Practical Round-Optimal Blind Signatures in the ROM from Standard Assumptions

Shuichi Katsumata[1], Michael Reichle[2], and Yusuke Sakai[3]

[1] PQShield Ltd. and AIST, Japan
[2] ETH Zürich, Switzerland. Work done while employed at Inria, Paris.
[3] AIST, Japan

Blind signatures serve as a foundational tool for privacy-preserving applications and have recently seen renewed interest due to new applications in blockchains and privacy-authentication tokens. With this, constructing practical *round-optimal* (i.e., signing consists of the minimum two rounds) blind signatures in the random oracle model (ROM) has been an active area of research, where several impossibility results indicate that either the ROM or a trusted setup is inherent.

In this work, we present two round-optimal blind signatures under standard assumptions in the ROM with different approaches: one achieves the smallest sum of the signature and communication sizes, while the other achieves the smallest signature size. Both of our instantiations are based on standard assumptions over asymmetric pairing groups, i.e., CDH, DDH, and/or SXDH. Our first construction is a highly optimized variant of the generic blind signature construction by Fischlin (CRYPTO'06) and has signature and communication sizes 447 B and 303 B, respectively. We progressively weaken the building blocks required by Fischlin and we result in the first blind signature where the sum of the signature and communication sizes fit below 1 KB based on standard assumptions. Our second construction is a semi-generic construction from a specific class of randomizable signature schemes that admits an *all-but-one* reduction. The signature size is only 96 B while the communication size is 2.2 KB. This matches the previously known smallest signature size while improving the communication size by several orders of magnitude. Finally, both of our constructions rely on a (non-black box) fine-grained analysis of the forking lemma that may be of independent interest.

1 Introduction

1.1 Background

Blind signature is an interactive signing protocol between a signer and a user with advanced privacy guarantees. At the end of the protocol, the user obtains a signature for his choice of message while the signer remains blind to the message she signed. To capture the standard notion of unforgeability, it is further required that a user interacting with the signer at most ℓ-times is not be able to produce valid signatures on more than ℓ distinct messages. The former and latter are coined as the *blindness* and *one-more unforgeability* properties, respectively.

Chaum introduced the notion of blind signatures [31] and showed its application to e-cash [31, 33, 69]. Since then, it has been an important building block for other applications such as anonymous credentials [24, 29], e-voting [32, 44], direct anonymous attenstation [25], and in more recent years, it has seen a renewed interest due to new applications in blockchains [83, 28] and privacy-preserving authentication tokens [81, 53].

Round-Optimality. One of the main performance measures for blind signatures is *round-optimality*, where the user and signer are required to only send one message each to complete the signing protocol. While this is an ideal feature for practical applications, unfortunately, there are a few impossibility results [64, 39, 70] on constructing round-optimal blind signatures in the plain model (i.e., without any trusted setup) from standard assumptions (*e.g.*, non-interactive assumptions and polynomial hardness). To circumvent this, cryptographers design round-optimal blind signatures by making a minimal relaxation of relying on the random oracle model (ROM) or the trusted setup model. Considering that trusted setups are a large obstacle for real-world deployment, in this work we focus on round-optimal blind signatures in the ROM under standard assumption[4]. We refer

[4] We note that all of our results favor well even when compared with schemes in the trusted setup model.

the readers to Appendix A for an overview on round optimal blind signatures under non-standard assumptions (*e.g.*, interactive or super polynomial hardness) or relying on stronger idealized models such as the generic group model.

Practical Round-Optimal Blind Signatures. Constructing a *practical* round-optimal blind signature has been an active area of research. In a seminal work, Fischlin [38] proposed the first generic round-optimal blind signature from standard building blocks. While the construction is simple, an efficient instantiation remained elusive since it required a non-interactive zero-knowledge (NIZK) proof for a relatively complex language.

Recently, in the lattice-setting, del Pino and Katsumata [34] showed a new lattice-tailored technique to overcome the inefficiency of Fischlin's generic construction and proposed a round-optimal blind signature with signature and communication sizes 100 KB and 850 KB.

A different approach that has recently accumulated attention is based on the work by Pointcheval [72] that bootstraps a specific class of blind signature schemes into a fully secure one (i.e., one-more unforgeable even if polynomially many concurrent signing sessions are started). This approach has been improved by Katz et al. [60] and Chairattana-Apirom et al. [30], and the very recent work by Hanzlik et al. [51] optimized this approach leading to a round-optimal blind signature based on the CDH assumption in the asymmetric pairing setting. One of their parameter settings provides a short signature size of 5 KB with a communication size 72 KB.

Finally, there are two constructions in the pairing setting with a trusted setup which can be instantiated in the ROM under standard assumptions [18, 4][5]. Blazy et al. [18] exploited the randomizability of Waters signature [82] and constructed a blinded version of Waters signature consisting of mere 2 group elements, i.e. 96 B. While it achieves the shortest signature size in the literature, since the user has to prove some relation to his message in a bit-by-bit manner, the communication scales linearly in the message length. For example for 256 bit messages, it requires more than 220 KB in communication. Abe et al. [4] use structure-preserving signatures (SPS) and Groth-Ostrovsky-Sahai (GOS) proofs [50] to instantiate the Fischlin blind signature with signatures of size 5.8 KB with around 1 KB of communication.

While round-optimal blind signatures in the ROM are coming close to the practical parameter regime, the signature and communication sizes are still orders of magnitude larger compared to those relying on non-standard assumptions or strong idealized models such as blind RSA [31, 13] or blind BLS [19]. Thus, we continue the above line of research to answer the following question:

How efficient can round-optimal blind signatures in the ROM be under standard assumptions?

1.2 Contributions

We present two round-optimal blind signatures based on standard group-based assumptions in the asymmetric pairing setting. The efficiency is summarized in Table 1, along with the assumptions we rely on. The first construction has signature and communication sizes 447 B and 303 B, respectively. It has the smallest communication size among all prior schemes and is the first construction where the sum of the signature and communication sizes fit below 1 KB. The second construction has signature and communication sizes 96 B and 2.2 KB, respectively. While it has a larger communication size compared to our first construction, the signature only consists of 2 group elements, matching the previously shortest by Blazy et al. [18] while simultaneously improving their communication size by around two orders of magnitude. Both constructions have efficient partially blind variants.

For our first construction, we revisit the generic blind signature construction by Fischlin [37]. We progressively weaken the building blocks required by Fischlin and show that the blind signature can be instantiated much more efficiently in the ROM than previously thought by a careful choice of the building blocks. At a high level, we show that the generic construction remains secure even if we replace the public-key encryption scheme (PKE) and online-extractable NIZK[6] with respectively a commitment scheme and a rewinding-extractable NIZK such as those offered by the standard

[5] Both [18, 4] require a trusted setup for a common reference string crs consisting of random group elements. We can remove the trusted setup by using a random oracle to sample crs.

[6] This is a type of NIZK where the extractor can extract a witness from the proofs output by the adversary in an *on-the-fly* manner.

Table 1. Comparison of Round-Optimal Blind Signatures in the ROM

Reference	Signature size	Communication size	Assumption
del Pino et al. [34]	100 KB	850 KB	DSMR, MLWE, MSIS
Blazy et al. [18]	96 B	220 KB †	SXDH, CDH
Abe et al. [4]	5.5 KB	1 KB	SXDH
Hanzlik et al. [51]‡	5 KB 9 KB	72 KB 36 KB	CDH
Ours: Section 3	447 B	303 B	SXDH
Ours: Section 5	96 B	2.2 KB	DDH, CDH

All group-based assumptions are in the asymmetric paring setting, and MLWE and MSIS denote the module version of the standard LWE and SIS, respectively. DSMR denotes the decisional small matrix ratio problem, which can be viewed as the module variant of the standard NTRU. (†): Communication of [18] scales linearly with the message size, and is given here for 256 bit messages. (‡): [51] offers tradeoffs between signature and communication sizes.

Fiat-Shamir transform [36, 73, 14]. While these modifications may seem insignificant on the surface, it accumulates in a large saving in the concrete signature and communication sizes. Moreover, our security proof requires overcoming new technical hurdles incurred by the rewinding-extraction and relies on a fined-grained analysis of a variant of the forking lemma.

For our second construction, we revisit the idea by Blazy et al. [18] relying on randomizable signatures. However, our technique is not a simple application of their idea as their construction relies on the specific structure of the Waters signature in a non-black-box manner. Our new insight is that a specific class of signature schemes with an *all-but-one* (ABO) reduction can be used in an almost black-box manner to construct round-optimal blind signatures, where ABO reductions are standard proof techniques to prove selective security of public-key primitives (see references in [67] for examples). Interestingly, we can cast the recent blind signature by del Pino and Katsumata [34] that stated to use lattice-tailored techniques as one instantiation of our methodology.

In the instantiation of our second construction, we use the Boneh-Boyen signature [20] that comes with an ABO reduction along with an online-extractable NIZK obtained via the Fiat-Shamir transform applied to Bulletproofs [27] and a Σ-protocol for some ElGamal related statements. To the best of our knowledge, this is the first time an NIZK that internally uses Bulletproofs was proven to be online-extractable in the ROM. Prior works either showed the non-interactive version of Bulletproofs to achieve the weaker rewinding extractability [9, 8] or the stronger online simulation extractability by further assuming the algebraic group model [45]. We believe the analysis of our online extractability to be novel and may be of independent interest.

1.3 Technical Overview

We give an overview of our contributions.

Fischlin's Round-Optimal Blind Signature. We review the generic construction by Fischlin [37] as it serves as a starting point for both of our constructions. The construction relies on a PKE, a signature scheme, and an NIZK. The blind signature's verification and signing keys $(\mathsf{bvk}, \mathsf{bsk})$ are identical to those of the underlying signature scheme $(\mathsf{vk}, \mathsf{sk})$. For simplicity, we assume a perfect correct PKE with uniform random encryption keys ek and that ek is provided to all the players as an output of the random oracle. The user first sends an encryption $c \leftarrow \mathsf{PKE}(\mathsf{ek}, m; r)$ of the message m. The signer then returns a signature $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, c)$ on the ciphertext c. The user then encrypts $\widehat{c} \leftarrow \mathsf{PKE}(\mathsf{ek}, c\|r\|\sigma; \widehat{r})$ and generates an NIZK proof π of the following fact where $(c, \sigma, r, \widehat{r})$ is the witness: $\widehat{c}$ encrypts (c, r, σ) under $\widehat{r}$; c encrypts the message m under r; and σ is a valid signature on c. The user outputs the blind signature $\sigma_{\mathsf{BS}} = (\widehat{c}, \pi)$.

It is not hard to see that the scheme is blind under the IND-CPA security of the PKE and the zero-knowledge property of the NIZK. The one-more unforgeability proof is also straight-forward: The reduction will use the adversary $\mathcal{A}$ against the one-more unforgeability game to break the `euf-cma` of the signature scheme. The reduction first programs the random oracle so that it knows the corresponding decryption key dk of the PKE. When $\mathcal{A}$ submits c to the blind signing

oracle, the reduction relays this to its signing oracle and returns $\mathcal{A}$ the signature σ it obtains. Moreover, it makes a list L of decrypted messages $m \leftarrow \mathsf{Dec}(\mathsf{dk}, c)$. When $\mathcal{A}$ outputs the forgeries $(\sigma_{\mathsf{BS},\mathsf{i}} = (\widehat{c}_i, \pi_i), m_i)_{i\in[\ell+1]}$, it searches a m_i such that $m_i \notin L$, which is guaranteed to exist since there are at most ℓ signing queries. The reduction then decrypts $(c_i, r_i, \sigma_i) \leftarrow \mathsf{Dec}(\mathsf{dk}, \widehat{c}_i)$. Since the PKE is perfectly correct and due to the soundness of the NIZK, c_i could not have been queried by $\mathcal{A}$ as otherwise $m_i \in L$, and hence, (c_i, σ_i) breaks `euf-cma` security.

Source of Inefficiency. There are two sources of inefficiency when trying to instantiate this generic construction. One is the use of a *layered* encryption: the NIZK needs to prove that c is a valid encryption of m on top of proving $\widehat{c}$ is a valid encryption of (c, r, σ). This contrived structure was required to bootstrap a sound NIZK to be *online-extractable*.[7] Specifically, the one-more unforgeability proof relied on the reduction being able to extract the (partial) witness (c_i, r_i, σ_i) in an on-the-fly manner from the outer encryption $\widehat{c}_i$ explicitly included in the blind signature. The other inefficiency stems from the heavy reliance on PKEs. As far as the correctness is concerned, the PKE seems replaceable by a computationally binding commitment scheme. This would be ideal since commitment schemes tend to be more size efficient than PKEs since decryptability is not required. However, without a PKE, it is not clear how the above proof would work.

First Construction. We explain our first construction, an optimized variant of Fischlin's generic construction.

Using Rewinding-Extractable NIZKs. The first step is to relax the online-extractable NIZK with a (single-proof) rewinding-extractable NIZK. Such an NIZK allows extracting a witness from a proof output by an adversary $\mathcal{A}$ by *rewinding* $\mathcal{A}$ on a fixed random tape. NIZKs obtained by compiling a Σ-protocol using the Fiat-Shamir transform is a representative example of an efficient rewinding-extractable NIZK. The net effect of this modification is that we can remove the layer of large encryption by $\widehat{c}$, thus making the statement simpler and allowing us to remove $\widehat{c}$ from σ_{BS}.

Let us check if this rewinding-extractable NIZK suffices in the above proof of one-more unforgeability. At first glance, the proof does not seem to work due to a subtle issue added by the rewinding extractor. Observe that the reduction now needs to simulate $\mathcal{A}$ in the *rewound execution* as well. In particular, after rewinding $\mathcal{A}$, $\mathcal{A}$ may submit a new c' to the blind signing oracle, which was not queried in the initial execution. The reduction relays this c' to its signing oracle as in the first execution to simulate the signature σ'. As before, we can argue that there exists a message m_i in the forgeries output by $\mathcal{A}$ in the *first* execution such that $m_i \notin L$, but we need to further argue that $m_i \notin L'$, where L' is the list of decrypted messages $\mathcal{A}$ submitted in the *rewound* execution. Namely, we need to argue that $m_i \notin L \cup L'$ for the reduction to break `euf-cma` security. However, a naive counting argument as done before no longer works because $|L \cup L'|$ can be large as 2ℓ, exceeding the number of forgeries output by $\mathcal{A}$, i.e., $\ell + 1$.

We can overcome this issue by taking a closer look at the internal of a particular class of rewinding-extractable NIZK. Specifically, throughout this paper, we focus on NIZKs constructed by applying the Fiat-Shamir transform on a Σ-protocol (or in more general a public-coin interactive protocol). A standard way to argue rewinding-extractability of a Fiat-Shamir NIZK is by relying on the forking lemma [73, 14], which states (informally) that if an event E happened in the first run, then it will happen in the rewound round with non-negligible probability. In the above context, we define E to be the event that the i-th message in $\mathcal{A}$'s forgeries satisfy $m_i \notin L$, where i is sampled uniformly random by the reduction at the outset of the game. Here, note that E is well-defined since the reduction can prepare the list L by decrypting $\mathcal{A}$'s signing queries. The forking lemma then guarantees that we also have $m_i \notin L'$ in the rewound execution.[8] This slightly more fine-grained analysis allows us to replace the online-extractable NIZK with a rewinding-extractable NIZK.

Issue with Using Commitments. The next step is to relax the PKE by a (computationally binding) commitment scheme. While the correctness and blindness hold without any issue, the one-more unforgeability proof seems to require a major reworking. The main reason is that without the reduction being able to decrypt $\mathcal{A}$'s signing queries c, we won't be able to define the list L. In particular, we can no longer define the event E, and hence, cannot invoke the forking lemma.

[7] Constructing an online extractable NIZK by adding a PKE on top of a sound NIZK is a standard method.

[8] For the keen readers, we note that we are guaranteed to have the same i-th message in both executions since these values are fixed at the forking point due to how the Fiat-Shamir transform works.

Thus, we are back to the situation where we cannot argue that the extracted witness (c_i, r_i, σ_i) from $\mathcal{A}$'s forgeries, is a valid forgery against the `euf-cma` security game. Even worse, $\mathcal{A}$ could potentially be breaking the computationally binding property of the commitment scheme by finding two message-randomness pairs (m_i, r_i) and (m'_i, r'_i) such that they both commit to c_i but $m_i \neq m'_i$. In such a case, extracting from a single proof does not seem sufficient since a reduction would need at least two extracted witnesses to break the binding of the commitment scheme.

To cope with the latter issue first, we extend the one-more unforgeability proof to rely on a *multi-proof* rewinding-extractable NIZK. In general, multi-proof rewinding-extractors run in exponential time in the number of proofs that it needs to extract from [78, 16]. However, in our situation, with a careful argument, we can prove that our extractor runs in strict polynomial time since $\mathcal{A}$ provides all the proofs to the extractor only at the end of the game. This is in contrast to the settings considered in [78, 16] where $\mathcal{A}$ can adaptively submit multiple proofs to the extractor throughout the game.

We note that the assumption we require has not changed: a Σ-protocol for the same relation as in the single-proof setting compiled into an NIZK via the Fiat-Shamir transform. To prove multi-proof rewinding-extractability of this Fiat-Shamir NIZK, we can no longer rely on the now standard general forking lemma by Bellare and Neven [14] that divorces the probabilistic essence of the forking lemma from any particular application context. A naive extension of the general forking lemma to the multi-forking setting will incur an exponential loss in the success probability. To provide a meaningful bound, we must take into account the extra structure offered by the Fiat-Shamir transform, and thus our analysis is akin to the more traditional forking lemma analysis by Pointcheval and Stern [73] or by Micali and Reyzin [66]. To the best of our knowledge, we provide the first formal analysis of the multi-proof rewinding-extractability of an NIZK obtained by applying the Fiat-Shamir transform to a Σ-protocol. We believe this analysis to be of independent interest.

Final Idea to Finish the Proof. Getting back to the proof of one-more unforgeability, the reduction now executes the multi-proof rewinding-extractor to extract all the witnesses $(c_i, r_i, \sigma_i)_{i\in[\ell+1]}$ from the forgeries. Relying on the binding of the commitment scheme, we are guaranteed that all the commitment c_i's are distinct. Moreover, since $\mathcal{A}$ only makes ℓ blind signature queries *in the first execution*, we further have that there exists at least one c_i in the forgeries which $\mathcal{A}$ did not submit in the first execution.

However, we are still stuck since it's unclear how to argue that this particular c_i was never queried by $\mathcal{A}$ in any of the rewound executions. Our next idea is to slightly strengthen the NIZK so that the proof π is statistically binding to a portion of the witness that contains the commitments.[9] We note that this is still strictly weaker and more efficiently instantiable compared to an online-extractable NIZK required by Fischlin's construction since we do not require the full list of witnesses to be efficiently extractable from the proofs in an online manner. We use this property to implicitly fix the commitments $(c_i)_{i\in[\ell+1]}$ included in the forgeries after the end of the first execution of $\mathcal{A}$. This will be the key property to completing the proof.

The last idea is for the reduction to randomize what it queries to its signing oracle. For this, we further assume the commitment scheme is randomizable, where we emphasize that this is done for ease of explanation and we do not strictly require such an assumption (see remark 1). When $\mathcal{A}$ submits a commitment c to the blind signing oracle, the reduction randomizes c to c' using some randomness rand and instead sends c' to its signing oracle. It returns the signature σ *and* rand to $\mathcal{A}$. $\mathcal{A}$ checks if c becomes randomized to c' using rand and if σ is a valid signature on c'. It then uses c' instead of c to generate the blind signature as before. The key observation is that the reduction is invoking its signing oracle with randomness outside of $\mathcal{A}$'s control. Since the commitments $(c_i)_{i\in[\ell+1]}$ were implicitly fixed at the end of the first execution, any randomized c' sampled in the subsequent rewound execution is independent of these commitments. Hence, the probability that the reduction queries c_i to the signing oracle in any of the rewound execution is negligible, thus constituting a valid forgery against the `euf-cma` security game as desired.

Instantiation. We instantiate the framework in the asymmetric pairing setting, i.e. we have groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of prime order p, some fixed generators $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$, and a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \mapsto \mathbb{G}_T$.

[9] At the Σ-protocol abstraction, we call this new property f-unique extraction. It is a strictly weaker property than the *unique response* property considered in the literature [37, 80].

For the commitment scheme, we choose Pedersen commitments ($\mathsf{C}_{\mathrm{Ped}}$) of the form $c = g_1^m \mathsf{pp}^r$, as $\mathsf{C}_{\mathrm{Ped}}$ is randomizable and consists of a single group element. Note that the public parameter $\mathsf{pp} \in \mathbb{G}_1$ is generated via a random oracle. We then need to choose an appropriate signature scheme that allows signing $\mathsf{C}_{\mathrm{Ped}}$ commitments. We choose SPS as all components of the scheme are group elements, in particular, the message space is $\mathbb{G}_1^\ell$, where ℓ is the message length. The most efficient choice in the standard model is [55] with signatures of size 335 Byte. Instead, we optimize KPW signatures [62] to a signature size of 223 Byte (from originally 382 Byte). Our optimized variant $\mathsf{S}_{\mathrm{KPW}}$ is no longer structure-preserving, as it consists of one element τ in $\mathbb{Z}_p$, but suffices for our applications. We refer to Section 4.2 for more details.

Note that $\mathsf{S}_{\mathrm{KPW}}$ would be an inefficient choice in the original Fischlin blind signature [38], as it requires encrypting the signature τ over $\mathbb{Z}_p$ to instantiate the online-extractable NIZK. In the pairing setting, this incurs an overhead in proof size linear in the security parameter λ [10]. The benefit of using our framework with the weaker rewinding-extractable NIZK is that we now only need to prove knowledge of τ, and thus can get away without encrypting it. Such an NIZK is possible with a single element in $\mathbb{Z}_p$ based on a Schnorr-type Σ-protocol (compiled with Fiat-Shamir). In the Σ-protocol, we further commit to group elements $(w_i)_i \in \mathbb{G}_1^n$ in the witness via ElGamal commitments (C_{EG}) of the form $E_i = (w_i \cdot \mathsf{pp}^{r_i}, g_1^{r_i})$, which the prover sends to the verifier in the first flow. In particular, this ensures f-unique extraction, as E_i fixes the commitment $c \in \{w_i\}_i$ statistically. Naively, this approach requires $2n$ group elements, where n is the number of group elements in the witness. Instead, we share the randomness among all commitments under different public parameters pp_i generated via a random oracle. The commitments remain secure but require only $n+1$ group elements. In particular, we set $E_i = (w_i \cdot \mathsf{pp}_i^s)$ and fix s via $S = g_1^s$. Then, we can open *all* commitments E_i in zero-knowledge with a *single* element in $\mathbb{Z}_p$, as knowledge of s is sufficient to recover the witness w_i from all E_i. Then, we compile our Σ-protocol with Fiat-Shamir to obtain a rewinding-based NIZK. We apply a well-known optimization to avoid sending some of the first flow α, and include the hash value $\beta \leftarrow \mathsf{H}(x, \alpha)$ in the proof explicitly. In total, compared to sending the witness to the verifier in the *clear*, our NIZK only has an overhead of 1 group element in $\mathbb{G}_1$ and 3 elements in $\mathbb{Z}_p$. The additional group element is S. The three additional $\mathbb{Z}_p$ elements are the hash value β, and values in the third flow required for (i) showing knowledge of s and (ii) linearizing a quadratic equation in the signature verification.

The instantiation of our framework achieves communication size of 303 Byte and signature size of 447 Byte.

Second Construction. We explain our second construction relying on randomizable signatures with an ABO reduction.

Getting Rid of NIZKs in the Signature. While the previous construction provides a small sum of signature and communication sizes, one drawback is that the blind signature has inherently a larger signature than those of the underlying signature scheme. The source of this large blind signature stems from using an NIZK to hide the underlying signature provided by the signer.

A natural approach used in the literature is to rely on techniques used to construct *randomizable* signature schemes [18, 41, 40, 61]. Informally, a randomizable signature scheme allows to publicly randomize the signature σ on a message m to a fresh signature σ'. Many standard group-based signature schemes (in the standard model and ROM) are known to satisfy this property, *e.g.*, [20, 82]. A failed attempt would be for the user to randomize the signature σ provided by the signer and output the randomized σ' as the blind signature. Clearly, this is not secure since the user is not hiding the message m, that is, σ and σ' are linkable through m thus breaking blindness. An idea to fix this would be to let the user send a commitment $c = \mathsf{Com}(m; r)$ to the signer and the signature signs the "message" c. However, unless the commitment c can be randomized consistently with σ, we would still need to rely on an NIZK to hide c. This calls for a signature scheme that is somehow compatible with commitments.

Signatures with All-But-One Reductions. Our main insight is that a specific class of signature schemes with an *all-but-one* (ABO) reduction is naturally compatible with blind signatures. An

[10] For instance, with ElGamal, the message is encrypted in the exponent and decryption would require a discrete logarithm computation. Thus, the message is typically encrypted bit-wise which incurs an overhead of $\log_2(p)$.

ABO reduction is a standard proof technique to prove selective security of public key primitives, *e.g.*, [21, 75, 49, 6], where a formal treatment can be found in [67]. In the context of signature schemes, this is a proof technique that allows the reduction to embed the challenge message m^* (i.e., the signature for which the adversary forges) into the verification key. The reduction can simulate any signatures on $m \neq m^*$, and when the adversary outputs a forgery on m^*, then the reduction can break some hard problems.

Let us now specify the class of signature scheme. We assume an additive homomorphic commitment scheme, that is, $\mathsf{Com}(m;r) + \mathsf{Com}(m';r') = \mathsf{Com}(m+m';r+r')$. We then assume a signature scheme where the signing algorithm $\mathsf{Sig}(\mathsf{sk}, m)$ can be rewritten as $\widehat{\mathsf{Sig}}(\mathsf{sk}, \mathsf{Com}(m;0)+u)$, where u is some fixed but random commitment included in the verification key. Namely, Sig first commits to the message m using no randomness, adds u to it, and proceeds with signing. Note that if $u = \mathsf{Com}(-m';r')$ for some (m', r'), then $\mathsf{Com}(m;0) + u = \mathsf{Com}(m-m';r')$. While contrived at first glance, this property is naturally satisfied by many of the signature schemes that admit an ABO reduction; the ABO reduction inherently requires embedding the challenge message m^* into the verification key in an unnoticeable manner and further implicitly requires message m submitted to the signing query to interact with the "committed" m^*. Specifically, the former hints at a need for an (implicit) commitment scheme and the later hints at the need for some operation between the commitments. Finally, to be used in the security proof, we assume there is a simulated signing algorithm $\widehat{\mathsf{SimSig}}$ along with a trapdoor td such that $\widehat{\mathsf{SimSig}}(\mathsf{td}, \mathsf{Com}(m-m';r'), m-m', r') = \widehat{\mathsf{Sig}}(\mathsf{sk}, \mathsf{Com}(m;0)+u)$ if and only if $m \neq m'$, where recall $u = \mathsf{Com}(-m';r')$. Specifically, $\widehat{\mathsf{SimSig}}$ can produce a valid signature if it knows the *non-zero* commitment message and randomness.

Let us explain the ABO reduction in slightly more detail. In the security proof, the reduction guesses (or the adversary $\mathcal{A}$ submits) a challenge message m^* that $\mathcal{A}$ will forge on. It then sets up the verification key while replacing the random commitment u to $u = \mathsf{Com}(-m^*;r^*)$ while also embedding a hard problem that it needs to solve. Due to the hiding property of the commitment scheme, this is unnoticeable from $\mathcal{A}$. Then, instead of using the real signing algorithm $\widehat{\mathsf{Sig}}$, the reduction uses the simulated signing algorithm $\widehat{\mathsf{SimSig}}$. As long as $m \neq m^*$, $\widehat{\mathsf{SimSig}}(\mathsf{td}, \mathsf{Com}(m-m^*;r^*), m-m^*, r^*)$ outputs a valid signature, and hence, can be used to simulate the signing oracle. Finally, given a forgery on m^*, the reduction is set up so that it can break a hard problem.

Turning it into a Blind Signature. To turn this into a blind signature, the key observation is that $\widehat{\mathsf{Sig}}$ is agnostic to the committed message and randomness of $\mathsf{Com}(m;0)+u$ — these are only used during the security proof when running $\widehat{\mathsf{SimSig}}$. Concretely, a user of a blind signature can generate a valid commitment $\mathsf{Com}(m;r)$, send it to the signer, and the signer can simply return $\sigma_r \leftarrow \widehat{\mathsf{Sig}}(\mathsf{sk}, \mathsf{Com}(m;r)+u)$. If the signature admits a way to map σ_r back to a normal signature σ for m, then we can further rely on the randomizability of the signature scheme to obtain a fresh signature σ' on the message m.

The proof of one-more unforgeability of this abstract blind signature construction is almost identical to the original ABO reduction with one exception. For the reduction to invoke the simulated $\widehat{\mathsf{SimSig}}$, recall it needs to know the message and randomness of the commitment $\mathsf{Com}(m;r)+u$. Hence, we modify the user to add an *online-extractable* NIZK to prove the correctness of the commitment $\mathsf{Com}(m;r)$ so that the reduction can extract (m,r). Here, we require online-extractability rather than rewinding-extractability since otherwise, the reduction will run exponentially in the number of singing queries [78, 16]. Also, this is why the communication size becomes larger compared with our first construction. Finally, when the adversary outputs a forgery including m^*, the reduction can break a hard problem as before. Here, we note that we can simply hash the messages m with a random oracle to obtain an adaptively secure scheme using the ABO reduction.

Interestingly, while the recent lattice-based blind signature by del Pino and Katsumata [34] stated to use lattice-tailored techniques to optimize Fischlin's generic construction, the construction and the proof of one-more unforgeability follows our above template, where they use the Agrawal-Boneh-Boyen signature [6] admitting an ABO reduction. The only difference is that since lattices do not have nice randomizable signatures, they still had to rely on an NIZK for the final signature. While we focused on ABO reductions where only one challenge message m^* can be embedded in the verification

key, the same idea naturally extends to all-but-*many* reductions. The blind signature by Blazy et al. [18] relying on the Waters signature can be viewed as one such instantiation. Finally, while we believe we can make the above approach formal using the ABO reduction terminology defined in [67], we focus on one class of instantiation in the main body for better readability. Nonetheless, we believe the above abstract construction will be useful when constructing round-optimal blind signatures from other assumptions.

