongly randomizable transcripts. More concretely:*

- $\mathsf{Sim}(\boldsymbol{x}, \gamma)$ *samples* $\boldsymbol{z} \leftarrow \mathcal{W}$*, sets* $\boldsymbol{A} = \gamma \cdot \boldsymbol{x} - \phi(\boldsymbol{z})$ *and outputs* $(A, \boldsymbol{z})$.
- $\mathsf{Rand}(\boldsymbol{x}, (\boldsymbol{A}^*, \gamma^*, \boldsymbol{z}^*))$ *samples* $\gamma' \leftarrow \mathcal{C}$ *and* $\boldsymbol{z}' \leftarrow \mathcal{W}$ *and outputs*

$$(\boldsymbol{A}, \gamma, \boldsymbol{z}) = (\boldsymbol{A}^* - \gamma' \boldsymbol{x} + \phi(\boldsymbol{A}'), \gamma^* + \gamma', \boldsymbol{z}^* + \boldsymbol{z}'). \tag{2.1}$$

Moreover, if ϕ is injective, then $\boldsymbol{\Sigma}_\phi$ has unique responses.

For completeness, we provide a proof in Appendix E.1

2.2 Non-Interactive Proof Systems

In this section we recall straightline-extractable non-interactive zero-knowledge proofs as defined in [KRW24]. As [KRW24], we consider NIZKs in the random oracle model and reuse some definitions almost verbatim. As in [KRS23], we additionally consider a common random string crs as input in our definitions. The crs can be derived in the random oracle by domain separation.

Definition 2.10 (Non-Interactive Proof System). *A* non-interactive proof system Π *for NP-relation $\mathcal{R}$ using a random oracle H is a pair $\Pi = (\mathsf{Prove}, \mathsf{Verify})$ of PPT algorithms with access to a random oracle and a CRS $\mathsf{crs} \in \{0,1\}^{\ell(\lambda)}$, where*

- $\mathsf{Prove}^{\mathsf{H}}(\mathsf{crs}, \mathbb{x}, \mathbb{w})$*: generates a proof π given $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$.*
- $\mathsf{Verify}^{\mathsf{H}}(\mathsf{crs}, \mathbb{x}, \pi)$*: verifies a proof π for statement $\mathbb{x}$ and outputs 0 or 1.*

We briefly define standard properties of non-interactive proof systems. A NIPS Π for $\mathcal{R}$ is *perfectly correct* if for any $\mathsf{crs} \in \{0,1\}^\ell$ and $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$ every generated proof is valid, i.e., Verify outputs 1.

We say Π is *zero-knowledge*, if there exists a simulator (which is allowed to choose crs and program the random oracle), such that no distinguisher can distinguish between an honest setup and a Prove oracle, and a simulated setup and a Sim oracle (which does not learn the witness, but still simulates convincing proofs).

Finally, we say Π is *straightline $\widetilde{\mathcal{R}}$-extractable* for a knowledge relation $\widetilde{\mathcal{R}}$ (which may differ from the correctness relation $\mathcal{R}$, but has $\mathcal{L}_{\mathcal{R}} = \mathcal{L}_{\widetilde{\mathcal{R}}}$), if there exists an extractor $(\mathsf{ExtSetup}, \mathsf{Ext})$ which is allowed to choose crs via $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{ExtSetup}(1^\lambda)$, such that: (1) $\mathsf{ExtSetup}$ is indistinguishable from uniform; (2) Given $(\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{ExtSetup}(1^\lambda)$ as the crs, for any accepting proof $(\mathbb{x}, \pi)$ an adversary

submits to a verification oracle, the extractor can provide a witness $\widetilde{\mathbb{w}}$ such that $(\mathbb{x}, \widetilde{\mathbb{w}}) \in \widetilde{\mathcal{R}}$, given only the trapdoor td and all list of all random oracle queries.

Looking ahead, we will need to straightline extract the to-be-signed message M, similar to [KRS23; KRW24]. As M will be additionally be ElGamal encrypted, we can efficiently realize this by using double encryption. That is, we choose $\mathsf{crs} \in \mathbb{G}$, and interpret $\mathsf{pk} = (G, \mathsf{crs}) \in \mathbb{G}^2$ as an ElGamal public key. (The extractor will remember the secret key sk.) We elaborate on this below.

Remark 2.11 (Efficient straightline extractability). In [KRW24], the Fischlin transformation is used to obtain straightline extractability for committed scalar message $m \in \mathbb{Z}_p$, at the cost of superconstant sized proofs. In our setting, the message $M \in \mathbb{G}$ is a group element. Straightline extraction of group elements is simple, using the "encryption to the sky" approach (see, *e.g.*, [KLN23; KRS23]): To prove $\mathcal{R}$, one proves encrypts M and includes ct'_M in the statement $\mathbb{x}'$. Now one proves the augmented relation $\mathcal{R}'$ which states that $\mathcal{R}$ holds and ct'_M contains the same M as in the witness $\mathbb{w}$. By putting a (dual-mode) ElGamal public key into crs, and keeping the secret key for extraction, this reduces straightline extractability of M to (non-straightline) soundness of Π. For Π derived by Fiat–Shamir for a linear map ϕ, this modification is very efficient as it just "combines" the original ϕ with a ϕ_{elg} for ElGamal encryption.

2.3 (Partially) Blind Signatures

Now, we define blind signatures [Cha82]. For brevity, we directly define their extension to partial blindness [AF96]. We follow closely the definitions from [KRW24].

Definition 2.12 (Partially Blind Signature Scheme). *A partially blind signature scheme with message space $\mathcal{M}$ and common message space $\mathcal{I}$ is a tuple of PPT algorithms $\mathsf{BS} = (\mathsf{KeyGen}, \mathsf{BSign}, \mathsf{BUser}, \mathsf{Verify})$ with the following syntax:*

- $\mathsf{KeyGen}(1^\lambda)$ *outputs a pair of keys* $(\mathsf{vk}, \mathsf{sk})$*. We assume* vk *can be efficiently computed from* sk*.*
- $\mathsf{BSign}(\mathsf{sk}, I) \rightleftarrows \mathsf{BUser}(\mathsf{vk}, m, I)$*:* BSign *takes as input a secret key* sk *and common message $I \in \mathcal{I}$.* BUser *takes as input a key* vk*, a message $m \in \mathcal{M}$ and common message $I \in \mathcal{I}$. After the execution,* BUser *returns a signature σ and we write $\sigma \leftarrow \langle \mathsf{BSign}(\mathsf{sk}, I), \mathsf{BUser}(\mathsf{vk}, m, I) \rangle$.*
- $\mathsf{Verify}(\mathsf{vk}, m, I, \sigma)$ *is deterministic and takes as input public key* vk*, message $m \in \mathcal{M}$, a common message I, and a signature σ, and outputs $b \in \{0, 1\}$.*

The security properties we demand from partially blind signatures are correctness, partial blindness and one-more (strong) unforgeability. As these notions are fairly standard, we only provide a brief overview and refer to Appendix B.2 for more details.

Correctness. An honest signing protocol execution yields a valid signature.

One-more (strong) unforgeability. The unforgeability guarantee of a (partially) blind signature scheme is that a valid signature can only be obtained via interaction with the signer. That is, one cannot output more valid signatures σ_i for distinct messages μ_i than the number of successfully completed signing sessions (one-more unforgeability, OMUF). We can strengthen the notion by demanding that the pairs (σ_i, μ_i) are distinct (one-more strong unforgeability, OMSUF)

Partial blindness. Partial blindness asserts that a (malicious) signer cannot link a concrete signing session with the obtained signature-message pair (σ_i, μ_i).

3 Baseline Signature Scheme

In this section, we introduce a signature on which we base our blind signature constructions in Sections 4 and 5. For this, we first introduce the functions Φ_{elg} and Φ_{dh} on which our signature is based, the derived Σ-protocols $\boldsymbol{\Sigma}_{\mathsf{elg}}, \boldsymbol{\Sigma}_{\mathsf{dh}}$. Then, we define the baseline signature scheme, and give some high-level intuition.

3.1 Preparations

Let us define two linear functions Φ_{elg} and Φ_{dh} to improve readability.

ElGamal Decryption. First, let Φ_{elg}, parameterized by $\mathsf{pk} = (G, H) \in \mathbb{G}^2$, be defined as follows.

$$\Phi_{\mathsf{elg}}^{\mathsf{pk}}(\mathbf{C}, (x, y)) = \begin{pmatrix} yH - xG \\ yC_1 - xC_0 \end{pmatrix}^{\mathsf{T}} = (x, y) \cdot \begin{pmatrix} -G & -C_0 \\ H & C_1 \end{pmatrix}. \tag{3.1}$$

If clear by context, we omit parameter pk. Observe that for fixed $\mathbf{C}$, the function Φ_{elg} is linear. We define the relation $\mathcal{R}_{\mathsf{elg}}$ with induced language $\mathcal{L}_{\mathsf{elg}}$ as

$$\mathcal{R}_{\mathsf{elg}} \coloneqq \big\{(\mathbb{x}, \mathbb{w}) \mid (0, M) = \Phi_{\mathsf{elg}}^{\mathsf{pk}}(\mathbf{C}, (x, y))\big\}, \tag{3.2}$$

where $\mathbb{x} = (\mathsf{pk}, \mathbf{C}, M) \in \mathbb{G}^5, \mathbb{w} = (x, y) \in \mathbb{Z}_p^2$. Note that $\mathcal{L}_{\mathsf{elg}}$ contains ElGamal ciphertexts $\mathbf{C}$ that encrypt to M with respect to scaled $\mathsf{pk}_y = (yG, yH)$ and the witness (x, y) is a scaled secret key $(s, 1)$. In particular, if $y \neq 0$, then $\mathbf{C}$ encrypts $1/y \cdot M$ with respect to pk. Finally, observe that $M \neq 0$, then $\mathbf{C}$ is *not* an encryption of 0 with respect to pk.

DDH. Second, we define the linear function Φ_{dh} parameterized by (G, D_1) such that

$$\Phi_{\mathsf{dh}}^{G, D_1}(d_2) = \begin{pmatrix} d_2 G \\ d_2 D_1 \end{pmatrix}^{\mathsf{T}}. \tag{3.3}$$

If clear by context, we omit parameter (G, D_1). We define the relation $\mathcal{R}_{\mathsf{dh}}$ with induced language $\mathcal{L}_{\mathsf{dh}}$ as

$$\mathcal{R}_{\mathsf{dh}} \coloneqq \big\{(\mathbb{x}, \mathbb{w}) \mid (D_2, D_3) = \Phi_{\mathsf{dh}}^{G, D_1}(d_2)\big\}, \tag{3.4}$$

where $\mathbb{x} = (G, D_1, D_2, D_3) \in \mathbb{G}^4, \mathbb{w} = d_2 \in \mathbb{Z}_p$. Note that $\mathcal{L}_{\mathsf{dh}}$ contains valid DDH tuples.

Interpretation: Σ-protocol for non-zero encryption. We will use the canonical Σ-protocols (Section 2.1) $\boldsymbol{\Sigma}_{\mathsf{elg}}$ and $\boldsymbol{\Sigma}_{\mathsf{dh}}$ derived from ϕ_{elg} and ϕ_{dh}. However, it can be helpful to interpret a part of our protocol as a *non-canonical* Σ-protocol $\boldsymbol{\Sigma}_{\mathsf{nez}}$ for non-zero message encryption, which we explain below. As $\boldsymbol{\Sigma}_{\mathsf{nez}}$ is not canonical, we do not use in our construction, but introduce it only for intuition and explanations.

Remark 3.1. Let $\boldsymbol{\Sigma}_{\mathsf{elg}} = \boldsymbol{\Sigma}_{\Phi_{\mathsf{elg}}}$ be the canonical Σ-protocol for Φ_{elg} as in Eq. (3.1) and $\mathcal{R}_{\mathsf{elg}}$ with $\mathbb{x} = (\mathsf{pk}, \mathbf{C})$. Consider the relation

$$\mathcal{R}_{\mathsf{nez}} \coloneqq \big\{((\mathsf{pk}, \mathbf{C}), (-s, 1)) \mid (0, M) = \Phi_{\mathsf{elg}}^{\mathsf{pk}}(\mathbf{C}, (x, y)) \wedge M \neq 0\big\}, \tag{3.5}$$

that is, statements consist of public key and ciphertexts *which encrypt a non-zero message*, and the witness is the respective (ElGamal) secret key. We now introduce $\boldsymbol{\Sigma}_{\mathsf{nez}}$, which is (implicitly) given in Section 3.1. Essentially, to prove $\mathcal{R}_{\mathsf{nez}}$, what we do is to scale the ElGamal secret key by $y \leftarrow \mathbb{Z}_p^\times$, which gives us an instance of $\mathcal{R}_{\mathsf{elg}}$ with witness (sy, y) and randomized message $M_\$ = yM$. Formally, we define $\boldsymbol{\Sigma}_{\mathsf{nez}}$ with challenge space $\mathbb{Z}_p$ as follows:

- $\mathsf{Init}((\mathsf{pk}, \mathbf{C}), (s, 1))$: Let $(0, M) = \Phi_{\mathsf{elg}}^{\mathsf{pk}}(\mathbf{C}, (s, 1))$. Sample $y \leftarrow \mathbb{Z}_p^\times$ and let $M_\$ = y \cdot M$. Let $\mathbb{x}_{\mathsf{elg}} = (\mathsf{pk}, \mathbf{C}, M_\$)$ and $\mathbb{w}_{\mathsf{elg}} = (sy, y)$. Let $(\mathsf{st}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{elg}}) = \mathsf{Init}((\mathbb{x}_{\mathsf{elg}}, \mathbb{w}_{\mathsf{elg}}))$. Output (st, A) where $\mathsf{st} = \mathsf{st}_{\mathsf{nez}}$ and $A = (M_\$, A_{\mathsf{elg}})$.
- $\mathsf{Resp}(\mathsf{st}, \gamma)$: Output $z = z_{\mathsf{elg}} = \mathsf{Resp}_{\mathsf{elg}}(\mathsf{st}_{\mathsf{elg}}, \gamma)$
- $\mathsf{Verify}(\mathbb{x}, A, \gamma, z)$: Parse $A = (M_\$, A_{\mathsf{elg}})$. Return 1 if $\mathsf{Verify}(\mathbb{x}_{\mathsf{elg}}, A_{\mathsf{elg}}, \gamma, z) = 1$. and $M_\$ \neq 0$.

Properties of Σ-protocols $\boldsymbol{\Sigma}_{\mathsf{elg}}, \boldsymbol{\Sigma}_{\mathsf{dh}}, \boldsymbol{\Sigma}_{\mathsf{nez}}$. The following lemma summarizes the core properties of our Σ-protocols.

Lemma 3.2. *$\boldsymbol{\Sigma}_{\mathsf{dh}}, \boldsymbol{\Sigma}_{\mathsf{elg}}, \boldsymbol{\Sigma}_{\mathsf{nez}}$ are Σ-protocols for relations $\mathcal{R}_{\mathsf{dh}}, \mathcal{R}_{\mathsf{elg}}, \mathcal{R}_{\mathsf{nez}}$, respectively, and each is 2-special sound, SVHZK and have strongly randomizable transcripts.*

3.2 Construction

Let $\boldsymbol{\Sigma}_{\mathsf{elg}} = (\mathsf{Init}_{\mathsf{elg}}, \mathsf{Resp}_{\mathsf{elg}}, \mathsf{Verify}_{\mathsf{elg}})$ and $\boldsymbol{\Sigma}_{\mathsf{dh}} = (\mathsf{Init}_{\mathsf{dh}}, \mathsf{Resp}_{\mathsf{dh}}, \mathsf{Verify}_{\mathsf{dh}})$ be Σ-protocols with challenge space $\mathbb{Z}_p$ for the relations $\mathcal{R}_{\mathsf{elg}}$ and $\mathcal{R}_{\mathsf{dh}}$ defined above, respectively. Denote by $\mathsf{Sim}_{\mathsf{elg}}$ the HVZK simulator of $\boldsymbol{\Sigma}_{\mathsf{elg}}$. We rely on some hash functions for our construction, later modeled as random oracles in the security proof. Let $\mathsf{H}_{\mathsf{ch}} : \{0,1\}^* \to \mathbb{Z}_p$ be a hash function to generate the challenge for the Fiat-Shamir transformation of OR-compiled Σ-protocols $\boldsymbol{\Sigma}_{\mathsf{elg}}$ and $\boldsymbol{\Sigma}_{\mathsf{dh}}$. Let $\mathsf{H}_{\mathsf{par}} : \{0,1\}^* \to (\{G\} \times \mathbb{G}^\times) \times (\mathbb{G}^2)$ be a random oracle whose outputs we view as $\mathsf{H}_{\mathsf{par}}(I) = (\mathsf{pk}_I, \mathbf{C}_I)$, an ElGamal public key $\mathsf{pk}_I = (G, \mathsf{pk}_{I,1}) \in (\mathbb{G}^\times)^2$, together with an ElGamal ciphertext $\mathbf{C}_I \in \mathbb{G}^2$ under this public key. We define the signature $\mathsf{S}_{\mathsf{neq}}$ in the following.

$\mathsf{S}_{\mathsf{neq}}$: Pairing-free signature

- $\mathsf{KeyGen}(1^\lambda)$:
  (1) Sample $D_1 \leftarrow \mathbb{G}$ and $d_2 \leftarrow \mathbb{Z}_p$.
  (2) Set $D_2 \coloneqq d_2 G$, $D_3 \coloneqq d_2 D_1$, and $\mathbf{D} \coloneqq (D_1, D_2, D_3)$.
  (3) Output $\mathsf{vk} \coloneqq \mathbf{D}$ and $\mathsf{sk} \coloneqq d_2$.
- $\mathsf{Sign}(\mathsf{sk}, I, M)$:
  (1) Set $(\mathsf{pk}_I, \mathbf{C}_I) \coloneqq \mathsf{H}_{\mathsf{par}}(I)$.
  (2) Set $\mathbf{C}_M \coloneqq (0, M)$ and $\mathbf{C} \coloneqq \mathbf{C}_I - \mathbf{C}_M$.
  (3) Sample $M_{\$} \leftarrow \mathbb{G}^\times$.
  (4) Compute a proof π as follows:
    (a) Let $\mathbb{x}_{\mathsf{dh}} \coloneqq (G, \mathbf{D})$ with $\mathbb{w}_{\mathsf{dh}} \coloneqq \mathsf{sk}$, and $\mathbb{x}_{\mathsf{elg}} \coloneqq (\mathsf{pk}_I, \mathbf{C}, M_{\$})$.
    (b) Sample $\gamma_{\mathsf{elg}} \leftarrow \mathbb{Z}_p$ and set $(\boldsymbol{A}_{\mathsf{elg}}, \boldsymbol{z}_{\mathsf{elg}}) \leftarrow \mathsf{Sim}_{\mathsf{elg}}(\mathbb{x}_{\mathsf{elg}}, \gamma_{\mathsf{elg}})$.
    (c) Run $(\boldsymbol{A}_{\mathsf{dh}}, \mathsf{st}_{\mathsf{dh}}) \leftarrow \mathsf{Init}_{\mathsf{dh}}(\mathbb{x}_{\mathsf{dh}}, \mathbb{w}_{\mathsf{dh}})$.
    (d) Set $\gamma \coloneqq \mathsf{H}_{\mathsf{ch}}(\mathbb{x}_{\mathsf{dh}}, \mathbb{x}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{dh}})$ and $\gamma_{\mathsf{dh}} \coloneqq \gamma - \gamma_{\mathsf{elg}}$.
    (e) Run $z_{\mathsf{dh}} \leftarrow \mathsf{Resp}_{\mathsf{dh}}(\mathsf{st}_{\mathsf{dh}}, \gamma_{\mathsf{dh}})$.
    (f) Set $\pi \coloneqq (\boldsymbol{A}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{dh}}, \gamma_{\mathsf{elg}}, \gamma_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{elg}}, z_{\mathsf{dh}})$.
  (5) Output $\sigma \coloneqq (M_{\$}, \pi)$.
- $\mathsf{Verify}(\mathsf{vk}, M, I, \sigma)$:
  (1) Parse σ as $\sigma = (M_{\$}, \boldsymbol{A}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{dh}}, \gamma_{\mathsf{elg}}, \gamma_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{elg}}, z_{\mathsf{dh}})$.
  (2) Set $(\mathsf{pk}_I, \mathbf{C}_I) \coloneqq \mathsf{H}_{\mathsf{par}}(I)$.
  (3) Set $\mathbf{C}_M \coloneqq (0, M)$ and $\mathbf{C} \coloneqq \mathbf{C}_I - \mathbf{C}_M$.
  (4) Let $\mathbb{x}_{\mathsf{dh}} \coloneqq (G, \mathbf{D})$ and $\mathbb{x}_{\mathsf{elg}} \coloneqq (\mathsf{pk}, \mathbf{C}, M_{\$})$.
  (5) Set $\gamma \coloneqq \mathsf{H}_{\mathsf{ch}}(\mathbb{x}_{\mathsf{dh}}, \mathbb{x}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{dh}})$.
  (6) Output 0 if $M_{\$} = 0$.
  (7) Output 0 if $\mathsf{Verify}_{\mathsf{dh}}(\mathbb{x}_{\mathsf{dh}}, \boldsymbol{A}_{\mathsf{dh}}, \gamma_{\mathsf{dh}}, z_{\mathsf{dh}}) = 0$.
  (8) Output 0 if $\mathsf{Verify}_{\mathsf{elg}}(\mathbb{x}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{elg}}, \gamma_{\mathsf{elg}}, \boldsymbol{z}_{\mathsf{elg}}) = 0$.
  (9) Output 0 if $\gamma \neq \gamma_{\mathsf{elg}} + \gamma_{\mathsf{dh}}$.
  (10) Otherwise, output 1.

Let us discuss the intuition behind the construction. The verification key vk is a DDH tuple and sk is the witness (w.r.t. the map ϕ_{dh}). The signature itself is an OR-proof for $\mathcal{R}_{\mathsf{dh}} \cup \mathcal{R}_{\mathsf{nez}}$, built from Σ-protocols $\boldsymbol{\Sigma}_{\mathsf{dh}}$ and $\boldsymbol{\Sigma}_{\mathsf{nez}}$, i.e., it asserts that either:

- vk is a DDH tuple, or
- $\mathbf{C}_I - (0, M)$ is *not* an encryption of 0 (under pk_I).

Evidently, the honest signer computes the Fiat–Shamir transformed OR-proof, using the SHVZK simulator $\mathsf{Sim}_{\mathsf{elg}}$ (implicitly, running SHVZK simulation $\mathsf{Sim}_{\mathsf{nez}}$). The resulting proof π, together with $M_{\$}$ is the signature σ. The verification procedure simply recomputes the statement for M in checks the validity the OR-proof.

For security, observe that we do not require M as input to H_{ch} (as $\mathbb{x}_{\mathsf{elg}}$ implicitly fixes M). Also, since $(\mathsf{pk}_I, \mathbf{C}_I)$ are generated by $\mathsf{H}_{\mathsf{par}}$, it is not (efficiently) possible to use this non-zero encryption branch. However, by letting the reduction program $\mathsf{H}_{\mathsf{par}}$, we have access to an all-but-one trapdoor, similar to [KRS23; KRW24].

We stress that the scheme $\mathsf{S}_{\mathsf{neq}}$ only serves as a baseline for our blind signature constructions $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ and $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ (cf. Figs. 1 and 2). As such, we will not analyze the scheme further but refer to the analysis of $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ and $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ instead.

4 Blind Signature in 4 Moves

In this section, we present our 4-move blind signature $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ which achieves one-more unforgeability and partial blindness.

4.1 Additional Preparations

Non-Interactive Proof System Let Π_{M} be a NIZK for the following relation

$$\mathcal{R}_{\mathsf{M}} \coloneqq \{(\mathbb{x}, \mathbb{w}) \mid \mathbf{C} = (0, M) + t \cdot \mathsf{pk}\} \tag{4.1}$$

using the common reference string $\mathsf{crs}_{\mathsf{M}}$ and random oracle H_{Π}, where $\mathbb{x} = (\mathsf{pk}, \mathbf{C}) \in \mathbb{G}^4$ and $\mathbb{w} = (M, t) \in \mathbb{G} \times \mathbb{Z}_p$. For our security analysis, it is sufficient if Π_{M} is straightline $\widetilde{\mathcal{R}}_{\mathsf{M}}$-extractable for the knowledge relation

$$\widetilde{\mathcal{R}}_{\mathsf{M}} \coloneqq \{(\mathbb{x} = (\mathsf{pk}_I, \mathbf{C}),\ \mathbb{w} = M) \mid \exists t \in \mathbb{Z}_p : \mathbb{w} = (M, t), (\mathbb{x}, \mathbb{w}) \in \mathcal{R}_{\mathsf{M}}\}. \tag{4.2}$$

A concrete instantiation (following Remark 2.11) uses $\mathsf{crs}_{\mathsf{M}} \in \mathbb{G}$, and defines $\mathsf{pk}_{\mathsf{ext}} = (G, \mathsf{crs}_{\mathsf{M}})$. During Prove, one encrypts M under $\mathsf{pk}_{\mathsf{ext}}$ as $C_{\mathsf{ext}} = (0, M) + t_{\mathsf{ext}} \cdot \mathsf{pk}_{\mathsf{ext}}$ and includes C_{ext} in the proof. Then one proves the following relation with a canonical Σ-protocol

$$\mathcal{R}_{\mathsf{ext}} \coloneqq \left\{ \begin{array}{l} (\mathbb{x} = (\mathsf{crs}, \mathsf{pk}_I, \mathbf{C}, C_{\mathsf{ext}}),\ \mathbb{w} = (M, t, t_{\mathsf{ext}})) : \\ \qquad \mathbf{C} = (0, M) + t \cdot \mathsf{pk}_I \ \wedge\ C_{\mathsf{ext}} = (0, M) + t_{\mathsf{ext}} \cdot \mathsf{pk}_{\mathsf{ext}} \end{array} \right\}. \tag{4.3}$$

With standard optimizations, the proof size is $3\mathbb{Z}_p + 3\mathbb{G}$.

4.2 Construction

We assume several random oracles (which can be obtained from a single random oracle by standard techniques). In our construction we use $\mathsf{H}_{\mathsf{crs}}, \mathsf{H}_{\mathsf{par}}, \mathsf{H}_M, \mathsf{H}_{\mathsf{ch}}$:

- We always set $\mathsf{crs}_{\mathsf{M}} = \mathsf{H}_{\mathsf{crs}}(0)$ for random oracle $\mathsf{H}_{\mathsf{crs}} \colon \{0\} \to \mathbb{G}$.
- We always set $(\mathsf{pk}_I, \mathbf{C}_I) = \mathsf{H}_{\mathsf{par}}(I)$ for random oracle $\mathsf{H}_{\mathsf{par}} \colon (\{G\} \times \mathbb{G}) \times \mathbb{G}^2$.
- We hash-then-sign via $M = \mathsf{H}_M(\mu)$, where $\mathsf{H}_M \colon \{0,1\}^* \to \mathbb{G}$ is a random oracle. We will often call M the (to-be-signed) "message" (although formally, M is the image of the actual message μ).
- We let $\mathsf{H}_{\mathsf{ch}} \colon \{0,1\}^* \to \mathbb{Z}_p$ be the random oracle used for the Fiat–Shamir transformation.

Additionally, we use the NIZK Π_{M} introduced above. With this, we can state $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ below.

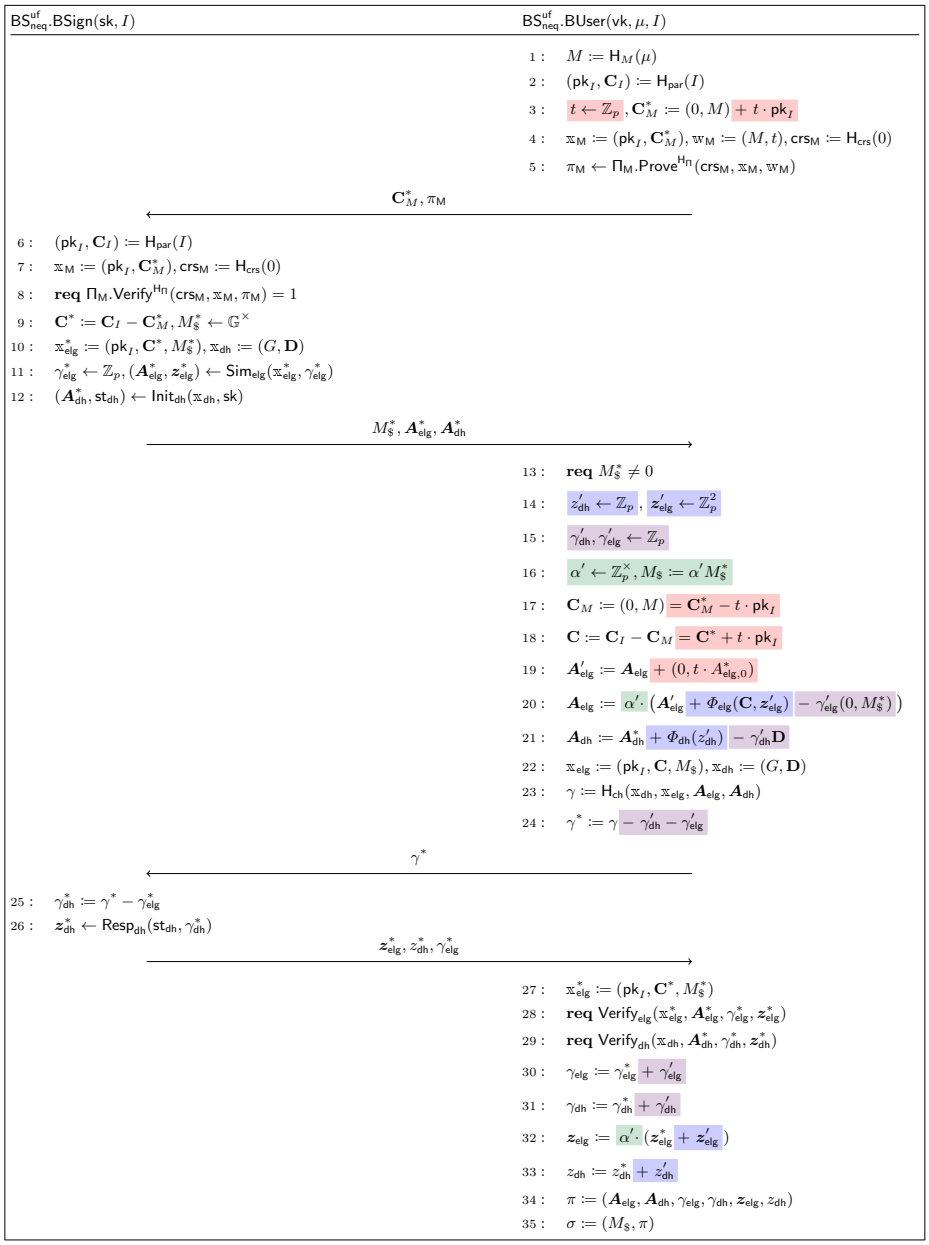

Fig. 1: The signing session for $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ for message $\mu \in \{0,1\}^*$ and common message $I \in \{0,1\}^*$. The signer and user abort (*i.e.*, output $\perp$) if $\mathbf{req}\ C$ is evaluated for a false condition C. Recall that $\mathsf{sk} = d_2$ is a witness for $\mathcal{L}_{\mathsf{dh}}$ membership of $\mathsf{vk} = \mathbf{D}$. The colors highlight terms for masking challenges γ, responses z, and statements $M_{\$}$ and $\mathbf{C}^*_M$.

$\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$: **Our 4-move pairing-free blind signature**
– $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{KeyGen}(1^\lambda)$: Output $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{S}_{\mathsf{neq}}.\mathsf{KeyGen}(1^\lambda)$. – $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{BSign}(\mathsf{sk}, I) \rightleftarrows \mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{BUser}(\mathsf{vk}, m, I)$: Proceeds in 4 moves and is given in Fig. 1. – $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{Verify}(\mathsf{vk}, \mu, I, \sigma)$: Output $b \leftarrow \mathsf{S}_{\mathsf{neq}}.\mathsf{Verify}(\mathsf{vk}, M, I, \sigma)$ for $M := \mathsf{H}_M(\mu)$.

Remark 4.1 (Optimizations). There are a few possible optimization which we omitted for readability:

- A standard optimization is to not include $(\boldsymbol{A}_{\mathsf{dh}}, \boldsymbol{A}_{\mathsf{elg}})$, as these can be recomputed from the statements and $(\gamma_{\mathsf{dh}}, \gamma_{\neq}, z_{\mathsf{dh}}, z_{\mathsf{elg}})$.
- The element $D_1 \leftarrow \mathbb{G}$ can be shared among all signers as part of a CRS, which reduces public key size to $2\mathbb{G}$.
- By switching to a proof for DLog instead of DDH, the public key can be reduced to one group element. The OMUF proof applies verbatim, except that the final step requires rewinding which increases the security loss

4.3 Security Analysis

We prove correctness, one-more unforgeability and partial blindness for $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$. Before we move towards the analysis, let us provide some useful lemmata.

Preparations The following two lemmata are repeatedly used in our proofs. In the first lemma, we recall soundness of OR-compiled Fiat-Shamir Σ-protocols. The proof is standard. For completeness, we include it in Appendix E.2.

Lemma 4.2 (Soundness of OR-proof). *For every H_{ch} query $X := (\mathbb{x}_{\mathsf{dh}}, \mathbb{x}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{dh}})$ with $\mathbb{x}_{\mathsf{dh}} \notin \mathcal{L}_{\mathsf{dh}}$ and $\mathbb{x}_{\mathsf{elg}} \notin \mathcal{L}_{\mathsf{elg}}$, there exists $(\gamma_{\mathsf{dh}}, \gamma_{\mathsf{elg}}, \boldsymbol{z}_{\mathsf{elg}}, z_{\mathsf{dh}})$ such that*

$$\begin{aligned} \tau_{\mathsf{dh}} &:= (\boldsymbol{A}_{\mathsf{dh}}, \gamma_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{dh}}) \text{ is valid for } \mathbb{x}_{\mathsf{dh}} \\ \tau_{\mathsf{elg}} &:= (\boldsymbol{A}_{\mathsf{elg}}, \gamma_{\mathsf{elg}}, z_{\mathsf{elg}}) \text{ is valid for } \mathbb{x}_{\mathsf{elg}} \\ \gamma &:= \mathsf{H}_{\mathsf{ch}}(X) = \gamma_{\mathsf{dh}} + \gamma_{\mathsf{elg}} \end{aligned}$$

with probability at most $1/p$. If Q_{ch} queries to H_{ch} were made, the probability that such a query X exists is at most Q_{ch}/p.

The next lemma captures a recurring random oracle reprogramming step.

Lemma 4.3 (Chosen plaintext setup). *Consider the following pair of experiments $(\mathsf{Exp}^0, \mathsf{Exp}^1)$ w.r.t. random oracle $\mathsf{H}_{\mathsf{par}}$ and an adversary $\mathcal{A}$ which makes at most Q_{par} queries. On a $\mathsf{H}_{\mathsf{par}}$ query, the experiment Exp^b operates as follows:*

- *Invoke $\mathcal{A}^{\mathsf{H}_{\mathsf{par}}}(1^\lambda)$*
- *Upon a fresh query $\mathsf{H}_{\mathsf{par}}(I)$:*

- *Sample* $\mathsf{pk}_{I,1} \leftarrow \mathbb{G}$*, set* $\mathsf{pk}_I \leftarrow (G, \mathsf{pk}_{I,1})$*, and send it to* $\mathcal{A}$*.*
- *Receive* M_I *from the challenger as a response.*
- *If* $b = 0$*, sample* $\mathbf{C}_I \leftarrow \mathbb{G}^2$*, program* $\mathsf{H}_{\mathsf{par}}(I) \coloneqq (\mathsf{pk}_I, \mathbf{C}_I)$*, and output* $\mathbf{C}_I$ *to* $\mathcal{A}$*.*
- *If* $b = 1$*,* $\mathbf{C}_I \leftarrow (0, M_I) + t_I \cdot \mathsf{pk}_I$ *where* $t_I \leftarrow \mathbb{Z}_p$*, program* $\mathsf{H}_{\mathsf{par}}(I) \coloneqq (\mathsf{pk}_I, \mathbf{C}_I)$*, and output* $\mathbf{C}_I$ *to* $\mathcal{A}$*.*

– $\mathcal{A}$ *eventually outputs a bit* b'*, which the experiment outputs.*

Then

$$|\Pr[\mathsf{Exp}^0(\mathcal{A}) = 1] - \Pr[\mathsf{Exp}^1(\mathcal{A}) = 1]| \leq \mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{A}}(\lambda, Q_{\mathsf{par}}).$$

The proof is simple and relegated to Appendix E.3.

Correctness First, we show correctness. This follows by inspection and we omit details. A formal proof is given in Appendix E.4.

Theorem 4.4 (Correctness). $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ *is perfectly correct.*

One-more Unforgeability Let us turn towards one-more unforgeability. The proof structure is quite similar to [KRW24], as we also simulate signing by using an all-but-one trapdoor. Technically, [KRW24] punctured the verification key, while we we puncture public parameters, namely public key and ciphertext output by $\mathsf{H}_{\mathsf{par}}$. In the proof, we guess the forgery common message $\overline{I}$ and guess (and program) the forgery (hashed) message $\overline{M} \coloneqq \mathsf{H}_M(\overline{I}, \overline{\mu})$. The we can puncture the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ branch for $\overline{I}$ by setting $(\mathsf{pk}^*_I, \mathbf{C}^*_I) = \mathsf{H}_{\mathsf{par}}(\overline{I})$ to an encryption of $\overline{M}$. This allows the reduction to change the signing to use the $\boldsymbol{\Sigma}_{\mathsf{elg}}$-OR-branch with witness (sk_I, M) for all queries, except for $(\overline{I}, \overline{M})$, for which no witness exists for the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ branch. In the final step, we switch vk to a non-DDH tuple, which makes the $\boldsymbol{\Sigma}_{\mathsf{dh}}$-OR-branch false. Since we forced $\mathcal{A}$ to forge for $(\overline{I}, \overline{M})$ (by guessing), a valid forgery would constitute a break of the soundness property of the Fiat–Shamir-compiled OR-proof.

We comment on an interesting subtlety here: We require the user to prove knowledge of M within $\mathbf{C}_M$, although the reduction will be in possession of sk_I, and could thus decrypt. However, in the step where we change set $M_{\overline{I}}$ from random to $\overline{M}$, we need to rely on IND-CPA security of $\mathbf{C}_I$, and thus must forget the secret key. But to perfectly emulate the hybrid game, we need still need to "decrypt" $\mathbf{C}_M$. Fortunately, the NIZK Π_{M} for this is very efficient (in the random oracle model): It is simply a proof of valid double encryption, for which it suffices to send $3\mathbb{Z}_p + 3\mathbb{G}$ elements additionally.

Theorem 4.5. *Denote by* p *the order of group* $\mathbb{G}$*. For any PPT adversary* $\mathcal{A}$ *that causes at most* $Q_{\mathsf{ch}}, Q_M, Q_{\mathsf{par}}, Q_\Pi$ *random oracle queries to* $\mathsf{H}_{\mathsf{ch}}, \mathsf{H}_M, \mathsf{H}_{\mathsf{par}}, \mathsf{H}_\Pi$*, respectively, to occur in the game, and that starts at most* Q_S *signing sessions, there are reductions* $\mathcal{B}_{\mathsf{crs}}$*,* $\mathcal{B}_{\mathsf{QDDH}}$ *whose running time is roughly that of the OMUF*

game, such that

$$\mathsf{AdvOMUF}^{\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}}_{\mathcal{A}}(\lambda) \leq \mathsf{AdvCRS}^{\Pi_{\mathsf{m}},\mathsf{ExtSetup}}_{\mathcal{B}_{\mathsf{crs}}}(\lambda, Q_{\Pi}) + \frac{Q_S}{p} + \frac{Q_M^2}{p} + \frac{1+Q_{\mathsf{ch}}}{p} + Q_M \cdot Q_{\mathsf{par}} \cdot \Big(\mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{B}_{\mathsf{dh}}}(\lambda, Q_{\mathsf{par}}) + \mathsf{AdvExt}^{\Pi_{\mathsf{m}},\mathsf{Ext}}_{\mathcal{B}_{\mathsf{ext}}}(\lambda, Q_{\Pi})\Big).$$

Proof. Let $\mathcal{A}$ be a PPT adversary against one-more unforgeability of $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$. Let $\mathbb{G}$ be a group of prime order p with generator G. For random oracle $\mathsf{H}_{\mathsf{xyz}} \in \{\mathsf{H}_M, \mathsf{H}_{\mathsf{ch}}, \mathsf{H}_{\mathsf{par}}, \mathsf{H}_{\Pi}, \mathsf{H}_{\mathsf{crs}}\}$, denote by Q_{xyz} the number of oracle queries to $\mathsf{H}_{\mathsf{xyz}}$. We use the convention that $\mathsf{H}_{\mathsf{xyz}}$ queries made by the game (*e.g.*, during signing queries or verification) count towards Q_{xyz}. Denote by Q_S the number of $\mathcal{A}$'s signing queries.

We proceed with a sequence of games Game i and denote by ε_i the advantage of $\mathcal{A}$ in Game i (*i.e.*, the probability that Game i outputs 1).

Game 0 (Honest). This game is the real one-more unforgeability game for scheme $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$. Recall that oracles $\mathsf{H}_M, \mathsf{H}_{\mathsf{ch}}, \mathsf{H}_{\mathsf{par}}, \mathsf{H}_{\Pi}, \mathsf{H}_{\mathsf{crs}}$ are modeled as random oracles. For convenience, let us recap the game below.

The game first samples $\mathsf{vk} = \mathbf{D}$ and $\mathsf{sk} = d_2$ via $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{KeyGen}$. That is, it samples $d_2 \leftarrow \mathbb{Z}_p$ and $D_1 \leftarrow \mathbb{G}$, then sets $D_3 \coloneqq d_2 D_1$ and $\mathbf{D} = (D_1, D_2, D_3)$. Next, the game sends vk to $\mathcal{A}$ and provides access to the random oracles and signing oracles $\mathcal{O}_{\mathsf{BSign}_1}, \mathcal{O}_{\mathsf{BSign}_2}$. In the end, $\mathcal{A}$ outputs a common message $\overline{I}$ and forgeries $(\overline{\mu}_j, \overline{\sigma}_j)_{j\in[Q_{\mathsf{frg}}]}$. The game outputs 1 iff $\mathcal{O}_{\mathsf{BSign}_2}$ was queried at most $Q_{\mathsf{frg}}-1$ times with common message $\overline{I}$, all messages $\{\overline{\mu}_j\}_{j\in[Q_{\mathsf{frg}}]}$ are pairwise-distinct, and all signatures verify (*i.e.*, $\mathsf{S}_{\mathsf{neq}}.\mathsf{Verify}(\mathsf{vk}, \overline{M}_j, \overline{I}, \overline{\sigma}_j) = 1$ with $\overline{M}_j \coloneqq \mathsf{H}_M(\overline{\mu}_j)$). We consistently mark values x associated to the forgeries with $\overline{x}$. We identify each signing session with a session identifier sid provided as input in $\mathcal{O}_{\mathsf{BSign}_1}$ and $\mathcal{O}_{\mathsf{BSign}_2}$. The signing oracles behave as follows:

- $\mathcal{O}_{\mathsf{BSign}_1}(\mathsf{sid}, I, \mathbf{C}^*_M, \pi_{\mathsf{M}})$: The game sets $(\mathsf{pk}_I, \mathbf{C}_I) \coloneqq \mathsf{H}_{\mathsf{par}}(I)$. Then, it verifies π_{M}, *i.e.*, it sets $\mathbb{x}_{\mathsf{M}} \coloneqq (\mathsf{pk}_I, \mathbf{C}^*_M)$ and $\mathsf{crs}_{\mathsf{M}} \coloneqq \mathsf{H}_{\mathsf{crs}}(0)$, and aborts if $\Pi_{\mathsf{M}}.\mathsf{Verify}^{\mathsf{H}_{\Pi}}(\mathsf{crs}_{\mathsf{M}}, \mathbb{x}_{\mathsf{M}}, \pi_{\mathsf{M}}) \neq 1$. Then, it sets up ciphertext $\mathbf{C}^* \coloneqq \mathbf{C}_I - \mathbf{C}^*_M$ and target $M^*_{\$} \leftarrow \mathbb{G}^{\times}$, and computes Σ-protocol messages for $\mathbb{x}^*_{\mathsf{elg}} \coloneqq (\mathsf{pk}_I, \mathbf{C}^*, M^*_{\$})$ and $\mathbb{x}_{\mathsf{dh}} \coloneqq (G, \mathbf{D})$. That is, it samples $\gamma^*_{\mathsf{elg}} \leftarrow \mathbb{Z}_p$ and sets $(\boldsymbol{A}^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}}) \leftarrow \mathsf{Sim}_{\mathsf{elg}}(\mathbb{x}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}})$, as well as $(\boldsymbol{A}^*_{\mathsf{dh}}, \mathsf{st}_{\mathsf{dh}}) \leftarrow \mathsf{Init}_{\mathsf{dh}}(\mathbb{x}_{\mathsf{dh}}, \mathsf{sk})$. The game stores $\gamma^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}}$ and $\mathsf{st}_{\mathsf{dh}}$ in its state for sid, and outputs
$$(M^*_{\$}, \boldsymbol{A}^*_{\mathsf{elg}}, \boldsymbol{A}^*_{\mathsf{dh}}).$$
- $\mathcal{O}_{\mathsf{BSign}_2}(\mathsf{sid}, \gamma^*)$: The game retrieves $\gamma^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}}$ and $\mathsf{st}_{\mathsf{dh}}$ from the state for sid (and aborts if this is not possible). Then, it sets $\gamma^*_{\mathsf{dh}} \coloneqq \gamma^* - \gamma^*_{\mathsf{elg}}$ and $\boldsymbol{z}^*_{\mathsf{dh}} \leftarrow \mathsf{Resp}_{\mathsf{dh}}(\mathsf{st}_{\mathsf{dh}}, \gamma^*_{\mathsf{dh}})$. The game empties its state for sid and outputs
$$(\boldsymbol{z}^*_{\mathsf{elg}}, z^*_{\mathsf{dh}}, \gamma^*_{\mathsf{elg}}).$$

By definition, we have
$$\mathsf{AdvOMUF}^{\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}}_{\mathcal{A}}(\lambda) = \varepsilon_0.$$

Game 1 (Abort if H_M collision). The game aborts its entire execution if there is a collision in H_M. By a standard birthday-bound argument, we have

$$|\varepsilon_0 - \varepsilon_1| \leq \frac{Q_M^2}{p}.$$

Game 2 (Extract M from π_{M}). Before sending vk to $\mathcal{A}$, the game sets $\mathsf{crs}_{\mathsf{M}} \leftarrow \mathsf{ExtSetup}(1^\lambda)$ and programs $\mathsf{H}_{\mathsf{crs}}(0) := \mathsf{crs}_{\mathsf{M}}$. Later, on every $\mathcal{O}_{\mathsf{BSign}_1}$ query of the form $(\mathsf{sid}, I, \mathbf{C}_M^*, \pi_{\mathsf{M}})$, after verifying π_{M}, the adversary extracts the message $M \leftarrow \mathsf{Ext}((\mathsf{td}, \mathcal{Q}), (\mathbb{x}_{\mathsf{M}}, \pi_{\mathsf{M}}))$ via π_{M}. Here, $\mathcal{Q}$ is a list containing all H_Π queries so far.

Note that while we already extract the message, we do not use the extracted value within the simulation yet. It is straightforward to construct a reduction $\mathcal{B}_2$ with running time similar to $\mathcal{A}$ such that

$$|\varepsilon_2 - \varepsilon_3| \leq \mathsf{AdvCRS}_{\mathcal{B}_2}^{\Pi_{\mathsf{m}}, \mathsf{ExtSetup}}(\lambda, Q_\Pi).$$

At this point, the game does *not* know whether M is actually a witness for relation $\widetilde{\mathcal{R}}_{\mathsf{M}}$. This can readily be verified via the secret key sk_I associated to pk_I. Yet, because the simulation cannot depend on sk_I for subsequent proof steps, we cannot yet add an explicit abort condition that that relies on sk_I. Nevertheless, it is useful to know the extracted message M for the next proof steps.

Game 3 (Guess $\overline{I}$). We guess the first query to $\mathsf{H}_{\mathsf{par}}$ such that the forgeries' common message $\overline{I}$ is provided as input. That is, the game samples $i_{I,\mathcal{A}} \leftarrow [Q_{\mathsf{par}}]$ at its start. When $\mathcal{A}$ outputs common message $\overline{I}$ and its forgeries, the game additionally checks whether $\overline{I}$ was queried for the first time to $\mathsf{H}_{\mathsf{par}}$ on the $i_{I,\mathcal{A}}$-th query. If not, the game aborts its entire execution.

Observe that such a query must exist, as we also count the game's $\mathsf{H}_{\mathsf{par}}$ queries and the game evaluates $\mathsf{H}_{\mathsf{par}}$ on input $\overline{I}$ when verifying the forgeries. As the guess $i_{I,\mathcal{A}}$ is hidden from $\mathcal{A}$, we have that

$$\varepsilon_2 \leq Q_{\mathsf{par}} \cdot \varepsilon_3.$$

We stress that at this point, the game knows $\overline{I}$ after the first $\mathsf{H}_{\mathsf{par}}$ query with input $\overline{I}$ was made. In particular, as the game evaluates $\mathsf{H}_{\mathsf{par}}$ on common message I during each $\mathcal{O}_{\mathsf{BSign}_1}$ query, the game knows the forgeries' common message $\overline{I}$ at latest when the first $\mathcal{O}_{\mathsf{BSign}_1}$ query with common message $\overline{I}$ is made.

Game 4 (Guess unsigned $\overline{M}$ in forgery). We guess the first query $i_{M,\mathcal{A}}$ to H_M such that the following two conditions hold:

(1) The input $X_{i_{M,\mathcal{A}}}$ to the $i_{M,\mathcal{A}}$-th H_M query is part of the *hashed* messages $\overline{M}_1, \ldots, \overline{M}_{Q_{\mathsf{frg}}}$ in $\mathcal{A}$'s forgeries and $X_{i_{M,\mathcal{A}}}$ was never queried to H_M before.
(2) No session with common message $\overline{I}$ is *completed* if $\overline{M} = \mathsf{H}_M(X_{i_{M,\mathcal{A}}})$ is extracted from proof π_{M} (cf. game 2).

Again, the game aborts its execution if the guess was incorrect.

If $\mathcal{A}$ is successful, then $\mathcal{A}$'s output contains Q_{frg} distinct messages $\overline{\mu}_j$. As H_M is collision-free (cf. game 1), there are also Q_{frg} distinct hashed messages $\overline{M}_j \coloneqq \mathsf{H}_M(\overline{\mu}_j)$. As there are at most $Q_{\mathsf{frg}} - 1$ completed sessions for common message $\overline{I}$, there must be at least one hashed message $\overline{M} \in \{\overline{M}_1, \ldots, \overline{M}_{Q_{\mathsf{frg}}}\}$ that was never extracted in any of these completed $Q_{\mathsf{frg}} - 1$ sessions. Consequently, such an index $i_{M,\mathcal{A}}$ must exist, and since the guess is hidden from $\mathcal{A}$, we have

$$\varepsilon_3 \leq Q_M \cdot \varepsilon_4.$$

In the following, we denote by $\overline{M}$ the output of the $i_{M,\mathcal{A}}$-th H_M query. Note that if $\mathcal{A}$ is successful, we can assume that $\overline{M}$ is known by the game from the start on.[6]

Game 5 (Abort if M_I^* is extracted). Initially, the game samples a random message $M_I^* \leftarrow \mathbb{G}$. Then, the game aborts its entire execution if M_I^* is extracted from π_{M} in $\mathcal{O}_{\mathsf{BSign}_1}$ for any common message I. That is, after setting $M \leftarrow \mathsf{Ext}((\mathsf{td}, \mathcal{Q}), (\mathbb{x}, \pi_{\mathsf{M}}))$ in $\mathcal{O}_{\mathsf{BSign}_1}$ (cf. game 3), the game checks if $M = M_I^*$. If so, the game aborts its entire execution, else it continues as before.