Instantiation. We instantiate the above framework with the Boneh-Boyen signature scheme S_{BB} [20, 22]. Recall that signatures of S_{BB} on a message $m \in \mathbb{Z}_p$ are of the form $\sigma = (\mathsf{sk} \cdot (u_1^m \cdot h_1)^r, g_1^r)$, where $u_1, h_1 \in \mathbb{G}_1$ are part of the verification key, sk is the secret key and $r \leftarrow \mathbb{Z}_p$ is sampled at random. We observe that S_{BB} is compatible with the Pedersen commitment scheme $\mathsf{C}_{\mathrm{Ped}}$ with generators u_1 and g_1. Roughly, the user commits to the message m via $c = u_1^m \cdot g_1^s$[11], where $s \leftarrow \mathbb{Z}_p$ blinds the message, proves that she committed to m honestly with a proof π generated via an appropriate online-extractable NIZK Π, and sends (c, π) to the signer. The signer checks π and signs c via $(\mu_0, \mu_1) \leftarrow (\mathsf{sk} \cdot (c \cdot h_1)^r, g_1^r)$. Note that as c shares the structure u_1^m with S_{BB} signatures on message m, the user can recompute a valid signature on m via $\sigma \leftarrow (\mu_0 \cdot \mu_1^{-s}, \mu_1)$. Before presenting σ to a verifier, the user rerandomizes σ to ensure blindness. We refer to section 5 for more details.

The main challenge is constructing an efficient *online-extractable* NIZK Π for the relation $\mathsf{R}_{\mathsf{bb}} = \{(x, w) : c = u_1^m \cdot g_1^s\}$, where $x = (c, u_1, g_1)$ and $w = (m, s)$. As we require online-extraction, a simple Σ-protocol showing $c = u_1^m \cdot g_1^s$ compiled via Fiat-Shamir is no longer sufficient as in our prior instantiation, as the extractor needs to rewind the adversary in order to extract (m, s). For example, we could instantiate Π with the (online-extractable) GOS proofs but such a proof has a size of around 400 KB. Another well-known approach is to additionally encrypt the witness (m, s) via a PKE and include the ciphertext into the relation; recall this method was used when explaining the Fischlin blind signature. The extractor can then use the secret key to decrypt the witnesses *online*. While a common choice for the PKE would be ElGamal encryption, this is insufficient since the extractor can only decrypt group elements g_1^m and g_1^s and not the witness in $\mathbb{Z}_p$ as required. To circumvent this, a common technique is to instead encrypt the binary decompositions $(m_i, s_i)_{i \in [\ell_2]}$ of m, s, respectively, with ElGamal, where $\ell_2 = \log_2(p)$. It then proves with a (non-online extractable) NIZK that $m = \sum_{i=1}^{\ell_2} m_i 2^{i-1}$ and $s = \sum_{i=1}^{\ell_2} s_i 2^{i-1}$ are valid openings of c, while also proving that m_i, s_i encrypted in the ElGamal ciphertexts are elements in $\{0, 1\}$, where the latter can be done via the equivalent identity $x \cdot (1 - x) = 0$. The extractor can now decrypt the ElGamal encryptions of m_i to $g_1^{m_i} \in \{g_1, 1_{\mathbb{G}_1}\}$ and efficiently decide whether m_i is 0 or 1. Similarly, it can recover the decomposition s_i. Unfortunately, this approach requires at least $2\ell_2$ ElGamal ciphertexts which amount to 32 KB alone. In fact, the bit-by-bit encryption of the witness is also the efficiency bottleneck of GOS proofs for $\mathbb{Z}_p$ witnesses.

We refine the above approach in multiple ways to obtain concretely efficient online-extractable NIZKs. Instead of using the binary decomposition, we observe that the extractor can still recover x from g_1^x if $x \in [0, B-1]$ is short, i.e., $B = \mathsf{poly}(\lambda)$. Thus, we let the prover encrypt the B-ary decompositions $(m_i, s_i)_{i \in [\ell]}$ of m and s, where $\ell = \log_B(p)$. For example, setting $B = 2^{32}$ allows the extractor to recover m_i via a brute-force calculation of the discrete logarithm, and the number of encryptions is reduced by a factor of 32. Concretely, we modify the prover to prove that an ElGamal ciphertext encrypts $(m_i, s_i)_{i \in [\ell]}$ such that (i) each m_i and s_i are in $[0, B-1]$, and (ii) $m = \sum_{i=1}^{\ell} m_i B^{i-1}$, $s = \sum_{i=1}^{\ell} s_i B^{i-1}$, and $c = u_1^m \cdot g_1^s$.

To instantiate our approach, we glue two different (non-online extractable) NIZKs Π_{rp} and Π_{ped} together, each being suitable to show relations (i) and (ii), respectively. For the range relation (i), we appeal to the batched variant of Bulletproofs [8] and turn it non-interactive with Fiat-Shamir. For the linear relation (ii), we use a standard NIZK with an appropriate Σ-protocol compiled with Fiat-Shamir. We further apply three optimizations to make this composition of NIZKs more efficient:

1. While Bulletproofs require committing to the decompositions $(m_i, s_i)_{i \in [\ell]}$ in Pedersen commitments, we use the shared structure of ElGamal ciphertexts and Pedersen commitments to avoid sending additional Pedersen commitments. This also makes the relation simpler since we do not have to prove consistency between the committed components in the ElGamal ciphertext and Pedersen commitment.

[11] In the actual construction, we further hash m by a random oracle; this effectively makes S_{BB} *adaptively* secure.

2. We use a more efficient discrete logarithm algorithm during extraction with runtime $\mathcal{O}(\sqrt{B})$, which allows us to choose more efficient parameters for the same level of security. This further reduces the number of encryptions by a factor 2.
3. We perform most of the proof in a more efficient elliptic curve $\widehat{\mathbb{G}}$ of same order p without pairing structure. As both the NIZKs Π_{rp} and Π_{ped} are not reliant on pairings, this reduces the size and efficiency of the NIZK considerably.

Proof of Instantiation. Finally, we analyze the security of the optimized online-extractable NIZK Π obtained by gluing Π_{rp} and Π_{ped} together. Correctness and zero-knowledge are straightforward. Also, online-extraction seems immediate on first sight. The extractor decrypts the decomposition, reconstructs the witness (m, s), and checks whether $c = u_1^m g_1^s$. To show why it works, we rely on the soundness of the range proof Π_{rp} to guarantee that the committed values are short. This allows the extractor to decrypt efficiently. Moreover, we rely on the soundness of Π_{ped} to guarantee that the decrypted values form a proper B-ary decompositions of an opening (m, s) of c. However, this high-level idea misses many subtle issues.

First, Bulletproofs are not well-established in the non-interactive setting in the ROM. While Attema et al. [9] show that special sound multi-round proof systems are knowledge sound (or rewinding-extractable) when compiled via Fiat-Shamir, Bulletproofs are only *computationally* special sound under the DLOG assumption. An easy fix for this is to relax the relation of the extracted witness. That is we use two different relations: one to be used by the prover and the other to be used by the extractor. We define an extracted witness w to be in the relaxed relation if either w is in the original relation *or* w is a DLOG solution with respect to (part of) the statement. With this relaxation, the interactive Bulletproofs becomes special sound for the relaxed relation since we can count the extracted DLOG solution as a valid witness. Observing that the result of [9] naturally translates to relaxed relations, we can conclude the non-interactive Bulletproofs to be rewinding-extractable in the ROM.

The second subtlety is more technical. For the formal proof, when the adversary submits a proof such that the online-extraction of Π fails, we must show that the adversary is breaking either the soundness of the underlying NIZKs Π_{rp} or Π_{ped}. Recall that Π_{rp} and Π_{ped} are glued together via the ElGamal ciphertext (cf. item 1). Specifically, each witness $w \in (m_i, s_i)_{i\in[\ell]}$ are encrypted as $c = (c_0, c_1) = (g^w \mathsf{pp}^r, g^r)$ with randomness $r \leftarrow \mathbb{Z}_p$, and Π_{rp} uses the partial "Pedersen part" c_0, while Π_{ped} uses the entire "ElGamal part" c. Thus one possibility for the online-extraction of Π failing is when the adversary breaks the tie between the two NIZKs by breaking the binding property of the Pedersen commitment. That is, if the adversary finds the DLOG between (g, pp), it can break the consistency between the two NIZKs in such a way that online-extraction of Π fails.

Put differently, to show that no adversary can trigger a proof for which the online-extraction of Π fails, we must show (at the minimum) that we can use such an adversary to extract a DLOG solution between (g, pp). This in particular implies that we have to *simultaneously* extract the witness w_0 of Π_{rp} containing one opening of c_0 and the witness w_1 of Π_{ped} containing the other opening of c_0 in order to break DLOG with respect to (g, pp), or equivalently to break the binding property of the Pedersen commitment. The issue with this is that we cannot conclude that both extractions succeed at the same time even if Π_{rp} and Π_{ped} *individually* satisfy the standard notion of rewinding-extractability. For instance, using the standard notion of rewinding-extractability, we cannot exclude the case where the adversary sets up the proofs π_0, π_1 of $\Pi_{\mathsf{rp}}, \Pi_{\mathsf{ped}}$, respectively, in such a way that if the extractor of Π_{rp} succeeds, then the extractor of Π_{ped} fails. We thus show in a careful non-black box analysis that the extraction of both proofs succeeds at the same time with non-negligible probability. To the best of our knowledge, this is the first time an NIZK that internally uses Bulletproofs is proven to be online-extractable in the ROM. We believe that our new analysis is of independent interest.

2 Preliminaries

2.1 Notation

Let $\lambda \in \mathbb{N}$ be the security parameter. A probabilistic polynomial time (PPT) algorithm $\mathcal{A}$ runs in time polynomial in the (implicit) security parameter λ. We write $\mathsf{Time}(\mathcal{A})$ for the runtime of

$\mathcal{A}$. A function $f(\lambda)$ is *negligible* in λ if it is $\mathcal{O}(\lambda^{-c})$ for every $c \in \mathbb{N}$. We write $f = \mathrm{negl}(\lambda)$ for short. Similarly, we write $f = \mathsf{poly}(\lambda)$ if $f(\lambda)$ is a polynomial with variable λ. If D is a probability distribution, $x \leftarrow D$ means that x is sampled from D and if S is a set, $x \leftarrow S$ means that x is sampled uniformly and independently at random from S. We also write $|S|$ for the cardinality of set S. Further, we write $D_0 \overset{c}{\approx} D_1$ for distributions D_0, D_1, if for all PPT adversaries $\mathcal{A}$, we have $|\Pr[x_0 \leftarrow D_0 : \mathcal{A}(1^\lambda, x_0) = 1] - \Pr[x_1 \leftarrow D_1 : \mathcal{A}(1^\lambda, x_1) = 1]| = \mathrm{negl}(\lambda)$. Similarly, we write $D_0 \overset{s}{\approx} D_1$ if the above holds even for unbounded adversaries. For some PPT algorithm $\mathcal{A}$, we write $\mathcal{A}^{\mathcal{O}}$ if $\mathcal{A}$ has oracle access to the oracle $\mathcal{O}$. If $\mathcal{A}$ performs some check, and the check fails, we assume that $\mathcal{A}$ outputs $\perp$ immediately. Generally, we assume that adversaries are implicitly stateful.

We denote with $[n]$ the set $\{1, \ldots, n\}$ for $n \in \mathbb{N}$. For any $\vec{h} = (h_1, \ldots, h_q)$ and $i \in [q]$, we denote $\vec{h}_{<i}$ as $(h_1, \ldots, h_{i-1})$ and $\vec{h}_{\geq i}$ as $(h_i, \ldots, h_q)$, where $\vec{h}_{<1}$ denotes an empty vector. Moreover, for any two vectors $\vec{h}$, $\vec{h'}$ of arbitrary length, we use $\vec{h} \| \vec{h'}$ to denote the concatenation of the two vectors. In particular, for any $i \in [q]$ and $\vec{h} \in \mathcal{H}^q$, we have $\vec{h} = \vec{h}_{<i} \| \vec{h}_{\geq i}$.

2.2 Groups and Pairings

Throughout this work, write $1_{\mathbb{G}}$ for the neutral element of some group $\mathbb{G}$ and use multiplicative notation. Also, we assume a PPT algorithm $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2) \leftarrow \mathsf{PGen}(1^\lambda)$ that on input 1^λ outputs descriptions of the groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of prime order p and a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \mapsto \mathbb{G}_T$, as well as generators g_1, g_2 of $\mathbb{G}_1, \mathbb{G}_2$, respectively. Recall that e is a pairing if e is non-degenerate, i.e. $e(g_1, g_2) \neq 1_{\mathbb{G}_T}$, and if e is bilinear, i.e. $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ for $a, b \in \mathbb{Z}_p$. We sometimes use implicit notation $[x]_k = g_k^x$ for $k \in [1, 2, T]$, $x \in \mathbb{Z}_p$ and $g_T = e(g_1, g_2)$. We extend the notation to matrices naturally, i.e. we write $[\boldsymbol{A}]_k = ([a_{i,j}]_k)$ for $\boldsymbol{A} = (a_{i,j}) \in \mathbb{Z}_p^{n \times m}$ and $k \in [1, 2, T]$. Note that while we mainly consider the asymmetric pairing setting, i.e. $\mathbb{G}_1 \neq \mathbb{G}_2$, all instantiations have a natural variant in the symmetric pairing setting with similar efficiency. Similarly, we assume a PPT algorithm $(\widehat{\mathbb{G}}, g) \leftarrow \mathsf{GGen}(1^\lambda, p)$ that on input 1^λ and prime order p, outputs a description of a group $\widehat{\mathbb{G}}$ of order p with generator g that is not equipped with a pairing. Generally, we assume that given the description(s), group operations, pairing evaluation and membership tests are efficient, and write $g \leftarrow \mathbb{G}$ for drawing elements from some group $\mathbb{G}$ at random. For readability, we leave PGen and GGen implicit in the rest of the work.

Instantiation. For our instantiations, we assume that the modulus p is of size 256 bit, and an element of $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ is of size $382, 763, 4572$ bit, respectively. These are common sizes of standard BLS curves [11] with security parameter $\lambda = 128$, in particular BLS12-381 [23]. For groups that require no pairing operation, we use a curve of order p and assume that elements are of size 256 bit. We generally write $\widehat{\mathbb{G}}$ for such groups.

Assumptions. In this paper, we use the following hardness assumptions.

Definition 1 (DLOG). *The discrete logarithm (DLOG) assumption in group $\mathbb{G}$ with generator g holds if for any PPT adversary $\mathcal{A}$, it holds that*

$$\Pr[x \leftarrow \mathbb{Z}_p : \mathcal{A}(g, g^x) = x] = \mathrm{negl}(\lambda)$$

Definition 2 (DDH). *The decisional Diffie-Hellman (DDH) assumption holds in group $\mathbb{G}$ with generator g if for any PPT adversary $\mathcal{A}$, it holds that*

$$|\Pr[a, b \leftarrow \mathbb{Z}_p : \mathcal{A}(g, g^a, g^b, g^{ab}) = 1] - \Pr[a, b, c \leftarrow \mathbb{Z}_p : \mathcal{A}(g, g^a, g^b, g^c) = 1]| = \mathrm{negl}(\lambda).$$

Definition 3 (SXDH). *The symmetric external Diffie-Hellman (SXDH) assumption holds in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ if the DDH assumption holds in $\mathbb{G}_1$ and in $\mathbb{G}_2$.*

Definition 4 (CDH). *The computational Diffie-Hellman (CDH) assumption holds in $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2)$ if for any PPT adversary $\mathcal{A}$, it holds that*

$$\Pr[a, b \leftarrow \mathbb{Z}_p : \mathcal{A}(g_1, g_2, g_1^a, g_2^a, g_1^b, g_2^b) = g^{ab}] = \mathrm{negl}(\lambda).$$

Explaining Group Elements as Random Strings. Our frameworks generally require that public parameters pp (of commitment schemes) and common random strings crs (of NIZKs) are random bit strings. For readability, we allow that pp and crs contain random group elements $g \leftarrow \mathbb{G}$ for some group $\mathbb{G}$. This is without loss of generality, as using explainable sampling, we can explain these elements as random strings.

Concretely, explainable sampling for $\mathbb{G}$ allows us to sample from the uniform distribution $\mathcal{U}_{\mathbb{G}}$ over $\mathbb{G}$ via $g \leftarrow \mathsf{SampleG}(1^\lambda; r)$ using ℓ_G bits of randomness, where $r \leftarrow \{0,1\}^{\ell_G}$. Importantly, there exists an algorithm $\mathsf{ExplainG}$ that given uniformly random $g \in \mathbb{G}$ outputs randomness $r' \leftarrow \mathsf{ExplainG}(1^\lambda, g)$ such that $g = \mathsf{SampleG}(1^\lambda; r')$ and $r \overset{s}{\approx} r'$. Note that such sampling techniques are known for elliptic curves, see for example [26].

Using this notation, a random string $\mathsf{crs} = (r_1, \ldots, r_n)$ of size $\{0,1\}^{n \cdot \ell_G}$ represents n group elements $g_i \leftarrow \mathsf{SampleG}(1^\lambda; r_i)$. Further, as long as all $g'_i \in \mathbb{G}$ are also distributed independently and uniformly at random, the bit string $\mathsf{crs}' = (r'_1, \ldots, r'_n)$ will have negligible statistical distance to crs, where $r'_i \leftarrow \mathsf{ExplainG}(1^\lambda, g'_i)$. Thus, we can safely replace crs with crs' in security reductions. The above also applies for $\mathsf{pp} = (r_1, \ldots, r_n)$. Similarly, we can instantiate random oracles that map into the group $\mathbb{G}$ with random oracles mapping into $\{0,1\}^{\ell_G}$ this way.

Throughout, we write $g \leftarrow \mathbb{G}$ short for $g \leftarrow \mathsf{SampleG}(1^\lambda)$. Also, we write $\mathsf{crs} = (g_1, \ldots, g_n)$ for short, where g_i are drawn uniformly at random. Note that we can replace crs with $\mathsf{crs}' = (g'_1, \ldots, g'_n)$ in proofs, if all g'_i are also distributed independently uniform over $\mathbb{G}$. We extend the notation above to tuples of random elements drawn from different groups. These techniques also apply for mixed crs or pp over different groups.

2.3 Commitment Scheme

A *commitment scheme* is a PPT algorithm $\mathsf{C} = \mathsf{C.Commit}$ such that

- $\mathsf{C.Commit}(\mathsf{pp}, m; r)$: given the public parameters $\mathsf{pp} \in \{0,1\}^{\ell_\mathsf{c}}$, message $m \in \mathcal{C}_\mathsf{msg}$ and randomness $r \in \mathcal{C}_\mathsf{rnd}$, computes a commitment $c \in \mathcal{C}_\mathsf{com}$, and outputs the pair (c, r),

Here, $\{0,1\}^{\ell_\mathsf{c}}$, $\mathcal{C}_\mathsf{msg}$, $\mathcal{C}_\mathsf{rnd}$, $\mathcal{C}_\mathsf{com}$, are public parameter, message, commitment randomness, and commitment spaces, respectively.[12] We do not explicitly define the opening algorithm since we can use the commitment randomness r as the decommitment (or opening) information and check if $c = \mathsf{Commit}(\mathsf{pp}, m; r)$ holds to verify that c is a valid commitment to message m.

Below, we first define the standard notions of binding and hiding, where note that correctness is implicit since we define the decommitment algorithm via Commit. While we define the computational variants below, we obtain the statistical variants by allowing the adversary $\mathcal{A}$ to be unbounded.

Definition 5 (Hiding). *A commitment scheme is* hiding *if for any PPT adversary $\mathcal{A}$, we have*

$$\mathsf{Adv}^{\mathrm{hide}}_{\mathcal{A}}(\lambda) = \left| \Pr\left[\begin{array}{l} \mathsf{pp} \leftarrow \{0,1\}^{\ell_\mathsf{c}},\ (m_0, m_1) \leftarrow \mathcal{A}(\mathsf{pp}), \\ m_0, m_1 \in \mathcal{C}_\mathsf{msg} \wedge\ (c, r) \leftarrow \mathsf{Commit}(\mathsf{pp}, m_\mathsf{coin}), \end{array} : \mathsf{coin} = \mathcal{A}(c) \right] - \frac{1}{2} \right| = \mathrm{negl}(\lambda).$$

Definition 6 (Binding). *A commitment scheme is* binding *if for any PPT adversary $\mathcal{A}$, we have*

$$\mathsf{Adv}^{\mathrm{bind}}_{\mathcal{A}}(\lambda) = \Pr\left[\begin{array}{l} \mathsf{pp} \leftarrow \{0,1\}^{\ell_\mathsf{c}},\ (m_0, m_1, r_0, r_1) \leftarrow \mathcal{A}(\mathsf{pp}), \\ m_0 \neq m_1 \in \mathcal{C}_\mathsf{msg} \wedge r_0, r_1 \in \mathcal{C}_\mathsf{rnd} \\ \wedge\ (c_b, r_b) = \mathsf{Commit}(\mathsf{pp}, m_b; r_b), b \in \{0,1\} \end{array} : c_0 = c_1 \right] = \mathrm{negl}(\lambda).$$

We further define rerandomizability. This allows rerandomizing a commitment to a new commitment on the same message. Moreover, we require that the new commitment has enough min-entropy for a random rerandomization randomness.

Definition 7 (Rerandomizability). *A commitment scheme is* rerandomizable *if there exist PPT algorithms* $(\mathsf{RerandCom}, \mathsf{RerandRand})$ *such that*

- $\mathsf{RerandCom}(\mathsf{pp}, c, \Delta r)$*: given the public parameter* pp*, a commitment $c \in \mathcal{C}_\mathsf{com}$, and a rerandomization randomness $\Delta r \in \mathcal{C}_\mathsf{rnd}$,* deterministically *outputs a rerandomized commitment $c' \in \mathcal{C}_\mathsf{com}$,*

[12] We assume uniform public parameters to improve readability. For our work it is sufficient that the distribution of the public parameters is explainable (see section 2.2).

– $\mathsf{RerandRand}(\mathsf{pp}, c, m, r, \Delta r)$*: on input of the public parameter* pp*, a commitment* $c \in \mathcal{C}_{\mathsf{com}}$*, a message* $m \in \mathcal{C}_{\mathsf{msg}}$*, a randomness* r*, and a rerandomization randomness* $\Delta r \in \mathcal{C}_{\mathsf{rnd}}$*, outputs a rerandomized randomness* r'*,*

and the following holds:

– *for all* $\mathsf{pp} \in \{0,1\}^{\ell_{\mathsf{C}}}$*,* $m \in \mathcal{C}_{\mathsf{msg}}$*,* $(c, r) \leftarrow \mathsf{Commit}(\mathsf{pp}, m)$*, and* $\Delta r \in \mathcal{C}_{\mathsf{rnd}}$*, if we compute* $c' = \mathsf{RerandCom}(\mathsf{pp}, c, \Delta r)$ *and* $r' \leftarrow \mathsf{RerandRand}(\mathsf{pp}, c, m, r, \Delta r)$*, then it holds that* $c' = c''$*, where* $(c'', r') = \mathsf{Com}(\mathsf{pp}, m; r')$*, and*
– *we have*

$$\max_{c, c' \in \mathcal{C}_{\mathsf{com}}} \Pr[\mathsf{pp} \leftarrow \{0,1\}^{\ell_{\mathsf{C}}}, \Delta r \leftarrow \mathcal{C}_{\mathsf{rnd}} : c' = \mathsf{RerandCom}(\mathsf{pp}, c, \Delta r)] = \mathrm{negl}(\lambda).$$

We note that any natural additive homomorphic commitment scheme satisfies rerandomizability if we define $\mathsf{RerandCom}(\mathsf{pp}, c, \Delta r) = c + \mathsf{Commit}(\mathsf{pp}, 0; \Delta r) = c'$. Observe that if $c = \mathsf{Commit}(\mathsf{pp}, m; r)$, the rerandomzied randomness is $r' = r + \Delta r$ since $c' = \mathsf{Commit}(\mathsf{pp}, m; r')$ by the homomorphic property. Moreover, c' has high min-entropy since $\mathsf{Commit}(\mathsf{pp}, 0)$ has high min-entropy for most natural commitment schemes. Finally, we note that while a computational variant of the high min-entropy property suffices for our generic construction, we use the statistical variant for simplicity and because our instantiation satisfies it.

2.4 Signature Scheme

We consider *deterministic* signature schemes; a scheme where the randomness of the signing algorithm is derived from the secret key and message. We can derandomize any signature scheme by using a pseudorandom function for generating the randomness used in the signing algorithm (see for example [59]). Formally, a signature scheme is a tuple of PPT algorithms $\mathsf{S} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ such that

– $\mathsf{KeyGen}(1^\lambda)$: generates a verification key vk and a signing key sk,
– $\mathsf{Sign}(\mathsf{sk}, m)$: given a signing key sk and a message $m \in \mathcal{S}_{\mathsf{msg}}$, *deterministically* outputs a signature σ,
– $\mathsf{Verify}(\mathsf{vk}, m, \sigma)$: given a verification key pk and a signature σ on message m, *deterministically* outputs a bit $b \in \{0,1\}$.

Here, $\mathcal{S}_{\mathsf{msg}}$ is the message space. We define the standard notion of correctness and `euf-cma` security

Definition 8 (Correctness). *A signature scheme is* correct*, if for all* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$*,* $m \in \mathcal{S}_{\mathsf{msg}}$*, and* $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$*, it holds that* $\mathsf{Verify}(\mathsf{vk}, m, \sigma) = 1$*.*

Definition 9 (EUF-CMA). *A signature scheme is* `euf-cma` *if for any PPT adversary* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\mathrm{euf}}_{\mathcal{A}}(\lambda) = \Pr\left[\begin{matrix}(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ (m, \sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}(\mathsf{sk}, \cdot)}(\mathsf{vk})\end{matrix} \quad : m \notin L \wedge \mathsf{Verify}(\mathsf{vk}, m, \sigma) = 1\right] = \mathrm{negl}(\lambda),$$

where L *is the list of messages* $\mathcal{A}$ *queried to the* Sign*-oracle.*

2.5 (Partially) Blind Signature Scheme

A partially blind signature scheme is a tuple of PPT algorithms $\mathsf{PBS} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ such that

– $\mathsf{KeyGen}(1^\lambda)$: generates the verification key bvk and signing key bsk,
– $\mathsf{User}(\mathsf{bvk}, \boxed{t}, m)$: given verification key bvk, common message $t \in \mathcal{BS}_t$, and message $m \in \mathcal{BS}_{msg}$, outputs a first message ρ_1 and a state st,
– $\mathsf{Signer}(\mathsf{bsk}, \boxed{t}, \rho_1)$: given signing key bsk, common message $t \in \mathcal{BS}_t$, and first message ρ_1, outputs a second message ρ_2,

- $\mathsf{Derive}(\mathsf{st}, \boxed{t}, \rho_2)$: given state st, common message $t \in \mathcal{BS}_t$, and second message ρ_2, outputs a signature σ,
- $\mathsf{Verify}(\mathsf{bvk}, \boxed{t}, m, \sigma)$: given verification key bvk, common message $t \in \mathcal{BS}_t$, and signature σ on message $m \in \mathcal{BS}_{msg}$, outputs a bit $b \in \{0,1\}$.

Here, $\mathcal{BS}_t$ and $\mathcal{BS}_{msg}$ are the message and tag spaces, respectively. In case the common message $t \in \mathcal{BS}_t$ is omitted from the syntax (or alternatively, always set to a fixed value), then we call the scheme to be a blind signature BS. We consider the standard security notions for blind signatures [54]. Below, we define correctness, partial blindness under *malicious keys*, and one-more unforgeability of a (partially) blind signature scheme. The definition for blind signature can be recovered by ignoring t, denoted with a box. Moreover, we assume the state is kept implicit in the following for better readability.