As M_I^* is never used within in the simulation (except for the abort condition), a union bound yields

$$|\varepsilon_4 - \varepsilon_5| \leq \frac{Q_S}{p}.$$

Game 6 (Setup $\mathbf{C}_I$ with specific messages). We now setup the ciphertexts $\mathbf{C}_I$ output by $\mathsf{H}_{\mathsf{par}}$ depending on whether the forgeries' common message $\overline{I}$ was queried. That is, on the first $\mathsf{H}_{\mathsf{par}}$ query with input $\overline{I}$, the game sets up $\mathsf{pk}_I \leftarrow \{G\} \times \mathbb{G}$ at random and encrypts $\overline{M}$ in $\mathbf{C}_I$, *i.e.*, computes ElGamal ciphertext $\mathbf{C}_I \coloneqq (0, \overline{M}) + t \cdot \mathsf{pk}_I$ for $t \leftarrow \mathbb{Z}_p$. The game outputs $(\mathsf{pk}_I, \mathbf{C}_I)$. On all other fresh $\mathsf{H}_{\mathsf{par}}$ queries on input I, the game sets up pk_I at random and encrypts M_I^* in $\mathbf{C}_I$, *i.e.*, sets $\mathbf{C}_I \coloneqq (0, M_I^*) + t \cdot \mathsf{pk}_I$, Again, the game outputs $(\mathsf{pk}_I, \mathbf{C}_I)$. Note that M_I^* is chosen as in Game 5.

Recall that in the previous game, all $\mathsf{H}_{\mathsf{par}}$ outputs $\mathbf{C}_I$ are uniform over $\mathbb{G}^2$. In this game, the ciphertexts $\mathbf{C}_I$ are setup with messages chosen by the game. This is exactly the setting in Lemma 4.3. As there are Q_{par} oracle queries in total, there is an adversary $\mathcal{B}_6$ on QDDH with running time roughly that of $\mathcal{A}$ such that

$$|\varepsilon_5 - \varepsilon_6| \leq \mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{B}_6}(\lambda, Q_{\mathsf{par}}).$$

As consequence of Game 5 and Game 6, the $\mathbf{C}_I$ output by $\mathsf{H}_{\mathsf{par}}$ are setup in two manners.

Remark 4.6. The ciphertexts $\mathbf{C}_I$ given by $(\mathsf{pk}_I, \mathbf{C}_I) = \mathsf{H}_{\mathsf{par}}(I)$ are setup as follows.

(1) For the forgery's common message $I = \overline{I}$, the ciphertext $\mathbf{C}_I$ encrypts the guessed message $\overline{M}$ (cf. Game 4).

[6] The game initially samples $\overline{M} \leftarrow \mathbb{G}$ and outputs $\overline{M}$ on the $i_{M,\mathcal{A}}$-th H_M query.

(2) For other common messages $I \neq \overline{I}$, the ciphertext $\mathbf{C}_I$ encrypts M_I^*. Also, M_I^* is never extracted within $\mathcal{O}_{\mathsf{BSign}_1}$ (else the game aborts).

In particular, $\mathbf{C}^* = \mathbf{C}_I - \mathbf{C}_M^*$ encrypts a non-zero value if $I = \overline{I}$ and $M \neq \overline{M}$ or if $I \neq \overline{I}$ (cf. Game 5 and Game 6).

Game 7 (Setup pk_I with known secret key). On every $\mathsf{H}_{\mathsf{par}}$ query, the game samples $\mathsf{sk}_I \leftarrow \mathbb{Z}_p$ and sets $\mathsf{pk}_{I,1} \coloneqq \mathsf{sk}_I \cdot G$. It computes $\mathbf{C}_I$ as in Game 6 and outputs $(\mathsf{pk}_I, \mathbf{C}_I)$ with $\mathsf{pk}_I \coloneqq (G, \mathsf{pk}_{I,1})$. Clearly, both games are identically distributed and we have

$$\varepsilon_6 = \varepsilon_7.$$

Game 8 (Abort if M is an invalid $\widetilde{\mathcal{R}}_{\mathsf{M}}$ witness). We now abort if the extracted message M is not a witness for relation $\widetilde{\mathcal{R}}_{\mathsf{M}}$. That is, after the game extracts M in $\mathcal{O}_{\mathsf{BSign}_1}$ from the proof π_{M} for statement $\mathbb{x}_{\mathsf{M}} = (\mathsf{pk}_I, \mathbf{C}_M^*)$, it decrypts $\mathbf{C}_M^*$ and aborts if the obtained message does not match M. More formally, the game sets $M' \leftarrow C_{M,1}^* - \mathsf{sk}_I \cdot C_{M,0}^*$ and aborts its entire execution if $M \neq M'$.

We can show that the abort probability is negligible under straightline $\widetilde{\mathcal{R}}_{\mathsf{M}}$-extractability of Π_{m}. It is easy to see that $M = M'$ iff $(\mathbb{x}_{\mathsf{M}}, M) \in \widetilde{\mathcal{R}}_{\mathsf{M}}$ (cf. Eq. (4.2)). In conclusion, we can construct an adversary $\mathcal{B}_8$ with running time roughly that of $\mathcal{A}$ such that

$$|\varepsilon_7 - \varepsilon_8| \leq \mathsf{AdvExt}_{\mathcal{B}_8}^{\Pi_{\mathsf{m}},\mathsf{Ext}}(\lambda, Q_{\mathsf{\Pi}})$$

Our next goal is to transition to a game where the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts are computed via the known witness, and the $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcripts are simulated. For this, it is important that the statement $\mathbb{x}_{\mathsf{elg}}^* = (\mathsf{pk}_I, \mathbf{C}^*, M_{\$}^*)$ is in the language $\mathcal{L}_{\mathsf{elg}}$. The abort conditions in previous games make sure that this is indeed true, except if

$$I = \overline{I} \qquad \text{and} \qquad M = \overline{M}. \tag{$\star$}$$

In case Eq. ($\star$) holds, the game still simulates the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcript.[7] Also, note that in order to compute $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts honestly, we need to find a witness for $\mathbb{x}_{\mathsf{elg}}^*$. For this, we setup $M_{\$}^*$ based on the message M^* encrypted in $\mathbf{C}^*$ as $M_{\$}^* = y \cdot M^*$ with known discrete logarithm y. Observe that then, the game knows a witness $(y \cdot \mathsf{sk}_I, y)$ for $\mathcal{R}_{\mathsf{elg}}$ in all signing sessions *except* if Eq. ($\star$) holds. We elaborate below.

Game 9 (Compute $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts honestly). Now, the game computes the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcript $(\boldsymbol{A}_{\mathsf{elg}}^*, \gamma_{\mathsf{elg}}^*, \boldsymbol{z}_{\mathsf{elg}})$ via the witness except if Eq. ($\star$) holds. More precisely, in $\mathcal{O}_{\mathsf{BSign}_1}$ after extracting M, the game sets $M^* \coloneqq \overline{M} - M$ if $I = \overline{I}$ and $M \neq \overline{M}$. Else, if $I \neq \overline{I}$, then it sets $M^* \coloneqq M_I^* - M$. Note that M^* is the message encrypted in $\mathbf{C}^*$. Then, the game sets $M_{\$}^* \coloneqq y \cdot M^*$ for $y \leftarrow \mathbb{Z}_p^\times$ and $\mathbb{w}_{\mathsf{elg}}^* \coloneqq (y \cdot \mathsf{sk}_I, y)$. If otherwise Eq. ($\star$) holds, then $M_{\$}^* \leftarrow \mathbb{G}^\times$ is still sampled at

[7] In this case, both transcripts are simulated and the game is not able to answer the $\mathcal{O}_{\mathsf{BSign}_2}$ oracle. But by definition of $\overline{M}$, the game aborts its execution if this occurs.

random. Note that $\mathbb{x}^*_{\mathsf{elg}} = (\mathsf{pk}_I, \mathbf{C}^*, M^*_\$)$ is set as before. Then, except if Eq. ($\star$) holds, the game samples $\gamma^*_{\mathsf{elg}} \leftarrow \mathbb{Z}_p$ and sets $(\boldsymbol{A}^*_{\mathsf{elg}}, \mathsf{st}_{\mathsf{elg}}) \leftarrow \mathsf{Init}_{\mathsf{elg}}(\mathbb{x}^*_{\mathsf{elg}}, \mathbb{w}^*_{\mathsf{elg}})$. Otherwise, it simulates $(\boldsymbol{A}^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}}) \leftarrow \mathsf{Sim}_{\mathsf{elg}}(\mathbb{x}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}})$ as before. In $\mathcal{O}_{\mathsf{BSign}_2}$, the game sets $\boldsymbol{z}^*_{\mathsf{elg}} \leftarrow \mathsf{Resp}_{\mathsf{elg}}(\mathsf{st}_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}})$ if Eq. ($\star$) holds. All other values are computed as in Game 8. Note that Eq. ($\star$) never occurs in $\mathcal{O}_{\mathsf{BSign}_2}$ due to the choice of $\overline{M}$ (cf. game 4).

First, let us show that $(\mathbb{x}^*_{\mathsf{elg}}, \mathbb{w}^*_{\mathsf{elg}}) \in \mathcal{R}_{\mathsf{elg}}$ (cf. Eq. (3.2)). Due to the abort condition introduced in Game 8, we know that $M = C^*_{M,1} - \mathsf{sk}_I \cdot C^*_{M,0}$. Also, recall that $\mathbf{C}^* = \mathbf{C}_I - \mathbf{C}^*_M$. Together with Remark 4.6, the above yields that

$$M^* = C^*_1 - \mathsf{sk}_I \cdot C^*_0.$$

Also, by construction we have $\mathsf{pk}_{I,1} = \mathsf{sk}_I \cdot \mathsf{pk}_{I,0}$. Multiplying both aforementioned equations by y yields that $(\mathbb{x}^*_{\mathsf{elg}}, \mathbb{w}^*_{\mathsf{elg}}) \in \mathcal{R}_{\mathsf{elg}}$. Thus, the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts $(\boldsymbol{A}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}})$ in Game 8 and Game 9 are identically distributed by HVZK. Also, by construction $M^*_\$$ is distributed uniform over $\mathbb{G}^\times$, as $M^* \neq 0$ (cf. Remark 4.6). In conclusion, we have

$$\varepsilon_8 = \varepsilon_9.$$

Game 10 (Simulate $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcripts). Now, the game simulates $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcript $(\boldsymbol{A}^*_{\mathsf{dh}}, \gamma^*_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{dh}})$ in all signing sessions. In particular, in $\mathcal{O}_{\mathsf{BSign}_1}$, the game samples $\gamma^*_{\mathsf{dh}} \leftarrow \mathbb{Z}_p$ and sets $(\boldsymbol{A}^*_{\mathsf{dh}}, z_{\mathsf{dh}}) \leftarrow \mathsf{Sim}_{\mathsf{dh}}(\mathbb{x}^*_{\mathsf{dh}}, \gamma^*_{\mathsf{dh}})$ instead of computing $\boldsymbol{A}_{\mathsf{dh}}$ via $\mathsf{Init}_{\mathsf{dh}}$. Further, the game does not sample γ^*_{elg} at random in $\mathcal{O}_{\mathsf{BSign}_1}$ anymore except if Eq. ($\star$) holds. Instead, the challenger sets $\gamma^*_{\mathsf{elg}} := \gamma^* - \gamma^*_{\mathsf{dh}}$ in $\mathcal{O}_{\mathsf{BSign}_2}$, and uses the simulated response z_{dh} computed in $\mathcal{O}_{\mathsf{BSign}_1}$. As the game aborts its execution if Eq. ($\star$) occurs in $\mathcal{O}_{\mathsf{BSign}_2}$, we leave the simulation behavior in $\mathcal{O}_{\mathsf{BSign}_2}$ unspecified in that case. Other than the above, the game behaves as in Game 9.

If Eq. ($\star$) does not hold in the signing session, then clearly γ^*_{elg} and γ^*_{dh} are distributed identically in Game 9 and Game 10. Further, the $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcript $(\boldsymbol{A}^*_{\mathsf{dh}}, \gamma^*_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{dh}})$ is identically distributed under HVZK. If Eq. ($\star$) holds, then both $\boldsymbol{\Sigma}_{\mathsf{dh}}$ and $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts are simulated. As noted above, it suffices to argue that the $\mathcal{O}_{\mathsf{BSign}_1}$ output $(M^*_\$, \boldsymbol{A}^*_{\mathsf{elg}}, \boldsymbol{A}^*_{\mathsf{dh}})$ in Game 10 are distributed as in Game 9. As the distribution of $\boldsymbol{A}^*_{\mathsf{elg}}$ and $M^*_\$$ remains unchanged, it suffices to inspect $\boldsymbol{A}_{\mathsf{dh}}$. By HVZK, a simulated $\boldsymbol{A}_{\mathsf{dh}}$ as in Game 10 and an honestly computed $\boldsymbol{A}_{\mathsf{dh}}$ as in Game 9 are distributed identically. Consequently, we have

$$\varepsilon_9 = \varepsilon_{10}.$$

Observe that in Game 10, the secret key $\mathsf{sk} = d_2$ is not required anymore for simulation.

Game 11 (Sample non-DDH tuple D). In this game, we change how the vk is setup. Instead of sampling a DDH tuple $\mathbf{D}$, the game samples $\mathbf{D} \leftarrow \mathbb{G}^3$ instead. Then, the game sets $\mathsf{vk} = \mathbf{D}$ and proceeds as in Game 10.

We can construct an adversary $\mathcal{B}_{11}$ against DDH with running time similar to $\mathcal{A}$ and with

$$|\varepsilon_{10} - \varepsilon_{11}| \leq \mathsf{AdvDDH}^{\mathbb{G}}_{\mathcal{B}_{11}}(\lambda).$$

Bounding $\mathcal{A}$'s advantage in Game 11: Denote by $\overline{\sigma}$ the forgery associated to message $\overline{M}$, *i.e.*, $\overline{\sigma} \coloneqq \overline{\sigma}_j$ for $j \in [Q_{\mathsf{frg}}]$ such that $\overline{M} = \mathsf{H}_M(\overline{\mu}_j)$. Also, let us parse $\overline{\sigma} = (\overline{M_\$}, \overline{\pi})$ with $\overline{\pi} = (\overline{\boldsymbol{A}}_{\mathsf{elg}}, \overline{\boldsymbol{A}}_{\mathsf{dh}}, \overline{\gamma}_{\mathsf{elg}}, \overline{\gamma}_{\mathsf{dh}}, \overline{\boldsymbol{z}}_{\mathsf{elg}}, \overline{z}_{\mathsf{dh}})$. Roughly, as $\overline{\pi}$ is an OR-proof for the language $\mathcal{L}_{\mathsf{dh}} \cup \mathcal{L}_{\mathsf{elg}}$ and as $\mathbb{x}_{\mathsf{dh}} \coloneqq (G, \mathbf{D}) \notin \mathcal{L}_{\mathsf{dh}}$ except with probability $1/p$, it must hold that $\overline{\mathbb{x}}_{\mathsf{elg}} \coloneqq (\mathsf{pk}_I, \overline{\mathbf{C}}, \overline{M_\$}) \in \mathcal{L}_{\mathsf{elg}}$ by soundness of π, where $\overline{\mathbf{C}} = \overline{\mathbf{C}_I - \mathbf{C}_M}$ with $\overline{\mathbf{C}_M} = (0, \overline{M})$. Further, as $\overline{M_\$} \neq 0$, the ciphertext $\overline{\mathbf{C}}$ is not an encryption of 0 as discussed in Section 3.1. But as both $\mathbf{C}_I$ and $\overline{\mathbf{C}_M}$ encrypt $\overline{M}$ by construction, $\mathcal{A}$ cannot win except with negligible probability.

In more detail, as $(G, \mathbf{D}) \notin \mathcal{L}_{\mathsf{dh}}$ except with probability $1/p$ and by Lemma 4.2, we have that $\overline{\mathbb{x}}_{\mathsf{elg}} \in \mathcal{L}_{\mathsf{elg}}$ except with probability probability $(1 + Q_{\mathsf{ch}})/p$. Then, by definition of $\mathcal{L}_{\mathsf{elg}}$ (cf. Eq. (3.2)), there exists $\mathbb{w}_{\mathsf{elg}} = (x, y) \in \mathbb{Z}_p^2$ such that $yH = xG$ and $\overline{M_\$} = y\overline{\mathbf{C}}_1 - x\overline{\mathbf{C}}_0$, where $\mathsf{pk}_I = (G, H)$.

Note that $y \neq 0$, as it must hold that $\overline{M_\$} \neq 0$. In more detail, assume for the sake of contradiction that $y = 0$. Then, it holds that $x = 0$, as otherwise $yH \neq xG$. Consequently, $0 = y\overline{\mathbf{C}}_1 - x\overline{\mathbf{C}}_0 = \overline{M_\$}$, which is a contradiction.

Dividing by y, we obtain $H = \mathsf{sk}_I \cdot G$, where $\mathsf{sk}_I = x/y$, and $1/y \cdot \overline{M_\$} = \overline{\mathbf{C}}_1 - \mathsf{sk}_I \cdot \overline{\mathbf{C}}_0$. That is, $\overline{\mathbf{C}}$ encrypts $1/y \cdot \overline{M_\$} \neq 0$. But as by construction, we know that $\mathbf{C}_I$ encrypts $\overline{M}$ (cf. Remark 4.6) and since $\mathbf{C}_M$ also encrypts $\overline{M}$, we have that $\overline{M_\$} = 0$. This is a contradiction. In summary, we obtain

$$\varepsilon_{11} \leq \frac{1 + Q_{\mathsf{ch}}}{p}. \qquad \square$$

Blindness Let us prove blindness of $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$. We follow the common proof technique of making indistinguishable changes to the game until we finally decouple the interaction and the generated signatures completely.

Theorem 4.7. $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ *is blind if DDH is hard in* $\mathbb{G}$. *More precisely, for any adversary* $\mathcal{A}$ *against blindness of* $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$, *there exists an adversary* $\mathcal{B}$ *against* QDDH *such that*

$$\mathsf{AdvBlind}^{\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}}_{\mathcal{A}}(\lambda, Q_{\mathsf{ch}}) \leq 2 \cdot \left(\mathsf{AdvZK}^{\Pi_{\mathsf{M}}}_{\mathcal{A}_{\mathsf{ZK}}}(\lambda) + \mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{A}_{\mathsf{DDH}}}(\lambda) + \frac{2}{p} \cdot Q_{\mathsf{ch}}\right)$$

Proof. We argue by game hops, where we gradually modify the game until the adversary has no information about the secret bit anymore.

Game 0 (Honest). This is the real blindness game, where $\mathcal{A}$ has access to the honest oracles $\mathcal{O}_0, \mathcal{O}_1$ and $b \leftarrow \{0, 1\}$ is the challenge bit. Recall that the adversary chooses a common message I and two messages (μ_0, μ_1) and then interacts with $\mathcal{O}_i(I, \mu_i)$, which run the protocol $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{BUser}(I, \mu_{i \oplus b})$ honestly and eventually outputs $\sigma_{i \oplus b}$. At the end of both interactions, $\mathcal{A}$ learns the signature pair (σ_0, σ_1). (If any $\sigma_i = \bot$, then $\mathcal{A}$ learns $(\bot, \bot)$ instead to avoid trivial attacks.) If the adversary guesses guess b' and $b' = b$, the game outputs 1 (i.e., $\mathcal{A}$ wins), else 0.

In the following games, we modify the behaviour of the oracles $\mathcal{O}_i$. We describe these as modifications to the protocol BUser in Fig. 1, which $\mathcal{O}_i$ executes. Let μ_i be the messages $\mathcal{A}$ inputs to $\mathcal{O}_i$, and let $M_i = \mathsf{H}_M(\mu_i)$ (for $i = 0, 1$).

Game 1 (Simulate π_M). We replace the proofs π_{M} of knowledge for $\mathbf{C}^*_M$ by a simulation. By a direct reduction to zero-knowledge of Π_{M}, we find an adversary $\mathcal{A}_{\mathsf{ZK}}$ such that

$$|\varepsilon_1 - \varepsilon_0| \leq \mathsf{AdvZK}^{\Pi_{\mathsf{M}}}_{\mathcal{A}_{\mathsf{ZK}}}(\lambda).$$

Game 2 (Abort if H_{ch} was queried before $\mathcal{A}$ received (σ_0, σ_1).). When the game queries $(\mathbb{x}_{\mathsf{elg}}, \mathbb{x}_{\mathsf{dh}}, A_{\mathsf{elg}}, A_{\mathsf{dh}})$ to H_{ch} on behalf of $\mathcal{O}_0$ and $\mathcal{O}_1$, abort if H_{ch} was queried on this before (by either $\mathcal{A}$ or the game itself). Due to masking with $\phi_{\mathsf{dh}}(z'_{\mathsf{dh}})$, from $\mathcal{A}$'s view the value A_{dh} is uniformly random in $\mathbb{G}$ (prior to receiving (σ_0, σ_1)). Hence we find

$$|\varepsilon_2 - \varepsilon_1| \leq \frac{2Q_{\mathsf{ch}}}{p}.$$

Game 3 (Program H_{ch}). Sample $\gamma^*, \gamma_{\mathsf{dh}}, \gamma_{\mathsf{elg}}$ uniformly at the start of $\mathcal{O}_0, \mathcal{O}_1$, and program $\mathsf{H}_{\mathsf{ch}}(\mathbb{x}_{\mathsf{dh}}, \mathbb{x}_{\mathsf{elg}}, A_{\mathsf{dh}}, A_{\mathsf{elg}}) \coloneqq \gamma_{\mathsf{dh}} + \gamma_{\mathsf{elg}}$. Due to the abort in Game 2, this change is purely conceptual, and thus

$$\varepsilon_3 = \varepsilon_2.$$

Game 4 (Use SHVZK simulation for $\mathbb{x}_{\mathsf{dh}}$). Next, we can replace the transcript $(\boldsymbol{A}_{\mathsf{dh}}, \gamma_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{dh}})$ by a SHVZK simulation. Indeed, since we program H_{ch} and thus know γ_{dh} beforehand, we can move the whole randomization code of the user to the final step BUser_2. By definition, the user computes its output $(A_{\mathsf{dh}}, \gamma_{\mathsf{dh}}, z_{\mathsf{dh}})$ as a randomization of the transcript $(A^*_{\mathsf{dh}}, \gamma^*_{\mathsf{dh}}, z^*_{\mathsf{dh}})$ (w.r.t. fixed statement $\mathbb{x}_{\mathsf{dh}}$), cf. Lemmas 2.9 and 3.2. By randomizability of transcripts for $\boldsymbol{\Sigma}_{\mathsf{dh}}$, we know that the randomized distribution of $(A_{\mathsf{dh}}, \gamma_{\mathsf{dh}}, z_{\mathsf{dh}})$ coincides with a SHVZK simulation for $\gamma_{\mathsf{dh}} \leftarrow \mathbb{Z}_p$. Overall, we find that

$$\varepsilon_4 = \varepsilon_3.$$

Observe that after this step, we do not use τ^*_{dh} anymore.

Game 5 (Use SHVZK simulation for $\mathbb{x}_{\mathsf{elg}}$ (and $\mathbb{x}_{\mathsf{nez}} = (M_{\$}, \mathbb{x}_{\mathsf{elg}})$)). In this game, we replace the transcript $(M_{\$}, \boldsymbol{A}_{\mathsf{elg}}, \gamma_{\mathsf{elg}}, \boldsymbol{z}_{\mathsf{elg}})$ by a SHVZK simulation. For this, we choose $M_{\$} \leftarrow \mathbb{G}^{\times}$ and let $\tau_{\mathsf{elg}} = (A_{\mathsf{elg}}, \gamma_{\mathsf{elg}}, z_{\mathsf{elg}})$ be computed as a SHVZK simulation of $\boldsymbol{\Sigma}_{\mathsf{elg}}$ for $\mathbb{x}_{\mathsf{elg}} = (\mathsf{pk}_I, \mathbf{C}_I, M_{\$})$.

To see that this change is perfectly indistinguishable, first observe that $M^*_{\$} \neq 0$ (or the user would abort), so that $M_{\$} = \alpha' \cdot M^*_{\$}$ is distributed uniformly in $\mathbb{G}^{\times}$. Hence, the distribution of $M_{\$}$ is unaffected. Second, and analogous to the previous game, we now use that $\boldsymbol{\Sigma}_{\mathsf{elg}}$ is SHVZK and randomizable, and that $\tau_{\mathsf{elg}} = (A_{\mathsf{elg}}, \gamma_{\mathsf{elg}}, z_{\mathsf{elg}})$ is a randomization of $\tau'_{\mathsf{elg}}(A'_{\mathsf{elg}}, \gamma'_{\mathsf{elg}}, z'_{\mathsf{elg}})$ where $\mathbb{x}'_{\mathsf{nez}} = (\mathsf{pk}_I, \mathbf{C}_M)$. Note that by correctness of $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ (cf. Theorem 4.4), if $\tau^*_{\mathsf{elg}} = (A^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}}, z^*_{\mathsf{elg}})$ verifies for $\mathbb{x}^*_{\mathsf{elg}} = (\mathsf{pk}_I, \mathbf{C}^*_M, M_{\$})$, then so does τ'_{elg} for $\mathbb{x}'_{\mathsf{elg}}$. Hence, by randomizability of $\boldsymbol{\Sigma}_{\mathsf{elg}}$, this change is perfectly indistinguishable and we find that

$$\varepsilon_5 = \varepsilon_4.$$

Observe that after this step, we do not use $\tau^*_{\mathsf{nez}} = (M^*_{\$}, \tau^*_{\mathsf{elg}})$ anymore.

Game 6 (Encrypt zero in $\mathbf{C}^*_M$). Finally, we replace the encryption of M in $\mathbf{C}^*$ by an encryption of 0. By a direct reduction to IND-CPA of ElGamal encryption, or equivalently QDDH, we find and adversary $\mathcal{A}_{\mathsf{dh}}$

$$|\varepsilon_6 - \varepsilon_5| \leq \mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{A}_{\mathsf{dh}}}(\lambda).$$

At this point, the game generates the signatures σ_i independently from the interaction with $\mathcal{A}$, and only after all signing interaction with $\mathcal{A}$ completed. Thus, the adversary's distinguishing advantage is 0.

By going these steps backward, we can switch the challenge bit $b = 0$ to $b = 1$ (potentially doubling the adversary's advantage). □

5 Blind Signature in 5 Moves

In this section, we present our 5-move blind signature $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ which achieves one-more *strong* unforgeability and partial blindness. Our construction $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ is obtained by modifying $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$, similar to how BS_2 is obtained from BS_1 in [CATZ24].

5.1 Preparations

We also rely on Π_{M} for relation $\mathcal{R}_{\mathsf{M}}$ from Section 4.1 (cf. Eq. (4.1)). Again, we assume that Π_{M} is straightline $\widetilde{\mathcal{R}}_{\mathsf{M}}$-extractable (cf. Eq. (4.2)).

5.2 Construction

Let $\mathsf{H}_{\mathsf{crs}}, \mathsf{H}_{\mathsf{par}}, \mathsf{H}_M, \mathsf{H}_{\mathsf{ch}}$ be defined as in Section 4.2. $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ is given below.

$\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$: Our 5-move pairing-free blind signature

- $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{KeyGen}(1^\lambda)$: Output $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{KeyGen}(1^\lambda)$.
- $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{BSign}(\mathsf{sk}, I) \rightleftarrows \mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{BUser}(\mathsf{vk}, m, I)$: Proceeds in 5 moves. We sketch the protocol below.
  - **Signer 1.** In $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{BSign}_1$, the signer sets $(\boldsymbol{A}^*_{\mathsf{dh}}, \mathsf{st}_{\mathsf{dh}}) \leftarrow \mathsf{Init}_{\mathsf{dh}}(\mathbb{x}_{\mathsf{dh}}, \mathsf{sk})$ for $\mathbb{x}_{\mathsf{dh}} = (G, \mathbf{D})$. The signer sends $\boldsymbol{A}^*_{\mathsf{dh}}$ to the user.
  - **User 1.** In $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{BUser}_1$, the user proceeds as in $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{BUser}_1$ except that sets $M := \mathsf{H}_M(\mu, \boldsymbol{A}_{\mathsf{dh}})$, where $\boldsymbol{A}^*_{\mathsf{dh}}$ is blinded to $\boldsymbol{A}_{\mathsf{dh}} := \boldsymbol{A}^*_{\mathsf{dh}} + \Phi_{\mathsf{dh}}(z'_{\mathsf{dh}}) - \gamma'_{\mathsf{dh}}\mathbf{D}$ for $z'_{\mathsf{dh}} \leftarrow \mathbb{Z}_p$, $\gamma'_{\mathsf{dh}} \leftarrow \mathbb{Z}_p$.
  - **Signer 2.** In $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{BSign}_2$, the signer proceeds as in $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{BSign}_1$ except that it reuses $(\boldsymbol{A}^*_{\mathsf{dh}}, \mathsf{st}_{\mathsf{dh}})$ from $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{BSign}_1$.
  - **User 2.** In $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{BUser}_2$, the user proceeds as in $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{BUser}_2$ except it reuses the blinded $\boldsymbol{A}_{\mathsf{dh}}$ from $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{BUser}_1$.
  - **Signer 3.** In $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{BSign}_3$, the singer proceeds as in $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{BSign}_2$.
  - **User 3.** In $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{BUser}_3$, the user proceeds in $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}.\mathsf{BUser}_3$.

As a $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ signing session is similar to a $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ signing session (cf. Section 4.2), we defer a formal description to Appendix D.

– $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{Verify}(\mathsf{vk}, \mu, I, \sigma)$:
  (1) Parse σ as $\sigma = (M_{\$}, \boldsymbol{A}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{dh}}, \gamma_{\mathsf{elg}}, \gamma_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{elg}}, z_{\mathsf{dh}})$.
  (2) Output $b \leftarrow \mathsf{S}_{\mathsf{neq}}.\mathsf{Verify}(\mathsf{vk}, M, I, \sigma)$ for $M \coloneqq \mathsf{H}_M(\mu, \boldsymbol{A}_{\mathsf{dh}})$.

Remark 5.1 (Partial blindness). Note that user and signer can agree on a common message I in the second move (*i.e.*, BUser_1) as the first move (*i.e.*, BSign_1) is independent from I. Our security analysis covers this variant.

Remark 5.2 (Optimizations). Almost all optimization for $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ also apply to $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ (cf. Remark 4.1), except for the standard optimization of only sending A_{dh}. In the protocol, A_{dh} is required to derive $\gamma = \mathsf{H}_{\mathsf{ch}}(\mu, A_{\mathsf{dh}})$. However, we can replace $A_{\mathsf{dh}} \in \mathbb{G}^2$ by $h_{\mathsf{dh}} = \mathsf{H}(A_{\mathsf{dh}})$ for another random oracle H in this derivation. Now, only one hash value $h_{\mathsf{dh}} \in \mathbb{H}$ instead of $A_{\mathsf{dh}} \in \mathbb{G}^2$ is needed in the signature.

5.3 Security Analysis

We prove correctness, one-more unforgeability and partial blindness for $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$. Before we move towards the analysis, let us provide some useful lemmata.

Preparations Before we proceed with the proof, we provide a useful lemma. The proof is given in Appendix E.5 and based on special soundness.

Lemma 5.3 (Unique Σ-protocol Challenges). *Let $\mathsf{pk} = (G, H)$ with $H \in \mathbb{G}$ and let $t \in \mathbb{Z}_p$. Let $\mathbb{x}_{\mathsf{elg}} = (\mathsf{pk}, \mathbf{C}, M_{\$})$ be a statement with $M_{\$} \in \mathbb{G}^{\times}$ and $\mathbf{C} = t \cdot \mathsf{pk}$. Then there exists at most one $\gamma_{\mathsf{elg}} \in \mathbb{Z}_p$ such that there exists a response $\boldsymbol{z}_{\mathsf{elg}}$ with $\boldsymbol{\Sigma}_{\mathsf{dh}}.\mathsf{Verify}(\mathbb{x}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{elg}}, \gamma_{\mathsf{elg}}, \boldsymbol{z}_{\mathsf{elg}}) = 1$.*

Correctness As the proof of correctness for $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ is almost identical to $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ (cf. Theorem 4.4), we omit details.

Theorem 5.4 (Correctness). $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ *is perfectly correct.*

One-more Strong Unforgeability We provide a high-level overview of the proof of OMSUF; a formal proof is in Appendix E.6. Let $(\overline{\mu}_j, \overline{\sigma}_j)_{j \in [Q_{\mathsf{frg}}]}$ be the forgeries output by $\mathcal{A}$ at the end of the game. In our analysis, we consider three types of adversaries, depending on the forgeries:

– Type (I) adversaries succeed only if they output distinct message-commitment pairs $(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})$. As now all $\overline{M}_j = \mathsf{H}_M(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})$ are pairwise-distinct, this is essentially attack standard OMUF. For this type of adversary, the proof for Theorem 4.5 (OMUF for $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$) is easily adapted.

– Type (II) adversaries succeed only if they reuse at least one $\boldsymbol{A}_{\mathsf{dh}}$ in the signature $\overline{\sigma}_i$, but do *not* reuse the corresponding γ_{dh}. In this case, we can appeal to special soundness of $\boldsymbol{\Sigma}_{\mathsf{dh}}$ to break DDH (in fact, extract a DLog). For this, we switch signing to the $\boldsymbol{\Sigma}_{\mathsf{elg}}$-branch, similar to Type (I) adversaries, with one difference: we do not puncture a message. That is, we proceed as in game 6 of the proof of Theorem 4.5, but we ensure that M_I differs from all extracted messages M in all sessions. Thus, we can always simulate the signing oracle without $\mathsf{sk} = d_2$, and as a Type (I) adversary outputs $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcripts with reused $\boldsymbol{A}_{\mathsf{dh}}$ but distinct challenges, we can extract the DLog of D_2 by special soundness.
– Type (III) adversaries succeed only if they reuse a complete transcript $\tau_{\mathsf{dh}} = (\boldsymbol{A}_{\mathsf{dh}}, \gamma_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{dh}})$ (and also $\overline{\mu}$) in the forgeries. In this case, we cannot appeal to 2-special soundness of $\boldsymbol{\Sigma}_{\mathsf{dh}}$. Thus, we turn to puncturing again: we guess the message $\overline{\mu}$ for which the transcript τ_{dh} is reused, and puncture the parameter $\mathbf{C}_I$ as in Type (I). As $\mathbf{C}_I$ is punctured and $\boldsymbol{\Sigma}_{\mathsf{elg}}$ (or more precisely, $\boldsymbol{\Sigma}_{\mathsf{nez}}$) is 2-special sound, there is at most one challenge $\Gamma(M_{\$}, \boldsymbol{A}_{\mathsf{elg}})$ which has an accepting z_{elg} (cf. Lemma 5.3). However, since a Type (III) adversary must reuse τ_{dh} to succeed, it can only succeed by guessing $\Gamma(M_{\$}, \boldsymbol{A}_{\mathsf{elg}})$, which succeeds only with probability $1/p$ per try.[8]

Note that the Types (I) to (III) cover all possible cases by the unique response property of $\boldsymbol{\Sigma}_{\mathsf{dh}}$ (cf. Definition 2.8) which can occur if an adversary succeeds in OMSUF. Consequently, by reducing each case to DDH (or DLog), we have shown OMSUF. The formal proof contains concrete bounds (cf. Appendix E.6).

Theorem 5.5. $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ *is OMSUF under relaxed knowledge soundness for* $\widetilde{\mathcal{R}}_{\mathsf{M}}$ *of* Π_{M} *and the* DDH *assumption.*

Blindness The blindness of $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ follows by exactly the same argument as our proof for $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$. Except that rerandomization of $\boldsymbol{\Sigma}_{\mathsf{dh}}$ and $\boldsymbol{\Sigma}_{\mathsf{elg}}$ occur in different rounds now, all steps of the proof of Theorem 4.7 apply verbatim. Thus, following theorem is an immediate corollary.

Theorem 5.6. $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ *is blind if DDH is hard in* $\mathbb{G}$*. The precise bound on the adversary's advantage carries over from Theorem 4.7.*

6 Achieving Statistical Blindness

Our protocols $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ and $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ are only computationally blind, because the user sends an ElGamal encryption of his message to the signer. Next, we briefly sketch how to achieve statistical blindness. A more detailed discussion is in Appendix C.

[8] This is formally imprecise, as the adversary has some freedom to choose γ_{elg} due to the OR-construction. However, it is true if γ_{elg} is fixed, e.g., by guessing the first occurrence in advance. Then because of the constraint $\gamma_{\mathsf{dh}} + \gamma_{\mathsf{elg}} = \mathsf{H}_{\mathsf{ch}}(A_{\mathsf{dh}}, A_{\mathsf{elg}})$, we get $\Gamma(M_{\$}, \boldsymbol{A}_{\mathsf{elg}}) \stackrel{!}{=} \gamma_{\mathsf{elg}} = \mathsf{H}_{\mathsf{ch}}(\mathbb{x}_{\mathsf{dh}}, \mathbb{x}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{dh}}, \boldsymbol{A}_{\mathsf{elg}}) - \gamma_{\mathsf{dh}}$.

The only obstruction to statistical blindness is that the user sends an *encryption* of the (hash of) the to-be-signed message $M = \mathsf{H}_M(\mu)$, and that the NIZK also needs to be statistically zero-knowledge. By replacing the encryption with dual-mode encryption, e.g., based on the dual-mode commitments used in Groth–Sahai proofs [GS08; GS12], this can be achieved. For uniformly random public keys, they are statistically hiding with overwhelming probability. Thus, we are able to prove statistical blindness, i.e., all game hops in the blindness proof are statistical. In the one-more (strong) unforgeability proofs, we merely need to add a step where we switch the public key to perfectly binding and efficiently extractable. The remaining proof steps are unchanged.

As the randomness (which is part of the witness) for dual-mode encryption based on DDH is $2\mathbb{Z}_p$ elements instead of one, the signature grows by one $\mathbb{Z}_p$ element and communication grows by $2\mathbb{Z}_p$ elements (due to the additional NIZK).

References

[Abe01] Masayuki Abe. "A Secure Three-Move Blind Signature Scheme for Polynomially Many Signatures". In: 2001, pp. 136–151. DOI: 10.1007/3-540-44987-6_9.

[Abe+10] Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. "Structure-Preserving Signatures and Commitments to Group Elements". In: 2010, pp. 209–236. DOI: 10.1007/978-3-642-14623-7_12.

[Abe+17] Masayuki Abe, Dennis Hofheinz, Ryo Nishimaki, Miyako Ohkubo, and Jiaxin Pan. "Compact Structure-Preserving Signatures with Almost Tight Security". In: 2017, pp. 548–580. DOI: 10.1007/978-3-319-63715-0_19.

[Abe+18] Masayuki Abe, Charanjit S. Jutla, Miyako Ohkubo, and Arnab Roy. "Improved (Almost) Tightly-Secure Simulation-Sound QA-NIZK with Applications". In: 2018, pp. 627–656. DOI: 10.1007/978-3-030-03326-2_21.

[Abe+23] Masayuki Abe, Dennis Hofheinz, Ryo Nishimaki, Miyako Ohkubo, and Jiaxin Pan. "Compact Structure-Preserving Signatures with Almost Tight Security". In: 36.4 (Oct. 2023), p. 37. DOI: 10.1007/s00145-023-09477-z.

[AEB20] Nabil Alkeilani Alkadri, Rachid El Bansarkhani, and Johannes Buchmann. "On Lattice-Based Interactive Protocols: An Approach with Less or No Aborts". In: 2020, pp. 41–61. DOI: 10.1007/978-3-030-55304-3_3.

[AF96] Masayuki Abe and Eiichiro Fujisaki. "How to Date Blind Signatures". In: 1996, pp. 244–251. DOI: 10.1007/BFb0034851.

[Agr+22] Shweta Agrawal, Elena Kirshanova, Damien Stehlé, and Anshu Yadav. "Practical, Round-Optimal Lattice-Based Blind Signatures". In: 2022, pp. 39–53. DOI: 10.1145/3548606.3560650.

[AHJ21] Nabil Alkeilani Alkadri, Patrick Harasser, and Christian Janson. "BlindOR: an Efficient Lattice-Based Blind Signature Scheme from OR-Proofs". In: 2021, pp. 95–115. DOI: 10.1007/978-3-030-92548-2_6.

[AO00] Masayuki Abe and Tatsuaki Okamoto. "Provably Secure Partially Blind Signatures". In: 2000, pp. 271–286. DOI: 10.1007/3-540-44598-6_17.

[BB04] Dan Boneh and Xavier Boyen. "Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles". In: 2004, pp. 223–238. DOI: 10.1007/978-3-540-24676-3_14.

[BB08] Dan Boneh and Xavier Boyen. "Short Signatures Without Random Oracles and the SDH Assumption in Bilinear Groups". In: 21.2 (Apr. 2008), pp. 149–177. DOI: 10.1007/s00145-007-9005-7.

[BCC04] Ernest F. Brickell, Jan Camenisch, and Liqun Chen. "Direct Anonymous Attestation". In: 2004, pp. 132–145. DOI: 10.1145/1030083.1030103.

[Bel+03] Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. "The One-More-RSA-Inversion Problems and the Security of Chaum's Blind Signature Scheme". In: *J. Cryptol.* 16.3 (2003), pp. 185–215. DOI: 10.1007/s00145-002-0120-1. URL: https://doi.org/10.1007/s00145-002-0120-1.

[Ben+21] Fabrice Benhamouda, Tancrède Lepoint, Julian Loss, Michele Orrù, and Mariana Raykova. "On the (in)security of ROS". In: 2021, pp. 33–53. DOI: 10.1007/978-3-030-77870-5_2.

[Ben+22] Fabrice Benhamouda, Tancrède Lepoint, Julian Loss, Michele Orrù, and Mariana Raykova. "On the (in)Security of ROS". In: 35.4 (Oct. 2022), p. 25. DOI: 10.1007/s00145-022-09436-0.

[Beu+23] Ward Beullens, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. "Lattice-Based Blind Signatures: Short, Efficient, and Round-Optimal". In: 2023, pp. 16–29. DOI: 10.1145/3576915.3616613.

[BL13] Foteini Baldimtsi and Anna Lysyanskaya. "Anonymous credentials light". In: 2013, pp. 1087–1098. DOI: 10.1145/2508859.2516687.

[Bla+13] Olivier Blazy, Georg Fuchsbauer, David Pointcheval, and Damien Vergnaud. "Short blind signatures". In: *Journal of computer security* 21.5 (2013), pp. 627–661.

[BLS04] Dan Boneh, Ben Lynn, and Hovav Shacham. "Short Signatures from the Weil Pairing". In: 17.4 (Sept. 2004), pp. 297–319. DOI: 10.1007/s00145-004-0314-9.

[Bol03] Alexandra Boldyreva. "Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme". In: 2003, pp. 31–46. DOI: 10.1007/3-540-36288-6_3.

[BR93] Mihir Bellare and Phillip Rogaway. "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols". In: 1993, pp. 62–73. DOI: 10.1145/168588.168596.

[Bra+24] Nicolas Brandt, Dennis Hofheinz, Michael Klooß, and Michael Reichle. "Tightly-Secure Blind Signatures in Pairing-Free Groups". In: (2024). URL: https://eprint.iacr.org/2024/XXX.

[Bra94] Stefan Brands. "Untraceable Off-line Cash in Wallets with Observers (Extended Abstract)". In: 1994, pp. 302–318. DOI: 10.1007/3-540-48329-2_26.

[Bus+22] Maxime Buser et al. "A Survey on Exotic Signatures for Post-Quantum Blockchain: Challenges & Research Directions". In: *ACM Comput. Surv.* (2022). Just accepted. ISSN: 0360-0300. DOI: 10.1145/3572771. URL: https://doi.org/10.1145/3572771.

[CA+22] Rutchathon Chairattana-Apirom, Lucjan Hanzlik, Julian Loss, Anna Lysyanskaya, and Benedikt Wagner. "PI-Cut-Choo and Friends: Compact Blind Signatures via Parallel Instance Cut-and-Choose and More". In: 2022, pp. 3–31. DOI: 10.1007/978-3-031-15982-4_1.

[CATZ24] Rutchathon Chairattana-Apirom, Stefano Tessaro, and Chenzhi Zhu. "Pairing-Free Blind Signatures from CDH Assumptions". In: 2024, pp. 174–209. DOI: 10.1007/978-3-031-68376-3_6.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. "Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols". In: 1994, pp. 174–187. DOI: 10.1007/3-540-48658-5_19.

[Cha82] David Chaum. "Blind Signatures for Untraceable Payments". In: 1982, pp. 199–203. DOI: 10.1007/978-1-4757-0602-4_18.

[Cha88] David Chaum. "Elections with Unconditionally-Secret Ballots and Disruption Equivalent to Breaking RSA". In: 1988, pp. 177–182. DOI: 10.1007/3-540-45961-8_15.

[CL01] Jan Camenisch and Anna Lysyanskaya. "An Efficient System for Non-transferable Anonymous Credentials with Optional Anonymity Revocation". In: 2001, pp. 93–118. DOI: 10.1007/3-540-44987-6_7.

[Cri+23] Elizabeth C. Crites, Chelsea Komlo, Mary Maller, Stefano Tessaro, and Chenzhi Zhu. "Snowblind: A Threshold Blind Signature in Pairing-Free Groups". In: 2023, pp. 710–742. DOI: 10.1007/978-3-031-38557-5_23.

[DHP24] Khue Do, Lucjan Hanzlik, and Eugenio Paracucchi. "M&M'S: Mix and Match Attacks on Schnorr-Type Blind Signatures with Repetition". In: 2024, pp. 363–387. DOI: 10.1007/978-3-031-58751-1_13.

[ElG85] Taher ElGamal. "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms". In: 31.4 (1985), pp. 469–472. DOI: 10.1109/TIT.1985.1057074.

[Esc+13] Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. "An Algebraic Framework for Diffie-Hellman Assumptions". In: 2013, pp. 129–147. DOI: 10.1007/978-3-642-40084-1_8.

[FHS15] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. "Practical Round-Optimal Blind Signatures in the Standard Model". In: 2015, pp. 233–253. DOI: 10.1007/978-3-662-48000-7_12.

[Fis06] Marc Fischlin. “Round-Optimal Composable Blind Signatures in the Common Reference String Model”. In: 2006, pp. 60–77. DOI: 10.1007/11818175_4.

[FKL18] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. “The Algebraic Group Model and its Applications”. In: 2018, pp. 33–62. DOI: 10.1007/978-3-319-96881-0_2.

[FOO92] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. “A practical secret voting scheme for large scale elections”. In: *AUSCRYPT*. Springer. 1992, pp. 244–251.

[FPS20] Georg Fuchsbauer, Antoine Plouviez, and Yannick Seurin. “Blind Schnorr Signatures and Signed ElGamal Encryption in the Algebraic Group Model”. In: 2020, pp. 63–95. DOI: 10.1007/978-3-030-45724-2_3.

[FS10] Marc Fischlin and Dominique Schröder. “On the Impossibility of Three-Move Blind Signature Schemes”. In: 2010, pp. 197–215. DOI: 10.1007/978-3-642-13190-5_10.

[Fuc+16] Georg Fuchsbauer, Christian Hanser, Chethan Kamath, and Daniel Slamanig. “Practical Round-Optimal Blind Signatures in the Standard Model from Weaker Assumptions”. In: 2016, pp. 391–408. DOI: 10.1007/978-3-319-44618-9_21.

[FW24] Georg Fuchsbauer and Mathias Wolf. “Concurrently Secure Blind Schnorr Signatures”. In: 2024, pp. 124–160. DOI: 10.1007/978-3-031-58723-8_5.

[Gar+11] Sanjam Garg, Vanishree Rao, Amit Sahai, Dominique Schröder, and Dominique Unruh. “Round Optimal Blind Signatures”. In: 2011, pp. 630–648. DOI: 10.1007/978-3-642-22792-9_36.

[GG14] Sanjam Garg and Divya Gupta. “Efficient Round Optimal Blind Signatures”. In: 2014, pp. 477–495. DOI: 10.1007/978-3-642-55220-5_27.

[Gha17] Essam Ghadafi. “Efficient Round-Optimal Blind Signatures in the Standard Model”. In: 2017, pp. 455–473. DOI: 10.1007/978-3-319-70972-7_26.

[GS08] Jens Groth and Amit Sahai. “Efficient Non-interactive Proof Systems for Bilinear Groups”. In: 2008, pp. 415–432. DOI: 10.1007/978-3-540-78967-3_24.

[GS12] Jens Groth and Amit Sahai. “Efficient Noninteractive Proof Systems for Bilinear Groups”. In: *SIAM J. Comput.* 41.5 (2012), pp. 1193–1232. DOI: 10.1137/080725386. URL: https://doi.org/10.1137/080725386.

[Hau+20] Eduard Hauck, Eike Kiltz, Julian Loss, and Ngoc Khanh Nguyen. “Lattice-Based Blind Signatures, Revisited”. In: 2020, pp. 500–529. DOI: 10.1007/978-3-030-56880-1_18.

[Hen+22] Scott Hendrickson, Jana Iyengar, Tommy Pauly, Steven Valdez, and Christopher A. Wood. *Private Access Tokens. Internet-Draft*

draft-private-access-tokens-01. Work in Progress. 2022. URL: https://datatracker.ietf.org/doc/draft-private-access-tokens/.

[HKL19] Eduard Hauck, Eike Kiltz, and Julian Loss. "A Modular Treatment of Blind Signatures from Identification Schemes". In: 2019, pp. 345–375. DOI: 10.1007/978-3-030-17659-4_12.

[HLW23] Lucjan Hanzlik, Julian Loss, and Benedikt Wagner. "Rai-Choo! Evolving Blind Signatures to the Next Level". In: 2023, pp. 753–783. DOI: 10.1007/978-3-031-30589-4_26.

[JLO97] Ari Juels, Michael Luby, and Rafail Ostrovsky. "Security of Blind Digital Signatures (Extended Abstract)". In: 1997, pp. 150–164. DOI: 10.1007/BFb0052233.

[Kat21] Shuichi Katsumata. "A New Simple Technique to Bootstrap Various Lattice Zero-Knowledge Proofs to QROM Secure NIZKs". In: 2021, pp. 580–610. DOI: 10.1007/978-3-030-84245-1_20.

[Kat+21] Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. "Round-Optimal Blind Signatures in the Plain Model from Classical and Quantum Standard Assumptions". In: 2021, pp. 404–434. DOI: 10.1007/978-3-030-77870-5_15.

[Kat+23] Shuichi Katsumata, Yi-Fu Lai, Jason T. LeGrow, and Ling Qin. "CSI-Otter: Isogeny-Based (Partially) Blind Signatures from the Class Group Action with a Twist". In: 2023, pp. 729–761. DOI: 10.1007/978-3-031-38548-3_24.

[KLN23] Markulf Kohlweiss, Anna Lysyanskaya, and An Nguyen. "Privacy-Preserving Blueprints". In: 2023, pp. 594–625. DOI: 10.1007/978-3-031-30617-4_20.

[KLR21] Jonathan Katz, Julian Loss, and Michael Rosenberg. "Boosting the Security of Blind Signature Schemes". In: 2021, pp. 468–492. DOI: 10.1007/978-3-030-92068-5_16.

[KLR24] Shuichi Katsumata, Yi-Fu Lai, and Michael Reichle. "Breaking Parallel ROS: Implication for Isogeny and Lattice-Based Blind Signatures". In: 2024, pp. 319–351. DOI: 10.1007/978-3-031-57718-5_11.

[KLX22] Julia Kastner, Julian Loss, and Jiayu Xu. "On Pairing-Free Blind Signature Schemes in the Algebraic Group Model". In: 2022, pp. 468–497. DOI: 10.1007/978-3-030-97131-1_16.

[KNR24] Julia Kastner, Ky Nguyen, and Michael Reichle. "Pairing-Free Blind Signatures from Standard Assumptions in the ROM". In: 2024, pp. 210–245. DOI: 10.1007/978-3-031-68376-3_7.

[KRS23] Shuichi Katsumata, Michael Reichle, and Yusuke Sakai. "Practical Round-Optimal Blind Signatures in the ROM from Standard Assumptions". In: 2023, pp. 383–417. DOI: 10.1007/978-981-99-8724-5_12.