Definition 10 (Correctness). *A partially blind signature scheme is* correct*, if for all messages* $(\boxed{t}, m) \in \mathcal{BS}_t \times \mathcal{BS}_{msg}$*,* $(\mathsf{bvk}, \mathsf{bsk}) \leftarrow \mathsf{KeyGen}(1^\lambda), (\rho_1, \mathsf{st}) \leftarrow \mathsf{User}(\mathsf{bvk}, \boxed{t}, m), \rho_2 \leftarrow \mathsf{Signer}(\mathsf{bsk}, \boxed{t}, \rho_1), \sigma \leftarrow \mathsf{Derive}(\mathsf{st}, \boxed{t}, \rho_2)$*, it holds that* $\mathsf{Verify}(\mathsf{bvk}, \boxed{t}, m, \sigma) = 1$*.*

Definition 11 (Partial Blindness Under Malicious Keys). *A partially blind signature scheme is* blind under malicious keys *if for any PPT adversary* $\mathcal{A}$*, we have*

$$\mathsf{Adv}^{\mathrm{blind}}_{\mathcal{A}}(\lambda) = \Pr\left[\begin{array}{l}(\mathsf{bvk}, \boxed{t}, m_0, m_1) \leftarrow \mathcal{A}(1^\lambda),\ \mathsf{coin} \leftarrow \{0,1\},\\ (\rho_{1,b}, \mathsf{st}_b) \leftarrow \mathsf{User}(\mathsf{bvk}, \boxed{t}, m_b) \text{ for } b \in \{0,1\},\\ (\rho_{2,\mathsf{coin}}, \rho_{2,1-\mathsf{coin}}) \leftarrow \mathcal{A}(\rho_{1,\mathsf{coin}}, \rho_{1-\mathsf{coin}}),\\ \sigma_b \leftarrow \mathsf{Derive}(\mathsf{st}_b, \boxed{t}, \rho_{2,b}) \text{ for } b \in \{0,1\},\\ \text{if } \exists b \text{ s.t. } \mathsf{Verify}(\mathsf{bvk}, \boxed{t}, m_b, \sigma_b) = 0\text{:}\\ \quad \text{then } \sigma_0 = \sigma_1 = \bot,\end{array} : \mathsf{coin} = \mathcal{A}(\sigma_0, \sigma_1)\right] - \frac{1}{2}\Bigg| = \mathrm{negl}(\lambda).$$

Definition 12 (One-more Unforgeability). *A blind signature scheme is* one-more unforgeable *if for any* $Q = \mathsf{poly}(\lambda)$ *and PPT adversary* $\mathcal{A}$ *that for each common message* $\boxed{t}$ *makes at most* Q *signing queries containing the same* $\boxed{t}$*, we have*

$$\mathsf{Adv}^{\mathrm{omuf}}_{\mathcal{A}}(\lambda) = \Pr\left[\begin{array}{l}(\mathsf{bvk}, \mathsf{bsk}) \leftarrow \mathsf{KeyGen}(1^\lambda)\\ \{\boxed{t}, (m_i, \sigma_i)\}_{i \in [Q+1]} \leftarrow \mathcal{A}^{\mathsf{Signer}(\mathsf{bsk}, \cdot, \cdot)}(\mathsf{bvk})\end{array} : \begin{array}{l}\forall i \neq j \in [Q+1] : m_i \neq m_j\\ \wedge\ \mathsf{Verify}(\mathsf{bvk}, \boxed{t}, m_i, \sigma_i) = 1\end{array}\right] = \mathrm{negl}(\lambda).$$

2.6 Σ-Protocol

Let R be an NP relation with statements x and witnesses w. We denote by $\mathscr{L}_{\mathsf{R}} = \{x \mid \exists w \text{ s.t. } (x,w) \in \mathsf{R}\}$ the language induced by R. A Σ-protocol for an NP relation R for language $\mathscr{L}_{\mathsf{R}}$ is a tuple of PPT algorithms $\Sigma = (\mathsf{Init}, \mathsf{Chall}, \mathsf{Resp}, \mathsf{Verify})$ such that

- $\mathsf{Init}(x, w)$: given a statement $x \in \mathscr{L}_{\mathsf{R}}$, and a witness w such that $(x,w) \in \mathsf{R}$, outputs a first flow message (i.e., commitment) α and a state st, where we assume st includes x, w,
- $\mathsf{Chall}()$: samples a challenge $\beta \leftarrow \mathcal{CH}$ (without taking any input),
- $\mathsf{Resp}(\mathsf{st}, \beta)$: given a state st and a challenge $\beta \in \mathcal{CH}$, outputs a third flow message (i.e., response) γ,
- $\mathsf{Verify}(x, \alpha, \beta, \gamma)$: given a statement $x \in \mathscr{L}_{\mathsf{R}}$, a commitment α, a challenge $\beta \in \mathcal{CH}$, and a response γ, outputs a bit $b \in \{0,1\}$.

Here, $\mathcal{CH}$ denotes the challenge space. We call the tuple (α, β, γ) the *transcript* and say that they are *valid for* x if $\mathsf{Verify}(x, \alpha, \beta, \gamma)$ outputs 1. When the context is clear, we simply say it is valid and omit x.

We first define the standard notions of correctness, honest-verifier zero-knowledge, and 2-special soundness.

Definition 13 (Correctness). *A* Σ*-protocol is* correct*, if for all* $(x,w) \in \mathsf{R}$*,* $(\alpha, \mathsf{st}) \leftarrow \mathsf{Init}(x, w)$*,* $\beta \in \mathcal{CH}$*, and* $\gamma \leftarrow \mathsf{Resp}(\mathsf{st}, \beta)$*, it holds that* $\mathsf{Verify}(x, \alpha, \beta, \gamma) = 1$*.*

Definition 14 (High Min-Entropy). *A Σ-protocol has* high min-entropy *if for all $(x, w) \in \mathsf{R}$ and (possibly unbounded) adversary $\mathcal{A}$, it holds that*

$$\Pr[(\alpha, \mathsf{st}) \leftarrow \mathsf{Init}(x, w), \alpha' \leftarrow \mathcal{A}(1^\lambda) : \alpha = \alpha'] = \mathrm{negl}(\lambda).$$

Definition 15 (HVZK). *A Σ-protocol is* honest-verifier zero-knowledge *(HVZK), if there exists a PPT zero-knowledge simulator Sim such that the distributions of $\mathsf{Sim}(x, \beta)$ and the honestly generated transcript with Init initialized with (x, w) are computationally indistinguishable for any $x \in \mathscr{L}_\mathsf{R}$, and $\beta \in \mathcal{CH}$, where the honest execution is conditioned on β being used as the challenge.*

Definition 16 (2-Special Soundness). *A Σ-protocol is* 2-special sound*, if there exists a* deterministic *PT extractor Ext such that given two valid transcripts $\{(\alpha, \beta_b, \gamma_b)\}_{b\in[2]}$ for statment x with $\beta_0 \neq \beta_1$, along with x, outputs a witness w such that $(x, w) \in \mathsf{R}$.*

Note that in the above, two valid transcripts for x with the same commitment and different challenges imply that statement x is in $\mathscr{L}_\mathsf{R}$. That is, we do not guarantee x to lie in $\mathscr{L}_\mathsf{R}$ when invoking Ext. While subtle, this allows us to invoke Ext properly within the security proof even if the reduction cannot decide if the statement x output by the adversary indeed lies in $\mathscr{L}_\mathsf{R}$.

In the following, we propose a new notion of *f-unique extraction*. The notion is similar to the *unique response* property [37, 80] which requires that given an incomplete transcript (α, β), there is at most one response γ such that the transcript $\tau = (\alpha, \beta, \gamma)$ is valid. We relax this in two ways. First, we require that given a transcript τ and another challenge β', it is impossible to find two different responses γ_0, γ_1, such $w_0 \neq w_1$, where w_b is the witness extracted from τ and $\tau_b = (\alpha, \beta', \gamma_b)$. We further relax this by only requiring this property for a portion of the witness, defined by a function f, i.e., we require $f(w_0) \neq f(w_1)$ instead of $w_0 \neq w_1$.

While it may seem like an unnatural property, this is satisfied by many natural sigma protocols. In particular, if the first flow α contains a perfectly binding commitment $c = \mathsf{Commit}(f(w); r)$ to $f(w)$, and the extractor extracts the appropriate r, then the Σ-protocol has f-unique extraction. We remark also that a statistical variant of f-unique extraction is sufficient for our purpose. We choose the definition below for simplicity and because our instantiation satisfies it. See section 3 for more details and concrete example of f-unique extraction.

Definition 17 (f-Unique Extraction). *For a (possibly non-efficient) function f, a Σ-protocol Σ has* f-unique extraction *if for any statement x, any transcript $\tau = (\alpha, \beta, \gamma)$ and challenge $\beta' \neq \beta$, there is no γ_0, γ_1, such that for $\tau_b = (\alpha, \beta', \gamma_b)$, we have*

$$f(\mathsf{Ext}(x, \tau, \tau_0)) \neq f(\mathsf{Ext}(x, \tau, \tau_1)).$$

2.7 Non-Interactive Zero Knowledge

Given a witness w for statement x, a non-interactive zero-knowledge (NIZK) proof system allows a prover to generate a proof π that attests that she *knows* some w' such that $(w', x) \in \mathsf{R}$. Proofs π can be verified for statement x *without* revealing anything but that the statement is true. Here, we quantify "knowledge of the witness" either via *adaptive knowledge soundness* or *online-extractability*. The former informally states that if an algorithm $\mathcal{A}$ can generate a valid proof-statement pair (x, π), then there exists some extractor that when given black-box access to $\mathcal{A}$, can extract some witness w s.t. $(x, w) \in \mathsf{R}$. The latter requires that the witness w can be extracted from (x, π) "on-the-fly" without disrupting $\mathcal{A}$. In this context, we require some random oracle H on which proving and verification rely. Further, we assume that the prover and verifier are supplied with a common random string crs. As we later aim to avoid such a crs in our blind signature framework, the crs will be the output of a random oracle.

More formally, an NIZK for a relation R is a tuple of oracle-calling PPT algorithms $(\mathsf{Prove}^\mathsf{H}, \mathsf{Verify}^\mathsf{H})$ such that:

- $\mathsf{Prove}^\mathsf{H}(\mathsf{crs}, x, w)$: receives a common random string $\mathsf{crs} \in \{0,1\}^\ell$, a statement x and a witness w, and outputs a proof π,
- $\mathsf{Verify}^\mathsf{H}(\mathsf{crs}, x, \pi)$: receives a statement x and a proof π, and outputs a bit $b \in \{0, 1\}$.

An NIZK satisfies correctness and zero-knowledge, where we call it *adaptive knowledge sound* if it satisfies definition 20 and *online-extractable* if it satisfies definition 21.

Definition 18 (Correctness). *An* NIZK *is* correct *if for any* $\mathsf{crs} \in \{0,1\}^\ell$, $(x,w) \in \mathsf{R}$, *and* $\pi \leftarrow \mathsf{Prove}^{\mathsf{H}}(\mathsf{crs}, x, w)$, *it holds that* $\mathsf{Verify}^{\mathsf{H}}(\mathsf{crs}, x, \pi) = 1$.

Definition 19 (Zero-Knowledge). *An* NIZK *is* zero-knowledge *if there exists a PPT simulator* $\mathsf{Sim} = (\mathsf{Sim}_{\mathsf{H}}, \mathsf{Sim}_\pi)$ *such that for any PPT adversary* $\mathcal{A}$, *it holds that*

$$\mathsf{Adv}^{\mathrm{zk}}_{\mathcal{A}}(\lambda) = \left|\Pr\left[\mathcal{A}^{\mathsf{H},\mathcal{P}}(\mathsf{crs}) = 1\right] - \Pr\left[\mathcal{A}^{\mathsf{Sim}_{\mathsf{H}},\mathcal{S}}(\mathsf{crs}) = 1\right]\right| = \mathrm{negl}(\lambda),$$

where $\mathcal{P}$ *and* $\mathcal{S}$ *are oracles that on input* (x,w) *return* $\perp$ *if* $(x,w) \notin \mathsf{R}$, *and else output* $\mathsf{Prove}^{\mathsf{H}}(\mathsf{crs}, x, w)$ *or* $\mathsf{Sim}_\pi(\mathsf{crs}, x)$ *respectively. Note that the probability is taken over* $\mathsf{crs} \leftarrow \{0,1\}^\ell$ *and the random choices of* H, *and both* $\mathsf{Sim}_{\mathsf{H}}$ *and* Sim_π *have a shared state.*

We define adaptive knowledge soundness. We remark that the soundness relation $\mathsf{R}_{\mathsf{lax}}$ can be different from the (correctness) relation R. We are typically interested in $\mathsf{R} \subseteq \mathsf{R}_{\mathsf{lax}}$ and call $\mathsf{R}_{\mathsf{lax}}$ the *relaxed* relation.

Definition 20 (Adaptive Knowledge Soundness). *An* NIZK *is* adaptively knowledge sound *for relation* $\mathsf{R}_{\mathsf{lax}}$ *if there exists positive polynomials* $p_{\mathsf{T}}, p_{\mathsf{P}}$ *and a PPT algorithm* Ext *such that for any* $\mathsf{crs} \in \{0,1\}^\ell$, *given oracle access to any PPT adversary* $\mathcal{A}$ *(with explicit random tape* ρ*) that makes* $Q_H = \mathsf{poly}(\lambda)$ *random oracle queries with*

$$\Pr[(x,\pi) \leftarrow \mathcal{A}^{\mathsf{H}}(\mathsf{crs}; \rho) : \mathsf{Verify}^{\mathsf{H}}(\mathsf{crs}, x, \pi) = 1] \geq \mu(\lambda),$$

we have

$$\Pr\begin{bmatrix}(x,\pi) \leftarrow \mathcal{A}^{\mathsf{H}}(\mathsf{crs}; \rho), \\ w \leftarrow \mathsf{Ext}(\mathsf{crs}, x, \pi, \rho, \vec{h}) \end{bmatrix} : (x,w) \in \mathsf{R}_{\mathsf{lax}} \Bigg] \geq \frac{\mu(\lambda) - \mathrm{negl}(\lambda)}{p_{\mathsf{P}}(\lambda, Q_H)},$$

where $\vec{h}$ *are the outputs of* H, *and the probability is over the random tape* ρ *and the random choices of* H. *Also, we require that the runtime of* Ext *is bounded by* $p_{\mathsf{T}}(\lambda, Q_H) \cdot \mathsf{Time}(\mathcal{A})$.

We define (multi)-online extractability similarly to [34]. We consider a slightly simplified definition where the runtime of the extractor Ext does not depend on the advantage μ of the adversary $\mathcal{A}$.

Definition 21 ((Multi)-Online Extractability). *An* NIZK *is* online-extractable *if for all PPT adversaries* $\mathcal{A}$, *there exists a PPT simulator* SimCRS *and extractor* Ext, *such that*

CRS Indistinguishability. *For any PPT adversary* $\mathcal{A}$, *we have*

$$\mathsf{Adv}^{\mathrm{crs}}_{\mathcal{A}}(\lambda) = |\Pr[\mathsf{crs} \leftarrow \{0,1\}^\ell : \mathcal{A}^{\mathsf{H}}(\mathsf{crs}) = 1] - \Pr[(\overline{\mathsf{crs}}, \mathsf{td}) \leftarrow \mathsf{SimCRS}(1^\lambda) : \mathcal{A}^{\mathsf{H}}(\overline{\mathsf{crs}}) = 1]| = \mathrm{negl}(\lambda).$$

Online Extractability. *There exists positive polynomials* $p_{\mathsf{T}}, p_{\mathsf{P}}$ *such that for any* $Q_H = \mathsf{poly}(\lambda)$ *and PPT adversary* $\mathcal{A}$ *that makes at most* Q_H *random oracle queries with*

$$\Pr\left[(\overline{\mathsf{crs}}, \mathsf{td}) \leftarrow \mathsf{SimCRS}(1^\lambda), \{(x_i, \pi_i)\}_{i \in [Q_S]} \leftarrow \mathcal{A}^{\mathsf{H}}(\overline{\mathsf{crs}}) : \forall i \in [Q_S] : \mathsf{Verify}^{\mathsf{H}}(\mathsf{crs}, x_i, \pi_i) = 1\right] \geq \mu(\lambda),$$

it holds that

$$\Pr\left[\begin{matrix}(\overline{\mathsf{crs}}, \mathsf{td}) \leftarrow \mathsf{SimCRS}(1^\lambda), \{(x_i, \pi_i)\}_{i \in [Q_S]} \leftarrow \mathcal{A}^{\mathsf{H}}(\overline{\mathsf{crs}}), \\ \{w_i \leftarrow \mathsf{Ext}(\overline{\mathsf{crs}}, \mathsf{td}, x_i, \pi_i)\}_{i \in [Q_S]}\end{matrix} : \begin{matrix}\forall i \in [Q_S] : (x_i, w_i) \in \mathsf{R} \\ \wedge\ \mathsf{Verify}^{\mathsf{H}}(\overline{\mathsf{crs}}, x_i, \pi_i) = 1\end{matrix}\right] \geq \frac{\mu(\lambda) - \mathrm{negl}(\lambda)}{p_{\mathsf{P}}(\lambda, Q_H)},$$

where the runtime of Ext *is upper bounded by* $p_{\mathsf{T}}(\lambda, Q_H) \cdot \mathsf{Time}(\mathcal{A})$.

2.8 Splitting Lemma

We review the standard splitting lemma from Pointcheval and Stern [73].

Lemma 1 (Splitting Lemma). *Let $\epsilon \in (0,1]$ and $A \subseteq X \times Y$ such that*

$$\Pr_{(x,y)\leftarrow X\times Y}[(x,y) \in A] \geq \epsilon.$$

For any $\alpha \in [0,\epsilon)$ define

$$B = \left\{ (x,y) \in X \times Y \middle| \Pr_{(x,y')\leftarrow X\times Y}[(x,y') \in A] \geq \epsilon - \alpha \right\}.$$

Then we have

$$\Pr_{(x,y)\leftarrow X\times Y}[(x,y) \in B \mid (x,y) \in A] \geq \frac{\alpha}{\epsilon}.$$

3 Optimizing the Fischlin Blind Signature

In this section, we provide an optimized generic construction of blind signatures compared with the Fischlin blind signature [38]. In particular, we relax the extractable (and perfect binding) commitment and multi-online extractable NIZK used as the central building block for the Fischlin blind signature by a computationally binding commitment and a standard rewinding-based NIZK built from a Σ-protocol satisfying f-unique extraction. As we show in Section 4, this relaxation allows us to minimize the sum of the communication and signature size. We construct a natural partially blind variant in Section 7.

3.1 Construction

Our generic construction is based on the building blocks $(\mathsf{C},\mathsf{S},\Sigma)$ that satisfy some specific requirements. If $(\mathsf{C},\mathsf{S},\Sigma)$ satisfies these requirements, then we call it $\mathsf{BS}_{\mathsf{Rnd}}$-*suitable*.

Definition 22 ($\mathsf{BS}_{\mathsf{Rnd}}$-Suitable $(\mathsf{C},\mathsf{S},\Sigma)$). *The tuple of schemes $(\mathsf{C},\mathsf{S},\Sigma)$ are called* $\mathsf{BS}_{\mathsf{Rnd}}$-suitable*, if it holds that*

- C *is a correct and hiding* rerandomizable *commitment scheme with public parameter, message, randomness, and commitment spaces $\{0,1\}^{\ell_{\mathsf{C}}}, \mathcal{C}_{\mathsf{msg}}, \mathcal{C}_{\mathsf{rnd}}$, and $\mathcal{C}_{\mathsf{com}}$, respectively, such that $\mathcal{C}_{\mathsf{msg}}$ is efficiently sampleable and $1/|\mathcal{C}_{\mathsf{msg}}| = \mathrm{negl}(\lambda)$,*
- S *is a correct and* $\mathtt{euf\text{-}cma}$ *secure* deterministic *signature scheme with message space $\mathcal{S}_{\mathsf{msg}}$ that contains $\mathcal{C}_{\mathsf{com}}$, i.e., $\mathcal{C}_{\mathsf{com}} \subseteq \mathcal{S}_{\mathsf{msg}}$ and we assume elements in $\mathcal{S}_{\mathsf{msg}}$ are efficiently checkable,*
- Σ *is a correct, HVZK, 2-special sound Σ-protocol with high min-entropy, and challenge space $\mathcal{CH}$ with $1/|\mathcal{CH}| = \mathrm{negl}(\lambda)$ for the relation*

$$\mathsf{R}_{\mathsf{rnd}} \coloneqq \{x = (\mathsf{pp},\mathsf{vk},\overline{m}), w = (\mu,c,r) \mid \mathsf{C.Commit}(\mathsf{pp},\overline{m};r) = (c,r) \ \wedge \ \mathsf{S.Verify}(\mathsf{vk},\mu,c) = 1\}.$$

We also require Σ to be f-unique extraction where $f(w) = c$, i.e., f outputs c and ignores (μ,r).

Overview. Let $(\mathsf{C},\mathsf{S},\Sigma)$ be $\mathsf{BS}_{\mathsf{Rnd}}$-suitable. Let $\mathsf{H}_{\mathsf{par}},\mathsf{H}_{\mathsf{M}},\mathsf{H}_{\beta}$ be a random oracles from $\{0,1\}^*$ into $\{0,1\}^{\ell_{\mathsf{C}}}, \mathcal{C}_{\mathsf{msg}}, \mathcal{CH}$, respectively. We now describe our framework $\mathsf{BS}_{\mathsf{Rnd}}[\mathsf{C},\mathsf{S},\Sigma]$, or $\mathsf{BS}_{\mathsf{Rnd}}$ for short.

For key generation, the signer samples $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{S.KeyGen}(1^\lambda)$ and publishes $\mathsf{bvk} = \mathsf{vk}$ and stores $\mathsf{bsk} = \mathsf{sk}$. Further, the verification key bvk (or rather the hash functions) *implicitly* specifies the public parameter pp for C via $\mathsf{pp} = \mathsf{H}_{\mathsf{par}}(0)$.

To sign a message m, the user first commits to $\mathsf{H}_{\mathsf{M}}(m)$ via C and sends the commitment c to the signer. The signer then rerandomizes the commitment c to c' via sampling a rerandomization randomness Δr, and signs c' via S. It then sends the signature μ to the user along with Δr. The user checks by recomputing c' from c and Δr, and checks if μ is a valid signature on c'. Finally, the final blind signature is a proof π for relation $\mathsf{R}_{\mathsf{ext}}$, generated via Σ using Fiat-Shamir. Note

that π is a non-interactive proof of knowledge of a signature μ on a commitment c' to message $\mathsf{H}_{\mathsf{M}}(m)$.

Construction. In more detail, we have the following, where we assume pp is provided to all of the algorithms for readability.

- $\mathsf{BS}_{\mathsf{Rnd}}.\mathsf{KeyGen}(1^\lambda)$: samples $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{S}.\mathsf{KeyGen}(1^\lambda)$ and outputs verification key $\mathsf{bvk} = \mathsf{vk}$ and signing key $\mathsf{bsk} = \mathsf{sk}$.
- $\mathsf{BS}_{\mathsf{Rnd}}.\mathsf{User}(\mathsf{bvk}, m)$: sets $\overline{m} \leftarrow \mathsf{H}_{\mathsf{M}}(m)$ and outputs the commitment $c \in \mathcal{C}_{\mathsf{com}}$ generated via $(c, r) \leftarrow \mathsf{C}.\mathsf{Commit}(\mathsf{pp}, \overline{m})$ as the first message and stores the randomness $\mathsf{st} = r \in \mathcal{C}_{\mathsf{rnd}}$.
- $\mathsf{BS}_{\mathsf{Rnd}}.\mathsf{Signer}(\mathsf{bsk}, c)$: checks if $c \in \mathcal{C}_{\mathsf{com}}$, samples a rerandomization randomness $\Delta r \leftarrow \mathcal{C}_{\mathsf{rnd}}$, rerandomizes the commitment c via $c' = \mathsf{C}.\mathsf{RerandCom}(\mathsf{pp}, c, \Delta r)$, signs $\mu \leftarrow \mathsf{S}.\mathsf{Sign}(\mathsf{sk}, c')$, and finally outputs the second message $\rho = (\mu, \Delta r)$.
- $\mathsf{BS}_{\mathsf{Rnd}}.\mathsf{Derive}(\mathsf{st}, \rho)$: parse $\mathsf{st} = r$, $\rho = (\mu, \Delta r)$ and checks $\Delta r \in \mathcal{C}_{\mathsf{rnd}}$. It then computes the randomized commitment $c'' = \mathsf{C}.\mathsf{RerandCom}(\mathsf{pp}, c, \Delta r)$ and randomized randomness $r' \leftarrow \mathsf{C}.\mathsf{RerandRand}(\mathsf{pp}, c, \overline{m}, r, \Delta r)$, and checks $\mathsf{S}.\mathsf{Verify}(\mathsf{vk}, c'', \mu) = 1$ and $c'' = \mathsf{C}.\mathsf{Commit}(\mathsf{pp}, \overline{m}; r')$. Finally, it outputs a signature $\sigma = \pi$, where $(\alpha, \mathsf{st}') \leftarrow \Sigma.\mathsf{Init}(x, w)$, $\beta \leftarrow \mathsf{H}_\beta(x, \alpha), \gamma \leftarrow \Sigma.\mathsf{Resp}(x, \mathsf{st}', \beta)$, $\pi = (\alpha, \beta, \gamma)$ with $x = (\mathsf{pp}, \mathsf{vk}, \overline{m})$, $w = (\mu, c'', r')$.
- $\mathsf{BS}_{\mathsf{Rnd}}.\mathsf{Verify}(\mathsf{bvk}, m, \sigma)$: parses $\sigma = \pi$ and $\pi = (\alpha, \beta, \gamma)$, sets $\overline{m} = \mathsf{H}_{\mathsf{M}}(m)$ and $x = (\mathsf{pp}, \mathsf{vk}, \overline{m})$, and outputs 1 if $\beta = \mathsf{H}_\beta(x, \alpha)$, $\Sigma.\mathsf{Verify}(x, \alpha, \beta, \gamma) = 1$, and otherwise outputs 0.

3.2 Correctness and Security

We prove correctness, blindness, and one-more unforgeability. The correctness of $\mathsf{BS}_{\mathsf{Rnd}}$ follows directly from the correctness of the underlying schemes $(\mathsf{C}, \mathsf{S}, \Sigma)$. Blindness follows mainly from the HVZK property of Σ and the hiding property of C. The only thing to be aware of is that the user needs to check the validity of the rerandomized commitment c'' by computing a rerandomized randomness using the randomness r used to compute the original commitment c. In order to invoke the hiding property of C on c, we rely on the correctness of the randomization property so that the reduction no longer needs to check the validity of c''.

The main technical challenge is the proof of one-more unforgeability. Here, we argue that we can use a successful attacker on the one-more unforgeability of $\mathsf{BS}_{\mathsf{Rnd}}$ to extract some forgery σ for S. The crux is to ensure that the extracted commitment was never signed during $\mathsf{Signer}(\mathsf{bsk}, \cdot)$ queries, even if we rewind the adversary in order to extract the witness from the forgery. We ensure this with two ideas: We first use f-unique extractionof Σ to argue that the forgeries implicitly fix the to-be-extracted commitments c_i's in the first execution of the adversary. We then use the fact that the reduction adds new randomness in the rewound executions outside of the adversary's control, that is, rerandomize the commitment c submitted to the signing oracle, to argue that the c_i's cannot appear in the list of rerandomized commitments. Finally, we note that we extract from all the proofs included in the forgeries since one extraction is not enough: since we're only using a computationally binding commitment, the adversary may be breaking the binding property, in which case, the reduction needs at least two witnesses. To this end, we perform a more fine-grained analysis of the standard forking lemma. More details can be found in section 1.3.

Theorem 1 (Correctness). *The scheme $\mathsf{BS}_{\mathsf{Rnd}}$ is correct.*

Proof. Let $\mathsf{pp} \leftarrow \mathsf{H}_{\mathsf{par}}(0)$, $m \in \{0,1\}^*$ and $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{S}.\mathsf{KeyGen}(1^\lambda)$. The user first computes $\overline{m} \leftarrow \mathsf{H}_{\mathsf{M}}(m)$ and $(c, r) \leftarrow \mathsf{C}.\mathsf{Commit}(\mathsf{pp}, \overline{m}; r)$ for some $r \leftarrow \mathcal{C}_{\mathsf{rnd}}$. Note that $c \in \mathcal{C}_{\mathsf{com}}$ as $\overline{m} \in \mathcal{C}_{\mathsf{msg}}$. The signer computes $\Delta r \leftarrow \mathcal{C}_{\mathsf{rnd}}$, $c' = \mathsf{C}.\mathsf{RerandCom}(\mathsf{pp}, c, \Delta r)$, and $\mu \leftarrow \mathsf{S}.\mathsf{Sign}(\mathsf{sk}, c')$, where $c' \in \mathcal{C}_{\mathsf{com}} \subseteq \mathcal{S}_{\mathsf{msg}}$. The user computes the randomized commitment $c'' = \mathsf{C}.\mathsf{RerandCom}(\mathsf{pp}, c, \Delta r)$ and randomized randomness $r' \leftarrow \mathsf{C}.\mathsf{RerandRand}(\mathsf{pp}, c, \overline{m}, r, \Delta r)$, and checks $\mathsf{S}.\mathsf{Verify}(\mathsf{vk}, c'', \mu) = 1$ and $c'' = \mathsf{C}.\mathsf{Commit}(\mathsf{pp}, \overline{m}; r')$. This holds since $\mathsf{RerandCom}$ is deterministic, and by the correctness of the *rerandomized* commitment (see definition 7) and of S. It then computes $\sigma = \pi$, where π is a proof generated using $x = (\mathsf{pp}, \mathsf{vk}, \overline{m}), w = (\sigma, c', r')$ and we have $(x, w) \in \mathsf{R}_{\mathsf{rnd}}$ due to the previous check. As $\overline{m} = \mathsf{H}_{\mathsf{M}}(m)$ and Σ is correct, we have $\Sigma.\mathsf{Verify}(x, \alpha, \beta, \gamma) = 1$ as desired.

Theorem 2 (Blindness). *The scheme $\mathsf{BS}_{\mathsf{Rnd}}$ is blind under malicious keys under the hiding and rerandomization properties of C and the high min-entropy and HVZK properties of Σ.*

Proof. Let $\mathcal{A}$ be a PPT adversary against blindness and Q_H denote the number of H_β queries. We define the following hybrids and denote by $\mathsf{Adv}^{\mathrm{H}_i}_{\mathcal{A}}(\lambda)$ the advantage of $\mathcal{A}$ in Hybrid i.