[KRW24] Michael Klooß, Michael Reichle, and Benedikt Wagner. "Practical Blind Signatures in Pairing-Free Groups". In: *To Appear in ASIACRYPT 2024. Available at ia.cr/2024/1378* (2024).

[Ks22] Yashvanth Kondi and abhi shelat. "Improved Straight-Line Extraction in the Random Oracle Model with Applications to Signature Aggregation". In: 2022, pp. 279–309. DOI: 10.1007/978-3-031-22966-4_10.

[KSD19] Mojtaba Khalili, Daniel Slamanig, and Mohammad Dakhilalian. "Structure-Preserving Signatures on Equivalence Classes from Standard Assumptions". In: 2019, pp. 63–93. DOI: 10.1007/978-3-030-34618-8_3.

[Lin08] Yehuda Lindell. "Lower Bounds and Impossibility Results for Concurrent Self Composition". In: 21.2 (Apr. 2008), pp. 200–249. DOI: 10.1007/s00145-007-9015-5.

[Mau15] Ueli Maurer. "Zero-knowledge proofs of knowledge for group homomorphisms". In: 77.2-3 (2015), pp. 663–676. DOI: 10.1007/s10623-015-0103-5.

[MSF10] Sarah Meiklejohn, Hovav Shacham, and David Mandell Freeman. "Limitations on Transformations from Composite-Order to Prime-Order Groups: The Case of Round-Optimal Blind Signatures". In: 2010, pp. 519–538. DOI: 10.1007/978-3-642-17373-8_30.

[Oka93] Tatsuaki Okamoto. "Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes". In: 1993, pp. 31–53. DOI: 10.1007/3-540-48071-4_3.

[Pas03] Rafael Pass. "On Deniability in the Common Reference String and Random Oracle Model". In: 2003, pp. 316–337. DOI: 10.1007/978-3-540-45146-4_19.

[Pas11] Rafael Pass. "Limits of provable security from standard assumptions". In: 2011, pp. 109–118. DOI: 10.1145/1993636.1993652.

[PK22] Rafaël del Pino and Shuichi Katsumata. "A New Framework for More Efficient Round-Optimal Lattice-Based (Partially) Blind Signature via Trapdoor Sampling". In: 2022, pp. 306–336. DOI: 10.1007/978-3-031-15979-4_11.

[Poi98] David Pointcheval. "Strengthened Security for Blind Signatures". In: 1998, pp. 391–405. DOI: 10.1007/BFb0054141.

[PS00] David Pointcheval and Jacques Stern. "Security Arguments for Digital Signatures and Blind Signatures". In: 13.3 (June 2000), pp. 361–396. DOI: 10.1007/s001450010003.

[PS97] David Pointcheval and Jacques Stern. "New Blind Signatures Equivalent to Factorization (Extended Abstract)". In: 1997, pp. 92–99. DOI: 10.1145/266420.266440.

[SC12] Jae Hong Seo and Jung Hee Cheon. "Beyond the Limitation of Prime-Order Bilinear Groups, and Round Optimal Blind Signatures". In: 2012, pp. 133–150. DOI: 10.1007/978-3-642-28914-9_8.

[Sch90] Claus-Peter Schnorr. "Efficient Identification and Signatures for Smart Cards". In: 1990, pp. 239–252. DOI: 10.1007/0-387-34805-0_22.

[Sch91] Claus-Peter Schnorr. "Efficient Signature Generation by Smart Cards". In: 4.3 (Jan. 1991), pp. 161–174. DOI: 10.1007/BF00196725.

[TZ22] Stefano Tessaro and Chenzhi Zhu. "Short Pairing-Free Blind Signatures with Exponential Security". In: 2022, pp. 782–811. DOI: 10.1007/978-3-031-07085-3_27.

[YL19] Xun Yi and Kwok-Yan Lam. "A New Blind ECDSA Scheme for Bitcoin Transaction Anonymity". In: 2019, pp. 613–620. DOI: 10.1145/3321705.3329816.

A Related Work

We give an overview of blind signature literature to complement our selective overview in Section 1.

Restricted Concurrency. Early constructions of blind signatures in the ROM [AO00; Oka93; PS00; Sch90] are based on Schnorr-style Σ-protocols. These constructions are efficient 3-move schemes in pairing-free groups, but proven secure for at most $\mathsf{polylog}$-many signing sessions [HKL19; KLX22; PS00]. The technique was applied in other settings, *e.g.*, hidden-order groups [PS97], lattices [AEB20; AHJ21; Hau+20], and isogenies [Kat+23], but the schemes inherit the polylog upper bound. This $\mathsf{polylog}$ upper bound on concurrent signing sessions is tight [Ben+21; DHP24; KLR24]: With the exception of [Hau+20], the aforementioned schemes are broken if $O(\lambda)$-many concurrent signing session are allowed.

Generic Group Model and ROM. In pairing-free generic groups and the ROM, there are 3-move blind signatures [Abe01; Cri+23; FPS20; KLX22; TZ22] that avoid the attack given in [Ben+21]. These constructions are practical, and notably [KLX22; TZ22] provide full concurrent security. The security argument in the above works relies on the algebraic group model (AGM) [FKL18] and random oracles. In the ROM, [BL13] shows security of [Abe01] without generic groups, albeit with limited concurrency.

Boosting Transforms. A line of work [CA+22; HLW23; KLR21] based on [Poi98] provide boosting transformations for blind signatures with limited concurrent security. Concrete constructions are given in the hidden-order or pairing setting. While [CA+22; KLR21] gives a generic framework, the resulting schemes are impractical.

Trusted Setup. In the pairing setting, there are several blind signatures with trusted setup [Abe+18; Bla+13; KSD19; MSF10; SC12] in the standard model. The setup assumption can be removed in [Abe+18; Bla+13] in the ROM.

Complexity Leveraging. There are blind signatures [Gar+11; GG14; Kat+21] that circumvent impossibility results to construct round-optimal blind signatures [FS10; Lin08; Pas11] in the standard model via complexity leveraging or by relying on both classical and post-quantum assumptions.

Interactive assumptions. There are several blind signatures [Abe+10; Agr+22; Bel+03; Beu+23; Bol03; Cha82] secure under interactive or q-type assumptions and in the ROM. Also, there are constructions in the standard model [Abe+10;

FHS15; Fuc+16; FW24; Gha17] which rely on tailored interactive hardness assumptions. Their security is based on strong interactive assumptions. Further, [Abe+10] first notices that Fischlin blind signatures can be instantiated by combining sufficiently algebraic signatures and GOS proofs.

Generic Frameworks. Juels, Luby, and Ostrovsky [JLO97] show that blind signatures can be constructed from generic MPC and one-way trapdoor permutations. Also, Fischlin [Fis06] gives a generic framework based on signatures, NIZKs and commitments. The latter framework can be instantiated efficiently in pairings, lattices and hidden order groups [**C:delKat22**; KNR24; KRS23] under standard assumption in the ROM.

B Omitted Preliminaries

B.1 Non-Interactive Proof Systems

Definition B.1 ((Perfect) Correctness). *Let* $\Pi = (\mathsf{Prove}, \mathsf{Verify})$ *be a non-interactive proof system for a relation* $\mathcal{R}$*. It is* perfectly correct *if for all* $(\mathbb{x}, \mathbb{w}) \in \mathcal{R}$ *it holds that*

$$\Pr[\mathsf{Verify}^{\mathsf{H}}(\mathsf{crs}, \mathbb{x}, \pi) = 1 \mid \pi \leftarrow \mathsf{Prove}^{\mathsf{H}}(\mathsf{crs}, \mathbb{x}, \mathbb{w})] = 1$$

where the probability is over the choice of H*,* crs*, and the randomness of* $\mathsf{Prove}, \mathsf{Verify}$*.*

Note that our definitions of zero-knowledge simulator and knowledge extractor are independent of an adversary, in particular, they are straightline by definition.

Definition B.2 (Zero-Knowledge). *Let* $\Pi = (\mathsf{Prove}, \mathsf{Verify})$ *be a non-interactive proof system for a relation* $\mathcal{R}$*. Let* Sim *be a PPT algorithm. Let* $\mathcal{A}$ *be an algorithm and let*

$$\mathsf{Real}^{\Pi}_{\mathcal{A}}(\lambda) := \Pr[b = 1 \mid b \leftarrow \mathcal{A}^{\mathsf{H}, \mathcal{O}_{\mathsf{Prove}}}(1^{\lambda}, \mathsf{crs})]$$
$$\mathsf{Ideal}^{\Pi}_{\mathcal{A}, \mathsf{Sim}}(\lambda) := \Pr[b = 1 \mid b \leftarrow \mathcal{A}^{\mathsf{H}, \mathcal{O}_{\mathsf{Sim}}}(1^{\lambda}, \mathsf{crs})]$$

Here, $\mathcal{A}$ *has (black-box) access to the random oracle* H *and to an oracle* $\mathcal{O}_{\mathsf{Prove}}$ *or* $\mathcal{O}_{\mathsf{Sim}}$*, which are as follows:*

- $\mathcal{O}_{\mathsf{Prove}}(\mathbb{x}, \mathbb{w})$*: Return* $\perp$ *if* $(\mathbb{x}, \mathbb{w}) \notin \mathcal{R}$*. Else, output* $\pi \leftarrow \mathsf{Prove}^{\mathsf{H}}(\mathsf{crs}, \mathbb{x}, \mathbb{w})$*.*
- $\mathcal{O}_{\mathsf{Sim}}(\mathbb{x}, \mathbb{w})$*: Return* $\perp$ *if* $(\mathbb{x}, \mathbb{w}) \notin \mathcal{R}$*. Else, output* $\pi \leftarrow \mathsf{Sim}^{\mathsf{H}}(\mathbb{x})$*.*

Furthermore, the simulator Sim *chooses* crs *and can program the random oracle* H *on any fresh input, i.e. if* $\mathsf{H}(m)$ *has not been queried before, then* Sim *is free to choose* $\mathsf{H}(m)$*, else, programming fails. We define the advantage of* $\mathcal{A}$ *against* Π *and* Sim *by* $\mathsf{AdvZK}^{\Pi, \mathsf{Sim}}_{\mathcal{A}}(\lambda) = |\mathsf{Real}^{\Pi}_{\mathcal{A}}(\lambda) - \mathsf{Ideal}^{\Pi}_{\mathcal{A}, \mathsf{Sim}}(\lambda)|$*. We call* Sim *a (straightline)* zero-knowledge simulator *for* Π*, if for any PPT* $\mathcal{A}$ *the advantage* $\mathsf{AdvZK}^{\Pi}_{\mathcal{A}}$ *is negligible. We say that* Π *is* zero-knowledge *if there is a zero-knowledge simulator for* Π*.*

For knowledge soundness, the extractor must compute a witness from an accepting proof, a potential crs trapdoor td, and all adversarial random oracle queries. In particular, extraction is straightline. We consider relaxed knowledge soundness, which means that the *knowledge relation* $\widetilde{\mathcal{R}}$ differs from the correctness relation $\mathcal{R}$.

Definition B.3 (Relaxed Knowledge Soundness). *Let* $\Pi = (\mathsf{Prove}, \mathsf{Verify})$ *be a non-interactive proof system for a relation* $\mathcal{R}$ *and let* $\widetilde{\mathcal{R}}$ *be an NP-relation. Let* $(\mathsf{ExtSetup}, \mathsf{Ext})$ *be a PPT algorithms. Let* $\mathcal{A}$ *be an oracle algorithm and let*

$$\mathsf{Real}_{\mathcal{A}}(\lambda) := \Pr[b = 1 \mid \mathsf{crs} \leftarrow \{0,1\}^{\ell(\lambda)};\ b \leftarrow \mathcal{A}^{\mathsf{H},\mathcal{O}_{\mathsf{Verify}}}(1^\lambda, \mathsf{crs}) = 1]$$
$$\mathsf{Ideal}_{\mathcal{A}}(\lambda) := \Pr[b = 1 \mid (\mathsf{crs}, \mathsf{td}) \leftarrow \mathsf{ExtSetup}(1^\lambda);\ b \leftarrow \mathcal{A}^{\mathsf{H},\mathcal{O}_{\mathsf{Ext}}}(1^\lambda, \mathsf{crs}) = 1]$$

Here, $\mathcal{A}$ *has (black-box) access to the random oracle* H *and to an oracle* $\mathcal{O}_{\mathsf{Prove}}$ *or* $\mathcal{O}_{\mathsf{Ext}}$*, which are as follows:*

- $\mathcal{O}_{\mathsf{Verify}}(\mathbb{x}, \pi)$*: Return* $\mathsf{Verify}(\mathbb{x}, \pi)$.
- $\mathcal{O}_{\mathsf{Ext}}(\mathbb{x}, \pi)$*: If* $\mathsf{Verify}(\mathbb{x}, \pi) = 1$ *and* $(\mathbb{x}, \mathbb{w}) \notin \widetilde{\mathcal{R}}$ *for* $\mathbb{w} \leftarrow \mathsf{Ext}(\mathsf{td}, \mathcal{Q}, \mathbb{x}, \pi)$*, return* 0*. Else, return* 1*. Here,* $\mathcal{Q}$ *denotes the set of* $\mathcal{A}$*'s* H *queries.*

The advantage of $\mathcal{A}$ *against knowledge soundness is* $\mathsf{AdvExt}^{\Pi,\widetilde{\mathcal{R}}}_{\mathcal{A}}(\lambda) := |\mathsf{Real}_{\mathcal{A}}(\lambda) - \mathsf{Ideal}_{\mathcal{A}}(\lambda)|$*. We denote by* $\mathsf{AdvCRS}^{\Pi,\mathsf{ExtSetup}}_{\mathcal{D}}(\lambda)$ *the (standard) distinguishing between real and trapdoored* crs*. We say that* Ext *is a* knowledge extractor *for* Π *and knowledge relation* $\widetilde{\mathcal{R}}$*, if for every PPT algorithm* $\mathcal{A}$ *(resp.* $\mathcal{D}$*), the advantage* $\mathsf{AdvExt}^{\Pi,\widetilde{\mathcal{R}}}_{\mathcal{A}}(\lambda)$ *(resp.* $\mathsf{AdvCRS}^{\Pi,\mathsf{ExtSetup}}_{\mathcal{D}}(\lambda)$*) is negligible in* λ*. We say that* Π *is straightline* $\widetilde{\mathcal{R}}$-extractable *if there is a knowledge extractor for* Π*.*

B.2 (Partially) Blind Signatures

For completeness, we provide formal definitions of security properties for blind signatures. In particular, we give correctness, (partial) blindness and one-more (strong) unforgeability.

Definition B.4 (Correctness). *A partially blind signature* BS *is (perfectly) correct if for all* $(\mathsf{vk}, \mathsf{sk}) \in \mathsf{KeyGen}(1^\lambda)$ *and all* $m \in \mathcal{M}, I \in \mathcal{I}$*, it holds that*

$$\Pr[\sigma \leftarrow \langle \mathsf{BSign}(\mathsf{sk}, I), \mathsf{BUser}(\mathsf{vk}, m, I) \rangle : \mathsf{Verify}(\mathsf{vk}, m, I, \sigma) = 1] = 1$$

The guarantee of a (partially) blind signature scheme for the signer is that a valid signature can only be obtained via interaction, *i.e.*, one cannot output more signatures than the number of successfully completed signing sessions (one-more unforgeability, OMUF). We consider both OMUF and OMSUF, the latter being one-more *strong* unforgeability, *i.e.*, the adversary succeeds also if it can produce a fresh signature on an already signed message.

Definition B.5 (One-More (Strong) Unforgeability). *Let* $\mathsf{BS} = (\mathsf{KeyGen}, \mathsf{BSign}, \mathsf{BUser}, \mathsf{Verify})$ *be a blind signature scheme. Let* $\mathcal{A}$ *be an algorithm playing the following game:*

(1) Run $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and let* $\mathcal{O}$ *be an interactive oracle running* $\mathsf{BSign}(\mathsf{sk}, \cdot)$.
(2) Run $(I, ((m_1, \sigma_1), \ldots, (m_k, \sigma_k))) \leftarrow \mathcal{A}^{\mathcal{O}}(\mathsf{vk})$ *where* $\mathcal{A}$ *can query* $\mathcal{O}$ *in an arbitrarily interleaved way.*
(3) Output 1 *if and only if* $\mathcal{A}$ *completed at most* $k-1$ *interactions with* $\mathcal{O}$ *on input* I*, and for each* $i \in [k]$ *it holds that* $\mathsf{Verify}(\mathsf{vk}, m_i, I, \sigma_i) = 1$*, and*
- *all* m_i *for* $i \in [k]$ *are pairwise distinct (for unforgeability), or*
- *all* (m_i, σ_i) *for* $i \in [k]$ *are pairwise distinct (for strong unforgeability).*

We denote by $\mathsf{AdvOMUF}^{\mathsf{BS}}_{\mathcal{A}}(\lambda)$ *(resp.* $\mathsf{AdvOMSUF}^{\mathsf{BS}}_{\mathcal{A}}(\lambda)$*) the probability that the above game outputs* 1*. We say that* BS *is one-more unforgeable (OMUF), if for every PPT algorithm* $\mathcal{A}$*, it holds that* $\mathsf{AdvOMUF}^{\mathsf{BS}}_{\mathcal{A}}(\lambda) = \mathrm{negl}(\lambda)$*. We define one-more strong unforgeable (OMSUF) analogously.*

The security guarantee for the user is that interactions are not linkable to the issued signatures (blindness). Our blindness notion is *malicious signer blindness*, i.e., the malicious signer can freely choose the public key and arbitrarily deviate from the protocol.

Definition B.6 (Partial Blindness). *Let* $\mathsf{BS} = (\mathsf{KeyGen}, \mathsf{BSign}, \mathsf{BUser}, \mathsf{Verify})$ *be a blind signature scheme. For an algorithm* $\mathcal{A}$ *and bit* $b \in \{0,1\}$*, consider the following game:*

(1) Run $(\mathsf{vk}, m_0, m_1, I, st) \leftarrow \mathcal{A}(1^\lambda)$.
(2) Let $\mathcal{O}_0$ *be an interactive oracle simulating* $\mathsf{BUser}(\mathsf{vk}, m_b, I)$ *and* $\mathcal{O}_1$ *be an interactive oracle simulating* $\mathsf{BUser}(\mathsf{vk}, m_{1-b}, I)$.
(3) Run $st' \leftarrow \mathcal{A}^{\mathcal{O}_0, \mathcal{O}_1}(st)$*, where* $\mathcal{A}$ *has arbitrary interleaved one-time access to* $\mathcal{O}_0$ *and* $\mathcal{O}_1$*. Let* σ_b, σ_{1-b} *be the local outputs of* $\mathcal{O}_0, \mathcal{O}_1$*, respectively.*
(4) If $\sigma_0 = \bot$ *or* $\sigma_1 = \bot$*, run* $b' \leftarrow \mathcal{A}(st', \bot, \bot)$*. Else, run* $b' \leftarrow \mathcal{A}(st', \sigma_0, \sigma_1)$.
(5) Output b'.

We denote by $\mathsf{AdvBlind}^{\mathsf{BS}}_{\mathcal{A}}(\lambda)$ *difference between the probability that the above game with* $b = 0$ *outputs* 1 *and the probability that the game with* $b = 1$ *outputs* 1*. We say that* BS *satisfies partial blindness if* $\mathsf{AdvBlind}^{\mathsf{BS}}_{\mathcal{A}}(\lambda) = \mathrm{negl}(\lambda)$.

C Achieving Statistical Blindness

Our protocols $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ and $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ are only computationally blind, because the user sends an ElGamal encryption of his message to the signer. By a minor modification, one can upgrade the protocols to statistical blindness: Instead of using ElGamal encryption, we can use its lossy or dual-mode version, which was also used in Groth–Sahai proofs [GS12]. Effectively, we generate public keys and encryptions as

$$\mathsf{pk} = \left(\begin{smallmatrix} G & Y_0 \\ X_1 & Y_1 \end{smallmatrix}\right) \quad \text{and} \quad \mathbf{C} = (0, M) + (r_0, r_1) \cdot \mathsf{pk}.$$

If pk is full rank, which happens for a random choice of (X_1, Y_0, Y_1) except with probability $1/p$, then $\boldsymbol{r} \cdot \mathsf{pk} \in \mathbb{G}^2$ is uniformly random. On the other hand, if we

setup $(Y_0, Y_1)^\mathsf{T} = t \cdot (G, X_1)^\mathsf{T}$, then $\mathbf{C}$ is an ElGamal encryption w.r.t. public key (G, X_1) and randomness $r_0 + r_1 t$. Both modes are indistinguishable under DDH.

It is now easy to adapt our protocol and proofs to this setting. The notable differences are:

(1) $\boldsymbol{z}_{\mathsf{elg}} \in \mathbb{Z}_p^2$ now consists of two scalars instead of one.
(2) The blindness is statistical if Π_{M} is statistically zero-knowledge.
(3) Programming the random oracles H_{ch} and $\mathsf{H}_{\mathsf{par}}$ is not required for blindness anymore, similar to [KRW24].

We elaborate on points Items (2) and (3) below.

Item (2) follows by observing that, in the current proof, only the first step (application of zero-knowledge, game 1) and the final step (application of IND-CPA, game 6) are computational steps. With statistical zero-knowledge and lossy encryption, both π_{M} and C_M^* are statistically independent of M.

For Item (3), observe that unless a query $(\mathsf{pk}_I, \mathbf{C}_I) = \mathsf{H}_{\mathsf{par}}(I)$ occurs for which pk_I is not of full rank, all C_M^* are statistically uniform and thus independent of M. Hence, we can brute-force a witness for $\boldsymbol{\Sigma}_{\mathsf{elg}}$ and replace the SHVZK simulation introduced in game 5 of the proof of blindness (Theorem 4.7) with honestly generated transcripts. By SHVZK, this is perfectly indistinguishable. But now, we do not need to program $(\gamma_{\mathsf{dh}}, \gamma_{\mathsf{elg}})$ anymore, and hence can undo the random oracle programming and aborts.

Remark C.1. As all steps, except zero-knowledge simulation and replacement of C_M^* by an encryption of 0 were statistical already in Theorem 4.7, this shows statistical closeness of the real execution and an (unbounded) execution which is statistically independent of the message. Statistical blindness follows. As we only program the ROM to prove an equality of (non-programmed) distributions, we find it plausible that statistical blindness does transfer to then quantum random oracle model (QROM) as well.

D Deferred Figures

We provide a description of the signing interaction of our 5-move blind signature $\mathsf{BS}_{\mathsf{neq}}^{\mathsf{suf}}$ in Fig. 2.

E Deferred Proofs

Here, we provide proofs that were deferred from the main body.

E.1 Proof of Lemma 2.9

Proof. We prove the claims individually.

2-special soundness: Observe that given a pair of accepting transcripts $(\boldsymbol{A}, \gamma_b, \boldsymbol{z}_b)$ for $b \in \{0, 1\}$, we see that $\boldsymbol{w} \coloneqq \frac{1}{\gamma_1 - \gamma_0}(\boldsymbol{z}_1 - \boldsymbol{z}_0)$ satisfies $\phi(\boldsymbol{w}) = \boldsymbol{x}$.

Special HVZK*:* Observe that $\boldsymbol{A}$ for an accepting transcript is *uniquely* defined by $(\boldsymbol{x}, \gamma, \boldsymbol{z})$ due to the verification equation, namely $\boldsymbol{A} = \gamma \cdot \boldsymbol{x} - \phi(\boldsymbol{z})$ must hold. Moreover, observe that $\boldsymbol{z} = \gamma \cdot \boldsymbol{w} + \boldsymbol{r}$ is uniformly distributed (since $\boldsymbol{r}$ is). Hence, we can sample $\boldsymbol{z} \leftarrow \mathcal{W}$ and set $\boldsymbol{A}$ as in the simulation without affecting the distribution. This is perfect SHVZK simulation.

Strongly randomizable transcripts: Clearly, randomization in (2.1) maps accepting transcripts to accepting transcripts. Now, observe that in (2.1) the challenge $\gamma = \gamma^* + \gamma'$ and response $\boldsymbol{z} = \boldsymbol{z}^* + \boldsymbol{z}'$ are evidently uniformly distributed (since γ' and $\boldsymbol{z}$ are). As $\boldsymbol{A}$ is uniquely defined given $(\boldsymbol{x}, \gamma, \boldsymbol{z})$, this shows that the complete transcripts is a uniformly random choice of accepting transcripts for $\boldsymbol{x}$. Also observe that the distribution is identical to that of SHVZK simulation (both for $\mathbb{x} \in \mathcal{L}_{\mathcal{R}_\phi} = \mathsf{im}(\phi)$ and $\mathbb{x} \notin \mathcal{L}_{\mathcal{R}_\phi}$).

Unique response: Given two accepting responses $\boldsymbol{z}, \boldsymbol{z}'$ for $(\boldsymbol{x}, \boldsymbol{A}, \gamma)$, we immediately find $\phi(\boldsymbol{z}) = \phi(\boldsymbol{z}')$, and by injectivity $\boldsymbol{z} = \boldsymbol{z}'$. □

Proof of Lemma 3.2

Proof. Since $\boldsymbol{\Sigma}_{\mathsf{dh}}$ and $\boldsymbol{\Sigma}_{\mathsf{elg}}$ are canonical Σ-protocols, we already know from Lemma 2.9 that they satisfy all properties. Thus, we concentrate on $\boldsymbol{\Sigma}_{\mathsf{nez}}$. The 2-special soundness property follows immediately from $\boldsymbol{\Sigma}_{\mathsf{elg}}$. For SHVZK it suffices to note that $M_\$$ is distributed as $M_\$ \leftarrow \mathbb{G}^\times$. The simulator $\mathsf{Sim}_{\mathsf{nez}}((\mathsf{pk}, \mathbf{C}), \gamma)$ thus simply samples $M_\$ \leftarrow \mathbb{G}^\times$, runs $(\boldsymbol{A}_{\mathsf{elg}}, z) \leftarrow \mathsf{Sim}_{\mathsf{elg}}((\mathsf{pk}, \mathbf{C}, M_\$), \gamma)$ and outputs $A = (M_\$, \boldsymbol{A}_{\mathsf{elg}})$ and z. Clearly, the simulation is perfect for $\mathbb{x} \in \mathcal{R}_{\mathsf{nez}}$.