- Hybrid 0 is identical to the real game.
- Hybrid 1 is the same as Hybrid 0, except the challenger aborts if the random oracle H_β was already queried on (x_b, α_b) before generating $\pi_b = (\alpha_b, \beta_b, \gamma_b)$ for $b \in \{0,1\}$. In more detail, the challenger runs $(\alpha_b, \mathsf{st}'_b) \leftarrow \Sigma.\mathsf{Init}(x_b, w_b)$, where x_b and w_b are defined as in the protocol, samples $\beta_b \leftarrow \mathcal{CH}$, runs $\gamma_b \leftarrow \Sigma.\mathsf{Resp}(x, \alpha_b, \mathsf{st}'_b, \beta_b)$. Then if (x_b, α_b) for either $b = 0$ or 1 were queried to H_β, the challenger aborts the game. Otherwise, the challenger programs $\mathsf{H}_\beta(x_b, \alpha_b) \leftarrow \beta_b$.
  Hybrids 0 and 1 differ only when the game aborts. Due to the high min-entropy of Σ, the probability that the random oracle is already defined on input (x_b, α_b) is bounded by $Q_H \cdot \mathrm{negl}(\lambda) = \mathrm{negl}(\lambda)$ for each $b \in \{0,1\}$. Taking the union bound on $b \in \{0,1\}$, we have that $|\mathsf{Adv}^{\mathrm{H}_0}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda)| \leq \mathrm{negl}(\lambda)$.
- Hybrid 2 is the same as Hybrid 1, except we omit the check on the rerandomized commitments. That is, when the challenger receives a second message $\rho_b = (\mu_b, \Delta r_b)$ from the adversary, it only computes the randomized commitment $c''_b = \mathsf{C.RerandCom}(\mathsf{pp}, c_b, \Delta r_b)$ and checks $\mathsf{S.Verify}(\mathsf{vk}, c''_b, \mu_b) = 1$. Recall in the previous hybrid, the challenger further computed the randomized randomness $r'_b \leftarrow \mathsf{C.RerandRand}(\mathsf{pp}, c_b, \overline{m}_b, r_b, \Delta r_b)$ and checked $c''_b = \mathsf{C.Commit}(\mathsf{pp}, \overline{m}_b; r'_b)$.
  Hybrids 1 and 2 differ only when $c''_b \neq \mathsf{C.Commit}(\mathsf{pp}, \overline{m}_b; r'_b)$. However, this can never occur due to the correctness of the rerandomized commitment (see definition 7). Hence, $\mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda)$.
- Hybrid 3 is the same as Hybrid 2, except the proofs π_b are simulated without the witness w_b. That is, the challenger generates a simulated transcript $(\alpha_b, \beta_b, \gamma_b)$ by sampling $\beta_b \leftarrow \mathcal{CH}$ and running $(\alpha_b, \gamma_b) \leftarrow \Sigma.\mathsf{Sim}(x_b, \beta_b)$.
  We can construct an adversary $\mathcal{B}_\Sigma$ such that $|\mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda)| \leq 2 \cdot \mathsf{Adv}^{\mathrm{hvzk}}_{\mathcal{B}_\Sigma}(\lambda)$. Essentially, $\mathcal{B}_\Sigma$ challenges $\mathcal{A}$ and uses the provided oracles to generate proofs and answer H_β queries. If the oracle outputs simulated proofs, the game is distributed identically to Hybrid 3. Else, the oracle outputs real proofs and behaves as in Hybrid 2.
- Hybrid 4 is the same as Hybrid 3, except the commitment-randomness pair $(c_b, r_b) \leftarrow \mathsf{C.Commit}(\mathsf{pp}, \mathsf{H}_\mathsf{M}(m_b))$ are instead computed as commitments to $\mathsf{H}_\mathsf{M}(0)$.
  It is straightforward to construct an adversary $\mathcal{B}_\mathsf{C}$ on the hiding property of C with $|\mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_4}_{\mathcal{A}}(\lambda)| \leq 2 \cdot \mathsf{Adv}^{\mathrm{hide}}_{\mathcal{B}_\mathsf{C}}(\lambda)$ by noticing that the challenger no longer requires the commitment randomness r_b to simulate $\mathcal{A}$ due to the modification we made in Hybrids 2 and 3. We note that the public parameters pp obtained from the hiding challenger can be programmed into $\mathsf{H}_\mathsf{par}(0)$.

In Hybrid 4, the value of coin is information-theoretically hidden from $\mathcal{A}$, as the commitments c_b and the proofs π_b are identically distributed for $b \in \{0,1\}$. Consequently, $\mathsf{Adv}^{\mathrm{H}_4}_{\mathcal{A}}(\lambda) = 0$. Also, the running time of the adversaries $\mathcal{B}_\mathsf{C}$ and $\mathcal{B}_\Sigma$ are roughly that of $\mathcal{A}$. Combining the inequalities yields the statement.

Theorem 3 (One-More Unforgeability). *The scheme BS_Rnd is one-more unforgeable under the binding and rerandomizability properties of C, euf-cma security of S, and the 2-special soundness and f-unique extraction properties of Σ.*

Proof. Let $\mathcal{A}$ be a PPT adversary against one-more unforgeability. Denote by Q_S the number of signing queries, by Q_M the number of H_M queries, and by Q_H the number of H_β queries. Recall that we model $\mathsf{H}_\mathsf{par}, \mathsf{H}_\mathsf{M}$, and H_β as random oracles, where we assume without loss of generality that $\mathcal{A}$ never repeats queries. In the end of the interaction with $\mathcal{A}$, that is after Q_S signing queries, $\mathcal{A}$ outputs $Q_S + 1$ forgeries $\{(m_i, \sigma_i)\}_{i \in [Q_S+1]}$. We write $\sigma_i = \pi_i$ and denote by c_i the Q_S first message queries to $\mathsf{BS}_\mathsf{Rnd}.\mathsf{Signer}(\mathsf{bsk}, \cdot)$ issued by $\mathcal{A}$. Note that if $\mathcal{A}$ is successful, then we have $\Sigma.\mathsf{Verify}(x_i, \alpha_i, \beta_i, \gamma_i) = 1$ and $\beta_i = \mathsf{H}_\beta(x_i, \alpha_i)$ for $\overline{m}_i = \mathsf{H}_\mathsf{M}(m_i)$, $x_i = (\mathsf{pp}, \mathsf{vk}, \overline{m}_i)$, and $\pi_i = (\alpha_i, \beta_i, \gamma_i)$. We first slightly alter the real game and remove subtle conditions to make the later proofs easier. We denote by $\mathsf{Adv}^{\mathrm{H}_i}_{\mathcal{A}}(\lambda)$ the advantage of $\mathcal{A}$ in Hybrid i for $i \in \{0,1\}$.

- Hybrid 0 is identical to the real game.
- Hybrid 1 is the same as Hybrid 0, except it aborts if there is a collision in H_M or H_β, or there is some (x_i, α_i) for $i \in [Q_S + 1]$ that was never queried to H_β.

It suffices to upper bound the abort probability. A collision in H_M (resp. H_β) happens with probability at most $Q_M^2/|\mathcal{C}_\mathsf{msg}|$ (resp. $Q_H^2/|\mathcal{CH}|$) (which follows for example from a union bound). Moreover, the probability that some fixed β_i of $\mathcal{A}$'s output equals to $\mathsf{H}_\beta(x_i, \alpha_i)$ is exactly $1/|\mathcal{CH}|$, if (x_i, α_i) was never queried to H_β. Thus, it follows that $\mathsf{Adv}^{\mathrm{H}_0}_{\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) + \frac{Q_M^2}{|\mathcal{C}_\mathsf{msg}|} + \frac{Q_H^2+1}{|\mathcal{CH}|} = \mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) + \mathrm{negl}(\lambda)$.

Description of Wrapper Algorithm $\mathcal{B}$. We now present a wrapper algorithm $\mathcal{B}$ that simulates the interaction between the challenger $\mathcal{G}$ and $\mathcal{A}$ in Hybrid 1. Looking ahead we apply a generalization of the standard forking lemma on $\mathcal{B}$ to extract the witnesses from all the proof (i.e. forgery) output by $\mathcal{A}$.

Notice that $\mathcal{G}$ is deterministic once the keys $(\mathsf{vk}, \mathsf{sk})$ of the (deterministic) signature scheme S, the Q_S rerandomization randomness in $\mathcal{C}_\mathsf{rnd}$, and the outputs of the random oracles $\mathsf{H}_\mathsf{par}, \mathsf{H}_\mathsf{M}, \mathsf{H}_\beta$ are determined. Since H_par is only used to generate the public parameter pp of the commitment scheme, we assume without loss of generality that only pp is given to $\mathcal{A}$ rather than access to H_par. We use coin to denote all the Q_M outputs of H_M and the random coins used by $\mathcal{A}$. We use $\vec{h} = (\widehat{\beta}_i, \Delta r_i)_{i\in[Q_H+Q_S]} \in (\mathcal{CH} \times \mathcal{C}_\mathsf{rnd})^{Q_H+Q_S}$ to explicitly denote the list that will be used to simulate the outputs of H_β and rerandomziation randomness sampled by $\mathcal{G}$. Here, we note that $\vec{h}$ is deliberately defined redundantly since $\mathcal{G}$ only needs Q_H hash outputs and Q_S rerandomziation randomness, rather than $Q_H + Q_S$ of them each. We also use $\widehat{\beta} \in \mathcal{CH}$ to denote the output of H_β to distinguish between the hash value β included in $\mathcal{A}$'s forgeries. We then define $\mathcal{B}$ as an algorithm that has oracle access to $\mathsf{S.Sign}(\mathsf{sk}, \cdot)$ as follows:

$\mathcal{B}^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(\mathsf{pp}, \mathsf{vk}, \vec{h}; \mathsf{coin})$: On input pp, vk, and $\vec{h} \in (\mathcal{CH}\times\mathcal{C}_\mathsf{rnd})^{Q_H+Q_S}$, $\mathcal{B}$ simulates the interaction between the challenger $\mathcal{G}$ and $\mathcal{A}$ in Hybrid 1. $\mathcal{B}$ invokes $\mathcal{A}$ on the randomness included in coin and simulates $\mathcal{G}$, where it runs the same code as $\mathcal{G}$ except for the following differences:

- It uses the provided pp and vk rather than generating it on its own;
- All Q_M random oracle queries to H_M are answered using the hash values include in coin;
- On the i-th ($i \in [Q_H]$) random oracle query to H_β, it retrieves an unused $(\widehat{\beta}_k, \Delta r_k)$ with the smallest index $k \in [Q_H + Q_S]$ and outputs $\widehat{\beta}_k$ and discards Δr_k;
- On the i-th ($i \in [Q_S]$) first message $c_i \in \mathcal{C}_\mathsf{com}$ from $\mathcal{A}$, it retrieves an unused $(\widehat{\beta}_k, \Delta r_k)$ with the smallest index $k \in [Q_H+Q_S]$ and discards $\widehat{\beta}_k$. It then computes $c_i' = \mathsf{C.RerandCom}(\mathsf{pp}, c_i, \Delta r_k)$, queries the signing oracle on c_i', obtains $\mu_i \leftarrow \mathsf{S.Sign}(\mathsf{sk}, c_i')$, and returns the second message $\rho_i = (\mu_i, \Delta r_k)$.

At the end of the game when $\mathcal{A}$ outputs the forgeries, $\mathcal{B}$ checks if the forgeries are valid and the added condition in Hybrid 1. If the check does not pass, then $\mathcal{B}$ outputs $((0)_{i\in[Q_S+1]}, \perp)$, i.e., $Q_S + 1$ zeros followed by a $\perp$. Otherwise, $\mathcal{B}$ finds the indices $I_i \in [Q_H + Q_S]$ such that $\mathsf{H}_\beta(x_i, \alpha_i) = \beta_i = \widehat{\beta}_{I_i}$ for $i \in [Q_S + 1]$, which are guaranteed to exist uniquely due to the modification we made in Hybrid 1. It then sets $\Lambda = (x_i, \alpha_i, \beta_i, \gamma_i)_{i\in[Q_S+1]}$ and outputs $((I_i)_{i\in[Q_S+1]}, \Lambda)$. It can be checked that $\mathcal{B}$ perfectly simulates the view of the challenger $\mathcal{G}$ in Hybrid 1. Therefore, $\mathcal{B}$ outputs $\Lambda \neq \perp$ with probability $\mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda)$.

Description of Forking Algorithm $\mathsf{F}_\mathcal{B}$. We now define a generalization of the standard forking algorithm F so that F keeps on rewinding $\mathcal{B}$ until some condition is satisfied. Concretely, F takes as input $(\mathsf{pp}, \mathsf{vk})$, has oracle access to $\mathsf{S.Sign}(\mathsf{sk}, \cdot)$, and invokes $\mathcal{B}$ internally as depicted in algorithm 1, where the number of repetition T is defined below.

We show that if $\mathcal{A}$ succeeds in breaking one-more unforgeability in Hybrid 1 with non-negligible probability, then we can set a specific number of repetition T so that the forking algorithm $\mathsf{F}_\mathcal{B}$ terminates in polynomial time and succeeds in outputting a non-$\perp$ with non-negligible probability. Formally, we have the following lemma.

Lemma 2. *Let $\epsilon = \mathsf{Adv}^{H_1}_{\mathcal{A}}(\lambda)$. Then, if we set $T = \left(\frac{\epsilon}{(Q_H+Q_S)(Q_S+2)^2}\right)^{-1} \cdot \log(2Q_S+2)$, $\mathsf{F}_\mathcal{B}$ outputs a non-$\perp$ with probability at least $\frac{\epsilon}{2(Q_S+2)^2}$.*

In particular, if ϵ is non-negligible, then $T = \mathsf{poly}(\lambda)$. Moreover, the running time of $\mathsf{F}_\mathcal{B}$ is at most (roughly) a factor $T \cdot (Q_S + 1) + 1$ more of $\mathcal{B}$ (or equivalently $\mathcal{A}$), so $\mathsf{F}_\mathcal{B}$ runs in polynomial time.

Algorithm 1 Description of the forking algorithm $\mathsf{F}_{\mathcal{B}}^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(\mathsf{pp},\mathsf{vk})$

Pick coin for $\mathcal{B}$ at random.
$\vec{h} \leftarrow (\mathcal{CH} \times \mathcal{C}_{\mathsf{rnd}})^{Q_H+Q_S}$
$\left((I_i)_{i\in[Q_S+1]}, \Lambda\right) \leftarrow \mathcal{B}^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(\mathsf{pp},\mathsf{vk},\vec{h};\mathsf{coin})$
if $\Lambda = \bot$ **then**
  return $\bot$ ▷ Return fail.
$D \coloneqq ()$ ▷ Prepare empty list.
for $j \in [Q_S+1]$ **do**
  $(c, \mathsf{flag}) \coloneqq (1, \bot)$
  while $c \in [T] \wedge \neg\mathsf{flag}$ **do**
    $\vec{h}^{(c)}_{j,\geq I_j} \leftarrow (\mathcal{CH} \times \mathcal{C}_{\mathsf{rnd}})^{Q_H+Q_S-I_j+1}$
    $\vec{h}^{(c)}_j \coloneqq \vec{h}_{<I_j} \| \vec{h}^{(c)}_{j,\geq I_j}$
    $\left((I^{(c)}_{j,i})_{i\in[Q_S+1]}, \Lambda^{(c)}_j\right) \leftarrow \mathcal{B}^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(\mathsf{pp},\mathsf{vk},\vec{h}^{(c)}_j;\mathsf{coin})$
    if $I^{(c)}_{j,j} = I_j$ **then**
      $D = D \cup (j, I_j, \Lambda^{(c)}_j)$
      $\mathsf{flag} = \top$ ▷ Break from while loop.
    $c = c + 1$
if $|D| < Q_S + 1$ **then** ▷ Check if $\mathcal{B}$ succeeds in all Q_S+1 run.
  return $\bot$ ▷ Return fail.
return (Λ, D)

Proof. Assume $\mathcal{B}$ outputs a valid $\Lambda = (x_i, \alpha_i, \beta_i, \gamma_i)_{i\in[Q_S+1]}$ in the first execution and denote this event as E. For $i \in [Q_S+1]$, we denote the tuple $(x_i, \alpha_i, \beta_i, \gamma_i)$ as the i-th *forgery*. For any $(i,k) \in [Q_S+1] \times [Q_H+Q_S]$, we denote $\mathsf{E}_{i,k}$ as the event that forgery is associated to the k-th hash query, i.e., the k-th entry of $\vec{h} \in (\mathcal{CH} \times \mathcal{C}_{\mathsf{rnd}})^{Q_H+Q_S}$ includes β_i. Here, note that $\forall i \in [Q_S+1]$, we have $\sum_{k\in[Q_H+Q_S]} \Pr[\mathsf{E}_{i,k}] = 1$. We define the set P_i as

$$P_i = \left\{ k \;\middle|\; \Pr[\mathsf{E}_{i,k} \mid \mathsf{E}] \geq \frac{1}{(Q_H+Q_S)(Q_S+2)} \right\},$$

where for any $k \in P_i$, we have $\Pr[\mathsf{E}_{i,k}] \geq \frac{\epsilon}{(Q_H+Q_S)(Q_S+2)}$. Let us define $\mathsf{E}^{\mathsf{good}}_i = \bigvee_{k\in P_i} \mathsf{E}_{i,k}$. Then, we have $\Pr\left[\mathsf{E}^{\mathsf{good}}_i \;\middle|\; \mathsf{E}\right] \geq \frac{Q_S+1}{Q_S+2}$, since there are at most (Q_H+Q_S) possible values of k's not in P_i and they can only account to a probability at most $(Q_H+Q_S) \times \frac{1}{(Q_H+Q_S)(Q_S+2)} = \frac{1}{Q_S+2}$.

Next, for any $(i,k) \in [Q_S+1] \times P_i$, let us define $X_{i,k} = R_{\mathsf{coin}} \times (\mathcal{CH} \times \mathcal{C}_{\mathsf{rnd}})^{k-1}$ and $Y_{i,k} = (\mathcal{CH} \times \mathcal{C}_{\mathsf{rnd}})^{Q_H+Q_S-k+1}$, where R_{coin} denotes the randomness space of coin. Here, note that $(\mathsf{x}_i, \vec{h}_{\geq k}) \in X_{i,k} \times Y_{i,k}$ can be parsed appropriately to be $(\mathsf{coin}, \vec{h})$, and defines all the inputs of $\mathcal{B}$, where we assume a fixed $(\mathsf{pp},\mathsf{vk})$. We further define $A_{i,k} \subseteq X_{i,k} \times Y_{i,k}$ to be the set of inputs that triggers event $\mathsf{E}_{i,k}$. Then using the splitting lemma (cf. lemma 1) with $\alpha = \frac{Q_S+1}{Q_S+2} \cdot \frac{\epsilon}{(Q_H+Q_S)(Q_S+2)}$, there exists a set $B_{i,k} \subset X_{i,k} \times Y_{i,k}$ such that

$$B_{i,k} = \left\{ (\mathsf{x}_i, \vec{h}_{\geq k}) \in X_{i,k} \times Y_{i,k} \;\middle|\; \Pr_{\vec{h'}_{\geq k} \leftarrow Y_{i,k}} \left[(\mathsf{x}_i, \vec{h'}_{\geq k}) \in A_{i,k} \right] \geq \frac{\epsilon}{(Q_H+Q_S)(Q_S+2)^2} \right\}, \quad (1)$$

and

$$\Pr_{(\mathsf{x}_i, \vec{h'}_{\geq k}) \leftarrow X_{i,k} \times Y_{i,k}} \left[(\mathsf{x}_i, \vec{h}_{\geq k}) \in B_{i,k} \;\middle|\; (\mathsf{x}, \vec{h}_{\geq k}) \in A_{i,k} \right] \geq \frac{Q_S+1}{Q_S+2}. \quad (2)$$

We are now ready to evaluate the success probability of the forking algorithm $\mathsf{F}_{\mathcal{B}}$. With probability ϵ, $\mathcal{B}$ outputs $((I_i)_{i\in[Q_S+1]}, \Lambda)$ in the first execution on input $(\mathsf{coin}, \vec{h}) \in R_{\mathsf{coin}} \times (\mathcal{CH} \times$

$\mathcal{C}_{\mathsf{rnd}})^{Q_H+Q_S}$. Then the probability that event $\mathsf{E}_i^{\mathsf{good}}$ occurrs for all $i \in [Q_S+1]$ is at least

$$\Pr\left[\forall i \in [Q_S+1],\ \mathsf{E}_i^{\mathsf{good}} \middle| \mathsf{E}\right] \geq 1 - \sum_{i\in[Q_S+1]} \Pr\left[\neg\mathsf{E}_i^{\mathsf{good}} \middle| \mathsf{E}\right] \geq \frac{1}{Q_S+2},$$

where the first inequality follows from the union bound and the second inequality follows from $\Pr\left[\mathsf{E}_i^{\mathsf{good}} \;\middle|\; \mathsf{E}\right] \geq \frac{Q_S+1}{Q_S+2}$.

Then, from eq. (2) and following the same union bound argument, $\mathsf{F}_{\mathcal{B}}$ samples a good input such that $(\mathsf{coin}, \vec{h}) \in B_{i,I_i}$ for all $i \in [Q_S+1]$ conditioned on $\mathsf{E}_i^{\mathsf{good}}$ for all $i \in [Q_S+1]$ with probability at least $\frac{1}{(Q_S+2)}$. Therefore, by eq. (1), if $\mathsf{F}_{\mathcal{B}}$ resamples $\vec{h}_{i,\geq I_i} \in Y_{i,I_i} = (\mathcal{CH} \times \mathcal{C}_{\mathsf{rnd}})^{Q_H+Q_S-I_i+1}$ conditioned on the set B_{i,I_i}, $\mathcal{B}$ succeeds on input $(\mathsf{coin}, \vec{h}_{i,<I_i} \| \vec{h}_{i,\geq I_i})$ with probability at least $\frac{\epsilon}{(Q_H+Q_S)(Q_S+2)^2}$. Conditioning on sampling an input $(\mathsf{coin}, \vec{h}) \in B_{i,I_i}$ for all $i \in [Q_S+1]$ and noting the independence of each rewinding, the probability that $\mathcal{B}$ succeeds in all j-th rewinding for $j \in [Q_S+1]$ is at least

$$\begin{aligned}\left(1 - \left(1 - \frac{\epsilon}{(Q_H+Q_S)(Q_S+2)^2}\right)^T\right)^{Q_S+1} &\geq \left(1 - \frac{1}{e^{\log(2Q_S+2)}}\right)^{Q_S+1} \\ &= \left(1 - \frac{1}{2(Q_S+1)}\right)^{Q_S+1} \geq \frac{1}{2}.\end{aligned}$$

Collecting all the bounds, we conclude that $\mathsf{F}_{\mathcal{B}}$ succeeds with probability at least $\frac{\epsilon}{2(Q_S+2)^2}$ as desired. Moreover, the running time of $\mathsf{F}_{\mathcal{B}}$ is roughly the same as running $\mathcal{B}$ for at most $T \cdot (Q_S+1)+1$ times, where the runtime of $\mathcal{B}$ is roughly the same as the runtime of $\mathcal{A}$.

Using $\mathsf{F}_{\mathcal{B}}$ to Break Binding of C or `euf-cma` of S. We are now ready to finish the proof. Assume $\epsilon = \mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda)$ is non-negligible. We use $\mathsf{F}_{\mathcal{B}}$ to extract the witnesses from the proofs output by $\mathcal{A}$ with non-negligible probability and show that such witnesses can be used to break either the binding of C or the `euf-cma` security of S. Thus establishing that $\epsilon = \mathrm{negl}(\lambda)$ by contradiction.

We define adversary $\mathcal{A}_{\mathsf{C},\mathsf{S}}$ on both the binding property of C and the `euf-cma` property of S as follows. Initially, $\mathcal{A}_{\mathsf{C},\mathsf{S}}$ obtains pp from the binding challenger. Further, she receives vk and oracle access to a signing oracle $\mathsf{S.Sign}(\mathsf{sk},\cdot)$ from the `euf-cma` challenger. Then, she runs the forking algorithm $R \leftarrow \mathsf{F}_{\mathcal{B}}^{\mathsf{S.Sign}(\mathsf{sk},\cdot)}(\mathsf{pp},\mathsf{vk})$. She checks $R \neq \perp$, and parses $R = (\Lambda, D)$, where $\Lambda = (x_i, \alpha_i, \beta_i, \gamma_i)_{i\in[Q_S+1]}$ and $D = (j, I_j, \Lambda_j)_{j\in[Q_S+1]}$. Due to lemma 2, $\mathsf{F}_{\mathcal{B}}$ runs in polynomial time and has non-negligible success probability. Below, we describe the second part of $\mathcal{A}_{\mathsf{C},\mathsf{S}}$ and analyze its success probability conditioned on $\mathsf{F}_{\mathcal{B}}$ succeeding. (If $R = \perp$, then $\mathcal{A}_{\mathsf{C},\mathsf{S}}$ outputs $\perp$ and aborts.)

For $j \in [Q_S+1]$, we denote by $(x'_j, \alpha'_j, \beta'_j, \gamma'_j)$ the j-th element of the tuple Λ_j. Moreover, note that the same coin and values $(\widehat{\beta}_1, \Delta r_1), \ldots (\widehat{\beta}_{I_j-1}, \Delta r_{I_j-1})$ are used for the initial run of $\mathcal{B}$ and the run of $\mathcal{B}$ where $\mathcal{B}$ outputs Λ_j. Thus, we have for all $j \in [Q_S+1]$ that $(x_i, \alpha_i) = (x'_i, \alpha'_i)$. Moreover, we have $\widehat{\beta}_{I_j} \neq \widehat{\beta}_{j,I_j}$, or equivalently $\beta_j \neq \beta'_j$ for all $j \in [Q_S+1]$ with probability at least $1 - \frac{Q_S+1}{|\mathcal{CH}|} = 1 - \mathrm{negl}(\lambda)$ since each hash outputs are sampled uniformly and independently at random. This allows $\mathcal{A}_{\mathsf{C},\mathsf{S}}$ to invoke 2-special soundness of Σ with overwhelming probability. For all $i \in [Q_S+1]$, she runs Ext on $(x_i, (\alpha_i, \beta_i, \gamma_i), (\alpha_i, \beta'_i, \gamma'_i))$ to extract a witness $w_i = (\mu_i, c_i, r_i)$ such that $\mathsf{C.Commit}(\mathsf{pp}, \overline{m}_i; r_i) = c_i \wedge \mathsf{S.Verify}(\mathsf{vk}, \mu_i, c_i) = 1$, where $x_i = (\mathsf{pp}, \mathsf{vk}, \overline{m}_i)$.

If there exists distinct $i, j \in [Q_S+1]$ with $c_i = c_j$, $\mathcal{A}_{\mathsf{C},\mathsf{S}}$ sends $(\overline{m}_i, \overline{m}_j, r_i, r_j)$ to the binding security game of C. Note that due to the check in Hybrid 1, the $(\overline{m}_i)_{i\in[Q_S+1]}$ are pairwise distinct, in particular $\overline{m}_i \neq \overline{m}_j$ but $\mathsf{C.Commit}(\mathsf{pp}, \overline{m}_i; r_i) = \mathsf{C.Commit}(\mathsf{pp}, \overline{m}_j; r_j)$. However, due to the binding property of C, this can happen with only negligible probability. Thus, the extracted commitments $(c_i)_{i\in[Q_S+1]}$ must be distinct with overwhelming probability.

In such a case, there must be at least one $i^* \in [Q_S+1]$ such that c_i^* was never queried to the signing oracle $\mathsf{S.Sign}(\mathsf{sk},\cdot)$ *in the first execution* of $\mathcal{B}$ or equivalently of $\mathcal{A}$. This is because due to the one-more unforgeability game, $\mathcal{A}$ only queries the signing oracle Q_S times. Thus, $\mathcal{A}_{\mathsf{C},\mathsf{S}}$ finds such i^* with the smallest index and outputs (μ_{i^*}, c_i^*) as a forgery against the `euf-cma` security of S.

It remains to show that what $\mathcal{A}_{\mathsf{C},\mathsf{S}}$ output is a valid forgery, i.e., $\mathcal{B}$ never queried c_i^* to the signing oracle in any of the *rewound executions*. To argue this, we first show that all the extracted

commitments $(c_i)_{i\in[Q_S+1]}$ are fixed *after the first execution ends* due to f-unique extraction (cf. definition 17). For any $(x_i, \tau_i := (\alpha_i, \beta_i, \gamma_i)) \in \Lambda$ defined in the first execution of $\mathcal{B}$, conditioning on $\mathsf{F}_{\mathcal{B}}$ succeeding, another valid transcript $(x_i, \tau_i' := (\alpha_i, \beta_i', \gamma_i')) \in \Lambda_i$ with $\beta_i \neq \beta_i'$ is guaranteed to exist with overwhelming probability. Due to f-unique extraction, for any such valid transcript the value $f(\mathsf{Ext}(x_i, \tau_i, \tau_i')) = c_i$ is identical, where recall f simply outputs the commitment included in the witness. Put differently, conditioning on $\mathsf{F}_{\mathcal{B}}$ succeeding, (x_i, τ_i) uniquely defines c_i with overwhelming probability. We emphasize that c_i does not need to be efficiently computable given only (x_i, τ_i); we only care if c_i is determined by (x_i, τ_i) in a statistical sense.

Now, assume $\mathcal{B}$ queried c_i^* to the signing oracle in one of the rewound executions. This means $\mathcal{A}$ outputs some c^* to $\mathcal{B}$ (or equivalently the simulated challenger $\mathcal{G}$ in Hybrid 1) and $\mathcal{B}$ computed $c_i^* = \mathsf{C.RerandCom}(\mathsf{pp}, c^*, \Delta r^*)$, where Δr^* is a fresh randomness sampled by $\mathsf{F}_{\mathcal{B}}$ to be used in the rewound execution. However, this cannot happen with all but negligible probability due to the rerandomizability of C since we have established above that Δr^* is sampled independently from c_i^*. Since there are at most $T \cdot (Q_S + 1)$ rewound executions, the probability that $\mathcal{B}$ queries c_i^* to the signing oracle during in one of the rewound execution is bounded by $T \cdot (Q_S + 1) \cdot \mathrm{negl}(\lambda) = \mathrm{negl}(\lambda)$, where we use $T = \mathsf{poly}(\lambda)$ due to lemma 2.

Thus, with overwhelming probability, what $\mathcal{A}_{\mathsf{C},\mathsf{S}}$ output is a valid forgery against the `euf-cma` security of S. However, due to the hardness of `euf-cma` security of S, this cannot happen with all but negligible probability. Combining all the arguments, we conclude that $\epsilon = \mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda)$ is negligible. This completes the proof.

Remark 1 (Removing the Rerandomizability Property). As briefly noted in our technical overview, an alternative approach to using rerandomizable commitment is to let the signer (i.e., $\mathsf{BS}_{\mathsf{Rnd}}.\mathsf{Signer}$) sample a random string rand and run $\mu \leftarrow \mathsf{S.Sign}(\mathsf{sk}, c \| \mathsf{rand})$ instead of $\mu \leftarrow \mathsf{S.Sign}(\mathsf{sk}, c')$, where $c' = \mathsf{C.RerandCom}(\mathsf{pp}, c, \Delta r)$ is the rerandomized commitment. The signer then sends $\rho = (\mu, \mathsf{rand})$ as the second message instead of $\rho = (\mu, \Delta r)$. By observing that rand has an identical effect as Δr in the security proof, it can be checked that the same proof can be used to show blindness and one-more unforgeability of this modified protocol. While this approach works for any commitment scheme, we chose not to since it requires a slightly larger NIZK proof due to the enlarged signing space of the underlying signature scheme S.

4 Instantiation of the Generic Construction

In this section, we instantiate the $\mathsf{BS}_{\mathsf{Rnd}}$-suitable schemes $(\mathsf{C}, \mathsf{S}, \Sigma)$ required for the construction of $\mathsf{BS}_{\mathsf{Rnd}}$ in section 3.

4.1 Commitments

We instantiate the commitment scheme C with Pedersen commitments. We also use a variant of ElGamal commitments in our instantiation of Σ.