For randomizable transcripts, let $\mathsf{Rand}_{\mathsf{elg}}$ be the canonical randomization algorithm for $\boldsymbol{\Sigma}_{\mathsf{nez}}$ and observe that for $\alpha \in \mathbb{Z}_p^\times$

$$\Phi_{\mathsf{elg}}(z_{\mathsf{elg}}) = \boldsymbol{A}_{\mathsf{elg}} + \gamma \cdot (0, M) \quad \Longrightarrow \quad \Phi_{\mathsf{elg}}(\alpha z_{\mathsf{elg}}) = \alpha \boldsymbol{A}_{\mathsf{elg}} + \gamma \cdot (0, \alpha M)$$

that is, multiplication of $\boldsymbol{A}_{\mathsf{elg}}$ and z_{elg} by α modifies the encrypted message in the statement from M to $M_\$ = \alpha M$. Observe that $M_\$$ is uniformly random in $\mathbb{G}^\times$ if $\alpha \leftarrow \mathbb{Z}_p^\times$. Thus, $\boldsymbol{\Sigma}_{\mathsf{nez}}$ is randomizable by first running $\mathsf{Rand}_{\mathsf{elg}}$ on the input transcript, and then (re)randomizing $M_\$$ with α' on the intermediate transcript. Note that the resulting distribution coincide with SHVZK (even for $\mathbb{x} \notin \mathcal{R}_{\mathsf{nez}}$). Also note that the correctness of randomization was explicitly verified in the Theorem 4.4 to assert correctness of $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$. Namely, $\mathsf{Rand}_{\mathsf{nez}}((\mathsf{pk}, \mathbf{C}), (M^*_\$, \boldsymbol{A}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}}, z^*_{\mathsf{elg}}))$ chooses $\alpha', \gamma', \boldsymbol{z}'_{\mathsf{elg}}$ and sets

- $M_\$ = \alpha' \cdot M^*_\$$
- $\boldsymbol{A}_{\mathsf{elg}} \coloneqq \alpha' \cdot \big(\boldsymbol{A}'_{\mathsf{elg}} + \Phi_{\mathsf{elg}}(\mathbf{C}, \boldsymbol{z}^*_{\mathsf{elg}}) - \gamma^*(0, M^*_\$)\big)$
- $\gamma = \gamma^* + \gamma'_{\mathsf{elg}}$ and $\boldsymbol{z}_{\mathsf{elg}} = \boldsymbol{z}^*_{\mathsf{elg}} + \boldsymbol{z}'_{\mathsf{elg}}$.

□

E.2 Proof of Lemma 4.2

Proof. By special soundness of $\boldsymbol{\Sigma}_{\mathsf{dh}}$ and because $\mathbb{x}_{\mathsf{dh}} \notin \mathcal{L}_{\mathsf{dh}}$, there is at most one challenge $\gamma_{\mathsf{dh}} \in \mathbb{Z}_p$ such that there exists an accepting response z_{dh}, *i.e.*, such that

τ_{dh} accepts. Similarly, by special soundness of $\boldsymbol{\Sigma}_{\mathsf{elg}}$ and as $\mathbb{x}_{\mathsf{elg}} \notin \mathcal{L}_{\mathsf{elg}}$, there exists at most one challenge γ_{elg} such that an accepting response z_{elg} exists. Thus, the pair $(\gamma_{\mathsf{dh}}, \gamma_{\mathsf{elg}})$ is fully determined by X. Consequently, the value γ is distributed uniformly over $\mathbb{Z}_p$ and at most with probability $1/p$, we have $\gamma = \gamma_{\mathsf{dh}} + \gamma_{\mathsf{elg}}$. The final statement follows by a union bound. □

E.3 Proof of Lemma 4.3

Proof. The proof is a straightforward reduction to QDDH, or rather IND-CPA security of ElGamal encryption. We argue by hybrid games. The first hybrid is $\mathsf{Exp}^0(\mathcal{A})$.

The second hybrid modifies the computation of $\mathbf{C}_I$ to $\mathbf{C}_I = \mathbf{C}'_I + (0, M_I)$ for $\mathbf{C}'_I \leftarrow \mathbb{G}^2$. As the distribution is unaffected, this change is does not affect the output probabilities.

The third and final change replaces $\mathbf{C}'_I \leftarrow \mathbb{G}^2$ by $\mathbf{C}'_I = t_I \cdot \mathsf{pk}_I$ for $t_I \leftarrow \mathbb{G}^2$. This the same as $\mathsf{Exp}^1(\mathcal{A})$. Observe that the tuples $(\mathsf{pk}_{I,1}, C_{I,0}, C_{I,1})$ are either uniformly random (except G), or DDH tuples. Hence, the indistinguishability is a direct reduction to QDDH with $Q = Q_{\mathsf{par}}$. □

E.4 Proof of Theorem 4.4

Proof. Denote by $\sigma = (M_\$, \pi)$ the output of a honest signing session. We need to show that $\pi = (\boldsymbol{A}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{dh}}, \gamma, \gamma_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{elg}}, z_{\mathsf{dh}})$ is a valid OR-proof for statements $\mathbb{x}_{\mathsf{dh}} := (G, \mathbf{D})$ and $\mathbb{x}_{\mathsf{elg}} := (\mathsf{pk}, \mathbf{C}, M_\$)$. Let $\gamma' := \mathsf{H}_{\mathsf{ch}}(\mathbb{x}_{\mathsf{dh}}, \mathbb{x}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{dh}})$ and $\gamma_{\mathsf{elg}} := \gamma' - \gamma_{\mathsf{dh}}$. By construction, we have $\gamma' = \gamma$, where γ is the challenge. It remains to show that

$$\mathsf{Verify}_{\mathsf{elg}}(\mathbb{x}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{elg}}, \gamma_{\mathsf{elg}}, \boldsymbol{z}_{\mathsf{elg}}) = 1 \ \wedge \ \mathsf{Verify}_{\mathsf{dh}}(\mathbb{x}_{\mathsf{dh}}, \boldsymbol{A}_{\mathsf{dh}}, \gamma_{\mathsf{dh}}, z_{\mathsf{dh}}) = 1.$$

We show that both transcripts verify below. Below, for some transcript $\tau_x = (\boldsymbol{A}_x, \gamma_x, \boldsymbol{z}_x)$, we denote $\tau_x[\boldsymbol{A}] = \boldsymbol{A}_x$, $\tau_x[\gamma] = \gamma_x$ and $\tau_x[\boldsymbol{z}] = \boldsymbol{z}_x$ for convenience.

Non-zero Encryption. Let us show that $\mathsf{Verify}_{\mathsf{elg}}(\mathbb{x}_{\mathsf{elg}}, \boldsymbol{A}_{\mathsf{elg}}, \gamma_{\mathsf{elg}}, \boldsymbol{z}_{\mathsf{elg}}) = 1$.

Note that by Lemma 2.9, the simulator $\mathsf{Sim}_{\mathsf{elg}}$ outputs a pair $(\boldsymbol{A}_{\mathsf{elg}}, \boldsymbol{z}_{\mathsf{elg}})$ such that

$$\Phi_{\mathsf{elg}}(\mathbf{C}^*, \boldsymbol{z}^*_{\mathsf{elg}}) = \boldsymbol{A}^*_{\mathsf{elg}} + \gamma^*_{\mathsf{elg}}(0, M^*_\$).$$

Denote by $\tau^* := (\boldsymbol{A}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}})$ the accepting transcript.

Let us show that after the transformation highlighted with , the modified transcript $\tau' := \tau^* + ((0, t \cdot A^*_{\mathsf{elg},0}), 0, 0)$ verifies with respect to $\mathbb{x}'_{\mathsf{elg}} = (\mathsf{pk}_I, \mathbf{C}, M^*_\$)$

where $\mathbf{C} = \mathbf{C}_I - \mathbf{C}_M = \mathbf{C}^* + t\mathsf{pk}_I$:

$$\begin{aligned}
\Phi_{\mathsf{elg}}(\mathbf{C}, \tau'[\boldsymbol{z}]) &= \Phi_{\mathsf{elg}}(\mathbf{C}^* + t\mathsf{pk}_I, \tau^*[\boldsymbol{z}]) \\
&= \tau^*[\boldsymbol{z}] \cdot \begin{pmatrix} \mathsf{pk}_{I,0} & C_0^* + t\mathsf{pk}_{I,0} \\ \mathsf{pk}_{I,1} & C_1^* + t\mathsf{pk}_{I,1} \end{pmatrix} \\
&= \tau^*[\boldsymbol{z}] \cdot \begin{pmatrix} \mathsf{pk}_I & \mathbf{C}^* \end{pmatrix} + \tau^*[\boldsymbol{z}] \cdot \begin{pmatrix} \mathbf{0} & t\mathsf{pk}_I \end{pmatrix} \\
&= \Phi_{\mathsf{elg}}(\mathbf{C}^*, \tau^*[\boldsymbol{z}]) + (0, t \cdot \tau^*[A_{\mathsf{elg},0}]) \\
&= \tau^*[\boldsymbol{A}] + (0, t \cdot \tau^*[A_{\mathsf{elg},0}])] + \tau^*[\gamma] \cdot (0, M_\$) \\
&= \tau'[\boldsymbol{A}] + \tau'[\gamma] \cdot (0, M_\$)
\end{aligned}$$

Now, let us show that after the transformation highlighted with , the modified transcript $\tau_1 \coloneqq \tau^* + (\Phi_{\mathsf{elg}}(\mathbf{C}^*, \boldsymbol{z}^*_{\mathsf{elg}}), 0, \boldsymbol{z}'_{\mathsf{elg}})$ still verifies with respect to $\mathbb{x}^*_{\mathsf{elg}}$:

$$\begin{aligned}
\Phi_{\mathsf{elg}}(\mathbf{C}^*, \tau_1[\boldsymbol{z}]) &= \Phi_{\mathsf{elg}}(\mathbf{C}^*, \boldsymbol{z}^*_{\mathsf{elg}} + \boldsymbol{z}'_{\mathsf{elg}}) \\
&= \Phi_{\mathsf{elg}}(\mathbf{C}^*, \boldsymbol{z}^*_{\mathsf{elg}}) + \Phi_{\mathsf{elg}}(\mathbf{C}^*, \boldsymbol{z}^*_{\mathsf{elg}}) \\
&= \boldsymbol{A}^*_{\mathsf{elg}} + \gamma^*_{\mathsf{elg}}(0, M^*_\$) + \Phi_{\mathsf{elg}}(\mathbf{C}^*, \boldsymbol{z}^*_{\mathsf{elg}}) \\
&= \tau_1[\boldsymbol{A}] + \tau_1[\gamma](0, M^*_\$).
\end{aligned}$$

Next, let us show that after the transformation highlighted with , the modified transcript $\tau_2 \coloneqq \tau_1 + (-\gamma'_{\mathsf{elg}}(0, M^*_\$), \gamma'_{\mathsf{elg}}, 0)$ still verifies with respect to $\mathbb{x}^*_{\mathsf{elg}}$:

$$\begin{aligned}
\Phi_{\mathsf{elg}}(\mathbf{C}^*, \tau_2[\boldsymbol{z}]) &= \Phi_{\mathsf{elg}}(\mathbf{C}^*, \tau_1[\boldsymbol{z}]) \\
&= \tau_1[\boldsymbol{A}] + \tau_1[\gamma](0, M^*_\$) \\
&= \tau_1[\boldsymbol{A}] + \gamma^*_{\mathsf{elg}}(0, M^*_\$) + \gamma'_{\mathsf{elg}}(0, M^*_\$) - \gamma'_{\mathsf{elg}}(0, M^*_\$) \\
&= \tau_1[\boldsymbol{A}] + (\gamma^*_{\mathsf{elg}} + \gamma'_{\mathsf{elg}})(0, M^*_\$) - \gamma'_{\mathsf{elg}}(0, M^*_\$) \\
&= \tau_2[\boldsymbol{A}] + \tau_2[\gamma](0, M^*_\$).
\end{aligned}$$

Next, let us show that after the transformation highlighted with , the modified transcript $\tau_3 \coloneqq (\alpha' \cdot \tau_2[A], \tau_2(\gamma), \alpha' \cdot \tau_2[z])$ still verifies with respect to $\mathbb{x}'_{\mathsf{elg}} = (\mathsf{pk}_I, \mathbf{C}^*, M_\$)$, where $M_\$ = \alpha' \cdot M^*_\$$:

$$\begin{aligned}
\Phi_{\mathsf{elg}}(\mathbf{C}^*, \tau_3[\boldsymbol{z}]) &= \Phi_{\mathsf{elg}}(\mathbf{C}^*, \alpha' \cdot \tau_2[z]) \\
&= \alpha' \cdot \Phi_{\mathsf{elg}}(\mathbf{C}^*, \tau_2[z]) \\
&= \alpha' \cdot (\tau_2[\boldsymbol{A}] + \tau_2[\gamma](0, M^*_\$)) \\
&= \alpha' \cdot \tau_2[\boldsymbol{A}] + \tau_2[\gamma](0, \alpha' \cdot M^*_\$) \\
&= \tau_3[\boldsymbol{A}] + \tau_3[\gamma](0, M_\$).
\end{aligned}$$

At this point, we have applied all the transformations, and every time, they mapped accepting transcripts to accepting transcripts. Thus, the non-zero encryption part of the protocol is correct.

DDH. This follows as above as for the transformations highlighted with ■ and ■ only linearity of the map is required, and we omit details. □

E.5 Proof of Lemma 5.3

Proof. Let us assume that there exist two such challenges $\gamma_{\mathsf{elg},1}$ and $\gamma_{\mathsf{elg},2}$. By special soundness, there exists $\mathbb{w}_{\mathsf{elg}} = (x, y) \in \mathbb{Z}_p^2$ such that $yH = xG$ and $M_\$ = yC_1 - xC_0$ by definition of $\mathcal{L}_{\mathsf{elg}}$ (cf. Eq. (3.2)). As previously, we can show that $y \neq 0$ (cf. final paragraph in the proof of Theorem 4.5). Dividing by y, we obtain $H = h \cdot G$, where $h \coloneqq x/y$, and $1/y \cdot M_\$ = C_1 - hC_0$. As $\mathbf{C} = t \cdot \mathsf{pk}$, we have $1/y \cdot M_\$ = tH - h(tG) = 0$. As $M_\$ \neq 0$, this is a contradiction. □

E.6 Proof of Theorem 5.5

Proof. Let $\mathcal{A}$ be a PPT adversary against strong one-more unforgeability of $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$. Let $\mathbb{G}$ be a group of prime order p with generator G. We use the same conventions and notations as in the proof of Theorem 4.5. That is, for random oracle $\mathsf{H}_{\mathsf{xyz}} \in \{\mathsf{H}_M, \mathsf{H}_{\mathsf{ch}}, \mathsf{H}_{\mathsf{par}}, \mathsf{H}_{\Pi}, \mathsf{H}_{\mathsf{crs}}\}$, denote by Q_{xyz} the number of oracle queries to $\mathsf{H}_{\mathsf{xyz}}$. Also, we assume that $\mathsf{H}_{\mathsf{xyz}}$ queries made by the game (*e.g.*, during signing queries or verification) count towards Q_{xyz}. Denote by Q_S the number of $\mathcal{A}$'s signing queries. We proceed with a sequence of games Game i and denote by ε_i the advantage of $\mathcal{A}$ in Game i (*i.e.*, the probability that Game i outputs 1).

Game 0 (Honest). This game is the real strong one-more unforgeability game for scheme $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$. Recall that oracles $\mathsf{H}_M, \mathsf{H}_{\mathsf{ch}}, \mathsf{H}_{\mathsf{par}}, \mathsf{H}_{\Pi}, \mathsf{H}_{\mathsf{crs}}$ are modeled as random oracles. For convenience, let us recap the game below.

The game first samples $\mathsf{vk} = \mathbf{D}$ and $\mathsf{sk} = d_2$ via $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}.\mathsf{KeyGen}$. That is, it samples $d_2 \leftarrow \mathbb{Z}_p$ and $D_1 \leftarrow \mathbb{G}$, then sets $D_3 \coloneqq d_2 D_1$ and $\mathbf{D} = (D_1, D_2, D_3)$. Next, the game sends vk to $\mathcal{A}$ and provides access to the random oracles and signing oracles $\mathcal{O}_{\mathsf{BSign}_2}, \mathcal{O}_{\mathsf{BSign}_3}, \mathcal{O}_{\mathsf{BSign}_3}$. In the end, $\mathcal{A}$ outputs a common message $\overline{I}$ and forgeries $(\overline{\mu}_j, \overline{\sigma}_j)_{j \in [Q_{\mathsf{frg}}]}$. The game outputs 1 iff $\mathcal{O}_{\mathsf{BSign}_3}$ was queried at most $Q_{\mathsf{frg}} - 1$ times with common message $\overline{I}$, all message-signature pairs $\{(\overline{\mu}_j, \overline{\sigma}_j)\}_{j \in [Q_{\mathsf{frg}}]}$ are pairwise-distinct, and all signatures verify (*i.e.*, $\mathsf{S}_{\mathsf{neq}}.\mathsf{Verify}(\mathsf{vk}, \overline{M}_j, \overline{I}, \overline{\sigma}_j) = 1$ with $\overline{M}_j \coloneqq \mathsf{H}_M(\overline{\mu}_j)$). We consistently mark values x associated to the forgeries with $\overline{x}$. In particular, we parse $\overline{\sigma}_j = (\overline{M_\$}_j, \overline{\pi}_j)$ with $\overline{\pi}_j = (\overline{\boldsymbol{A}}_{\mathsf{elg},j}, \overline{\boldsymbol{A}}_{\mathsf{dh},j}, \overline{\gamma}_{\mathsf{elg},j}, \overline{\gamma}_{\mathsf{dh},j}, \overline{\boldsymbol{z}}_{\mathsf{elg},j}, \overline{z}_{\mathsf{dh},j})$. Also, we denote $\overline{\pi}_{\mathsf{xyz},j} \coloneqq (\overline{\boldsymbol{A}}_{\mathsf{xyz},j}, \overline{\gamma}_{\mathsf{xyz},j}, \overline{\boldsymbol{z}}_{\mathsf{xyz},j})$ for $\mathsf{xyz} \in \{\mathsf{elg}, \mathsf{dh}\}$.

We identify each signing session with a session identifier sid provided as input in $\mathcal{O}_{\mathsf{BSign}_2}$ and $\mathcal{O}_{\mathsf{BSign}_3}$. The signing oracles behave as follows:

- $\mathcal{O}_{\mathsf{BSign}_2}(\mathsf{sid})$: The game sets $\mathbb{x}_{\mathsf{dh}} \coloneqq (G, \mathbf{D})$ and $(\boldsymbol{A}^*_{\mathsf{dh}}, \mathsf{st}_{\mathsf{dh}}) \leftarrow \mathsf{Init}_{\mathsf{dh}}(\mathbb{x}_{\mathsf{dh}}, d_2)$, where $\mathsf{vk} = \mathbf{D}$ and $\mathsf{sk} = d_2$. Note that common message I is not yet specified.

(This can be delayed until $\mathcal{O}_{\mathsf{BSign}_3}$.) The game stores $\mathsf{st}_{\mathsf{dh}}$ in its state for sid and outputs

$$\boldsymbol{A}^*_{\mathsf{dh}}.$$

– $\mathcal{O}_{\mathsf{BSign}_3}(\mathsf{sid}, I, \mathbf{C}^*_M, \pi_{\mathsf{M}})$: The game aborts if the state for sid does not contain (exactly) $\mathsf{st}_{\mathsf{dh}}$, else it sets $(\mathsf{pk}_I, \mathbf{C}_I) \coloneqq \mathsf{H}_{\mathsf{par}}(I)$. Then, it verifies π_{M}, *i.e.*, it sets $\mathbb{x}_{\mathsf{M}} \coloneqq (\mathsf{pk}_I, \mathbf{C}^*_M)$ and $\mathsf{crs}_{\mathsf{M}} \coloneqq \mathsf{H}_{\mathsf{crs}}(0)$, and aborts if $\Pi_{\mathsf{M}}.\mathsf{Verify}^{\mathsf{H}_\Pi}(\mathsf{crs}_{\mathsf{M}}, \mathbb{x}_{\mathsf{M}}, \pi_{\mathsf{M}}) \neq 1$. Then, it sets up ciphertext $\mathbf{C}^* \coloneqq \mathbf{C}_I - \mathbf{C}^*_M$ and target $M^*_{\$} \leftarrow \mathbb{G}^\times$, and computes Σ-protocol messages for $\mathbb{x}^*_{\mathsf{elg}} \coloneqq (\mathsf{pk}_I, \mathbf{C}^*, M^*_{\$})$. That is, it samples $\gamma^*_{\mathsf{elg}} \leftarrow \mathbb{Z}_p$ and sets $(\boldsymbol{A}^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}}) \leftarrow \mathsf{Sim}_{\mathsf{elg}}(\mathbb{x}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}})$. Note that the first flow $\mathbb{x}_{\mathsf{dh}} \coloneqq (G, \mathbf{D})$ was already output in $\mathcal{O}_{\mathsf{BSign}_2}$. The game further stores γ^*_{elg} and $\boldsymbol{z}^*_{\mathsf{elg}}$ in its state for sid, and outputs

$$(M^*_{\$}, \boldsymbol{A}^*_{\mathsf{elg}}, \boldsymbol{A}^*_{\mathsf{dh}}).$$

– $\mathcal{O}_{\mathsf{BSign}_3}(\mathsf{sid}, \gamma^*)$: The game retrieves $\gamma^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}}$ and $\mathsf{st}_{\mathsf{dh}}$ from the state for sid (and aborts if this is not possible). Then, it sets $\gamma^*_{\mathsf{dh}} \coloneqq \gamma^* - \gamma^*_{\mathsf{elg}}$ and $\boldsymbol{z}^*_{\mathsf{dh}} \leftarrow \mathsf{Resp}_{\mathsf{dh}}(\mathsf{st}_{\mathsf{dh}}, \gamma^*_{\mathsf{dh}})$. The game empties its state for sid and outputs

$$(\boldsymbol{z}^*_{\mathsf{elg}}, z^*_{\mathsf{dh}}, \gamma^*_{\mathsf{elg}}).$$

By definition, we have

$$\mathsf{AdvOMSUF}^{\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}}_{\mathcal{A}}(\lambda) = \varepsilon_0.$$

Before we proceed with our game sequence, let us define three different games Game 0.1, Game 0.2, Game 0.3 by modifying the win condition in Game 0 such that

$$\varepsilon_0 \leq \varepsilon_{0.1} + \varepsilon_{0.2} + \varepsilon_{0.3}.$$

Game 0.1 (Distinct messages). This game is identical to Game 0, except that after adversary $\mathcal{A}$ outputs its forgeries $(\overline{\mu}_j, \overline{\sigma}_j)_{j \in [Q_{\mathsf{frg}}]}$, the game checks that

$$\text{all message-commitment pairs } \{(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})\}_{j \in [Q_{\mathsf{frg}}]} \text{ are pairwise distinct,} \quad \text{(I)}$$

that is, each forgery consists of distinct message and $\boldsymbol{\Sigma}_{\mathsf{dh}}$-commitment pairs.

Game 0.2 ($\boldsymbol{A}_{\mathsf{dh}}$ reuse with distinct γ_{dh}). This game is identical to Game 0, except that after adversary $\mathcal{A}$ outputs its forgeries $(\overline{\mu}_j, \overline{\sigma}_j)_{j \in [Q_{\mathsf{frg}}]}$, the game checks that

$$\exists j, k \in [Q_{\mathsf{frg}}] : j \neq k, \overline{\mu}_j = \overline{\mu}_k, \overline{\boldsymbol{A}}_{\mathsf{dh},j} = \overline{\boldsymbol{A}}_{\mathsf{dh},k}, \overline{\gamma}_{\mathsf{dh},j} \neq \overline{\gamma}_{\mathsf{dh},k}, \quad \text{(II)}$$

that is, the j-th and k-th forgery share the same message and $\boldsymbol{\Sigma}_{\mathsf{dh}}$ commitments, but the $\boldsymbol{\Sigma}_{\mathsf{dh}}$ challenges are distinct.

Game 0.3 ($\boldsymbol{A}_{\mathsf{dh}}$ reuse with same γ_{dh}). This game is identical to Game 0, except that after adversary $\mathcal{A}$ outputs its forgeries $(\overline{\mu}_j, \overline{\sigma}_j)_{j \in [Q_{\mathsf{frg}}]}$, the game checks that

$$\exists j, k \in [Q_{\mathsf{frg}}] : j \neq k, \overline{\mu}_j = \overline{\mu}_k, \overline{\boldsymbol{A}}_{\mathsf{dh},j} = \overline{\boldsymbol{A}}_{\mathsf{dh},k}, \overline{\gamma}_{\mathsf{dh},j} = \overline{\gamma}_{\mathsf{dh},k}, \quad \text{(III)}$$

that is, the j-th and k-th forgery share the same message and $\boldsymbol{\Sigma}_{\mathsf{dh}}$ commitments, and the $\boldsymbol{\Sigma}_{\mathsf{dh}}$ challenges are identical.