Rerandomizable Commitment Scheme. We recall Pedersen commitments $\mathsf{C}_{\mathrm{Ped}}$ [71] with public parameters $\mathsf{pp} \in \mathbb{G}$ over a group $\mathbb{G}$ with generator g:

- $\mathsf{C}_{\mathrm{Ped}}.\mathsf{Commit}(\mathsf{pp}, m; r)$: outputs commitment $c = g^m \cdot \mathsf{pp}^r$ and opening $r \leftarrow \mathbb{Z}_p$,

Pedersen commitments are correct, computationally binding under the DLOG assumption and statistically hiding. Further, note that it is rerandomizable via:

- $\mathsf{C}_{\mathrm{Ped}}.\mathsf{RerandCom}(\mathsf{pp}, c, \Delta r)$: outputs $c' \leftarrow c \cdot \mathsf{pp}^{\Delta r}$
- $\mathsf{C}_{\mathrm{Ped}}.\mathsf{RerandRand}(\mathsf{pp}, c, m, r, \Delta r)$: outputs $r' \leftarrow \Delta r + r$

ElGamal. We recall ElGamal commitments C_{EG} [35] with public parameters $\mathsf{pp} \in \mathbb{G}$ over a group $\mathbb{G}$ with generator g:

- $\mathsf{C}_{\mathrm{EG}}.\mathsf{Commit}(\mathsf{pp}, m; r)$: outputs commitment $c = (c_1, c_2) = (m \cdot \mathsf{pp}^r, g^r)$ and opening $r \leftarrow \mathbb{Z}_p$,

ElGamal commitments are correct, perfectly binding and computationally hiding under DDH. Also, it is extractable with the following simulator and extractor:

- $\mathsf{C}_{\mathrm{EG}}.\mathsf{SimSetup}(1^\lambda)$: samples $x \leftarrow \mathbb{Z}_p$, and outputs public parameters $\mathsf{pp} = g^x$ and trapdoor $\mathsf{td} = x$.
- $\mathsf{C}_{\mathrm{EG}}.\mathsf{Ext}(\mathsf{pp}, \mathsf{td}, c)$: extracts $m \leftarrow c_1/c_2^{\mathsf{td}}$.

Note that the second part $c_2 = g^r$ of the commitment can be reused across multiple commitments, if each commitment uses different public parameters, i.e. $\mathsf{pp}_i \leftarrow \mathbb{G}$, as below.

Remark 2 (ElGamal with Message Space $\mathbb{G}^n$). Let $n \in \mathbb{N}$ and $\mathsf{pp}_i \leftarrow \mathbb{G}$ for $i \in [n]$. We can commit to messages $(m_1, \ldots, m_n) \in \mathbb{G}^n$ via $R \leftarrow g^r$ and $c_i = m_i \mathsf{pp}_i^r$. The (non-compact) vector commitment $(R, c_1, \ldots, c_2)$ remains correct, perfectly binding with message space $\mathbb{G}^n$ and hiding under DDH [12]. Extraction is possible as before with the trapdoor $\mathsf{td} = (x_i)_{i \in [n]}$ with $g^{x_i} = \mathsf{pp}_i$ via $m_i \leftarrow c_i / R^{x_i}$.

Remark 3 (ElGamal with Message Space $\mathbb{Z}_p$). We can commit to a message $m \in \mathbb{Z}_p$ via $c = (g^m \mathsf{pp}_i^r, g^r)$. This variant remains perfectly binding with message space $\mathbb{Z}_p$ and hiding under DDH. Note that this variant is not extractable for message space $\mathbb{Z}_p$. If the message m is of polynomial size, i.e. $|m| \leq B$ for some bound $B = \mathsf{poly}(\lambda)$, extraction is possible via $g \leftarrow c_1/c_2^{\mathsf{td}}$ and then calculating the discrete logarithm m of g. This can be done in polynomial time by trying all $m' \in \mathbb{Z}_p$ with $|m'| \leq B$ with brute-force.

4.2 Signature Scheme

For the signature scheme S, we use a variant of the Kiltz-Pan-Wee (KPW) structure-preserving signature (SPS) scheme [62] in the asymmetric pairing setting. The message space of KPW is $\mathbb{G}_1^\ell$, where $\ell \in \mathbb{N}$ is the message length.

Any SPS must contain at least three group elements, and at least one in each $\mathbb{G}_2$ and in $\mathbb{G}_1$ [3]. But as the bit size of elements in $\mathbb{G}_2$ is larger than the bit size of elements in $\mathbb{G}_1$ and $\mathbb{Z}_p$, removing elements in $\mathbb{G}_2$ in the signature is desirable. For $\mathsf{BS}_{\mathsf{Rnd}}$, we do not require the full structure-preserving property of KPW, as we can design efficient Σ-protocols for signature verification, even if the signature contains elements in $\mathbb{Z}_p$.

Indeed, KPW signatures contain an element σ_4 in $\mathbb{G}_2$. We observe that we can safely replace σ_4 with its discrete logarithm τ. Further, we can omit two more elements in $\mathbb{G}_1$ for free, as they can be recomputed via τ and the remaining signature elements.

Our optimized variant is given below.

- $\mathsf{S}_{\mathrm{KPW}}.\mathsf{KeyGen}(1^\lambda)$: samples $a, b \leftarrow \mathbb{Z}_p$ and sets $\boldsymbol{A} \leftarrow (1, a)^\top$ and $\boldsymbol{B} \leftarrow (1, b)^\top$. It samples $\boldsymbol{K} \leftarrow \mathbb{Z}_p^{(\ell+1)\times 2}$, $\boldsymbol{K}_0$, $\boldsymbol{K}_1 \leftarrow \mathbb{Z}_p^{2\times 2}$ and sets $\boldsymbol{C} \leftarrow \boldsymbol{K}\boldsymbol{A}$. It sets $(\boldsymbol{C}_0, \boldsymbol{C}_1) \leftarrow (\boldsymbol{K}_0\boldsymbol{A}, \boldsymbol{K}_1\boldsymbol{A})$, $(\boldsymbol{P}_0, \boldsymbol{P}_1) \leftarrow (\boldsymbol{B}^\top \boldsymbol{K}_0, \boldsymbol{B}^\top \boldsymbol{K}_1)$, $\mathsf{vk} \leftarrow ([\boldsymbol{C}_0]_2, [\boldsymbol{C}_1]_2, [\boldsymbol{C}]_2, [\boldsymbol{A}]_2)$, and $\mathsf{sk} \leftarrow (\boldsymbol{K}, [\boldsymbol{P}_0]_1, [\boldsymbol{P}_1]_1, [\boldsymbol{B}]_1)$. It outputs $(\mathsf{vk}, \mathsf{sk})$.
- $\mathsf{S}_{\mathrm{KPW}}.\mathsf{Sign}(\mathsf{sk}, [\boldsymbol{m}]_1)$: samples $r, \tau \leftarrow \mathbb{Z}_p$ and sets $\sigma_1 \leftarrow [(1, \boldsymbol{m}^\top)\boldsymbol{K} + r(\boldsymbol{P}_0 + \tau \boldsymbol{P}_1)]_1 \in \mathbb{G}_1^2$, $\sigma_2 \leftarrow [r\boldsymbol{B}^\top]_1 \in \mathbb{G}_1^2$, and $\sigma_3 \leftarrow \tau \in \mathbb{Z}_p$. It outputs $(\sigma_1, \sigma_2, \sigma_3)$.
- $\mathsf{S}_{\mathrm{KPW}}.\mathsf{Verify}(\mathsf{vk}, [\boldsymbol{m}]_1, (\sigma_1, \sigma_2, \sigma_3))$: checks $e(\sigma_1, [\boldsymbol{A}]_2) = e([(1, \boldsymbol{m}^\top)]_1, [\boldsymbol{C}]_2) \cdot e(\sigma_2, [\boldsymbol{C}_0]_2 \cdot \tau[\boldsymbol{C}_1]_2)$.

We show that $\mathsf{S}_{\mathrm{KPW}}$ is `euf-cma` under the SXDH assumption in Theorem 4. The proof relies on the computational core lemma of [63]. $\mathsf{S}_{\mathrm{KPW}}$ can be made deterministic via a pseudorandom function.

Theorem 4. *The scheme $\mathsf{S}_{\mathrm{KPW}}$ is correct and `euf-cma` secure under the SXDH assumption.*

Proof. Correctness follows from a simple calculation. For `euf-cma` security, we first recall the computational core lemma of [63].

Lemma 3. *For all adversaries $\mathcal{A}$, there exists an adversary $\mathcal{B}$ with almost the same running time satisfying*

$$\Pr\left[b = b' : \begin{array}{l} a,\, b \leftarrow \mathbb{Z}_p,\ \boldsymbol{A} \leftarrow (1,a)^\top,\ \boldsymbol{B} \leftarrow (1,b)^\top \\ \boldsymbol{K}_0,\ \boldsymbol{K}_1 \leftarrow \mathbb{Z}_p^{2\times 2} \\ (\boldsymbol{P}_0, \boldsymbol{P}_1) \leftarrow (\boldsymbol{B}^\top \boldsymbol{K}_0, \boldsymbol{B}^\top \boldsymbol{K}_1) \\ \mathsf{pk} \leftarrow ([\boldsymbol{P}_0]_1, [\boldsymbol{P}_1]_1, [\boldsymbol{B}]_1, \boldsymbol{K}_0\boldsymbol{A}, \boldsymbol{K}_1\boldsymbol{A}, \boldsymbol{A}) \\ b \leftarrow \{0,1\},\ b' \leftarrow \mathcal{A}^{\mathcal{O}_b, \mathcal{O}^*}(\mathsf{pk}) \end{array}\right] \leq \frac{1}{2} + 2Q\mathsf{Adv}^{\mathrm{sxdh}}_{\mathcal{B}}(\lambda) + Q/p$$

where Q is the number of queries $\mathcal{A}$ makes to $\mathcal{O}_b$, $\mathcal{A}$ is not allowed to make the same query to both $\mathcal{O}_b$ and $\mathcal{O}^*$, and

- *$\mathcal{O}_b(\tau)$: on a query $\tau \in \mathbb{Z}_p$ returns $([b\mu\boldsymbol{a}^\perp + r(\boldsymbol{P}_0 + \tau\boldsymbol{P}_1)]_1, [r\boldsymbol{B}^\top]_1) \in (\mathbb{G}_1^2)^2$ with $\mu \leftarrow \mathbb{Z}_p$, $r \leftarrow \mathbb{Z}_p$, where $\boldsymbol{a}^\perp \in \mathbb{Z}_p^2$ is a non-zero vector that satisfies $\boldsymbol{a}^\perp\boldsymbol{A} = \boldsymbol{0}$,*
- *$\mathcal{O}^*(\tau^*)$: on a query $\tau^* \in \mathbb{Z}_p$ returns $[\boldsymbol{K}_0 + \tau^*\boldsymbol{K}_1]_2$ with only a single query to $\mathcal{O}^*$ allowed for $\mathcal{A}$,*

Now let $\mathcal{A}$ be an adversary against the `euf-cma` security of the scheme $\mathsf{S}_{\mathrm{KPW}}$. After Q signing queries, $\mathcal{A}$ outputs a forgery $([\boldsymbol{m}^*]_1, \sigma^*)$. We follow the proof structure of [62], but adapt it to our optimizations using lemma 3. We define the following hybrids and denote by $\mathsf{Adv}^{\mathrm{H}_i}_{\mathcal{A}}(\lambda)$ the advantage of $\mathcal{A}$ in Hybrid i.

- Hybrid 0 is the real game.
- Hybrid 1 is the same as Hybrid 0, except the verification check (to verify the final output of $\mathcal{A}$), is replaced with $\mathsf{S}_{\mathrm{KPW}}.\mathsf{Verify}^*$, defined as follows. On input of verification key vk, message $[\boldsymbol{m}]_1$, and signature $\sigma = (\sigma_1, \sigma_2, \sigma_3)$, $\mathsf{S}_{\mathrm{KPW}}.\mathsf{Verify}^*$ checks $e(\sigma_1, [\boldsymbol{1}]_2) = e([(1, \boldsymbol{m}^\top)]_1, [\boldsymbol{K}]_2) \cdot e(\sigma_2, [\boldsymbol{K}_0]_2 \cdot \sigma_3[\boldsymbol{K}_1]_2)$.
  Note that for any $([\boldsymbol{m}]_1, \sigma)$, we have
  $$\begin{aligned} & e(\sigma_1, [\boldsymbol{A}]_2) = e([(1, \boldsymbol{m}^\top)]_1, [\boldsymbol{C}]_2) \cdot e(\sigma_2, [\boldsymbol{C}_0]_2 \cdot \tau[\boldsymbol{C}_1]_2) \\ \iff & e(\sigma_1, [\boldsymbol{A}]_2) = e([(1, \boldsymbol{m}^\top)]_1, [\boldsymbol{K}\boldsymbol{A}]_2) \cdot e(\sigma_2, [\boldsymbol{K}_0\boldsymbol{A}]_2 \cdot \tau[\boldsymbol{K}_1\boldsymbol{A}]_2) \\ \impliedby & e(\sigma_1, [1]_2) = e([(1, \boldsymbol{m}^\top)]_1, [\boldsymbol{K}]_2) \cdot e(\sigma_2, [\boldsymbol{K}_0 + \tau\boldsymbol{K}_1]_2) \end{aligned}$$
  Thus, if $([\boldsymbol{m}]_1, \sigma)$ passes $\mathsf{S}_{\mathrm{KPW}}.\mathsf{Verify}$ but not $\mathsf{S}_{\mathrm{KPW}}.\mathsf{Verify}^*$, we have that $\sigma_1 - ([(1, m^\top)\boldsymbol{K}]_1 + \sigma_2(\boldsymbol{K}_0 + \tau\boldsymbol{K}_1)) \in \mathbb{G}_1^2$ is a non-zero vector in the kernel of $\boldsymbol{A}$. Finding such a vector is hard under the SXDH assumption, and we can construct an adversary $\mathcal{B}_{\mathrm{dh}}$ such that $|\mathsf{Adv}^{\mathrm{H}_0}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda)| \leq \mathsf{Adv}^{\mathrm{sxdh}}_{\mathcal{B}_{\mathrm{dh}}}(\lambda)$.
- Hybrid 2 is the same as Hybrid 1, except if the chosen tags $\tau_1, \ldots, \tau_Q$ during the signing queries are not all distinct.
  A simple union bound implies that $|\mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda)| \leq Q^2/p$.
- Hybrid 3 is the same as Hybrid 2, except it samples $i^* \leftarrow [Q+1]$ and aborts if i^* is not the smallest index such that $\tau_i = \sigma_3^*$, where $\tau_{Q+1} = \sigma_3^*$ is the tag from the forgery.
  As there are Q queries, we have $\mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda) \geq 1/(Q+1)\mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda)$.
- Hybrid 4 is the same as Hybrid 3, except the signature queries are answered as follows. On input of $[\boldsymbol{m}]_1$ in the i-th signing query, samples $r, \tau, \mu \leftarrow \mathbb{Z}_p$, and set $\mu \leftarrow 0$ if $\tau = \tau^*$. Then, set $\sigma_1 \leftarrow [(1, \boldsymbol{m}^\top)\boldsymbol{K} + \mu\boldsymbol{a}^\perp + r(\boldsymbol{P}_0 + \tau\boldsymbol{P}_1)]_1$, $\sigma_2 \leftarrow [r\boldsymbol{B}^\top]_1$, and $\sigma_3 \leftarrow \tau \in \mathbb{Z}_p$, where $\boldsymbol{a}^\perp$ is a non-zero vector in the kernel of $\boldsymbol{A}$ such that $\boldsymbol{a}^\perp\boldsymbol{A} = \boldsymbol{0}$. Finally, outputs $(\sigma_1, \sigma_2, \sigma_3)$.
  Hybrid 3 and Hybrid 4 are indistinguishable which we can show by constructing an adversary $\mathcal{B}_4$ against lemma 3. $\mathcal{B}_4$ first samples $\boldsymbol{K}$ and receives $([\boldsymbol{P}_0]_1, [\boldsymbol{P}_1]_1, [\boldsymbol{B}]_1, \boldsymbol{K}_0\boldsymbol{A}, \boldsymbol{K}_1\boldsymbol{A}, \boldsymbol{A})$ with which she sets up the verification key vk for $\mathcal{A}$. $\mathcal{B}_4$ answers the i-th signing query for $i \neq i^*$ via $(\sigma_1, \sigma_2, \sigma_3) \leftarrow ([(1, \boldsymbol{m}^\top)\boldsymbol{K}]_1 \cdot \sigma_1', \sigma_2)$, where $(\sigma_1', \sigma_2) \leftarrow \mathcal{O}_b(\tau)$. The i^*-th signing query is answered honestly. A quick calculation shows that for $b = 0$ (resp. $b = 1$) the queries are answered as in Hybrid 3 (resp. 4). Note that $\mathsf{S}_{\mathrm{KPW}}.\mathsf{Verify}^*$ can be simulated via a query to $\mathcal{O}^*(\tau^*)$. Thus, we have $|\mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_4}_{\mathcal{A}}(\lambda)| \leq 2Q\mathsf{Adv}^{\mathrm{sxdh}}_{\mathcal{B}}(\lambda) + Q/p$, where $\mathcal{B}$ is the adversary from lemma 3.
- Hybrid 5 is the same as Hybrid 4, except $\boldsymbol{K}$ is computed as $\boldsymbol{K} \leftarrow \boldsymbol{K'} + \boldsymbol{u}\boldsymbol{a}^\perp$ for $\boldsymbol{K'} \leftarrow \mathbb{Z}_p^{(\ell+1)\times 2}, \boldsymbol{u} \leftarrow \mathbb{Z}_q^{\ell+1}$. Clearly, Hybrid 5 and Hybrid 4 are identically distributed and we have $\mathsf{Adv}^{\mathrm{H}_4}_{\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathrm{H}_5}_{\mathcal{A}}(\lambda)$.

Finally, observe that as in [62], the verification key vk and signing queries for $\tau_i \neq \tau^*$ leak no information about $\boldsymbol{u}$, as $\boldsymbol{C} = (\boldsymbol{K'} + \boldsymbol{u}\boldsymbol{a}^\perp)\boldsymbol{A} = \boldsymbol{K'}\boldsymbol{A}$, and $(1, \boldsymbol{m}^\top)(\boldsymbol{K'} + \boldsymbol{u}\boldsymbol{a}^\perp) + \mu\boldsymbol{a}^\perp$ is identically distributed to $(1, \boldsymbol{m}^\top)\boldsymbol{K'} + \mu'\boldsymbol{a}^\perp$. The i^*-th signing query leaks $(1, \boldsymbol{m}^\top)(\boldsymbol{K'} + \boldsymbol{u}\boldsymbol{a}^\perp)$, captured by $(1, \boldsymbol{m}^\top)\boldsymbol{u}$. To provide a valid forgery, $\mathcal{A}$ needs to compute

$$\boldsymbol{v} \coloneqq (1, \boldsymbol{m^*}^\top)(\boldsymbol{K'} + \boldsymbol{u}\boldsymbol{a}^\perp).$$

Given $(1, \boldsymbol{m}^\top)\boldsymbol{u}$, for any adaptively chosen $\boldsymbol{m^*} \neq \boldsymbol{m}$, we have that $(1, \boldsymbol{m}^{*\top})\boldsymbol{u}$ is uniformly random over $\mathbb{Z}_p$ in the view of $\mathcal{A}$. Thus, $\boldsymbol{v}$ is also uniformly random over $\mathbb{Z}_p$ and we have $\mathsf{Adv}^{\mathrm{H}_5}_{\mathcal{A}}(\lambda) \leq 1/p$.

4.3 Σ-protocol

Now, we instantiate the Σ-protocol Σ for relation $\mathsf{R}_{\mathsf{rnd}}$, where C is instantiated with $\mathsf{C}_{\mathrm{Ped}}$ over $\mathbb{G}_1$ and S instantiated with $\mathsf{S}_{\mathrm{KPW}}$. In this context, the relation $\mathsf{R}_{\mathsf{rnd}}$ can be written as

$$\begin{aligned}\mathsf{R}_{\mathsf{rnd}} = \{(x, w) : c = g_1^{\overline{m}}\mathsf{pp}^r \wedge \\ e(\boldsymbol{\mu_1}, [\boldsymbol{A}]_2) = e((g_1, c), [\boldsymbol{C}]_2) \cdot e(\boldsymbol{\mu_2}, [\boldsymbol{C}_0]_2 \cdot \mu_3[\boldsymbol{C}_1]_2)\},\end{aligned}$$

where $x = (\mathsf{pp}, \mathsf{vk}, \overline{m}), w = (\mu, c, r)$ for $\mathsf{vk} = ([\boldsymbol{C}_0]_2, [\boldsymbol{C}_1]_2, [\boldsymbol{C}]_2, [\boldsymbol{A}]_2)$ and $\mu = (\boldsymbol{\mu_1}, \boldsymbol{\mu_2}, \mu_3)$.

We now provide the Σ-protocol, denoted Σ_{rnd}. For readability, we further assume that random generators $(\mathsf{pp}_1, \ldots, \mathsf{pp}_5)$ are setup without trapdoors via a random oracle for simplicity. (In the construction, we let these values be output by the hash functions H_{pp} in addition to pp.) On a high level, the prover commits to the witness in ElGamal commitments under different public parameters pp_i and shared randomness $S = g_1^s$, and then shows that the committed values satisfy the required relations. Note that the verification equation of μ is quadratic, as both $\boldsymbol{\mu_2}$ and μ_3 are witnesses. For this part, we introduce a new witness $\omega = s \cdot \mu_3$.

- $\Sigma_{\mathsf{rnd}}.\mathsf{Init}(\mathsf{crs}, x, w)$: for (crs, x, w) defined as above, denote $e_1 = c, e_2 = \mu_{1,1}, e_3 = \mu_{1,2}, e_4 = \mu_{2,1}, e_5 = \mu_{2,2}$ and $\tau = \mu_3, \omega = s \cdot \tau$. First, sets $S = g_1^s$ for $s \leftarrow \mathbb{Z}_p$ and commits to e_i with shared randomness via $E_i = e_i\mathsf{pp}_i^s$. Samples additive masks $\widetilde{s}, \widetilde{r}, \widetilde{\omega}, \widetilde{\tau} \leftarrow \mathbb{Z}_p$ and sets

$$\begin{aligned}&D_{\overline{m}} = \mathsf{pp}_1^{-\widetilde{s}} \cdot \mathsf{pp}^{-\widetilde{r}}, && D_s = g_1^{-\widetilde{s}}, && D_\omega = S^{\widetilde{\tau}} g_1^{-\widetilde{\omega}} \\ &\boldsymbol{D_\mu} = e((\mathsf{pp}_2^{-\widetilde{s}}, \mathsf{pp}_3^{-\widetilde{s}}), [\boldsymbol{A}]_2)^{-1} \cdot e((1_{\mathbb{G}_1}, \mathsf{pp}_1^{-\widetilde{s}}), [\boldsymbol{C}]_2) \\ &\qquad \cdot e((\mathsf{pp}_4^{-\widetilde{s}}, \mathsf{pp}_5^{-\widetilde{s}}), [\boldsymbol{C_0}]_2) \cdot e((E_4^{\widetilde{\tau}}\mathsf{pp}_4^{-\widetilde{\omega}}, E_5^{\widetilde{\tau}}\mathsf{pp}_5^{-\widetilde{\omega}}), [\boldsymbol{C_1}]_2).\end{aligned}$$

  Outputs $\alpha = (S, E_1, \ldots, E_5, D_{\overline{m}}, D_s, D_\omega, \boldsymbol{D_\mu})$ and stores $(\widetilde{s}, \widetilde{r}, \widetilde{\omega}, \widetilde{\tau})$ in st,
- $\Sigma_{\mathsf{ext}}.\mathsf{Chall}()$: samples a challenge $\beta \leftarrow \mathbb{Z}_p$,
- $\Sigma_{\mathsf{ext}}.\mathsf{Resp}(\mathsf{st}, \beta)$: sets $\gamma_r = \beta \cdot r + \widetilde{r}, \gamma_s = \beta \cdot s + \widetilde{s}, \gamma_\tau = \beta \cdot \tau + \widetilde{\tau}, \gamma_\omega = \beta \cdot \omega + \widetilde{\omega}$, and outputs the response $\gamma = (\gamma_r, \gamma_s, \gamma_\tau, \gamma_\omega)$,
- $\Sigma_{\mathsf{ext}}.\mathsf{Verify}(\mathsf{crs}, x, \alpha, \beta, \gamma)$: checks the following equations

$$\begin{aligned}&D_{\overline{m}} = E_1^\beta \cdot \mathsf{pp}_1^{-\gamma_s} g_1^{-\beta \cdot \overline{m}} \mathsf{pp}^{-\gamma_r}, && D_s = S^\beta g_1^{-\gamma_s}, && D_\omega = S^{\gamma_\tau} \cdot g_1^{-\gamma_\omega}, \\ &\boldsymbol{D_\mu} = e(\boldsymbol{F_1}, [\boldsymbol{A}]_2)^{-1} \cdot e(\boldsymbol{F_m}, [\boldsymbol{C}]_2) \cdot e(\boldsymbol{F_2}, [\boldsymbol{C_0}]_2) \cdot e(\boldsymbol{F_3}, [\boldsymbol{C_1}]_2),\end{aligned}$$

  where $\boldsymbol{F_1} = (E_2^\beta \cdot \mathsf{pp}_2^{-\gamma_s}, E_3^\beta \cdot \mathsf{pp}_3^{-\gamma_s}), \boldsymbol{F_m} = (g_1^\beta, E_1^\beta \cdot \mathsf{pp}_1^{-\gamma_s}), \boldsymbol{F_2} = (E_4^\beta \cdot \mathsf{pp}_4^{-\gamma_s}, E_5^\beta \cdot \mathsf{pp}_5^{-\gamma_s}), \boldsymbol{F_3} = (E_4^{-\gamma_\tau} \cdot \mathsf{pp}_4^{-\gamma_\omega}, E_5^{-\gamma_\tau} \cdot \mathsf{pp}_5^{-\gamma_\omega})$, and outputs 1 iff all checks succeed. Note that the first equation checks that the commitment c committed in E_1 is a valid $\mathsf{C}_{\mathrm{Ped}}$ commitments to $\overline{m}$, the second equation fixes s (and thus the values committed in E_i) and the third equation fixes $\omega = s \cdot \tau$ which allows to open the commitments E_4^τ and E_5^τ in zero-knowledge. Finally the last equation checks whether the committed signature μ is valid. For this, we rewrite $e(\boldsymbol{\mu_2}, [\boldsymbol{C}_0]_2 \cdot \mu_3[\boldsymbol{C}_1]_2) = e(\boldsymbol{\mu_2}, [\boldsymbol{C}_0]_2) \cdot e(\boldsymbol{\mu_2}^{\mu_3}, [\boldsymbol{C}_1]_2)$, and use that (E_5^τ, E_6^τ) commits to $\boldsymbol{\mu_2}^{\mu_3}$.

Theorem 5 (Correctness). *The scheme Σ_{rnd} is correct.*

Proof. Note that we have $E_i^\beta \mathsf{pp}_i^{-\gamma_s} = e_i^\beta \cdot \mathsf{pp}_i^{-\widetilde{s}}$ and $E_i^{-\gamma_\tau}\mathsf{pp}_i^{-\gamma_\omega} = e_i^{\beta \cdot \tau} \cdot E_i^{-\widetilde{\tau}} \cdot \mathsf{pp}_i^{-\widetilde{\omega}}$. With these identities, the identities in $\Sigma_{\mathsf{rnd}}.\mathsf{Verify}$ follow from a straightforward calculation, and we omit details.

Theorem 6 (HVZK). *The scheme* Σ_{rnd} *is HVZK under the* DDH *assumption.*

Proof. Let $x = (\mathsf{pp}, \mathsf{vk}, \overline{m})$ and $\beta \leftarrow \mathbb{Z}_p$. We define the simulator Sim as follows. On input of (x, β), samples $s \leftarrow \mathbb{Z}_p$, and sets $S = g_1^s, E_i \leftarrow \mathsf{pp}_i^s$ for $i \in [5]$. Then, samples $\gamma = (\gamma_r, \gamma_s, \gamma_\tau, \gamma_\omega) \leftarrow \mathbb{Z}_p^4$ and computes $D_{\overline{m}}, D_s, D_\omega, \boldsymbol{D_\mu}$ via the identities in $\Sigma_{\mathsf{ext}}.\mathsf{Verify}$. Finally, sets $\alpha = (S, E_1, \ldots, E_5, D_{\overline{m}}, D_s, D_\omega, \boldsymbol{D_\mu})$ and outputs the transcript (α, β, γ).

To show that Sim outputs transcripts that are indistinguishable from real transcripts, we define the following hybrids and denote by $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_i}(\lambda)$ the advantage of $\mathcal{A}$ in Hybrid i.

- Hybrid 0 outputs honestly generated transcripts.
- Hybrid 1 is the same as Hybrid 0, except the elements $(D_r, D_s, D_\omega, \boldsymbol{D_\mu})$ are generated as in Sim. A quick calculation shows that Hybrid 0 and Hybrid 1 are identically distributed, and we have $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_0}(\lambda) = \mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_1}(\lambda)$.
- Hybrid 2 is the same as Hybrid 1, except γ is computed as in Sim. Again, it is easy to check that $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_1}(\lambda) = \mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_2}(\lambda)$.
- Hybrid 3 is the same as Hybrid 2, except the commitments $(S, E_1, \ldots, E_5)$ are commitments to $1_{\mathbb{G}_1}$ instead. If $\mathcal{A}$ can distinguish between Hybrid 2 and Hybrid 3, we can construct an adversary $\mathcal{B}$ against the DDH assumption, as $(S, E_1, \ldots, E_5)$ is hiding under DDH (see remark 2). Here, we use the fact that the public parameters $(\mathsf{pp}_1, \ldots, \mathsf{pp}_5)$ are output by a random oracle which is programmed with the challenge pp_i appropriately.

As Hybrid 3 outputs simulated transcripts, the statement follows.

Theorem 7 (Special Soundness). *The scheme* Σ_{ext} *is 2-special sound.*

Proof. Let $x = (\mathsf{pp}, \mathsf{vk}, \overline{m})$. We define the extractor Ext as follows. First, Ext receives as input two valid transcripts (α, β, γ) and $(\alpha, \beta', \gamma')$ with $\beta \neq \beta'$. Next, Ext parses $\alpha = (S, E_1, \ldots, E_5, D_{\overline{m}}, D_s, D_\omega, \boldsymbol{D_\mu})$ and $\gamma = (\gamma_r, \gamma_s, \gamma_\tau, \gamma_\omega), \gamma' = (\gamma'_r, \gamma'_s, \gamma'_\tau, \gamma'_\omega)$. From the identities in $\Sigma_{\mathsf{ext}}.\mathsf{Verify}$, we obtain the identities

$$
\begin{aligned}
S^{\beta'} \cdot g_1^{-\gamma'_s} &= S^{\beta} \cdot g_1^{-\gamma_s}, \\
E_1^{\beta'} \cdot \mathsf{pp}_1^{-\gamma'_s} g_1^{-\beta' \cdot \overline{m}} \mathsf{pp}^{-\gamma'_r} &= E_1^{\beta} \cdot \mathsf{pp}_1^{-\gamma_s} g_1^{-\beta \cdot \overline{m}} \mathsf{pp}^{-\gamma_r} \\
S^{\gamma'_\tau} \cdot g_1^{-\gamma'_\omega} &= S^{\gamma_\tau} \cdot g_1^{-\gamma_\omega}, \\
e(\boldsymbol{F_1'}, [\boldsymbol{A}]_2)^{-1} \cdot e(\boldsymbol{F_m'}, [\boldsymbol{C}]_2) &= e(\boldsymbol{F_1}, [\boldsymbol{A}]_2)^{-1} \cdot e(\boldsymbol{F_m}, [\boldsymbol{C}]_2) \\
\cdot e(\boldsymbol{F_2'}, [\boldsymbol{C_0}]_2) \cdot e(\boldsymbol{F_3'}, [\boldsymbol{C_1}]_2) \quad &\cdot e(\boldsymbol{F_2}, [\boldsymbol{C_0}]_2) \cdot e(\boldsymbol{F_3}, [\boldsymbol{C_1}]_2),
\end{aligned}
$$

where $\boldsymbol{F_1} = (E_2^\beta \cdot \mathsf{pp}_2^{-\gamma_s}, E_3^\beta \cdot \mathsf{pp}_3^{-\gamma_s}), \boldsymbol{F_m} = (g_1^\beta, E_1^\beta \cdot \mathsf{pp}_1^{-\gamma_s}), \boldsymbol{F_2} = (E_4^\beta \cdot \mathsf{pp}_4^{-\gamma_s}, E_5^\beta \cdot \mathsf{pp}_5^{-\gamma_s}), \boldsymbol{F_3} = (E_4^{-\gamma_\tau} \cdot \mathsf{pp}_4^{-\gamma_\omega}, E_5^{-\gamma_\tau} \cdot \mathsf{pp}_5^{-\gamma_\omega})$ and similarly for $\boldsymbol{F_1'}$, $\boldsymbol{F_m'}$, $\boldsymbol{F_2'}$ and $\boldsymbol{F_3'}$. We denote $\Delta x = (\gamma_x - \gamma'_x)$ for $x \in \{r, s, \tau, \omega\}$ and $\Delta\beta = (\beta - \beta')$. Note that $\Delta\beta \neq 0$. We further denote $s = \Delta\gamma_s/\Delta\beta, r = \Delta\gamma_r/\Delta\beta, \tau = \Delta\gamma_\tau/\Delta\beta$ and $\omega = \Delta\gamma_\omega/\Delta\beta$.

From the first equation, we obtain $S^{\Delta\beta} \cdot g_1^{-\Delta\gamma_s} = 1_{\mathbb{G}_1}$. Taking both sides to the power of $1/\Delta\beta$ yields $S = g_1^{\Delta\gamma_s/\Delta\beta} = g_1^s$. Similarly, we obtain from the second equation that $E_1^{\Delta\beta} \cdot \mathsf{pp}_1^{-\Delta\gamma_s} g_1^{-\Delta\beta\overline{m}} \cdot \mathsf{pp}^{-\Delta\gamma_r} = 1_{\mathbb{G}_1}$, and thus $E_1 = g_1^{\overline{m}} \cdot \mathsf{pp}^r \cdot \mathsf{pp}_1^s$. Consequently, we have $c = g_1^{\overline{m}} \cdot \mathsf{pp}^r$, for $c = E_1 \cdot \mathsf{pp}_1^{-s}$ As above, the third equation yields $S^\tau = g_1^\omega$, so $\omega = s \cdot \tau$.

We can now recompute the value e_i committed in E_i via $e_i \leftarrow E_i \mathsf{pp}_i^{-s}$. Note that that $\boldsymbol{F_1}(\boldsymbol{F_1'})^{-1} = (E_2^{\Delta\beta} \cdot \mathsf{pp}_2^{-\Delta\gamma_s}, E_3^{\Delta\beta} \cdot \mathsf{pp}_3^{-\Delta\gamma_s})$ and thus $(\boldsymbol{F_1}(\boldsymbol{F_1'})^{-1})^{1/\Delta\beta} = (E_2 \cdot \mathsf{pp}_3^{-s}, E_3 \cdot \mathsf{pp}_4^{-s}) = (e_3, e_4)$. Similarly, we have $(\boldsymbol{F_m}(\boldsymbol{F_m'})^{-1})^{1/\Delta\beta} = (g_1, e_1)$, $(\boldsymbol{F_2}(\boldsymbol{F_2'})^{-1})^{1/\Delta\beta} = (e_4, e_5)$ and $(\boldsymbol{F_3}(\boldsymbol{F_3'})^{-1})^{1/\Delta\beta} = (e_4^\tau, e_5^\tau)$. Finally, the last identity implies:

$$
e((e_2, e_3), [\boldsymbol{A}]_2) = e((g_1, e_1), [\boldsymbol{C}]_2) \cdot e((e_4, e_5), [\boldsymbol{C_0}]_2) \cdot e((e_4^\tau, e_5^\tau), [\boldsymbol{C_1}]_2)
$$

As $e((e_4^\tau, e_5^\tau), [\boldsymbol{C_1}]_2) = e((e_4, e_5), \tau[\boldsymbol{C_1}]_2)$, it follows that $(\boldsymbol{\mu_1} = (e_2, e_3), \boldsymbol{\mu_2} = (e_4, e_5), \mu_3 = \tau)$ is a valid signature on message $e_1 = c$. Also, we know $c = g_1^{\overline{m}} \cdot \mathsf{pp}^r$, as desired.

Theorem 8. *The* Σ*-protocol* Σ_{ext} *has high min-entropy.*

Proof. Observe that, for example, D_s is distributed uniformly random in $\mathbb{G}_1$. It follows that the advantage of any adversary in the min-entropy game is at most $1/p = \mathrm{negl}(\lambda)$.

Theorem 9. *The Σ-protocol Σ_{ext} has f-unique extraction, where $f(\mu, c, r) = c$.*

Proof. As c is committed via an ElGamal commitment, even the first flow α of a transcript (α, β, γ) fixes c perfectly. The statement follows.

4.4 Optimizations and Efficiency

We analyze the efficiency of $\mathsf{BS}_{\mathsf{Rnd}}$ when instantiated with the above schemes.

Optimizations. Note that in the construction, we apply Fiat-Shamir to the Σ_{rnd}. It is well-known that the values $(D_{\overline{m}}, D_s, D_\omega, \boldsymbol{D_\mu})$ can be omitted from the proof, as the identities can be recomputed as in $\Sigma_{\mathsf{rnd}}.\mathsf{Verify}$ and checked via β.

Efficiency. The scheme $\mathsf{BS}_{\mathsf{Rnd}}$ is secure under SXDH. The user sends 1 element in $\mathbb{G}_1$ and 1 element in $\mathbb{Z}_p$, the signer sends 4 elements in $\mathbb{G}_1$ and 1 element in $\mathbb{Z}_p$ and the final signature contains 6 elements in $\mathbb{G}_1$ and 5 elements in $\mathbb{Z}_p$. Consequently, the total communication is 303 Byte and signatures are of size 447 Byte for $\lambda = 128$.

5 Blind Signatures based on Boneh-Boyen Signature

In this section, we provide a blind signature based on randomizable signatures. Compared to the optimized generic construction of the Fischlin blind signature in section 3, the resulting signature size is much smaller since it only consists of one signature of the underlying randomizable signature scheme. The construction also relies on an online-extractable NIZK which can be instantiated efficiently by carefully combining Bulletproofs and another NIZK for an ElGamal commitment (see section 6). In Section 7.2 we show how to adapt the scheme for a partially blind variant, where we modify the Boneh-Boyen signature [20, 22] in order to embed the common message into the verification key.

5.1 Construction

We focus on the asymmetric pairing setting. We note that there is also a natural variant of this scheme in the symmetric setting and we omit details. First, we recall the Boneh-Boyen signatures [20, 22] in the asymmetric setting. While this is implicit in our proof, we note the following is known to be selectively secure in the standard model under the CDH assumption:

- $\mathsf{S}_{\mathsf{BB}}.\mathsf{KeyGen}(1^\lambda)$: samples $\alpha, \beta, \gamma \leftarrow \mathbb{Z}_p$, and sets $u_1 = g_1^\alpha, u_2 = g_2^\alpha, h_1 = g_1^\gamma, h_2 = g_2^\gamma, v = e(g_1, g_2)^{\alpha\beta}$, and outputs $\mathsf{vk} = (u_1, u_2, h_1, h_2, v)$ and $\mathsf{sk} = g_1^{\alpha\beta}$,
- $\mathsf{S}_{\mathsf{BB}}.\mathsf{Sign}(\mathsf{sk}, m)$: samples $r \in \mathbb{Z}_p$ and outputs $(\sigma_1, \sigma_2) = (\mathsf{sk} \cdot (u_1^m h_1)^r, g_1^r) \in \mathbb{G}_1^2$,
- $\mathsf{S}_{\mathsf{BB}}.\mathsf{Verify}(\mathsf{vk}, m, (\sigma_1, \sigma_2))$: outputs 1 if $e(\sigma_1, g_2) = v \cdot e(\sigma_2, u_2^m h_2)$, and otherwise outputs 0.

Overview. We present our framework for blind signatures based on S_{BB}. Let Π be an online-extractable NIZK proof system, with random oracle $\mathsf{H}_{\mathsf{zk}} : \{0,1\}^* \mapsto \{0,1\}^{\ell_{\mathsf{zk}}}$ and common reference string crs of length ℓ_{crs} for the relation

$$\mathsf{R}_{\mathsf{bb}} := \{x = (c, u_1, g_1), w = (\overline{m}, s) \mid c = u_1^{\overline{m}} \cdot g_1^s\}.$$

Let $\mathsf{H}_{\mathsf{M}}, \mathsf{H}_{\mathsf{crs}}$ be a random oracles mapping into $\mathbb{Z}_p, \{0,1\}^{\ell_{\mathsf{crs}}}$ respectively. The framework $\mathsf{BS}_{\mathsf{BB}}[\Pi]$, or $\mathsf{BS}_{\mathsf{BB}}$ for short, broadly works as follows. The verification and signing keys are identical to the ones of S_{BB}, that is $\mathsf{vk} = (u_1, u_2, h_1, h_2, v)$ and $\mathsf{sk} = g_1^{\alpha\beta}$. Additionally, the oracle $\mathsf{H}_{\mathsf{crs}}$ implicitly defines a common random string $\mathsf{crs} = \mathsf{H}_{\mathsf{crs}}(0)$ for Π. In order to obtain a signature on message m, the user first commits to $\overline{m} = \mathsf{H}_{\mathsf{M}}(m)$ in a Pedersen commitment $c \leftarrow u_1^{\overline{m}} \cdot g_1^s \in \mathbb{G}_1$, where $s \leftarrow \mathbb{Z}_p$. Then, computes a proof π via Π showing that c was computed honestly, and sends $\rho_1 = (c, \pi)$ to the signer. The signer then checks the proof and sends $\rho_2 = (\rho_{2,0}, \rho_{2,1}) \leftarrow (\mathsf{sk} \cdot (c \cdot h_1)^r, g_1^r)$ to the user, where $r \leftarrow \mathbb{Z}_p$. Finally, the user checks that the ρ_2 is valid with respect to c and that

$(\rho_{2,0}, \rho_{2,1})$ are consistent, and then derives a re-randomized $\mathsf{S_{BB}}$ signature on $\overline{m} = \mathsf{H_M}(m)$ via $\sigma = (\rho_{2,0}/\rho_{2,1}^s \cdot (u_1^{\overline{m}} h_1)^{r'}, \rho_{2,1} \cdot g_1^{r'})$.

Construction. In more detail, we have the following, where we assume that crs is provided to all of the algorithms for readability.

- $\mathsf{BS_{BB}.KeyGen}(1^\lambda)$: outputs $(\mathsf{bvk}, \mathsf{bsk}) \leftarrow \mathsf{S_{BB}.KeyGen}(1^\lambda)$, where $\mathsf{bvk} = (u_1, u_2, h_1, h_2, v)$ with $u_1 = g_1^\alpha, u_2 = g_2^\alpha, h_1 = g_1^\gamma, h_2 = g_2^\gamma, v = e(g_1, g_2)^{\alpha\beta}$ and $\mathsf{bsk} = g_1^{\alpha\beta}$.
- $\mathsf{BS_{BB}.User}(\mathsf{bvk}, m)$: checks validity of the verification key bvk via $e(u_1, g_2) = e(g_1, u_2)$ and $e(h_1, g_2) = e(g_1, h_2)$, sets $\overline{m} \leftarrow \mathsf{H_M}(m)$ and computes a Pedersen commitment $c = u_1^{\overline{m}} g_1^s \in \mathbb{G}_1$ to $\overline{m}$ and a proof $\pi \leftarrow \Pi.\mathsf{Prove}^{\mathsf{H_{zk}}}(\mathsf{crs}, x, w)$, where $s \leftarrow \mathbb{Z}_p$, $x = (c, u_1, g_1)$, and $w = (\overline{m}, s)$. It outputs the first message $\rho_1 = (c, \pi)$ and stores the randomness $\mathsf{st} = s$.
- $\mathsf{BS_{BB}.Signer}(\mathsf{bsk}, \rho_1)$: parses $\rho_1 = (c, \pi)$, checks $\Pi.\mathsf{Verify}^{\mathsf{H_{zk}}}(\mathsf{crs}, x, \pi) = 1$ and outputs the second message $\rho_2 = (\rho_{2,0}, \rho_{2,1}) \leftarrow (\mathsf{sk} \cdot (c \cdot h_1)^r, g_1^r) \in \mathbb{G}_1^2$, where $r \leftarrow \mathbb{Z}_p$.
- $\mathsf{BS_{BB}.Derive}(\mathsf{st}, \rho_2)$: parses $\mathsf{st} = s$ and $\rho_2 = (\rho_{2,0}, \rho_{2,1})$, checks $e(\rho_{2,0}, g_2) = v \cdot e(\rho_{2,1}, u_2^{\overline{m}} g_2^s \cdot h_2)$, and outputs the signature $\sigma = (\rho_{2,0}/\rho_{2,1}^s \cdot (u_1^{\overline{m}} h_1)^{r'}, \rho_{2,1} \cdot g_1^{r'}) \in \mathbb{G}_1^2$ for $r' \leftarrow \mathbb{Z}_p$.
- $\mathsf{BS_{BB}.Verify}(\mathsf{bvk}, m, \sigma)$: sets $\overline{m} \leftarrow \mathsf{H_M}(m)$ and outputs $b \leftarrow \mathsf{S_{BB}.Verify}(\mathsf{bvk}, \overline{m}, \sigma)$.

5.2 Correctness and Security

We prove correctness, blindness and one-more unforgeability. Correctness follows from a simple calculation. Blindness follows from the zero-knowledge property of Π, and as c statistically hides the message and σ is re-randomized. The proof follows a similar all-but-one reduction as the underlying Boneh-Boyen signature. The only difference is that we modify the Boneh-Boeyn signature which is selectively secure in the standard model, to be adaptively secure in the ROM, and to use the (multi)-online extractor to extract randomness of c submitted by the adversary. Concretely, the reduction first guesses a query $\overline{m}^* = \mathsf{H_M}(m^*)$ and embeds a CDH challenge into vk such that it can sign all values in $\mathbb{Z}_p \setminus \{\overline{m}^*\}$. For each signing query, the reduction extracts the randomness of c from the proof π, simulates the signing of m as in the original `euf-cma` proof of $\mathsf{S_{BB}}$, and finally reapplies the randomness of c to the intermediate signature. If the extracted message is $\overline{m}^*$, the reduction aborts. Here, we crucially require that Π is online-extractable. In the end, the reduction hopes to receive a valid signature on $\overline{m}^*$ with which it can solve CDH. More details can be found in section 1.3.

Theorem 10 (Correctness). *The scheme* $\mathsf{BS_{BB}}$ *is correct.*

Proof. Let $(\mathsf{bvk}, \mathsf{bsk}) \leftarrow \mathsf{BS_{BB}.KeyGen}(1^\lambda)$, where $\mathsf{bvk} = (u_1, u_2, h_1, h_2, v)$ and $\mathsf{sk} = g_1^{\alpha\beta}$ with $u_1 = g_1^\alpha, u_2 = g_2^\alpha, h_1 = g_1^\gamma, h_2 = g_2^\gamma, v = e(g_1, g_2)^{\alpha\beta}$ and $\mathsf{bsk} = g_1^{\alpha\beta}$. Let $\rho_1 = (c = (u_1^{\overline{m}} g_1^s), \pi) \leftarrow \mathsf{BS_{BB}.User}(\mathsf{bvk}, m)$ for any message m and $\overline{m} \leftarrow \mathsf{H_M}(m)$, where note that the check on the verification key performed by the user passes by construction. Under the correctness of Π, the proof π verifies, and we have $\rho_2 = (\rho_{2,0}, \rho_{2,1}) = (\mathsf{sk} \cdot (c \cdot h_1)^r, g_1^r) \leftarrow \mathsf{BS_{BB}.Signer}(\mathsf{bsk}, \rho_1)$. Note that the check in $\mathsf{BS_{BB}.Derive}(\mathsf{st}, \rho_2)$ passes as

$$\begin{aligned} e(\rho_{2,0}, g_2) &= e(\mathsf{sk} \cdot (c \cdot h_1)^r, g_2) \\ &= e(\mathsf{sk}, g_2) \cdot e(c \cdot h_1, g_2)^r \\ &= v \cdot e(u_1^{\overline{m}} g_1^s \cdot h_1, g_2)^r \\ &= v \cdot e(g_1^r, u_2^{\overline{m}} g_2^s \cdot h_2) = v \cdot e(\rho_{2,1}, u_2^{\overline{m}} g_2^s \cdot h_2). \end{aligned}$$

Then, we can verify that $\sigma = (\sigma_0, \sigma_1) \leftarrow \mathsf{BS_{BB}.Derive}(\mathsf{st}, \rho)$ indeed outputs a valid $\mathsf{S_{BB}}$ signature:

$$\begin{aligned} (\sigma_0, \sigma_1) &= ((\mathsf{sk} \cdot (u_1^{\overline{m}} g_1^s \cdot h_1)^r)/g_1^{rs} \cdot (u_1^{\overline{m}} h_1)^{r'}, \quad g_1^r \cdot g_1^{r'}) \\ &= (\mathsf{sk} \cdot (u_1^{\overline{m}} h_1)^r \cdot (u_1^{\overline{m}} h_1)^{r'}, \quad g_1^{r+r'}) \\ &= (\mathsf{sk} \cdot (u_1^{\overline{m}} h_1)^{r+r'}, \quad g_1^{r+r'}). \end{aligned}$$

Theorem 11 (Blindness). *The scheme* $\mathsf{BS_{BB}}$ *is blind under malicious keys under the zero-knowledge property of* Π.

Proof. Let $\mathcal{A}$ be a PPT adversary against blindness. We define the following hybrids and denote by $\mathsf{Adv}^{\mathrm{H}_i}_{\mathcal{A}}(\lambda)$ the advantage of $\mathcal{A}$ in Hybrid i.

- Hybrid 0 is identical to the real game.
- Hybrid 1 is the same as Hybrid 0, except the H_{zk} queries and the proofs (π_0, π_1) are instead simulated via $\mathsf{Sim} = (\mathsf{Sim}_{\mathsf{H}_{\mathsf{zk}}}, \mathsf{Sim}_{\pi})$ of Π. In more detail, for every query q to the random oracle H_{zk}, it outputs $\mathsf{Sim}_{\mathsf{H}_{\mathsf{zk}}}(q)$. After receiving (bvk, m_0, m_1) from $\mathcal{A}$, it checks the validity of the verification key bvk and sets $\pi_b \leftarrow \mathsf{Sim}_{\pi}(\mathsf{crs}, x_b)$ for $x_b = (c_b, u_1, g_1)$, where $c_b = u_1^{\overline{m}_b} g_1^{s_b}$ and $\overline{m}_b = \mathsf{H}_{\mathsf{M}}(m_b)$
  We can construct an adversary $\mathcal{B}_{\Pi}$ against the zero-knowledge property of Π with advantage $\mathsf{Adv}^{\mathrm{zk}}_{\mathcal{B}_{\Pi}}(\lambda) \geq |\mathsf{Adv}^{\mathrm{H}_0}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda)|$. $\mathcal{B}_{\Pi}$ simulates the view to $\mathcal{A}$ by programming the received crs into H_{zk}. It then uses the provided oracles to answer H_{zk} queries and to generate proofs (π_0, π_1). Finally, $\mathcal{B}_{\Pi}$ forwards the guess of $\mathcal{A}$ to its challenger. If the oracle outputs simulated proofs, the game is distributed identically to Hybrid 1. Else, the oracle outputs real proofs and behaves as in Hybrid 0. Thus, we have

$$|\mathsf{Adv}^{\mathrm{H}_0}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda)| \leq \mathsf{Adv}^{\mathrm{zk}}_{\mathcal{B}_{\Pi}}(\lambda).$$

- Hybrid 2 is the same as Hybrid 1, except that the (inefficient) challenger recovers the signing key bsk, and prepares the signatures (σ_0, σ_1) on its own. In more detail, the challenger brute force searches the exponent $\alpha, \beta, \gamma \in \mathbb{Z}_p$ such that $u_1 = g_1^{\alpha}$, $h_1 = g_1^{\gamma}$, and $v = e(g_1, g_2)^{\alpha\beta}$ in the verification key bvk output by $\mathcal{A}$. When $\mathcal{A}$ returns the second message $\rho_{b,2} = (\rho_{b,2,0}, \rho_{b,2,1})$, it first checks if $e(\rho_{b,2,0}, g_2) = v \cdot e(\rho_{b,2,1}, u_2^{\overline{m}_b} g_2^{s_b} \cdot h_2)$ as in the previous Hybrid 1. If so, it runs $\sigma_b \leftarrow \mathsf{S}_{\mathsf{BB}}.\mathsf{Sign}(\mathsf{sk}, \overline{m}_b)$, where $\mathsf{sk} = g_1^{\alpha\beta}$ and $\overline{m}_b = \mathsf{H}_{\mathsf{M}}(m_b)$. Otherwise, it is the same as the previous Hybrid 1.
  We show that Hybrids 1 and 2 are perfectly indistinguishable. Since the check performed by the challenger passes, we have the following for both Hybrids:

$$\begin{aligned} & e(\rho_{b,2,0}, g_2) = v \cdot e(\rho_{b,2,1}, u_2^{\overline{m}_b} g_2^{s_b} \cdot h_2) \\ \Leftrightarrow \quad & e(\rho_{b,2,0}, g_2) = e(g_1, g_2)^{\alpha\beta} \cdot e(\rho_{b,2,1}, g_2)^{\alpha\overline{m}_b + s_b + \gamma}. \end{aligned}$$

  If we set $\rho_{b,2,1} = g_1^{r_b^*}$, then we have $\rho_{b,2,0} = g_1^{\alpha\beta + r_b^* \cdot (\alpha\overline{m}_b + s_b + \gamma)\cdot} = \mathsf{sk} \cdot (c_b \cdot h_1)^{r_b^*}$, where $\mathsf{sk} = g_1^{\alpha\beta}$. In Hybrid 1, the challenger then outputs the signature

$$\begin{aligned} \sigma_b &= (\rho_{2,b,0} / \rho_{2,b,1}^{s_b} \cdot (u_1^{\overline{m}_b} h_1)^{r'_b}, \rho_{2,b,1} \cdot g_1^{r'_b}) \\ &= (\mathsf{sk} \cdot (u_1^{\overline{m}_b} g_1^{s_b} \cdot h_1)^{r_b^*} \cdot g_1^{-r_b^* s_b} \cdot (u_1^{\overline{m}_b} h_1)^{r'_b}, g_1^{r_b^* + r'_b}) \\ &= (\mathsf{sk} \cdot (u_1^{\overline{m}_b} h_1)^{r_b^* + r'_b}, g_1^{r_b^* + r'_b}), \end{aligned}$$

  where $r'_b \leftarrow \mathbb{Z}_p$. Since r'_b is information-theoretically hidden from $\mathcal{A}$, σ_b is identically distributed as a signature output by $\mathsf{S}_{\mathsf{BB}}.\mathsf{Sign}(\mathsf{sk}, \overline{m}_b)$ in Hybrid 2. Hence, we have $\mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda)$.
- Hybrid 3 is the same as Hybrid 2, except that the challenger samples a random $c_b \leftarrow \mathbb{G}_1$, simulates a proof $\pi_b \leftarrow \mathsf{Sim}_{\pi}(\mathsf{crs}, x_b)$ for $x_b = (c_b, u_1, g_1)$, and outputs $\rho_{b,1} = (c_b, \pi_b)$ as the first message. It can be checked that Hybrids 2 and 3 are perfectly indistinguishable by noticing that $s_b \leftarrow \mathbb{Z}_p$ is information-theoretically hidden from $\mathcal{A}$ due to the modifications we made in Hybrids 1 and 2. Namely, in Hybrid 2, we have $c_b = u_1^{\overline{m}_b} g_1^{s_b}$ where s_b is uniform over $\mathbb{Z}_p$ from the view of $\mathcal{A}$. Hence, sampling c_b uniform over $\mathbb{G}$ results in the same distribution. Hence, we have $\mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda)$.

In Hybrid 3, the value of coin is information-theoretically hidden from $\mathcal{A}$, as the commitments c_b and the proofs π_b are identically distributed for $b \in \{0, 1\}$. Consequently, $\mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda) = 0$. Also, the running time of the adversaries $\mathcal{B}_{\Pi}$ is roughly that of $\mathcal{A}$. Combining the inequalities yields the statement.

Theorem 12 (Unforgeability). *The scheme* $\mathsf{BS}_{\mathsf{BB}}$ *is one-more unforgeable under the* CDH *assumption and the online-extractability of* Π.

Proof. Let $\mathcal{A}$ be a PPT adversary against one-more unforgeability of $\mathsf{BS}_{\mathsf{BB}}$. Let Ext be the extractor and $\mathsf{Sim}_{\mathsf{crs}}$ the simulator of Π from definition 21. We denote by Q_S the number of signing queries, by Q_M the number of H_{M} queries, and by Q_H the number of H_{zk} queries. Recall that we model $\mathsf{H}_{\mathsf{crs}}, \mathsf{H}_{\mathsf{zk}}$ and H_{M} as random oracles, where we assume without loss of generality that $\mathcal{A}$ never repeats queries. We denote by q_j the j-th query to H_{M} for $j \in [Q_M]$. After Q_S signing queries, $\mathcal{A}$ outputs $Q_S + 1$ forgeries $\{(m_i, \sigma_i)\}_{i=1}^{Q_S+1}$. We write $\sigma_i = (\sigma_{i,1}, \sigma_{i,2})$, and denote by $\rho_{1,i} = (c_i, \pi_i)$ the Q_S first message queries to $\mathsf{BS}_{\mathsf{BB}}.\mathsf{Signer}(\mathsf{bsk}, \cdot)$ issued by $\mathcal{A}$. We define the following hybrids and denote by $\mathsf{Adv}^{\mathrm{H}_i}_{\mathcal{A}}(\lambda)$ the advantage of $\mathcal{A}$ in Hybrid i.