Case distinction: Clearly, the conditions in Eqs. (I), (II) and (III) cover all possible forgery types of $\mathcal{A}$. Thus, we have $\varepsilon_0 \leq \varepsilon_{0.1} + \varepsilon_{0.2} + \varepsilon_{0.3}$. Lemmas E.1 to E.3 yields that there are reductions $\mathcal{B}_{\mathsf{crs}}$, $\mathcal{B}_{\mathsf{QDDH}}, \mathcal{B}_{\mathsf{dl}}$ such that

$$\begin{aligned}\mathsf{AdvOMSUF}^{\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}}_{\mathcal{A}}(\lambda) \leq\ & 2\mathsf{AdvCRS}^{\Pi_{\mathsf{m}},\mathsf{ExtSetup}}_{\mathcal{B}_{\mathsf{crs}}}(\lambda, Q_{\Pi}) + \mathsf{AdvDLOG}^{\mathbb{G}}_{\mathsf{dl}}(\lambda) \\ & + (Q_M + 1) \cdot Q_{\mathsf{par}} \cdot \Big(\mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{B}_{\mathsf{dh}}}(\lambda, Q_{\mathsf{par}}) + \mathsf{AdvExt}^{\Pi_{\mathsf{m}},\mathsf{Ext}}_{\mathcal{B}_{\mathsf{ext}}}(\lambda, Q_{\Pi})\Big) \\ & + \frac{2Q_S}{p} + \frac{Q_M^2}{p} + \frac{1 + Q_{\mathsf{ch}}}{p} + \frac{Q_M \cdot Q_{\mathsf{ch}}^2}{p}.\end{aligned}$$

Lemma E.1. *There are reductions $\mathcal{B}_{\mathsf{crs}}$, $\mathcal{B}_{\mathsf{QDDH}}$ whose running time is roughly that of the OMSUF game, such that*

$$\begin{aligned}\varepsilon_{0.1} \leq\ & \mathsf{AdvCRS}^{\Pi_{\mathsf{m}},\mathsf{ExtSetup}}_{\mathcal{B}_{\mathsf{crs}}}(\lambda, Q_{\Pi}) + \frac{Q_S}{p} + \frac{Q_M^2}{p} + \frac{1 + Q_{\mathsf{ch}}}{p} \\ & + Q_M \cdot Q_{\mathsf{par}} \cdot \Big(\mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{B}_{\mathsf{dh}}}(\lambda, Q_{\mathsf{par}}) + \mathsf{AdvExt}^{\Pi_{\mathsf{m}},\mathsf{Ext}}_{\mathcal{B}_{\mathsf{ext}}}(\lambda, Q_{\Pi})\Big).\end{aligned}$$

Lemma E.2. *There are reductions $\mathcal{B}_{\mathsf{crs}}$, $\mathcal{B}_{\mathsf{QDDH}}, \mathcal{B}_{\mathsf{dl}}$ whose running time is roughly that of the OMSUF game, such that*

$$\begin{aligned}\varepsilon_{0.2} \leq\ & \mathsf{AdvCRS}^{\Pi_{\mathsf{m}},\mathsf{ExtSetup}}_{\mathcal{B}_{\mathsf{crs}}}(\lambda, Q_{\Pi}) + \frac{Q_S}{p} + \mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{B}_{\mathsf{dh}}}(\lambda, Q_{\mathsf{par}}) \\ & + \mathsf{AdvExt}^{\Pi_{\mathsf{m}},\mathsf{Ext}}_{\mathcal{B}_{\mathsf{ext}}}(\lambda, Q_{\Pi}) + \mathsf{AdvDLOG}^{\mathbb{G}}_{\mathsf{dl}}(\lambda).\end{aligned}$$

Lemma E.3. *There is a reduction $\mathcal{B}_{\mathsf{QDDH}}$ whose running time is roughly that of the OMSUF game, such that*

$$\varepsilon_{0.3} \leq Q_M \cdot \big(\mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{B}_{\mathsf{dh}}}(\lambda, Q_{\mathsf{par}}) + \frac{Q_{\mathsf{ch}}^2}{p}\big).$$

This concludes the proof of Theorem 5.5. □

Below, we prove the remaining Lemmas E.1 to E.3 in Appendices E.7 to E.9, respectively.

E.7 Proof of Lemma E.1

Proof. Observe that if all pairs $(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})$ are pairwise distinct, then so are the messages $\overline{M}_j = \mathsf{H}_M(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})$ with high probability. Now, essentially the same argument as in the proof of Theorem 4.5 gives us the statement. Most of the proof is almost verbatim, but we elaborate below for completeness. We highlight when the argument differs via .

Game 1 (Honest). The game is identical to Game 0.1 and we have

$$\varepsilon_{0.1} = \varepsilon_1.$$

Game 2 (Abort if H_M collision). The game aborts its entire execution if there is a collision in H_M. By a standard birthday-bound argument, we have

$$|\varepsilon_1 - \varepsilon_2| \leq \frac{Q_M^2}{p}.$$

Game 3 (Extract M from π_{M}). Before sending vk to $\mathcal{A}$, the game sets $\mathsf{crs}_{\mathsf{M}} \leftarrow \mathsf{ExtSetup}(1^\lambda)$ and programs $\mathsf{H}_{\mathsf{crs}}(0) := \mathsf{crs}_{\mathsf{M}}$. Later, on every $\mathcal{O}_{\mathsf{BSign}_2}$ query of the form $(\mathsf{sid}, I, \mathbf{C}_M^*, \pi_{\mathsf{M}})$, after verifying π_{M}, the adversary extracts the message $M \leftarrow \mathsf{Ext}((\mathsf{td}, \mathcal{Q}), (\mathbb{x}_{\mathsf{M}}, \pi_{\mathsf{M}}))$ via π_{M}. Here, $\mathcal{Q}$ is a list containing all H_Π so far.

Note that while we already extract the message, we do not use the extracted value within the simulation yet. It is straightforward to construct a reduction $\mathcal{B}_3$ with running time similar to $\mathcal{A}$ such that

$$|\varepsilon_2 - \varepsilon_3| \leq \mathsf{AdvCRS}_{\mathcal{B}_3}^{\Pi_{\mathsf{m}}, \mathsf{ExtSetup}}(\lambda, Q_\Pi).$$

At this point, the game does *not* know whether M is actually a witness for relation $\widetilde{\mathcal{R}}_{\mathsf{M}}$. This can readily be verified via the secret key sk_I associated to pk_I. Yet, because the simulation cannot depend on sk_I for subsequent proof steps, we cannot yet add an explicit abort condition that that relies on sk_I. Nevertheless, it is useful to know the extracted message M for the next proof steps.

Game 4 (Guess $\overline{I}$). We guess the first query to $\mathsf{H}_{\mathsf{par}}$ such that the forgeries' common message $\overline{I}$ is provided as input. That is, the game samples $i_{I,\mathcal{A}} \leftarrow [Q_{\mathsf{par}}]$ at its start. When $\mathcal{A}$ outputs common message $\overline{I}$ and its forgeries, the game additionally checks whether $\overline{I}$ was queried for the first time to $\mathsf{H}_{\mathsf{par}}$ on the $i_{I,\mathcal{A}}$-th query. If not, the game aborts its entire execution.

Observe that such a query must exist, as we also count the game's $\mathsf{H}_{\mathsf{par}}$ queries and the game evaluates $\mathsf{H}_{\mathsf{par}}$ on input $\overline{I}$ when verifying the forgeries. As the guess $i_{I,\mathcal{A}}$ is hidden from $\mathcal{A}$, we have that

$$\varepsilon_3 \leq Q_{\mathsf{par}} \cdot \varepsilon_4.$$

We stress that at this point, the game knows $\overline{I}$ after the first $\mathsf{H}_{\mathsf{par}}$ query with input $\overline{I}$ was made. In particular, as the game evaluates $\mathsf{H}_{\mathsf{par}}$ on common message I during each $\mathcal{O}_{\mathsf{BSign}_2}$ query, the game knows the forgeries' common message $\overline{I}$ at latest when the first $\mathcal{O}_{\mathsf{BSign}_2}$ query with common message $\overline{I}$ is made.

Game 5 (Guess unsigned $\overline{M}$ in forgery). We guess the first query $i_{M,\mathcal{A}}$ to H_M such that the following two conditions hold:

(1) The input $X_{i_{M,\mathcal{A}}}$ to the $i_{M,\mathcal{A}}$-th H_M query is part of $\mathcal{A}$'s forgeries and $X_{i_{M,\mathcal{A}}}$ was never queried to H_M beforehand, *i.e.*, there exists $j \in [Q_{\mathsf{frg}}]$ such that $X_{i_{M,\mathcal{A}}} = (\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})$.
(2) No session with common message $\overline{I}$ is *completed* if $\overline{M} = \mathsf{H}_M(X_{i_{M,\mathcal{A}}})$ is extracted from proof π_{M} (cf. game 3).

Again, the game aborts its execution if the guess was incorrect.

Here, we crucially rely on Eq. (I). If $\mathcal{A}$ is successful, then $\mathcal{A}$'s output contains Q_{frg} distinct message-commitment pairs $\{(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})\}_{j\in[Q_{\mathsf{frg}}]}$. As H_M is collision-free (cf. game 2), there are also Q_{frg} distinct hashed messages $\overline{M}_j \coloneqq \mathsf{H}_M(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})$. As there are at most $Q_{\mathsf{frg}} - 1$ completed sessions for common message $\overline{I}$, there must be at least one hashed message $\overline{M} \in \{\overline{M}_1, \ldots, \overline{M}_{Q_{\mathsf{frg}}}\}$ that was never extracted in any of these completed $Q_{\mathsf{frg}} - 1$ sessions. Consequently, such an index $i_{M,\mathcal{A}}$ must exist, and since the guess is hidden from $\mathcal{A}$, we have

$$\varepsilon_4 \leq Q_M \cdot \varepsilon_5.$$

In the following, we denote by $\overline{M}$ the output of the $i_{M,\mathcal{A}}$-th H_M query. Note that if $\mathcal{A}$ is successful, we can assume that $\overline{M}$ is known by the game from the start on.[9]

Game 6 (Abort if M_I^* is extracted). Initially, the game samples a random message $M_I^* \leftarrow \mathbb{G}$. Then, the game aborts its entire execution if M_I^* is extracted from π_{M} in $\mathcal{O}_{\mathsf{BSign}_2}$ for any common message I. That is, after setting $M \leftarrow \mathsf{Ext}((\mathsf{td}, \mathcal{Q}), (\mathbb{x}, \pi_{\mathsf{M}}))$ in $\mathcal{O}_{\mathsf{BSign}_2}$ (cf. game 4), the game checks if $M = M_I^*$. If so, the game aborts its entire execution, else it continues as before.

As M_I^* is never used within in the simulation (except for the abort condition), a union bound yields

$$|\varepsilon_5 - \varepsilon_6| \leq \frac{Q_S}{p}.$$

Game 7 (Setup $\mathbf{C}_I$ with specific messages). We now setup the ciphertexts $\mathbf{C}_I$ output by $\mathsf{H}_{\mathsf{par}}$ depending on whether the forgeries' common message $\overline{I}$ was queried. That is, on the first $\mathsf{H}_{\mathsf{par}}$ query with input $\overline{I}$, the game sets up $\mathsf{pk}_I \leftarrow \{G\} \times \mathbb{G}$ at random and encrypts $\overline{M}$ in $\mathbf{C}_I$, *i.e.*, computes ElGamal ciphertext $\mathbf{C}_I \coloneqq (0, \overline{M}) + t \cdot \mathsf{pk}_I$ for $t \leftarrow \mathbb{Z}_p$. The game outputs $(\mathsf{pk}_I, \mathbf{C}_I)$. On all other fresh $\mathsf{H}_{\mathsf{par}}$ queries on input I, the game sets up pk_I at random and encrypts M_I^* in $\mathbf{C}_I$, *i.e.*, sets $\mathbf{C}_I \coloneqq (0, M_I^*) + t \cdot \mathsf{pk}_I$. Again, outputs $(\mathsf{pk}_I, \mathbf{C}_I)$. Note that M_I^* is chosen as in Game 6.

Recall that in the previous game, all $\mathsf{H}_{\mathsf{par}}$ outputs $\mathbf{C}_I$ are uniform over $\mathbb{G}^2$. In this game, the ciphertexts $\mathbf{C}_I$ are setup with messages chosen by the game. This is exactly the setting in Lemma 4.3. As there are Q_{par} oracle queries in total, there is an adversary $\mathcal{B}_7$ on QDDH with running time roughly that of $\mathcal{A}$ such that

$$|\varepsilon_6 - \varepsilon_7| \leq \mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{B}_7}(\lambda, Q_{\mathsf{par}}).$$

As consequence of Game 6 and Game 7, the $\mathbf{C}_I$ output by $\mathsf{H}_{\mathsf{par}}$ are setup in two manners. Note that Remark 4.6 also holds in this proof. That is, the ciphertexts $\mathbf{C}_I$ given by $(\mathsf{pk}_I, \mathbf{C}_I) = \mathsf{H}_{\mathsf{par}}(I)$ are setup as follows.

(1) For the forgery's common message $I = \overline{I}$, the ciphertext $\mathbf{C}_I$ encrypts the guessed message $\overline{M}$ (cf. Game 5).
(2) For other common messages $I \neq \overline{I}$, the ciphertext $\mathbf{C}_I$ encrypts M_I^*. Also, M_I^* is never extracted within $\mathcal{O}_{\mathsf{BSign}_2}$ (else the game aborts).

[9] The game initially samples $\overline{M} \leftarrow \mathbb{G}$ and outputs $\overline{M}$ on the $i_{M,\mathcal{A}}$-th H_M query.

In particular, $\mathbf{C}^* = \mathbf{C}_I - \mathbf{C}^*_M$ encrypts a non-zero value if $I = \overline{I}$ and $M \neq \overline{M}$ or if $I \neq \overline{I}$ (cf. Game 6 and Game 7).

Game 8 (Setup pk_I with known secret key). On every $\mathsf{H}_{\mathsf{par}}$ query, the game samples $\mathsf{sk}_I \leftarrow \mathbb{Z}_p$ and sets $\mathsf{pk}_{I,1} \coloneqq \mathsf{sk}_I \cdot G$. It computes $\mathbf{C}_I$ as in Game 7 and outputs $(\mathsf{pk}_I, \mathbf{C}_I)$ with $\mathsf{pk}_I \coloneqq (G, \mathsf{pk}_{I,1})$. Clearly, both games are identically distributed and we have

$$\varepsilon_7 = \varepsilon_8.$$

Game 9 (Abort if M is an invalid $\widetilde{\mathcal{R}}_{\mathsf{M}}$ witness). We now abort if the extracted message M is not a witness for relation $\widetilde{\mathcal{R}}_{\mathsf{M}}$. That is, after the game extracts M in $\mathcal{O}_{\mathsf{BSign}_2}$ from the proof π_{M} for statement $\mathbb{x}_{\mathsf{M}} = (\mathsf{pk}_I, \mathbf{C}^*_M)$, it decrypts $\mathbf{C}^*_M$ and aborts if the obtained message does not match M. More formally, the game sets $M' \leftarrow C^*_{M,1} - \mathsf{sk}_I \cdot C^*_{M,0}$ and aborts its entire execution if $M \neq M'$.

We can show that the abort probability is negligible under straightline $\widetilde{\mathcal{R}}_{\mathsf{M}}$-extractability of Π_{m}. It is easy to see that $M = M'$ iff $(\mathbb{x}_{\mathsf{M}}, M) \in \widetilde{\mathcal{R}}_{\mathsf{M}}$ (cf. Eq. (4.2)). In conclusion, we can construct an adversary $\mathcal{B}_9$ with running time roughly that of $\mathcal{A}$ such that

$$|\varepsilon_8 - \varepsilon_9| \leq \mathsf{AdvExt}^{\Pi_{\mathsf{m}},\mathsf{Ext}}_{\mathcal{B}_9}(\lambda, Q_{\Pi})$$

Our next goal is to transition to a game where the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts are computed via the known witness, and the $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcripts are simulated. For this, it is important that the statement $\mathbb{x}^*_{\mathsf{elg}} = (\mathsf{pk}_I, \mathbf{C}^*, M^*_{\$})$ is in the language $\mathcal{L}_{\mathsf{elg}}$. The abort conditions in previous games make sure that this is indeed true, except if

$$I = \overline{I} \qquad \text{and} \qquad \overline{M} = M. \tag{$\star$}$$

In case Eq. ($\star$) holds, the game still simulates the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcript.[10] Also, note that in order to compute $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts honestly, we need to find a witness for $\mathbb{x}^*_{\mathsf{elg}}$. For this, we setup $M^*_{\$}$ based on the message M^* encrypted in $\mathbf{C}^*$ and by randomizing M^* with known discrete logarithm y. Observe that then, the game knows a witness $(y \cdot \mathsf{sk}_I, y)$ for $\mathcal{R}_{\mathsf{elg}}$ in all signing sessions *except* if Eq. ($\star$) holds. We elaborate below.

Game 10 (Compute $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts honestly). Now, the game computes the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcript $(\boldsymbol{A}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}}, \boldsymbol{z}_{\mathsf{elg}})$ via the witness except if Eq. ($\star$) holds. More precisely, in $\mathcal{O}_{\mathsf{BSign}_2}$ after extracting M, the game sets $M^* \coloneqq \overline{M} - M$ if $I = \overline{I}$ and $M \neq \overline{M}$. Else, if $I \neq \overline{I}$, then it sets $M^* \coloneqq M^*_I - M$. Note that M^* is the message encrypted in $\mathbf{C}^*$. Then, the game sets $M^*_{\$} \coloneqq y \cdot M^*$ for $y \leftarrow \mathbb{Z}^{\times}_p$ and $\mathbb{w}^*_{\mathsf{elg}} \coloneqq (\mathsf{sk}_I, y)$. If otherwise Eq. ($\star$) holds, then $M^*_{\$} \leftarrow \mathbb{G}^{\times}$ is still sampled at random. Note that $\mathbb{x}^*_{\mathsf{elg}} = (\mathsf{pk}_I, \mathbf{C}^*, M^*_{\$})$ is set as before. Then, except if Eq. ($\star$) holds, the game samples $\gamma^*_{\mathsf{elg}} \leftarrow \mathbb{Z}_p$ and sets $(\boldsymbol{A}^*_{\mathsf{elg}}, \mathsf{st}_{\mathsf{elg}}) \leftarrow \mathsf{Init}_{\mathsf{elg}}(\mathbb{x}^*_{\mathsf{elg}}, \mathbb{w}^*_{\mathsf{elg}})$. Otherwise, it simulates $(\boldsymbol{A}^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}}) \leftarrow \mathsf{Sim}_{\mathsf{elg}}(\mathbb{x}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}})$ as before. In $\mathcal{O}_{\mathsf{BSign}_3}$, the

[10] In this case, both transcripts are simulated and the game is not able to answer the $\mathcal{O}_{\mathsf{BSign}_3}$ oracle. But by definition of $\overline{M}$, the game aborts its execution if this occurs.

game sets $\boldsymbol{z}^*_{\mathsf{elg}} \leftarrow \mathsf{Resp}_{\mathsf{elg}}(\mathsf{st}_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}})$ if Eq. $(\star)$ holds. All other values are computed as in Game 9. Note that Eq. $(\star)$ never occurs in $\mathcal{O}_{\mathsf{BSign}_3}$ due to the choice of $\overline{M}$ (cf. game 5).

First, let us show that $(\mathbb{x}^*_{\mathsf{elg}}, \mathbb{w}^*_{\mathsf{elg}}) \in \mathcal{R}_{\mathsf{elg}}$ (cf. Eq. (3.2)). Due to the abort condition introduced in Game 9, we know that $M = C^*_{M,1} - \mathsf{sk}_I \cdot C^*_{M,0}$. Also, recall that $\mathbf{C}^* = \mathbf{C}_I - \mathbf{C}^*_M$. Together with Remark 4.6, the above yields that

$$M^* = C^*_1 - \mathsf{sk}_I \cdot C^*_0.$$

Also, by construction we have $\mathsf{pk}_{I,1} = \mathsf{sk}_I \cdot \mathsf{pk}_{I,0}$. Multiplying both aforementioned equations by y yields that $(\mathbb{x}^*_{\mathsf{elg}}, \mathbb{w}^*_{\mathsf{elg}}) \in \mathcal{R}_{\mathsf{elg}}$. Thus, the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts $(\boldsymbol{A}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}})$ in Game 9 and Game 10 are identically distributed by HVZK. Also, by construction $M^*_{\$}$ is distributed uniform over $\mathbb{G}^\times$, as $M^* \neq 0$ (cf. Remark 4.6). In conclusion, we have

$$\varepsilon_9 = \varepsilon_{10}.$$

Game 11 (Simulate $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcripts). Now, the game simulates $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcript $(\boldsymbol{A}^*_{\mathsf{dh}}, \gamma^*_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{dh}})$ in all signing sessions. As $\boldsymbol{A}^*_{\mathsf{dh}}$ is already output in the additional $\mathcal{O}_{\mathsf{BSign}_1}$ oracle, this step slightly differs from the OMUF proof of $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$. In particular, in $\mathcal{O}_{\mathsf{BSign}_1}$, the game samples $\gamma^*_{\mathsf{dh}} \leftarrow \mathbb{Z}_p$ and sets $(\boldsymbol{A}^*_{\mathsf{dh}}, z_{\mathsf{dh}}) \leftarrow \mathsf{Sim}_{\mathsf{dh}}(\mathbb{x}^*_{\mathsf{dh}}, \gamma^*_{\mathsf{dh}})$ instead of computing $\boldsymbol{A}_{\mathsf{dh}}$ via $\mathsf{Init}_{\mathsf{dh}}$. It outputs the simulated $\boldsymbol{A}^*_{\mathsf{dh}}$. In $\mathcal{O}_{\mathsf{BSign}_2}$, the game does not sample γ^*_{elg} at random except if Eq. $(\star)$ holds. Instead, the challenger sets $\gamma^*_{\mathsf{elg}} \coloneqq \gamma^* - \gamma^*_{\mathsf{dh}}$ in $\mathcal{O}_{\mathsf{BSign}_3}$, and uses the simulated response z_{dh} computed in $\mathcal{O}_{\mathsf{BSign}_1}$. As if Eq. $(\star)$ occurs in $\mathcal{O}_{\mathsf{BSign}_3}$, the game aborts its execution, we leave the simulation behavior in $\mathcal{O}_{\mathsf{BSign}_3}$ unspecified in that case. Other than the above, the game behaves as in Game 10.

Even though the $\boldsymbol{\Sigma}_{\mathsf{dh}}$ commitment $\boldsymbol{A}^*_{\mathsf{dh}}$ is output in $\mathcal{O}_{\mathsf{BSign}_1}$, the proof is as in game 10 in the OMUF proof of $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$. If Eq. $(\star)$ does not hold in the signing session, then clearly γ^*_{elg} and γ^*_{dh} are distributed identically in Game 10 and Game 11. Further, the $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcript $(\boldsymbol{A}^*_{\mathsf{dh}}, \gamma^*_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{dh}})$ is identically distributed under HVZK. If Eq. $(\star)$ holds, then both $\boldsymbol{\Sigma}_{\mathsf{dh}}$ and $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts are simulated. Here, it suffices to argue that the $\mathcal{O}_{\mathsf{BSign}_1}$ and $\mathcal{O}_{\mathsf{BSign}_2}$ outputs $\boldsymbol{A}^*_{\mathsf{dh}}$ and $(M^*_{\$}, \boldsymbol{A}^*_{\mathsf{elg}})$, respectively, in Game 11 are distributed as in Game 10. As the distribution of $\boldsymbol{A}^*_{\mathsf{elg}}$ and $M^*_{\$}$ remains unchanged, it suffices to inspect $\boldsymbol{A}_{\mathsf{dh}}$. By HVZK, a simulated $\boldsymbol{A}_{\mathsf{dh}}$ as in Game 11 and an honestly computed $\boldsymbol{A}_{\mathsf{dh}}$ as in Game 10 are distributed identically. Consequently, we have

$$\varepsilon_{10} = \varepsilon_{11}.$$

Observe that in Game 11, the secret key $\mathsf{sk} = d_2$ is not required anymore for simulation.

Game 12 (Sample non-DDH tuple $\mathbf{D}$). In this game, we change how the vk is setup. Instead of sampling a DDH tuple $\mathbf{D}$, the game samples $\mathbf{D} \leftarrow \mathbb{G}^3$ instead. Then, the game sets $\mathsf{vk} = \mathbf{D}$ and proceeds as in Game 11.

We can construct an adversary $\mathcal{B}_{12}$ against DDH with running time similar to $\mathcal{A}$ and with

$$|\varepsilon_{11} - \varepsilon_{12}| \leq \mathsf{AdvDDH}^{\mathbb{G}}_{\mathcal{B}_{12}}(\lambda).$$

Bounding $\mathcal{A}$'s advantage in Game 12 : Denote by $\overline{\sigma}$ the forgery associated to message $\overline{M}$, *i.e.*, $\overline{\sigma} \coloneqq \overline{\sigma}_j$ for $j \in [Q_{\mathsf{frg}}]$ such that $\overline{M} = \mathsf{H}_M(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})$. Also, let us parse $\overline{\sigma} = (\overline{M_{\$}}, \overline{\pi})$ with $\overline{\pi} = (\overline{\boldsymbol{A}}_{\mathsf{elg}}, \overline{\boldsymbol{A}}_{\mathsf{dh}}, \overline{\gamma}_{\mathsf{elg}}, \overline{\gamma}_{\mathsf{dh}}, \overline{\boldsymbol{z}}_{\mathsf{elg}}, \overline{z}_{\mathsf{dh}})$. Roughly, as $\overline{\pi}$ is an OR-proof for the language $\mathcal{L}_{\mathsf{dh}} \cup \mathcal{L}_{\mathsf{elg}}$ and as $\overline{\mathbb{x}}_{\mathsf{dh}} \coloneqq (G, \mathbf{D}) \notin \mathcal{L}_{\mathsf{dh}}$ except with probability $1/p$, it must hold that $\overline{\mathbb{x}}_{\mathsf{elg}} \coloneqq (\mathsf{pk}_I, \overline{\mathbf{C}}, \overline{M_{\$}}) \in \mathcal{L}_{\mathsf{elg}}$ by soundness of π, where $\overline{\mathbf{C}} = \overline{\mathbf{C}_I} - \overline{\mathbf{C}_M}$ with $\overline{\mathbf{C}_M} = (0, \overline{M})$. Further, as $\overline{M_{\$}} \neq 0$, the ciphertext $\overline{\mathbf{C}}$ is not an encryption of 0 But as both $\mathbf{C}_I$ and $\overline{\mathbf{C}_M}$ encrypt $\overline{M}$ by construction, $\mathcal{A}$ cannot win except with negligible probability. This can be shown as in final paragraph Theorem 4.5's proof and we omit further details. We have

$$\varepsilon_{12} \leq \frac{1 + Q_{\mathsf{ch}}}{p}.$$

Lemma E.1 follows by collecting all above bounds. □

E.8 Proof of Lemma E.2

Proof. Recall that in this case, there are distinct $j, k \in [Q_{\mathsf{frg}}]$ such that the j-th and k-th forgery share the same message and $\boldsymbol{\Sigma}_{\mathsf{dh}}$ commitments, but the $\boldsymbol{\Sigma}_{\mathsf{dh}}$ challenges are distinct. Then, the transcripts $\pi_{\mathsf{dh},j}$ $\pi_{\mathsf{dh},k}$ fulfil the conditions to invoke special soundness. In particular, we obtain a witness $\mathbb{w}$ that $\mathbb{x}_{\mathsf{dh}} = (G, \mathbf{D}) \in \mathcal{L}_{\mathsf{dh}}$. This allows to break the DLog assumption if we manage to sign without $\mathbb{w}$. For this, it suffices to sign via the nez branch instead of the dh branch.