- Hybrid 0 is identical the real game.
- Hybrid 1 is the same as Hybrid 0, except it samples $(\overline{\mathsf{crs}}, \tau) \leftarrow \mathsf{Sim}_{\mathsf{crs}}(1^\lambda)$ and programs $\overline{\mathsf{crs}}$ into the random oracle $\mathsf{H}_{\mathsf{crs}}$ via $\mathsf{H}_{\mathsf{crs}}(0) \leftarrow \overline{\mathsf{crs}}$. It is easy to construct an adversary $\mathcal{B}_{\mathsf{crs}}$ against the CRS indistinguishability of Π such that $\mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\mathrm{H}_0}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{crs}}_{\mathcal{B}_{\mathsf{crs}}}(\lambda)$.
- Hybrid 2 is the same as Hybrid 1, except $(\overline{m}_i, s_i)$ is extracted from all the proofs $\{\pi_i\}_{i\in[Q_S]}$ using Ext_{Π}. Specifically, when $\mathcal{A}$ provides the signing query $\rho_{1,i} = (c_i, \pi_i)$, the challenger runs $w_i \leftarrow \mathsf{Ext}(\mathsf{crs}, \mathsf{td}, x_i, \pi_i)$, where $x_i = (c_i, u_1, g_1)$. It parses $w_i = (\overline{m}_i, s_i)$ and aborts if $u_1^{\overline{m}_i} \cdot g_1^{s_i} \neq c_i$. Otherwise, it is defined identically to the previous Hybrid 1.
  We can construct an adversary $\mathcal{B}_{\mathsf{Ext}}$ against the multi-online extractability of Π with $\mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda) \geq \frac{\mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) - \mathrm{negl}(\lambda)}{p_{\mathsf{P}}(\lambda, Q_H)}$ with additional runtime overhead $p_{\mathsf{T}}(\lambda, Q_H) \cdot \mathsf{Time}(\mathcal{A})$, where p_{P} and p_{T} are polynomials as defined in definition 21. In more detail, let us first consider $\mathcal{B}'_{\mathsf{Ext}}$ that receives $\overline{\mathsf{crs}}$ from its challenger, and simulates the challenger of Hybrid 2 for $\mathcal{A}$, after programming $\overline{\mathsf{crs}}$ into $\mathsf{H}_{\mathsf{crs}}$ and by answering H_{zk} queries via its provided oracle. At the end of the game, $\mathcal{B}'_{\mathsf{Ext}}$ outputs the pairs $\{(x_i, \pi_i)\}_{i\in[Q_S]}$, where $x_i = (c_i, u_1, h_1)$. Because $\mathcal{A}$ succeeds with probability $\mathsf{Adv}^{\mathrm{H}_1}_{\lambda}$, the pre-condition of definition 21 holds, and the above bound follows.
  Observe that the output w_i of Ext is not used anywhere in Hybrid 2. Also, aborting in case of extraction failure and the runtime of Ext does not impact the success probability of $\mathcal{A}$ in Hybrid 2. Therefore, we can equally define an adversary $\mathcal{B}_{\mathsf{Ext}}$ that runs Ext during the game instead of at the end. Specifically, this is identical to the description of the Hybrid 2 challenger.
- Hybrid 3 is the same as Hybrid 2, except it aborts if there is a collision in H_{M} or if there is some message m_i in $\mathcal{A}$'s output that was never queried to H_{M}.
  The abort probability is upper bounded by $\frac{Q_M^2+1}{p}$, as a collision in H_{M} happens with probability at most Q_M^2/p and the probability that some S_{BB} signature σ_i is valid for a random message $\overline{m} \in \mathbb{Z}_p$ is $1/p$. It follows that $\mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda) - \frac{Q_M^2+1}{p}$.
- Hybrid 4 is the same as Hybrid 3, except it guesses a query index j^* of H_{M} for which it outputs some random $\overline{m}^* \in \mathbb{Z}_p$, and aborts if either some sign query contains $\overline{m}^*$ or if $\mathcal{A}$ provides no forgery for some message that hashes to $\overline{m}^*$. More concretely, before Hybrid 3 interacts with $\mathcal{A}$, it samples $\overline{m}^* \leftarrow \mathbb{Z}_p$ and $j^* \leftarrow [Q_M]$. For the j^*-th query to H_{M}, it sets $\mathsf{H}_{\mathsf{M}}(q_{j^*}) \leftarrow \overline{m}^*$. At the i-th signing query, the signer aborts if the extracted witness is $(\overline{m}^*, s_i)$, else proceeds as usual. Also, Hybrid 3 aborts if $\overline{m}^* \notin \{\mathsf{H}_{\mathsf{M}}(m_i) \mid i \in [Q_S + 1]\}$, where m_i are the messages from the forgeries output of $\mathcal{A}$.
  A simple calculation yields $\mathsf{Adv}^{\mathrm{H}_4}_{\mathcal{A}}(\lambda) \geq \frac{1}{Q_M}\mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda)$, where E_3 denotes the event that $\mathcal{A}$ is successful in Hybrid 3.

$$\begin{aligned}
\mathsf{Adv}^{\mathrm{H}_4}_{\mathcal{A}}(\lambda) &= \Pr[\overline{m}^* \notin \{\mathsf{Ext}(\mathsf{crs}, \mathsf{td}, x_i, \pi_i)\}_{i\in[Q_S]} \wedge \overline{m}^* \in \{\mathsf{H}_{\mathsf{M}}(m_i)\}_{i=\in[Q_S+1]} \mid E_3] \cdot \Pr[E_3] \\
&= \Pr[\overline{m}^* \in \{\mathsf{H}_{\mathsf{M}}(m_i)\}_{i\in[Q_S+1]} \setminus \{\mathsf{Ext}(\mathsf{crs}, \mathsf{td}, x_i, \pi_i)\}_{i\in[Q_S]} \mid E_3] \cdot \Pr[E_3] \\
&\geq \Pr[\overline{m}^* = \overline{m}^\star \mid E_3] \cdot \Pr[E_3] \\
&= 1/Q_M \cdot \mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda)
\end{aligned}$$

  Here, we use for the inequality that there is no collision in H_{M}, and thus there exists at least one $\overline{m}^\star \in \{\mathsf{H}_{\mathsf{M}}(m_i)\}_{i\in[Q_S+1]} \setminus \{\mathsf{Ext}(\mathsf{crs}, \mathsf{td}, x_i, \pi_i)\}_{i\in[Q_S]}$. In the last equality, we use that j^* is uniform in $\mathcal{A}$'s view.
- Hybrid 5 is the same as Hybrid 4, except it sets up a punctured verification key and simulates signing without knowing the full sk. Specifically, it samples $g_1 \leftarrow \mathbb{G}_1, g_2 \leftarrow \mathbb{G}_2$ and sets $A_1 \leftarrow g_1^\alpha, A_2 \leftarrow g_2^\alpha, B_1 \leftarrow g_1^\beta, B_2 \leftarrow g_2^\beta, u_1 = A_1, u_2 = A_2, h_1 = u_1^{-\overline{m}^*} \cdot g_1^\delta, h_2 = u_2^{-\overline{m}^*} \cdot g_2^\delta, v = e(A_1, B_2)$, where $\overline{m}^*$ is the j^*-th H_{M} output and $\alpha, \beta, \delta \leftarrow \mathbb{Z}_p$, and sends $\mathsf{bvk} =$

$(g_1, g_2, u_1, u_2, h_1, h_2, v)$ to $\mathcal{A}$. Note that implicitly $\gamma = -\overline{m}^* \cdot \alpha + \delta \in \mathbb{Z}_p$. For each signing query $\rho_{i,1} = (c_i, \pi_i)$ with extracted witness $w_i = (\overline{m}_i, s_i)$, the challenger samples some $\tilde{r_i} \leftarrow \mathbb{Z}_p$, and sets $\rho_{i,2,1} = g_1^{\tilde{r_i}} \cdot B_1^{-1/(\overline{m}_i - \overline{m}^*)}$ and $\rho_{i,2,0} = A^{(\overline{m}_i - \overline{m}^*)\tilde{r}} \cdot g_1^{(s_i+\delta)\tilde{r_i}} \cdot B_1^{-(s_i+\delta)/(\overline{m}_i - \overline{m}^*)}$. It then returns $\mathcal{A}$ the second message as $\rho_{i,2} = (\rho_{i,2,0}, \rho_{i,2,1})$. Otherwise, it is defined identically as in the previous Hybrid 4.
Clearly, bvk is distributed identically in Hybrids 4 and 5. Also, $\rho_{i,2,0} = g_1^{\tilde{r_i} - \beta/(\overline{m}_i - \overline{m}^*)}$ is distributed identically in Hybrids 4 and 5, as $r_i = \tilde{r_i} - \beta/(\overline{m}_i - \overline{m}^*)$ is distributed uniformly over $\mathbb{Z}_p$ for a uniform $\tilde{r_i}$. The same holds for $\rho_{i,2,1}$, as $\overline{m}_i \neq \overline{m}^*$ due to the abort condition and

$$\begin{aligned} g_1^{\alpha\beta}(c_i \cdot h_1)^{r_i} &= g_1^{\alpha\beta}(A^{\overline{m}_i} \cdot g_1^{s_i} \cdot A^{-\overline{m}^*} \cdot g_1^{\delta})^{\tilde{r_i} - \beta/(\overline{m}_i - \overline{m}^*)} \\ &= g_1^{\alpha\beta}(A^{\overline{m}_i - \overline{m}^*} \cdot g_1^{s_i+\delta})^{\tilde{r_i} - \beta/(\overline{m}_i - \overline{m}^*)} \\ &= g_1^{\alpha\beta} A^{-\beta} \cdot A^{(\overline{m}_i - \overline{m}^*)\tilde{r}} \cdot g_1^{(s_i+\delta)\tilde{r_i}} \cdot B^{-(s_i+\delta)/(\overline{m}_i - \overline{m}^*)} \\ &= A^{(\overline{m}_i - \overline{m}^*)\tilde{r}} \cdot g_1^{(s_i+\delta)\tilde{r_i}} \cdot B^{-(s_i+\delta)/(\overline{m}_i - \overline{m}^*)} = \rho_{i,2,1}. \end{aligned}$$

Thus, $\mathsf{Adv}^{\mathrm{H}_5}_{\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathrm{H}_4}_{\mathcal{A}}(\lambda)$.

We now show that we construct an adversary $\mathcal{B}_{\mathsf{CDH}}$ with $\mathsf{Adv}^{\mathrm{CDH}}_{\mathcal{A}_{\mathsf{CDH}}}(\lambda) = \mathsf{Adv}^{\mathrm{H}_5}_{\mathcal{A}}(\lambda)$. First, $\mathcal{B}_{\mathsf{CDH}}$ receives CDH-tuple $(g_1, g_2, A_1, A_2, B_1, B_2)$, and uses these values to simulate the challenger of Hybrid 5 to $\mathcal{A}$. After $\mathcal{A}$ outputs the forgeries $\{(m_i, \sigma_i)\}_{i=1}^{Q_S+1}$, $\mathcal{B}_{\mathsf{CDH}}$ outputs $\sigma_{i^*,1}/\sigma_{i^*,2}^{\delta}$ to its challenger, where i^* such that $\mathsf{H}_{\mathsf{M}}(m_{i*}) = \overline{m}^*$ and $\sigma_{i^*} = (\sigma_{i^*,1}, \sigma_{i^*,2})$.

Note that due to the abort conditions in Hybrid 5, the probability that $\mathcal{B}_{\mathsf{CDH}}$ outputs such a value $\sigma_{i^*,1}/\sigma_{i^*,2}^{\delta}$ with $e(\sigma_{i^*,1}, g_2) = v \cdot e(\sigma_{i^*,2}, u_2^{\overline{m}_{i^*}} \cdot h_2)$ is at least $\mathsf{Adv}^{\mathrm{H}_5}_{\mathcal{A}}(\lambda)$. In that case, we have

$$\begin{aligned} & e(\sigma_{i^*,1}, g_2) = v \cdot e(\sigma_{i^*,2}, u_2^{\overline{m}_{i^*}} \cdot h_2) \\ \Longrightarrow\ & e(\sigma_{i^*,1}, g_2) = v \cdot e(\sigma_{i^*,2}, A_2^{\overline{m}^*} \cdot A_2^{-\overline{m}^*} g_2^{\delta}) \\ \Longrightarrow\ & e(\sigma_{i^*,1}, g_2) = v \cdot e(\sigma_{i^*,2}, g_2^{\delta}) \\ \Longrightarrow\ & e(\sigma_{i^*,1}/\sigma_{i^*,2}^{\delta}, g_2) = v \end{aligned}$$

Thus, $\sigma_{i^*,1}/\sigma_{i^*,2}^{\delta} = g_1^{\alpha\beta}$ as desired. The statement follows after collecting all the above bounds on the success probability and runtime.

6 Instantiation of the Framework based on Boneh-Boyen

Here, we instantiate the online-extractable NIZK Π and analyze the efficiency of the blind signature $\mathsf{BS}_{\mathsf{BB}}[\Pi]$. First, we present our instantiation of Π. Π is a NIZK for showing knowledge of an opening of a Pedersen commitment $c = u_1^m g_1^s$, where $u_1, g_1 \in \mathbb{G}_1$ and $m, s \in \mathbb{Z}_p$,[13] i.e. for the relation

$$\mathsf{R}_{\mathsf{bb}} \coloneqq \{x = (c, u_1, g_1), w = (m, s) \mid c = u_1^m g_1^s\}.$$

On a high level, we follow the well-known paradigm of combining an extractable commitment scheme (or equivalently a PKE) with a rewinding-based (non-online-extractable) NIZK Π' to construct an online-extractable NIZK Π. In the paradigm, the prover commits to the witness (m, s) using the extractable commitments, and adds a proof π via Π' to ensure that she indeed committed to openings of c, i.e. that $c = u_1^m g_1^s$ and (m, s) are committed in the extractable commitments. The online extractor can recover (m, s) from π by extracting the commitments via an appropriate trapdoor. Note that the soundness of the NIZK Π' guarantees consistency of the committed values and the openings of c. Indeed, in case the online extraction fails with non-negligible probability, we obtain a contradiction to the soundness of the underlying NIZK Π' via rewinding.

In the group setting, a common choice is ElGamal commitments. Note that we require the variant of ElGamal with message space $\mathbb{Z}_p$, i.e. $c_m = (g_1^m \mathsf{pp}^{r_m}, g_1^{r_m})$ and $(c_s = g_1^s \mathsf{pp}^{r_s}, g_1^{r_s})$ (see

[13] While we used $\overline{m}$ in the previous section, we omit the bar and simply denote m for readability.

remark 3). Unfortunately, we can only extract (m, s) if the values are short, i.e. $m, s \in [0, B-1]$ for $B = \mathsf{poly}(\lambda)$. A common technique to circumvent this issue is to instead commit to the binary decomposition $(m_i)_i$ and $(s_i)_i$, where $m = \sum_{i=1}^{\lceil \log_2 p \rceil} m_i 2^{i-1}$ and $s = \sum_{i=1}^{\lceil \log_2 p \rceil} s_i 2^{i-1}$, and show that the committed values $(m_i)_i$ and $(s_i)_i$ are bits and form valid decompositions of m and s, respectively. Unfortunately, this approach requires $2 \cdot \lceil \log_2(p) \rceil$ ElGamal commitments, and thus at least 1024 group elements for $\lambda = 256$. These elements already amount to more than 32 KB alone.

We instead commit to the B-ary decomposition $(m_i)_{i \in [\ell]}$ and $(s_i)_{i \in [\ell]}$ of $m = \sum_{i=1}^{\ell} m_i B^{i-1}$ and $s = \sum_{i=1}^{\ell} s_i B^{i-1}$, respectively, where $\ell = \lceil \log_B p \rceil$. As before, we show consistency of the committed values with the openings of c via a NIZK that ensures $c = u_1^m g_1^s$, $m = \sum_{i=1}^{\ell} m_i B^{i-1}$ and $s = \sum_{i=1}^{\ell} s_i B^{i-1}$, and a range proof which ensures that all ElGamal committed m_i and s_i lie in the range $[0, B-1]$. This approach improves efficiency considerably. For example for $B = 2^{32}$, we have $\ell = 8$ and require only 32 group elements for the ElGamal commitments, instead of 1024.

We instantiate the NIZK Π by composing two NIZKs together: one for proving the ElGamal commitments (Π_{ped}) and the other for the range proof (Π_{rp}). For the first NIZK Π_{ped}, we use the Fiat-Shamir transformation on an appropriate Σ-protocol Σ_{ped}. For the second NIZK Π_{rp} for the range proof, we would like to use the Fiat-Shamir transformation on a variant of Bulletproofs [27] that shows range membership for multiple Pedersen commitments ([8], Appendix F.2). However, Bulletproofs are not well-established in the non-interactive setting. Recently, [45] shows that non-interactive Bulletproofs are sound in the AGM and ROM, but as the proof relies on the AGM, it is not sufficient for our purpose. Attema, Fehr and Klooß show in another recent result [9] that the Fiat-Shamir transformation is sound for multi-round interactive proof systems Σ_{int}, if Σ_{int} is standard *special* sound, i.e. given a valid *transcript tree*, it is possible to recover the witness unconditionally. Unfortunately, Bulletproofs and its variant [8] satisfy only *computational* special soundness which is insufficient to apply the result of [9] directly. To this end, we show that for an appropriate *relaxed* special soundness relation, we can nonetheless apply the Fiat-Shamir transform on Bulletproofs and its variant [8] using the result of [9]. While the resulting NIZK Π_{rp} for the range proof satisfies a relaxed notion of special soundness, this is sufficient for our purpose.

Equipped with the above tools, we can instantiate Π. We apply three further optimizations:

1. The (interactive) range proof of [8] requires the witness $\boldsymbol{e} = (m_1, \ldots, m_\ell, s_1, \ldots, s_\ell)$ to be committed in the Pedersen commitments. Note that we can reuse the ElGamal commitments E_i as Pedersen commitments for the range proof, where $(E_i = g_1^{e_i} \mathsf{pp}^{r_i}, R_i = g_1^{r_i})_{i \in [2\ell]}$ are already required for online extraction. However, as the extractor knows the trapdoor td such that $g_1^{\mathsf{td}} = \mathsf{pp}$, we need to be careful in the security analysis, as td also allows to equivocate Pedersen commitments (and thus potentially break soundness of the range proof). This subtlety is reflected in the relaxed soundness relation. Fortunately, we can analyze the extraction probability without knowing td and the proof goes through. We provide more details in section 6.2.
2. During extraction, we use a more efficient algorithm to compute the discrete logarithm in $\mathcal{O}(\sqrt{B})$. This allows us to choose better parameters, as a larger bound B improves efficiency but impacts the runtime of the extractor.
3. Observe that pairing groups are generally larger and slower than simple prime-order groups. Thus, we move the parts that are not required to be in $\mathbb{G}_1$ into a group $\widehat{\mathbb{G}}$ of the same order p as $\mathbb{G}_1$ (to maintain algebraic consistency). As both NIZKs Π_{ped} and Π_{rp} are not reliant on pairings, we can perform both almost exclusively in $\widehat{\mathbb{G}}$.

In the following, we first present the Σ-protocol Σ_{ped} (which we will later transform into a NIZK Π_{ped} via Fiat-Shamir) and the NIZK Π_{rp} for the range proof. We then combine both proof systems into an online-extractable NIZK Π for the relation R_{bb} and analyze its security. In more detail, when online-extraction of Π fails, it is easy to show that we can extract a witness from *at least one* of the proofs generated by Π_{ped} and Π_{rp} by relying on the rewinding-extraction of Π_{ped} and the adaptive knowledge soundness of Π_{rp} (cf. definition 20) in an independent manner. What is non-trivial is to show that both extractions succeed *simultaneously*. Such simultaneous extraction is necessary since the two NIZKs are glued together by the Pedersen commitment $E_i = g_1^{e_i} \mathsf{pp}^{r_i}$: each NIZK may be using a different opening to construct the proof, in which case we need to extract from both proofs to break binding. For instance, using the standard notion of rewinding-extractability, we cannot exclude the case where the adversary sets up the proofs π_0, π_1 of $\Pi_{\mathsf{rp}}, \Pi_{\mathsf{ped}}$, respectively, in such

a way that if the extractor of Π_{rp} succeeds, then the extractor of Π_{ped} fails. We show through a careful non-black analysis of the underlying NIZKs that there is a non-negligible probability of the rewinding-extraction of Π_{ped} and the adaptive knowledge soundness of Π_{rp} succeeding at the same time. Finally, we evaluate the efficiency of $\mathsf{BS}_{\mathsf{BB}}[\Pi]$. For readability, we mark elements $\widehat{g}$ in $\widehat{\mathbb{G}}$ with a hat.

6.1 Σ-protocol Σ_{ped} for the Decomposition

We first present a Σ-protocol Σ_{ped} for the relation $\mathsf{R}_{\mathsf{ped}}$ where

$$\mathsf{R}_{\mathsf{ped}} = \{(x,w) : c = u_1^m g_1^s, E_i = \widehat{g}^{e_i}\widehat{\mathsf{pp}}^{r_i}, R_i = \widehat{g}^{r_i}, \prod_{i\in[\ell]} E_i^{B^{i-1}} = \widehat{g}^m \cdot \widehat{\mathsf{pp}}^{t_m}, \prod_{i\in[\ell]} E_{i+\ell}^{B^{i-1}} = \widehat{g}^s \cdot \widehat{\mathsf{pp}}^{t_s}\},$$

where $x = (c, u_1, g_1, \widehat{g}, \widehat{\mathsf{pp}}, (E_i, R_i)_{i\in[2\ell]}, B)$ and $w = (m, s, (e_i, r_i)_{i\in[2\ell]}, t_m, t_s)$. Note that the relation shows that $m = \sum_{i=1}^{\ell} e_i B^{i-1}$ and $s = \sum_{i=1}^{\ell} e_{i+\ell} B^{i-1}$ under the DLOG assumption. The protocol is given below.

- $\Sigma_{\mathsf{ped}}.\mathsf{Init}(x,w)$: for (x,w) as above, samples additive masks $\widetilde{m}, \widetilde{s}, \widetilde{e}_i, \widetilde{r}_i, \widetilde{t}_m, \widetilde{t}_s \leftarrow \mathbb{Z}_p$, sets

$$D_c = u_1^{\widetilde{m}} g_1^{\widetilde{s}}, D_{e_i} = \widehat{g}^{\widetilde{e}_i}\widehat{\mathsf{pp}}^{\widetilde{r}_i}, D_{r_i} = \widehat{g}^{\widetilde{r}_i},$$
$$D_m = \widehat{g}^{\widetilde{m}}\widehat{\mathsf{pp}}^{\widetilde{t_m}}, D_s = \widehat{g}^{\widetilde{s}}\widehat{\mathsf{pp}}^{\widetilde{t_s}},$$

  where $i \in [2\ell]$, outputs $\alpha = (D_c, (D_{e_i}, D_{r_i})_{i\in[2\ell]}, D_m, D_s)$, and stores $(\widetilde{m}, \widetilde{s}, (\widetilde{e}_i, \widetilde{r}_i)_{i\in[2\ell]}, \widetilde{t}_m, \widetilde{t}_s)$ in st.
- $\Sigma_{\mathsf{ped}}.\mathsf{Chall}()$: samples a challenge $\beta \leftarrow \mathbb{Z}_p$,
- $\Sigma_{\mathsf{ped}}.\mathsf{Resp}(\mathsf{st}, \beta)$: sets $\gamma_k = \beta \cdot k + \widetilde{k}$ for $k \in \{m, s, e_1, \ldots, e_{2\ell}, r_1, \ldots, r_{2\ell}, t_m, t_s\}$, and outputs the response $\gamma = (\gamma_m, \gamma_s, (\gamma_{e_i}, \gamma_{e_i})_{i\in[2\ell]}, \gamma_{t_m}, \gamma_{t_s})$,
- $\Sigma_{\mathsf{ped}}.\mathsf{Verify}(x, \alpha, \beta, \gamma)$: checks the following equations

$$D_c = u_1^{\gamma_m} \cdot g_1^{\gamma_s} \cdot c^{-\beta}, D_{e_i} = \widehat{g}^{\gamma_{e_i}}\widehat{\mathsf{pp}}^{\gamma_{r_i}} \cdot E_i^{-\beta}, D_{r_i} = \widehat{g}^{\gamma_{r_i}} \cdot R_i^{-\beta},$$
$$D_m = \widehat{g}^{\gamma_m}\widehat{\mathsf{pp}}^{\gamma_{t_m}} \cdot (\prod_{i=1}^{\ell} E_i^{B^{i-1}})^{-\beta}, D_s = \widehat{g}^{\gamma_s}\widehat{\mathsf{pp}}^{\gamma_{t_s}} \cdot (\prod_{i=1}^{\ell} E_{i+\ell}^{B^{i-1}})^{-\beta},$$

  and outputs 1 if and only if all checks succeed. Note that the first three equations open the commitments c and (E_i, R_i), and the last two equations show the products hold. Also, observe that the equations are well-defined, as both $\widehat{\mathbb{G}}$ and $\mathbb{G}_1$ have order p.

We now show that Σ_{ped} is correct, HVZK, 2-special sound and has high min-entropy.

Theorem 13. *The Σ-protocol Σ_{ped} is correct.*

Proof. Let (α, β, γ) be a honest transcript for (x,w), where $x = (c, u_1, g_1, \widehat{g}, \widehat{\mathsf{pp}}, (E_i, R_i)_{i\in[2\ell]}, B)$ and $w = (m, s, (e_i, r_i)_{i\in[2\ell]}, t_m, t_s)$. We use the notation from above. We show that the first and fourth check pass, the remaining identities follow similarly.

$$u_1^{\gamma_m} \cdot g_1^{\gamma_s} \cdot c^{-\beta} = u_1^{\beta\cdot m+\widetilde{m}} \cdot g_1^{\beta\cdot s+\widetilde{s}} \cdot c^{-\beta} =$$
$$(u_1^m \cdot g_1^s)^{\beta} \cdot u_1^{\widetilde{m}} g_1^{\widetilde{s}} \cdot c^{-\beta} = c^{\beta} \cdot u_1^{\widetilde{m}} g_1^{\widetilde{s}} \cdot c^{-\beta} = D_c$$

$$\widehat{g}^{\gamma_m}\widehat{\mathsf{pp}}^{\gamma_{t_m}} \cdot (\prod_{i=1}^{\ell} E_i^{B^{i-1}})^{-\beta} = \widehat{g}^{\beta\cdot m+\widetilde{m}}\widehat{\mathsf{pp}}^{\beta\cdot t_m+\widetilde{t_m}} \cdot (\prod_{i=1}^{\ell} E_i^{B^{i-1}})^{-\beta} =$$
$$(\widehat{g}^m \cdot \widehat{\mathsf{pp}}^{t_m})^{\beta} \cdot \widehat{g}^{\widetilde{m}}\widehat{\mathsf{pp}}^{\widetilde{t_m}} \cdot (\prod_{i=1}^{\ell} E_i^{B^{i-1}})^{-\beta} = \widehat{g}^{\widetilde{m}}\widehat{\mathsf{pp}}^{\widetilde{t_m}} = D_m$$

Theorem 14. *The Σ-protocol Σ_{ped} is HVZK.*

Proof. Let $x = (c, u_1, g_1, \widehat{g}, \widehat{\mathsf{pp}}, (E_i, R_i)_{i\in[2\ell]}, B)$ and $\beta \in \mathbb{Z}_p$. We define the simulator Sim as follows. On input (x, β), samples $\gamma = (\gamma_m, \gamma_s, (\gamma_{e_i}, \gamma_{e_i})_{i\in[2\ell]}, \gamma_{t_m}, \gamma_{t_s}) \leftarrow \mathbb{Z}_p^{4+4\ell}$ and computes $(D_c, (D_{e_i}, D_{r_i})_{i\in[2\ell]}, D_m, D_s)$ via the identities in $\Sigma_{\mathsf{ped}}.\mathsf{Verify}$. Finally, sets $\alpha = (D_c, (D_{e_i}, D_{r_i})_{i\in[2\ell]}, D_m, D_s)$ and outputs the transcript (α, β, γ).

To show that Sim outputs transcripts that are indistinguishable from real transcripts, we define the following hybrids and denote by $\mathsf{Adv}^{\mathrm{H}_i}_{\mathcal{A}}(\lambda)$ the advantage of $\mathcal{A}$ in Hybrid i.

- Hybrid 0 outputs honestly generated transcripts.
- Hybrid 1 is the same as Hybrid 0, except the elements $(D_c, (D_{e_i}, D_{r_i})_{i\in[2\ell]}, D_m, D_s)$ are generated as in Sim. It is easy to check that Hybrid 0 and Hybrid 1 are identically distributed, and we have $\mathsf{Adv}^{\mathrm{H}_0}_{\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda)$.
- Hybrid 2 is the same as Hybrid 1, except γ is computed as in Sim. As the values $\widetilde{k}$ serves as one-time pad for $\beta \cdot k$, where $k \in \{m, s, e_1, \ldots, e_{2\ell}, r_1, \ldots, r_{2\ell}, t_m, t_s\}$, it follows that Hybrid 1 and Hybrid 2 are identically distributed. Thus, we have $\mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda)$.

As Hybrid 2 outputs simulated transcripts, the statement follows.

Theorem 15. *The Σ-protocol Σ_{ped} is 2-special sound.*

Proof. Let $x = (c, u_1, g_1, \widehat{g}, \widehat{\mathsf{pp}}, (E_i, R_i)_{i\in[2\ell]}, B)$. We define the extractor Ext as follows. On input valid transcripts (α, β, γ) and $(\alpha, \beta', \gamma')$ with $\beta \neq \beta'$. Ext parses $\alpha = (D_c, (D_{e_i}, D_{r_i})_{i\in[2\ell]}, D_m, D_s)$ and $\gamma = (\gamma_m, \gamma_s, (\gamma_{e_i}, \gamma_{e_i})_{i\in[2\ell]}, \gamma_{t_m}, \gamma_{t_s}), \gamma' = (\gamma'_m, \gamma'_s, (\gamma'_{e_i}, \gamma'_{e_i})_{i\in[2\ell]}, \gamma'_{t_m}, \gamma'_{t_s})$. As both transcripts are valid, we obtain the following identities via the verification identities

$$
\begin{aligned}
u_1^{\gamma_m} \cdot g_1^{\gamma_s} \cdot c^{-\beta} &= u_1^{\gamma'_m} \cdot g_1^{\gamma'_s} \cdot c^{-\beta'},\\
\widehat{g}^{\gamma_{e_i}} \widehat{\mathsf{pp}}^{\gamma_{r_i}} \cdot E_i^{-\beta} &= \widehat{g}^{\gamma'_{e_i}} \widehat{\mathsf{pp}}^{\gamma'_{r_i}} \cdot E_i^{-\beta'},\\
\widehat{g}^{\gamma_{r_i}} \cdot R_i^{-\beta} &= \widehat{g}^{\gamma'_{r_i}} \cdot R_i^{-\beta'},\\
\widehat{g}^{\gamma_m} \widehat{\mathsf{pp}}^{\gamma_{t_m}} \cdot (\prod_{i=1}^{\ell} E_i^{B^{i-1}})^{-\beta} &= \widehat{g}^{\gamma'_m} \widehat{\mathsf{pp}}^{\gamma'_{t_m}} \cdot (\prod_{i=1}^{\ell} E_i^{B^{i-1}})^{-\beta'},\\
\widehat{g}^{\gamma_s} \widehat{\mathsf{pp}}^{\gamma_{t_s}} \cdot (\prod_{i=1}^{\ell} E_{i+\ell}^{B^{i-1}})^{-\beta} &= \widehat{g}^{\gamma'_s} \widehat{\mathsf{pp}}^{\gamma'_{t_s}} \cdot (\prod_{i=1}^{\ell} E_{i+\ell}^{B^{i-1}})^{-\beta'},
\end{aligned}
$$

We denote $\Delta k = (\gamma_k - \gamma'_k)$ and $k = \Delta k / \Delta\beta$ for $k \in \{m, s, e_1, \ldots, e_{2\ell}, r_1, \ldots, r_{2\ell}, t_m, t_s\}$ and $\Delta\beta = (\beta - \beta')$. Note that $\Delta\beta \neq 0$. The extractor finally outputs $(m, s, e_1, \ldots, e_{2\ell}, r_1, \ldots, r_{2\ell}, t_m, t_s)$.