Game 1 (Honest). The game is identical to Game 0.2 and we have

$$\varepsilon_{0.2} = \varepsilon_1.$$

Game 2 (Extract M from π_{M}). Before sending vk to $\mathcal{A}$, the game sets $\mathsf{crs}_{\mathsf{M}} \leftarrow \mathsf{ExtSetup}(1^\lambda)$ and programs $\mathsf{H}_{\mathsf{crs}}(0) \coloneqq \mathsf{crs}_{\mathsf{M}}$. Later, on every $\mathcal{O}_{\mathsf{BSign}_2}$ query of the form $(\mathsf{sid}, I, \mathbf{C}^*_M, \pi_{\mathsf{M}})$, after verifying π_{M}, the adversary extracts the message $M \leftarrow \mathsf{Ext}((\mathsf{td}, \mathcal{Q}), (\mathbb{x}_{\mathsf{M}}, \pi_{\mathsf{M}}))$ via π_{M}. Here, $\mathcal{Q}$ is a list containing all H_Π so far.

Note that while we already extract the message, we do not use the extracted value within the simulation yet. It is straightforward to construct a reduction $\mathcal{B}_2$ with running time similar to $\mathcal{A}$ such that

$$|\varepsilon_2 - \varepsilon_3| \leq \mathsf{AdvCRS}^{\Pi_{\mathsf{m}}, \mathsf{ExtSetup}}_{\mathcal{B}_2}(\lambda, Q_\Pi).$$

Game 3 (Abort if M^*_I is extracted). Initially, the game samples a random message $M^*_I \leftarrow \mathbb{G}$. Then, the game aborts its entire execution if M^*_I is extracted from π_{M} in $\mathcal{O}_{\mathsf{BSign}_2}$ for any common message I. That is, after setting $M \leftarrow \mathsf{Ext}((\mathsf{td}, \mathcal{Q}), (\mathbb{x}, \pi_{\mathsf{M}}))$ in $\mathcal{O}_{\mathsf{BSign}_2}$, the game checks if $M = M^*_I$. If so, the game aborts its entire execution, else it continues as before.

As M^*_I is never used within in the simulation (except for the abort condition), a union bound yields

$$|\varepsilon_2 - \varepsilon_3| \leq \frac{Q_S}{p}.$$

Game 4 (Encrypt M_I^* in $\mathbf{C}_I$). We now encrypt M_I^* in all ciphertexts $\mathbf{C}_I$ output by $\mathsf{H}_{\mathsf{par}}$. That is, on every fresh $\mathsf{H}_{\mathsf{par}}$ query, the game sets up $\mathsf{pk}_I \leftarrow \{G\} \times \mathbb{G}^\times$ at random and encrypts M_I^* in $\mathbf{C}_I$, *i.e.*, computes ElGamal ciphertext $\mathbf{C}_I \coloneqq (0, M_I^*) + t \cdot \mathsf{pk}_I$ for $t \leftarrow \mathbb{Z}_p$. The game outputs $(\mathsf{pk}_I, \mathbf{C}_I)$.

Recall that in the previous game, all $\mathsf{H}_{\mathsf{par}}$ outputs $\mathbf{C}_I$ are uniform over $\mathbb{G}^2$. In this game, the ciphertexts $\mathbf{C}_I$ are setup with a message chosen by the game. This is exactly the setting in Lemma 4.3. As there are Q_{par} oracle queries in total, there is an adversary $\mathcal{B}_4$ on QDDH with running time roughly that of $\mathcal{A}$ such that

$$|\varepsilon_3 - \varepsilon_4| \leq \mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{B}_4}(\lambda, Q_{\mathsf{par}}).$$

As consequence of Game 3 and Game 4, the $\mathbf{C}_I$ ciphertexts output by $\mathsf{H}_{\mathsf{par}}$ all encrypt M_I^*.

Game 5 (Abort if M is an invalid $\widetilde{\mathcal{R}}_{\mathsf{M}}$ witness). We now abort if the extracted message M is not a witness for relation $\widetilde{\mathcal{R}}_{\mathsf{M}}$. That is, after the game extracts M in $\mathcal{O}_{\mathsf{BSign}_2}$ from the proof π_{M} for statement $\mathbb{x}_{\mathsf{M}} = (\mathsf{pk}_I, \mathbf{C}_M^*)$, it decrypts $\mathbf{C}_M^*$ and aborts if the obtained message does not match M. More formally, the game sets $M' \leftarrow C_{M,1}^* - \mathsf{sk}_I \cdot C_{M,0}^*$ and aborts its entire execution if $M \neq M'$.

We can show that the abort probability is negligible under straightline $\widetilde{\mathcal{R}}_{\mathsf{M}}$-extractability of Π_{m}. It is easy to see that $M = M'$ iff $(\mathbb{x}_{\mathsf{M}}, M) \in \widetilde{\mathcal{R}}_{\mathsf{M}}$ (cf. Eq. (4.2)). In conclusion, we can construct an adversary $\mathcal{B}_5$ with running time roughly that of $\mathcal{A}$ such that

$$|\varepsilon_4 - \varepsilon_5| \leq \mathsf{AdvExt}^{\Pi_{\mathsf{m}},\mathsf{Ext}}_{\mathcal{B}_5}(\lambda, Q_{\Pi})$$

Game 6 (Setup pk_I with known secret key). On every $\mathsf{H}_{\mathsf{par}}$ query, the game samples $\mathsf{sk}_I \leftarrow \mathbb{Z}_p$ and sets $\mathsf{pk}_{I,1} \coloneqq \mathsf{sk}_I \cdot G$. It computes $\mathbf{C}_I$ as in Game 5 and outputs $(\mathsf{pk}_I, \mathbf{C}_I)$ with $\mathsf{pk}_I \coloneqq (G, \mathsf{pk}_{I,1})$. Clearly, both games are identically distributed and we have

$$\varepsilon_5 = \varepsilon_6.$$

Our next goal is to transition to a game where the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts are computed via the known witness, and the $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcripts are simulated. As in previous proofs, we need that $\mathbb{x}^*_{\mathsf{elg}} = (\mathsf{pk}_I, \mathbf{C}^*, M^*_{\$})$ is in the language $\mathcal{L}_{\mathsf{elg}}$. The abort conditions in game 3 ensures this.

Game 7 (Compute $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts honestly). Now, the game computes the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcript $(\boldsymbol{A}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}}, \boldsymbol{z}_{\mathsf{elg}})$ via the witness. More precisely, in $\mathcal{O}_{\mathsf{BSign}_2}$ after extracting M, the game computes the message $M^* \coloneqq M_I^* - M$ encrypted in $\mathbf{C}^*$. Then, the game sets $M^*_{\$} \coloneqq y \cdot M^*$ for $y \leftarrow \mathbb{Z}_p^\times$ and $\mathbb{w}^*_{\mathsf{elg}} \coloneqq (\mathsf{sk}_I, y)$, and sets $\mathbb{x}^*_{\mathsf{elg}} = (\mathsf{pk}_I, \mathbf{C}^*, M^*_{\$})$ as before. Next, the game samples $\gamma^*_{\mathsf{elg}} \leftarrow \mathbb{Z}_p$ and sets $(\boldsymbol{A}^*_{\mathsf{elg}}, \mathsf{st}_{\mathsf{elg}}) \leftarrow \mathsf{Init}_{\mathsf{elg}}(\mathbb{x}^*_{\mathsf{elg}}, \mathbb{w}^*_{\mathsf{elg}})$, and then proceeds as before. In $\mathcal{O}_{\mathsf{BSign}_3}$, the game sets $\boldsymbol{z}^*_{\mathsf{elg}} \leftarrow \mathsf{Resp}_{\mathsf{elg}}(\mathsf{st}_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}})$.

We can show, *e.g.*, as in the proof of Lemma E.1 (cf. game 10) that $(\mathbb{x}^*_{\mathsf{elg}}, \mathbb{w}^*_{\mathsf{elg}}) \in \mathcal{R}_{\mathsf{elg}}$ and $M^* \in \mathbb{G}^\times$. By HVZK, the $\boldsymbol{\Sigma}_{\mathsf{elg}}$ transcripts $(\boldsymbol{A}^*_{\mathsf{elg}}, \gamma^*_{\mathsf{elg}}, \boldsymbol{z}^*_{\mathsf{elg}})$ and $M^*_{\$}$ in Game 6 and Game 7 are identically distributed. In conclusion, we have

$$\varepsilon_6 = \varepsilon_7.$$

Game 8 (Simulate $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcripts). Now, the game simulates $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcript $(\boldsymbol{A}^*_{\mathsf{dh}}, \gamma^*_{\mathsf{dh}}, \boldsymbol{z}_{\mathsf{dh}})$ in all signing sessions. The changes are as in game 11 in the proof of Lemma E.1 and we omit details. As previously, we have

$$\varepsilon_7 = \varepsilon_8.$$

Observe that in Game 8, the secret key $\mathsf{sk} = d_2$ is not required anymore for simulation.

Bounding $\mathcal{A}$'s advantage in Game 8: We construct an adversary $\mathcal{B}$ on the DLog assumption. In particular, $\mathcal{B}$ obtains DLog challenge $X \in \mathbb{G}$. Then, $\mathcal{B}$ simulates Game 8 to $\mathcal{A}$ except that it sets up $\mathsf{vk} = (D_1, D_2, D_3)$ with $D_1 \coloneqq d_1 G$, $D_2 \coloneqq X$ and $D_3 \coloneqq d_1 \cdot D_2$, where $d_1 \leftarrow \mathbb{Z}_p$. Finally, $\mathcal{A}$ outputs forgeries such that Eq (II) holds. That is, there are distinct $j, k \in [Q_{\mathsf{frg}}]$ such that $\overline{\mu}_j = \overline{\mu}_k$ and $\overline{\boldsymbol{A}}_{\mathsf{dh},j} = \overline{\boldsymbol{A}}_{\mathsf{dh},k}$ but $\overline{\gamma}_{\mathsf{dh},j} \neq \overline{\gamma}_{\mathsf{dh},k}$. Adversary $\mathcal{B}$ sets $\mathbb{w}_{\mathsf{dh}} \leftarrow \mathsf{Ext}_{\mathsf{dh}}(\mathbb{x}_{\mathsf{dh}}, \overline{\pi}_{\mathsf{dh},j}, \overline{\pi}_{\mathsf{dh},k})$ and outputs $\mathbb{w}_{\mathsf{dh}}$, where $\mathbb{x}_{\mathsf{dh}} = (G, \mathbf{D})$ and $\mathsf{Ext}_{\mathsf{dh}}$ is the extractor of $\boldsymbol{\Sigma}_{\mathsf{dh}}$ due to special soundness (cf. Definition 2.4).

If $\mathcal{A}$ is successful, then $\overline{\pi}_{\mathsf{dh},j}$ and $\overline{\pi}_{\mathsf{dh},k}$ share $\boldsymbol{\Sigma}_{\mathsf{dh}}$ commitment $\overline{\boldsymbol{A}}_{\mathsf{dh},j} = \overline{\boldsymbol{A}}_{\mathsf{dh},k}$ but have distinct challenges $\overline{\gamma}_{\mathsf{dh},j} \neq \overline{\gamma}_{\mathsf{dh},k}$. Thus, it holds that $(\mathbb{w}_{\mathsf{dh}}, \mathbb{x}_{\mathsf{dh}}) \in \mathcal{R}_{\mathsf{dh}}$ and consequently, the value $\mathbb{w}_{\mathsf{dh}}$ is the DLog of $D_2 = X$. Consequently, we have

$$\varepsilon_8 \leq \mathsf{AdvDLOG}^{\mathbb{G}}_{\mathcal{B}}(\lambda).$$

Lemma E.2 follows by collecting all above bounds. □

E.9 Proof of Lemma E.3

Proof. Recall that in in Game 0.3, the j-th and k-th forgery share the same message $\overline{\mu}_j = \overline{\mu}_k$, $\boldsymbol{\Sigma}_{\mathsf{dh}}$ commitments $\overline{\boldsymbol{A}}_{\mathsf{dh},j} = \overline{\boldsymbol{A}}_{\mathsf{dh},k}$ and $\boldsymbol{\Sigma}_{\mathsf{dh}}$ challenges $\overline{\gamma}_{\mathsf{dh},j} = \overline{\gamma}_{\mathsf{dh},k}$. Then as $\boldsymbol{\Sigma}_{\mathsf{dh}}$ has unique responses (cf. Definition 2.8), it follows that both $\boldsymbol{\Sigma}_{\mathsf{dh}}$ transcripts are identical, *i.e.*, $\overline{\pi}_{\mathsf{dh},j} = \overline{\pi}_{\mathsf{dh},k}$. Intuitively, this means that the adversary does not "use" the dh branch to produce the forgery. Thus, the adversary must use the elg branch in a non-trivial manner to come up with the forgeries. If we now puncture the elg branch for $\overline{M} \coloneqq \mathsf{H}_M(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})$, then this should not be possible. We formalize this intuition below.

Game 1 (Honest). The game is identical to Game 0.3 and we have

$$\varepsilon_{0.3} = \varepsilon_1.$$

Game 2 (Guess duplicated $\overline{M}$ in forgery). We guess the first query $i_{M,\mathcal{A}}$ to H_M such that the following condition holds:

(1) The input $X_{i_{M,\mathcal{A}}}$ to the $i_{M,\mathcal{A}}$-th H_M query was never queried to H_M beforehand.
(2) The input $X_{i_{M,\mathcal{A}}}$ is part of $\mathcal{A}$'s forgeries twice and fulfils III, *i.e.*, there exists distinct $j, k \in [Q_{\mathsf{frg}}]$ such that $X_{i_{M,\mathcal{A}}} = (\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j}) = (\overline{\mu}_k, \overline{\boldsymbol{A}}_{\mathsf{dh},k})$ and $\overline{\gamma}_{\mathsf{dh},j} = \overline{\gamma}_{\mathsf{dh},k}$.

Again, the game aborts its execution if the guess was incorrect.

If $\mathcal{A}$ is successful, then such a query must exists by Eq. (III) and since the guess is hidden from $\mathcal{A}$, we have

$$\varepsilon_1 \leq Q_M \cdot \varepsilon_2.$$

In the following, we denote by $\overline{M}$ the output of the $i_{M,\mathcal{A}}$-th H_M query. As before, we can assume that $\overline{M}$ is known by the game from the start on if $\mathcal{A}$ is successful.

Game 3 (Encrypt $\overline{M}$ in $\mathbf{C}_I$). We now encrypt $\overline{M}$ in all ciphertexts $\mathbf{C}_I$ output by $\mathsf{H}_{\mathsf{par}}$. That is, on every fresh $\mathsf{H}_{\mathsf{par}}$ query, the game sets up $\mathsf{pk}_I \leftarrow \{G\} \times \mathbb{G}^\times$ at random and encrypts $\overline{M}$ in $\mathbf{C}_I$, *i.e.*, computes ElGamal ciphertext $\mathbf{C}_I \coloneqq (0, \overline{M}) + t \cdot \mathsf{pk}_I$ for $t \leftarrow \mathbb{Z}_p$. The game outputs $(\mathsf{pk}_I, \mathbf{C}_I)$.

Recall that in the previous game, all $\mathsf{H}_{\mathsf{par}}$ outputs $\mathbf{C}_I$ are uniform over $\mathbb{G}^2$. In this game, the ciphertexts $\mathbf{C}_I$ are setup with a message chosen by the game. This is exactly the setting in Lemma 4.3. As there are Q_{par} oracle queries in total, there is an adversary $\mathcal{B}_3$ on QDDH with running time roughly that of $\mathcal{A}$ such that

$$|\varepsilon_2 - \varepsilon_3| \leq \mathsf{AdvQDDH}^{\mathbb{G}}_{\mathcal{B}_3}(\lambda, Q_{\mathsf{par}}).$$

As consequence of Game 2 and Game 3, the $\mathbf{C}_I$ ciphertexts output by $\mathsf{H}_{\mathsf{par}}$ all encrypt $\overline{M}$.

Bounding $\mathcal{A}$'s advantage in Game 3: Let us assume that $\mathcal{A}$ is successful in Game 3. Then, there are two distinct indices $j, k \in [Q_{\mathsf{frg}}]$ such that $(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j}) = (\overline{\mu}_k, \overline{\boldsymbol{A}}_{\mathsf{dh},k})$ and $\overline{\gamma}_{\mathsf{dh},j} = \overline{\gamma}_{\mathsf{dh},k}$. Also, we know that $\mathbf{C}_I$ encrypts $\overline{M} = \mathsf{H}_M(\overline{\mu}_j, \overline{\boldsymbol{A}}_{\mathsf{dh},j})$.

Denote $\overline{\mathbf{C}} \coloneqq \mathbf{C}_I - (0, \overline{M}) = t \cdot \mathsf{pk}_I$ for $t \in \mathbb{Z}_p$ as sampled in $\mathsf{H}_{\mathsf{par}}$. By construction, the statements $\overline{\mathbb{x}}_{\mathsf{elg},j} = (\mathsf{pk}_I, \overline{\mathbf{C}}, \overline{M_{\$}}_j)$ and $\overline{\mathbb{x}}_{\mathsf{elg},k} = (\mathsf{pk}_I, \overline{\mathbf{C}}, \overline{M_{\$}}_k)$ are associated to $\overline{\pi}_{\mathsf{elg},j}$ and $\overline{\pi}_{\mathsf{elg},k}$, respectively. By $\boldsymbol{\Sigma}_{\mathsf{dh}}$'s unique response property (cf. Definition 2.8), it holds that $\overline{\pi}_{\mathsf{dh},j} = \overline{\pi}_{\mathsf{dh},k}$. Let us denote $\overline{\boldsymbol{A}}_{\mathsf{dh}} \coloneqq \overline{\boldsymbol{A}}_{\mathsf{dh},j}$ and $\overline{\gamma}_{\mathsf{dh}} \coloneqq \overline{\gamma}_{\mathsf{dh},j}$. We know that

$$\overline{\gamma}_j = \mathsf{H}_{\mathsf{ch}}(\mathbb{x}_{\mathsf{dh}}, \overline{\mathbb{x}}_{\mathsf{elg},j}, \overline{\boldsymbol{A}}_{\mathsf{elg},j}, \overline{\boldsymbol{A}}_{\mathsf{dh}}) = \overline{\gamma}_{\mathsf{elg},j} + \overline{\gamma}_{\mathsf{dh}}, \tag{E.1}$$

$$\overline{\gamma}_k = \mathsf{H}_{\mathsf{ch}}(\mathbb{x}_{\mathsf{dh}}, \overline{\mathbb{x}}_{\mathsf{elg},k}, \overline{\boldsymbol{A}}_{\mathsf{elg},k}, \overline{\boldsymbol{A}}_{\mathsf{dh}}) = \overline{\gamma}_{\mathsf{elg},k} + \overline{\gamma}_{\mathsf{dh}}, \tag{E.2}$$

where $\mathbb{x}_{\mathsf{dh}} = (G, \mathbf{D})$. As $(\overline{\mu}_j, \overline{\sigma}_j) \neq (\overline{\mu}_k, \overline{\sigma}_k)$, we know by $\boldsymbol{\Sigma}_{\mathsf{elg}}$'s unique response property that $(\overline{M_{\$}}_j, \overline{\boldsymbol{A}}_{\mathsf{elg},j}) \neq (\overline{M_{\$}}_k, \overline{\boldsymbol{A}}_{\mathsf{elg},k})$ must hold.[11] Consequently, we have that $\overline{\gamma}_j$ and $\overline{\gamma}_k$ are distributed independently. As both transcripts $\overline{\pi}_{\mathsf{elg},j}$ and $\overline{\pi}_{\mathsf{elg},k}$ are valid and as $\overline{M_{\$}}_j, \overline{M_{\$}}_k \in \mathbb{G}^\times$, it must hold that $\overline{\gamma}_{\mathsf{elg},j}$ and $\overline{\gamma}_{\mathsf{elg},k}$ are the unique challenges specified as in Lemma 5.3. By substracting Eq. (E.2) from Eq. (E.1), we obtain

$$\overline{\gamma}_j - \overline{\gamma}_k = \overline{\gamma}_{\mathsf{elg},j} - \overline{\gamma}_{\mathsf{elg},k}.$$

[11] As $\mu_j = \mu_k$, we have $\overline{\sigma}_j \neq \overline{\sigma}_k$. If we had $(\overline{M_{\$}}_j, \overline{\boldsymbol{A}}_{\mathsf{elg},j}) = (\overline{M_{\$}}_k, \overline{\boldsymbol{A}}_{\mathsf{elg},k})$, then as $\boldsymbol{A}_{\mathsf{dh},j} = \boldsymbol{A}_{\mathsf{dh},k}$ are identical, the challenges $\overline{\gamma}_j$ and $\overline{\gamma}_k$ derived via H_{ch} as in Eqs. (E.1) and (E.2) were identical. Then, as $\gamma_{\mathsf{dh},j} = \gamma_{\mathsf{dh},k}$, we have $\gamma_{\mathsf{elg},j} = \gamma_{\mathsf{elg},k}$. The unique response property of $\boldsymbol{\Sigma}_{\mathsf{elg}}$ now implies $\overline{\sigma}_j = \overline{\sigma}_k$ and yields a contradiction.

As the right side is determined by the input to H_{ch} and the H_{ch}-outputs on the left side is uniform, the probability that such a pair of H_{ch} queries exists is at most Q_{ch}^2/p. This follows, *e.g.*, via a simple union bound. In total, we have

$$\varepsilon_3 \leq \frac{Q_{\mathsf{ch}}^2}{p}.$$

Lemma E.3 follows by collecting all above bounds. □

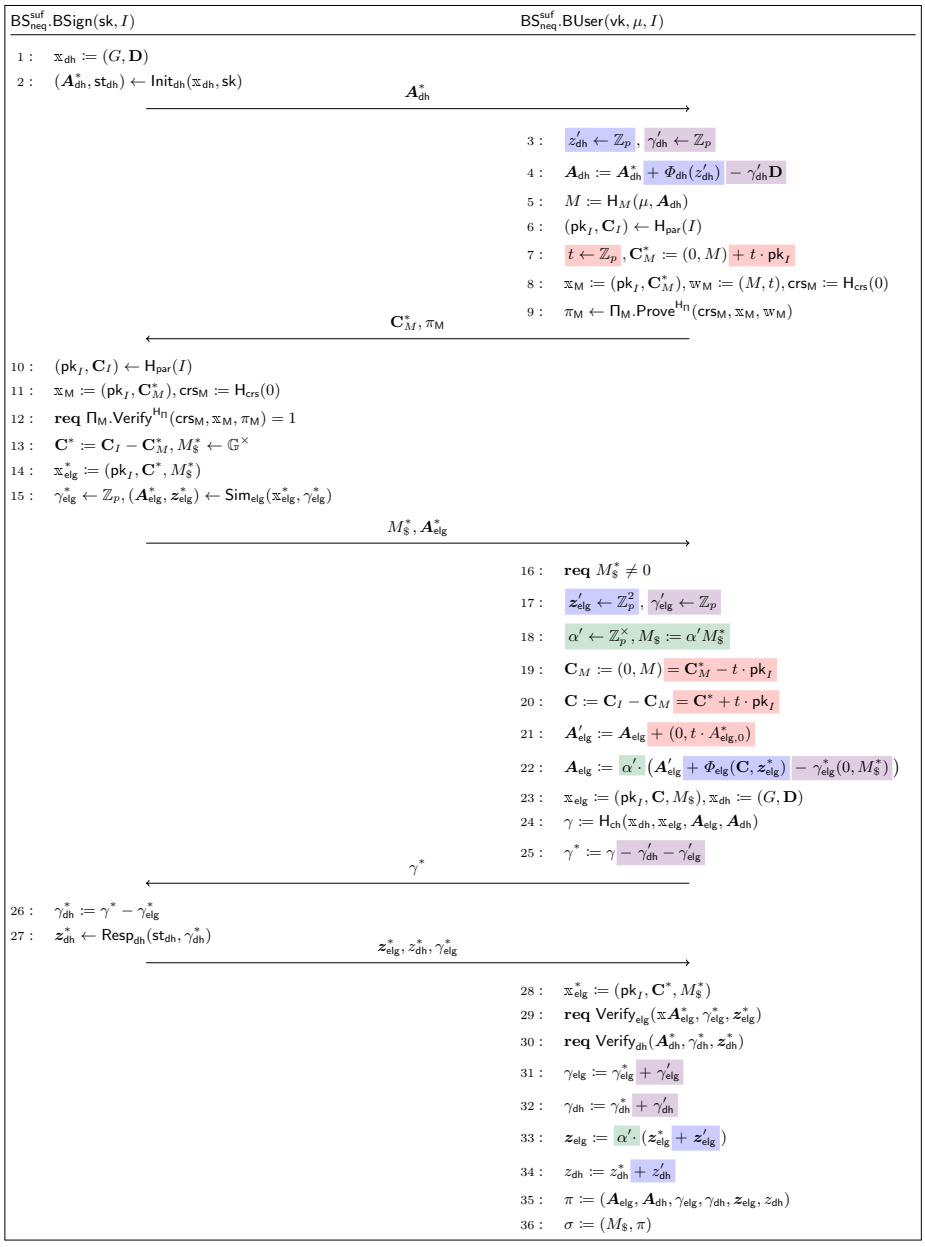

Fig. 2: The signing session for $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ for message $\mu \in \{0,1\}^*$ and common message $I \in \{0,1\}^*$. The signer and user abort (*i.e.*, output $\perp$) if $\mathbf{req}\ C$ is evaluated for a false condition C. Recall that $\mathsf{sk} = d_2$ is a witness for $\mathcal{L}_{\mathsf{dh}}$ membership of $\mathsf{vk} = \mathbf{D}$. We follow the color conventions in Fig. 1. The main difference between $\mathsf{BS}^{\mathsf{suf}}_{\mathsf{neq}}$ and $\mathsf{BS}^{\mathsf{uf}}_{\mathsf{neq}}$ is that the message M is derived from H_M on input $(\mu, \boldsymbol{A}_{\mathsf{dh}})$ instead of just μ.