From the first equation, we obtain $u_1^{\gamma_m - \gamma'_m} \cdot g_1^{\gamma_s - \gamma'_s} \cdot c^{-(\beta-\beta')} = 1_{\mathbb{G}_1}$. Taking both sides to the power of $1/\Delta\beta$ yields $u_1^{\Delta m/\Delta\beta} \cdot g_1^{\Delta s/\Delta\beta} = c$. By definition of m and s, we obtain $c = u_1^m g_1^s$ as desired. Similarly, we obtain from the second and third equation that $\widehat{g}^{e_i} \widehat{\mathsf{pp}}^{r_i} = E_i$ and $\widehat{g}^{r_i} = R_i$, respectively. The fourth equation yields that $\widehat{g}^{\gamma_m - \gamma'_m} \widehat{\mathsf{pp}}^{\gamma_{t_m} - \gamma'_{t_m}} \cdot (\prod_{i=1}^{\ell} E_i^{B^{i-1}})^{-(\beta-\beta')} = 1_{\widehat{\mathbb{G}}}$. Again, multiplying with $1/\Delta\beta$ in the exponent yields $\prod_{i\in[\ell]} E_i^{B^{i-1}} = \widehat{g}^m \cdot \widehat{\mathsf{pp}}^{t_m}$, and similarly the fifth equation yields $\prod_{i\in[\ell]} E_{i+\ell}^{B^{i-1}} = \widehat{g}^s \cdot \widehat{\mathsf{pp}}^{t_s}$. This concludes the proof.

Theorem 16. *The Σ-protocol Σ_{ped} has high min-entropy.*

Proof. Observe that all D_{r_i} are distributed uniformly random in $\widehat{\mathbb{G}}$. It follows that the advantage of any adversary in the min-entropy game is at most $1/p^{\ell} = \mathrm{negl}(\lambda)$.

6.2 Range Proof Π_{rp} for the Decomposition

We now describe the NIZK Π_{rp} for the range proof for the vectors committed in E_i. We start with an appropriate multi-round interactive proof system and obtain Π_{rp} via the Fiat-Shamir transformation. We follow the definitions for multi-round interactive proof systems and the notion of $(\boldsymbol{k}, \boldsymbol{N})$-special soundness with vectors $\boldsymbol{k}$ and $\boldsymbol{N}$ of [9]. We use the definitions of correctness, HVZK of [8]. Since we

can rely on the result of [9, 8] in a black-box manner, we refer the readers to [9, 8] for the formal definitions.

Let H_{rp} be a random oracle mapping into $\mathbb{Z}_p$ [14]. Π_{rp} is a NIZK with random oracle H_{rp} for the relation

$$\mathsf{R}_{\mathsf{rp}} = \{(x, w) : \ E_i = \widehat{g}^{e_i} \cdot \widehat{\mathsf{pp}}^{r_i}, e_i \in [0, B-1] \text{ for } i \in [2\ell]\},$$

with $x = (B, (E_i)_{i\in[2\ell]})$ and $w = ((e_i, r_i)_{i\in[2\ell]})$, where B is a power of two. We obtain Π_{rp} by applying the Fiat-Shamir transformation as described in [9] to the multi-round interactive proof system $\Sigma^{2\ell}_{\mathsf{rp}}$ with $\mathsf{crs} = (\widehat{g}, \widehat{\mathsf{pp}}, (\widehat{g}_i)_{i\in[\ell_{\mathsf{rp}}]})$ from [8] (Appendix F.2), for appropriate $\ell_{\mathsf{rp}} \in \mathbb{N}$.

Denote with $\mathsf{R}_{\mathsf{dlog}} = \{(\mathsf{crs}, w^*)\}$ the relation that contains all non-trivial DLOG relations w^* for crs, i.e. computing w^* for random crs allows to solve the DLOG assumption (see [8] for more details). Via Theorem 14 of [8], we can show that Π_{rp} is correct and zero-knowledge. Moreover, we can show adaptive knowledge soundness for the relaxed relation

$$\mathsf{R}_{\mathsf{lax}} \coloneqq \{(x, w) : \ (x, w) \in \mathsf{R}_{\mathsf{rp}} \text{ or } (\mathsf{crs}, w) \in \mathsf{R}_{\mathsf{dlog}}\},$$

using Theorem 4 of [9], which is sufficient for our purpose. We sketch the proof below.

Theorem 17. *The* NIZK Π_{rp} *for relation* R_{rp} *is correct, zero-knowledge and adaptively knowledge sound for the relaxed relation* $\mathsf{R}_{\mathsf{lax}} \supseteq \mathsf{R}_{\mathsf{rp}}$.

Proof (Sketch). Correctness follows directly, as $\Sigma^{2\ell}_{\mathsf{rp}}$ is correct (Theorem 14, [8]) and the Fiat-Shamir transformation retains correctness of the interactive protocol. For showing zero-knowledge, observe that the intermediate prover outputs in $\Sigma^{2\ell}_{\mathsf{rp}}$ have high min-entropy. Thus, with all but negligible probability, these outputs were never queried to the random oracle. Consequently, the simulator can simulate a proof π by first simulating a transcript with challenge vector $\boldsymbol{\beta}$ using the HVZK property of $\Sigma^{2\ell}_{\mathsf{rp}}$, and then programming the random oracle with $\boldsymbol{\beta}$ accordingly.

Note that [9] shows *computational* special soundness of $\Sigma^{2\ell}_{\mathsf{rp}}$ for the relation R_{rp}. Here, computational special soundness means that either some w such that $(x, w) \in \mathsf{R}_{\mathsf{rp}}$ or w^* such that $(w^*, \mathsf{crs}) \in \mathsf{R}_{\mathsf{dlog}}$ is extracted from a transcript tree. The latter happens with negligible probability under the DLOG assumption, so w is extracted with overwhelming probability. However, to apply Theorem 4 from [9] to Π_{rp}, we require standard special soundness of $\Sigma^{2\ell}_{\mathsf{rp}}$, and thus we make the witness w^* explicit in the relation $\mathsf{R}_{\mathsf{lax}}$. For the relation $\mathsf{R}_{\mathsf{lax}}$, the interactive proof system $\Sigma^{2\ell}_{\mathsf{rp}}$ is $(\boldsymbol{k}, \boldsymbol{N})$-special sound with vectors $\boldsymbol{k} = (2\ell+1, 4n\ell+1, 2n\ell+3, 2, 2, n_1, \ldots, n_\mu)$ and $\boldsymbol{N} = (N_i)_{i=1}^{5+\mu}$, where $n = \log_2(B), \mu = \lceil \log_2(4n\ell+4) \rceil - 1, n_i = 3$ and $N_i = p$. As [9] never requires correctness or HVZK of the proof system, it is fine that R_{rp} and $\mathsf{R}_{\mathsf{lax}}$ are different. Thus, we can apply Theorem 4 in [9] to show adaptive knowledge soundness [15]. Note that the knowledge error $\mathsf{Er}(\boldsymbol{k}, \boldsymbol{N})$ is negligible in λ, following the notation of [9], because:

$$\begin{aligned}
\mathsf{Er}(\boldsymbol{k}, \boldsymbol{N}) &= 1 - \prod_{i=1}^{5+\mu} (1 - \frac{k_i - 1}{N_i}) \\
&\leq 1 - \prod_{i=1}^{5+\mu} (1 - \frac{4n\ell}{p}) \\
&= \mathrm{negl}(\lambda)
\end{aligned}$$

As we also have $\prod_{i=1}^{5+\mu} k_i \leq (5n\ell)^5 \cdot 3^{\log_2(8n\ell)} = \mathcal{O}((n\ell)^6) = \mathsf{poly}(\lambda)$, the extractor runs in time $\mathsf{poly}(\lambda)$ as desired. The statement follows.

[14] Note that technically, Π_{rp} requires a tuple of hash function $(\mathsf{H}_i)_{i\in[5+\mu]}$ mapping into $\mathbb{Z}_p$, where $\mu = \lceil \log_2(4\log_2(B)\ell + 4) \rceil - 1$. With sufficient input separation, we view $(\mathsf{H}_i)_{i\in[5+\mu]}$ as a single random oracle H_{rp}, for example if we query $\mathsf{H}_{\mathsf{rp}}(i, q)$ instead of $\mathsf{H}_i(q)$.

[15] Technically, our definition of adaptive knowledge soundness (cf. definition 20) differs slightly from the definition in [9]. Ours allows us to prove online-extractability for our construction Π later, and it is easy to check that the extractor of [9] suffices for our definition (cf. remark 3, [AFK22]).

6.3 Online-extractable NIZK Π for R_{bb}

We are now ready to instantiate the online-extractable NIZK Π for the relation R_{bb} with $\mathsf{crs} = (\widehat{g}, \widehat{\mathsf{pp}}, (\widehat{g}_i)_{i\in[\ell_{\mathsf{rp}}]})$. Let H_{rp} be the random oracle of Π_{rp} and H_β be a random oracle mapping into $\mathbb{Z}_p$. We denote by $\mathsf{H}_{\mathsf{bb}} = (\mathsf{H}_{\mathsf{rp}}, \mathsf{H}_\beta)$ the random oracle of Π [16]. Let $B = \mathsf{poly}(\lambda)$ be a power of two. The scheme is given below.

- $\Pi.\mathsf{Prove}^{\mathsf{H}_{\mathsf{bb}}}(\mathsf{crs}, x, w)$: on input crs, $x = (c, u_1, g_1)$ and $w = (m, s)$, decomposes $m = \sum_{i=1}^{\ell} m_i B^{i-1}$, $s = \sum_{i=1}^{\ell} s_i B^{i-1}$, and computes $R_i = \widehat{g}^{r_i}, E_i = \widehat{g}^{e_i}\widehat{\mathsf{pp}}_i^{r_i}$ for $i \in [2\ell]$, where $\boldsymbol{e} = (m_1, \ldots, m_\ell, s_1, \ldots, s_\ell)$ and $r_i \leftarrow \mathbb{Z}_p$. Then, sets $t_m \leftarrow \sum_{i=1}^{\ell} r_i B^{i-1}$ and $t_s \leftarrow \sum_{i=1}^{\ell} r_{i+\ell} B^{i-1}$, and computes

$$\pi_0 \leftarrow \Pi_{\mathsf{rp}}.\mathsf{Prove}^{\mathsf{H}_{\mathsf{rp}}}(\mathsf{crs}, x_0, w_0),$$

  for statement $x_0 = (B, (E_i)_{i\in[2\ell]})$ and witness $w_0 = ((e_i, r_i)_{i\in[2\ell]})$, and

$$\begin{aligned}
&(\alpha, \mathsf{st}) \leftarrow \Sigma_{\mathsf{ped}}.\mathsf{Init}(x_1, w_1),\\
&\beta \leftarrow \mathsf{H}_\beta(x_1, \alpha),\\
&\gamma \leftarrow \Sigma_{\mathsf{ped}}.\mathsf{Resp}(x_1, \mathsf{st}, \beta),\\
&\pi_1 \leftarrow (\alpha, \beta, \gamma),
\end{aligned}$$

  for statement $x_1 = (c, u_1, g_1, \widehat{g}, \widehat{\mathsf{pp}}, (E_i, R_i)_{i\in[2\ell]}, B)$ and witness $w_1 = (m, s, (e_i, r_i)_{i\in[2\ell]}, t_m, t_s)$. Outputs $\pi = (\pi_0, \pi_1, (E_i, R_i)_{i\in[2\ell]})$.
- $\Pi.\mathsf{Verify}^{\mathsf{H}_{\mathsf{bb}}}(\mathsf{crs}, x, \pi)$: on input crs, $x = (c, u_1, g_1)$ and $\pi = (\pi_0, \pi_1, (E_i, R_i)_{i\in[2\ell]})$, checks

$$\begin{aligned}
&\Pi_{\mathsf{rp}}.\mathsf{Verify}^{\mathsf{H}_{\mathsf{rp}}}(\mathsf{crs}, x_0, \pi_0) = 1,\\
&\mathsf{H}_\beta(x_0, \alpha) = \beta' \ \wedge \ \beta = \beta',\\
&\Sigma_{\mathsf{ped}}.\mathsf{Verify}(x_1, \alpha, \beta, \gamma) = 1,
\end{aligned}$$

  where $\pi_1 = (\alpha, \beta, \gamma)$ and x_0, x_1 are defined as above, and outputs 1 iff all checks succeed.

We show that Π is correct, zero-knowledge under the DDH assumption and online-extractable under the DLOG assumption. Correctness follows immediately from the correctness of Π_{rp} and Σ_{ped}. Also, zero-knowledge is easy to show via the hiding property of ElGamal commitments, the zero-knowledge property of Π_{rp} (cf. theorem 17) and the HVZK and high min-entropy property of Σ_{ped} (cf. theorems 14 and 16).

The proof for multi-proof extractability is more intricate. Roughly, the extractor embeds a trapdoor td for the commitment scheme in the crs. Then, given a statement-proof pair (x, π) with $x = (c, u_1, g_1)$ and $\pi = (\pi_0, \pi_1, (E_i, R_i)_{i\in[2\ell]})$, it decrypts the witnesses $(e_i)_i$ from the ElGamal commitment $(E_i, R_i)_i$ and tries to check if the extracted witness reconstructs to a witness in the relation R_{bb}. We expect that this is possible, as the range proof guarantees that the committed values are short and Σ_{ped} proves the linear relations in the exponents.

For the sake of exposition, below we only consider extracting from a single pair $(x, \pi) \leftarrow \mathcal{A}(\mathsf{crs})$ generated by some adversary $\mathcal{A}$. The argument generalizes to Q_S pairs in a straightforward manner. Note that (x, π) defines statement-proof pairs (x_0, π_1) for Π_{rp} and (x_1, π_1) for Σ_{ped} as in $\Pi.\mathsf{Verify}$.

For the sake of contradiction, let us assume that extraction fails. We first try to extract a witness $w_0 = (e_i', r_i')_i$ from π_0 via the knowledge extractor of Π_{rp}, and a witness $w_1 = (m, s, (e_i, r_i)_i)$ from π_1 from two related transcripts obtained via rewinding $\mathcal{A}$. Here, it is important that $\mathcal{A}$ is run with the same random tape $\mathsf{coin}_{\mathcal{A}}$ for both extractions to guarantee that the statements x_0 and x_1 share the commitments $(E_i)_i$. For now, let us assume that both extractions succeed, i.e. $(x_0, w_0) \in \mathsf{R}_{\mathsf{rp}}$ and $(x_1, w_1) \in \mathsf{R}_{\mathsf{ped}}$. Assuming the soundness of Π_{rp}, we have $e_i' \in [0, B-1]$. Moreover, assuming the soundness of the non-interactive Σ_{ped}, the extracted $(e_i)_i$ form the B-ary decomposition of a valid opening of c. Then, under the assumption that extraction fails, we must have $e_i' \neq e_i$ for some i. However, this breaks the binding property of the Pedersen commitment implicitly defined by the ElGamal commitments. In particular, we found a DLOG relation for the tuple $(\widehat{g}, \widehat{\mathsf{pp}}_i)$. Note that

[16] Note that as in section 6.2, we can see H_{bb} as a single random oracle mapping into $\mathbb{Z}_p$ (to fit our definition). For readability, we allow H_{bb} to be a tuple of random oracles in the security proof.

while the extracted DLOG relation is a trapdoor information td the extractor uses to extract the witnesses, this will not be an issue since we do not need td to analyze the success probability of the adversary.

It remains to show that extraction of w_0 and w_1 succeeds. Recall that we assumed that the extraction of w_0 and w_1 succeeds simultaneously, even if we initially run $\mathcal{A}$ on a shared random coin. But these events are dependent, and applying adaptive knowledge soundness of Π_{rp} and a general forking lemma independently is not sufficient. Instead, we first extract w_0 with the extractor of Π_{rp}. This step has a high success probability due to knowledge soundness of Π_{rp}. Then, we define a specialized forking algorithm that first runs $\mathcal{A}$ on the same randomness (and same initial random oracle choices), and then rewinds $\mathcal{A}$ to obtain related transcripts. Finally, a careful non-black box analysis of the forking algorithm, similar to [73], allows us to conclude that the algorithm succeeds in finding two related transcripts.

Theorem 18. *The* NIZK Π *is correct.*

Proof. By construction, it holds that $(x_0, w_0) \in \mathsf{R}_{\mathsf{rp}}$. Similarly, we have $(x_1, w_1) \in \mathsf{R}_{\mathsf{ped}}$, as

$$\begin{aligned}\prod_{i\in[\ell]} E_i^{B^{i-1}} &= \prod_{i\in[\ell]} (\widehat{g}^{e_i}\widehat{\mathsf{pp}}_i^{r_i})^{B^{i-1}} \\ &= \widehat{g}^{\sum_{i=1}^{\ell} m_i B^{i-1}} \cdot \widehat{\mathsf{pp}}^{\sum_{i=1}^{\ell} r_i B^{i-1}} \\ &= \widehat{g}^{m} \cdot \widehat{\mathsf{pp}}^{t_m},\end{aligned}$$

and similarly $\prod_{i\in[\ell]} E_{i+\ell}^{B^{i-1}} = \widehat{g}^{s} \cdot \widehat{\mathsf{pp}}^{t_s}$.

Theorem 19. *The* NIZK Π *is zero-knowledge under the zero-knowledge property of* Π_{rp}*, the HVZK and high min-entropy property of* Σ_{ped}*, and under the* DDH *assumption in* $\widehat{\mathbb{G}}$*.*

Proof. Denote by $\mathsf{Sim}_0 = (\mathsf{Sim}_{\mathsf{H}_{\mathsf{rp}}}, \mathsf{Sim}_{\pi_0})$ the simulator of Π_{rp} and by $\Sigma_{\mathsf{ped}}.\mathsf{Sim}_1$ the simulator of Σ_{ped}. We define the simulator $\mathsf{Sim} = (\mathsf{Sim}_{\mathsf{H}_{\mathsf{bb}}}, \mathsf{Sim}_{\pi})$ of Π as follows. $\mathsf{Sim}_{\mathsf{H}_{\mathsf{bb}}}$ prepares an empty list L. For every new query q to the random oracle H_β, it returns a random element $\beta \leftarrow \mathbb{Z}_p$ and stores (q, β) in L, and answers old queries consistently via L. For every query q to the random oracle H_{rp}, it returns $\mathsf{Sim}_{\mathsf{H}_{\mathsf{rp}}}(q)$. Now, for each proof query $(x, w) \in \mathsf{R}_{\mathsf{bb}}$, Sim_{π} sets $(E_i, R_i) \leftarrow (\widehat{\mathsf{pp}}^{r_i}, \widehat{g}^{r_i})$ for random $r_i \leftarrow \mathbb{Z}_p$ and $i \in [2\ell]$, and prepares the two statements x_0, x_1 as in the real protocol. It then simulates $\pi_0 \leftarrow \mathsf{Sim}_{\pi_0}(\mathsf{crs}_0, x_0)$ and $\pi_0 = (\alpha, \beta, \gamma) \leftarrow \Sigma_{\mathsf{ped}}.\mathsf{Sim}_1(x_1, \beta)$, where $\beta \leftarrow \mathbb{Z}_p$. If H_β was already queried on input (x_1, α), then Sim_{π} outputs $\perp$. Otherwise, Sim_{π} outputs the simulated proof $\pi = (\pi_0, \pi_1, (E_i, R_i)_{i\in[2\ell]})$, and $\mathsf{Sim}_{\mathsf{H}_{\mathsf{bb}}}$ stores $((x_1, \alpha), \beta_1)$ in the list L.

Let $\mathcal{A}$ be PPT adversary on the zero-knowledge property of Π and let $Q_{\mathsf{H}_{\mathsf{rp}}}, Q_{\mathsf{H}_\beta}, Q_S$ denote the number of $\mathsf{H}_{\mathsf{rp}}, \mathsf{H}_\beta, \mathsf{Sim}_{\pi}$ queries, respectively. Without loss of generality, we assume that $\mathcal{A}$ never queries H_{rp} and H_β twice on the same input. We define the following hybrids and denote by $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_i}(\lambda)$ the probability that $\mathcal{A}$ outputs 1 in Hybrid i.

- Hybrid 0 is identical to real game, where proofs are honestly generated. Specifically, the proof oracle outputs on input (x, w) the value $\perp$ if $(x, w) \notin \mathsf{R}_{\mathsf{bb}}$, and else the value $\Pi.\mathsf{Prove}^{\mathsf{H}_{\mathsf{bb}}}(\mathsf{crs}, x, w)$. By definition, $\mathcal{A}$ outputs 1 with probability $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_0}(\lambda)$.
- Hybrid 1 is identical to Hybrid 0, except we simulate the proofs π_0 using the simulator $\mathsf{Sim}_0 = (\mathsf{Sim}_{\mathsf{H}_{\mathsf{rp}}}, \mathsf{Sim}_{\pi_0})$ of Π_{rp}. In more detail, for every query $(x, w) \in \mathsf{R}_{\mathsf{bb}}$, the challenger computes $(E_i, R_i)_{i\in[2\ell]}$ as before, but sets $\pi_0 \leftarrow \mathsf{Sim}_{\pi_0}(\mathsf{crs}, x_0)$ for $x_0 = (B, (E_i)_{i\in[2\ell]})$. The simulator still generates the proof π_1 honestly, and outputs $\pi = (\pi_0, \pi_1, (E_i, R_i)_{i\in[2\ell]})$. Similarly, for every query q to H_{rp}, outputs $\mathsf{Sim}_{\mathsf{H}_{\mathsf{rp}}}(q)$.
  We can construct an adversary $\mathcal{B}_{\Pi_{\mathsf{rp}}}$ against the zero-knowledge property of Π_{rp} with advantage $\mathsf{Adv}_{\mathcal{B}_{\Pi_{\mathsf{rp}}}}^{\mathrm{zk}} \geq |\mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_0}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_1}(\lambda)|$. Concretely, $\mathcal{B}_{\Pi_{\mathsf{rp}}}$ challenges $\mathcal{A}$ and uses the provided oracles to generate the proofs π_0 and answer the H_{rp} queries. In the end, $\mathcal{B}_{\Pi_{\mathsf{rp}}}$ outputs the bit b received from $\mathcal{A}$. If the provided oracle generate real proofs, then $\mathcal{B}_{\Pi_{\mathsf{rp}}}$ simulates Hybrid 0 to $\mathcal{A}$, else it simulates Hybrid 1, and thus we have

$$|\mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_0}(\lambda) - \mathsf{Adv}_{\mathcal{A}}^{\mathrm{H}_1}(\lambda)| \leq \mathsf{Adv}_{\mathcal{B}_{\Pi_{\mathsf{rp}}}}^{\mathrm{zk}}.$$

– Hybrid 2 is the same as Hybrid 1, except for every query $(x, w) \in \mathsf{R}_{\mathsf{bb}}$, the simulator aborts if the random oracle H_β was already queried on its input when generating π_1. Concretely, after generating π_0 as in Hybrid 1, the simulator sets $(\alpha, \mathsf{st}) \leftarrow \Sigma_{\mathsf{ped}}.\mathsf{Init}(x_1, w_1)$ for (x_1, w_1) defined as before. It then draws $\beta \leftarrow \mathbb{Z}_p$ and finishes the generation of $\pi_1 = (\alpha, \beta, \gamma)$ via $\gamma \leftarrow \Sigma_{\mathsf{ped}}.\mathsf{Resp}(x_1, \mathsf{st}, \beta)$. Finally, it checks if (x_1, α) was already queried to H_β and aborts if so. Otherwise, it programs $\mathsf{H}_\beta(x_1, \alpha) \leftarrow \beta$.
Hybrids 1 and 2 differ only when the game aborts. Due to the high min-entropy of Σ_{ped}, the probability that the random oracle is already defined on input (x_1, α) is bounded by $Q_{\mathsf{H}_\beta} \cdot \mathrm{negl}(\lambda)$. Further, there are at most Q_S queries to Sim_π, and because $Q_S Q_{\mathsf{H}_\beta} \cdot \mathrm{negl}(\lambda) = \mathrm{negl}(\lambda)$, we have

$$|\mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda)| \leq \mathrm{negl}(\lambda).$$

– Hybrid 3 is the same as Hybrid 2, except for every query $(x, w) \in \mathsf{R}_{\mathsf{bb}}$, the proofs π_1 are simulated without the witness w_1. That is, the simulator generates a simulated proof $\pi_1 = (\alpha, \beta, \gamma)$ by setting $\beta \leftarrow \mathbb{Z}_p$ and running $(\alpha, \gamma) \leftarrow \Sigma_{\mathsf{ped}}.\mathsf{Sim}(x_1, \beta)$, and programs H_β accordingly.
We can construct an adversary $\mathcal{B}_{\Sigma_{\mathsf{ped}}}$ against the HVZK property of Σ_{ped}. Concretely, for every $(x, w) \in \mathsf{R}_{\mathsf{bb}}$, $\mathcal{B}_{\Sigma_{\mathsf{ped}}}$ obtains the transcript (α, β, γ). If (x_1, α) was already queried to H_β, then it aborts as in the previous game. Otherwise, it uses $\pi_1 = (\alpha, \beta, \gamma)$ to generate π as in Hybrid 2. If $\mathcal{B}_{\Sigma_{\mathsf{ped}}}$ receives simulated proofs, the game is distributed as in Hybrid 3, else it is distributed as in Hybrid 2. Consequently, we have

$$|\mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda)| \leq Q_S \cdot \mathsf{Adv}^{\mathrm{hvzk}}_{\mathcal{B}_{\Sigma_{\mathsf{ped}}}}(\lambda).$$

– Hybrid 4 is the same as Hybrid 3, except for every query $(x, w) \in \mathsf{R}_{\mathsf{bb}}$, the commitments $(E_i, R_i)_{i \in [2\ell]}$ are commitments to 0.
As the openings of $(E_i, R_i)_{i \in [2\ell]}$ are not required anymore for generating π_0 and π_1, we can construct an adversary $\mathcal{B}_{\mathsf{DDH}}$ against the hiding property of the ElGamal commitments (which holds under DDH). We thus have

$$|\mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{H}_4}_{\mathcal{A}}(\lambda)| \leq 2\ell \cdot \mathsf{Adv}^{\mathrm{ddh}}_{\mathcal{B}_{\mathsf{DDH}}}(\lambda)$$

Note that the description of the simulator in Hybrid 4 is identical to $\mathsf{Sim} = (\mathsf{Sim}_{\mathsf{H}_{\mathsf{bb}}}, \mathsf{Sim}_\pi)$. Collecting the above bounds yields the statement.

Theorem 20. *The* NIZK Π *is multi-online extractable under adaptive knowledge soundness of* Π_{rp}*, the 2-special soundness property of* Σ_{ped}*, and under the* DLOG *assumption in* $\widehat{\mathbb{G}}$*.*

Proof. The simulator and extractor are given below. Roughly, the extractor extracts the B-ary decomposition of (m, s) from the commitments $(E_i, R_i)_{i \in [2\ell]}$, then recomputes and outputs (m, s) if $c = u_1^m g_1^s$. If the reconstruction of (m, s) fails or the supplied proof is invalid, it outputs $\perp$.

– $\mathsf{SimCRS}(1^\lambda)$: sets $\overline{\mathsf{crs}} = (\widehat{g}, \widehat{\mathsf{pp}}, (\widehat{g}_i)_{i \in [\ell_{\mathsf{rp}}]})$ with $\widehat{g}, \widehat{g}_i \leftarrow \widehat{\mathbb{G}}, \mathsf{td} \leftarrow \mathbb{Z}_p$ and $\widehat{\mathsf{pp}} \leftarrow \widehat{g}^{\mathsf{td}}$, and outputs $(\overline{\mathsf{crs}}, \mathsf{td})$,
– $\mathsf{Ext}(\overline{\mathsf{crs}}, \mathsf{td}, x, \pi)$: on input $\overline{\mathsf{crs}}$, trapdoor td, and proof $\pi = (\pi_0, \pi_1, (E_i, R_i)_{i \in [2\ell]})$ for statement $x = (c, u_1, g_1)$, the extractor first checks if π is valid via $\Pi.\mathsf{Verify}^{\mathsf{H}_{\mathsf{bb}}}(\overline{\mathsf{crs}}, x, \pi) = 1$. Then, it sets $F_i \leftarrow E_i \cdot R_i^{-\mathsf{td}}$ and checks that there is some $e_i \in [0, B-1]$ such that $F_i = \widehat{g}^{e_i}$. If so, it sets $m = \sum_{i=1}^{\ell} e_i B^{i-1}$ and $s = \sum_{i=1}^{\ell} e_{i+\ell} B^{i-1}$, checks $c = u_1^m \cdot g_1^s$, and finally outputs $w = (m, s)$. If any of the above checks fails, it outputs $\perp$.

Note that both SimCRS and Ext are PPT because the exponent of F_i can be brute-forced in polynomial time, as $B = \mathsf{poly}(\lambda)$. Also, if Ext does not output $\perp$, Ext is guaranteed to output w such that $(x, w) \in \mathsf{R}_{\mathsf{bb}}$ by construction.

We show that Π has CRS indistinguishability with the simulator SimCRS. Observe that $\mathsf{SimCRS}(1^\lambda)$ and the random variable $\mathsf{crs} \leftarrow (\widehat{g}, \widehat{\mathsf{pp}}, (\widehat{g}_i)_{i \in [\ell_{\mathsf{rp}}]})$ with $\widehat{g}, \widehat{\mathsf{pp}}, \widehat{g}_i \leftarrow \widehat{\mathbb{G}}$ are identically distributed. It follows that for any PPT adversary $\mathcal{A}$, we have $\mathsf{Adv}^{\mathrm{crs}}_{\mathcal{A}}(\lambda) = 0$.

It remains to show that Π is (multi-)online-extractable with the extractor Ext. We denote by H_0 the random oracle H_{rp} and by H_1 the random oracle H_β. Let $\mathcal{A}$ be an adversary making at most Q_0, Q_1 queries to $\mathsf{H}_0, \mathsf{H}_1$ respectively. For $(\overline{\mathsf{crs}}, \mathsf{td}) \leftarrow \mathsf{SimCRS}(1^\lambda), \{(x_i, \pi_i)\}_{i \in [Q_S]} \leftarrow \mathcal{A}^{\mathsf{H}_{\mathsf{bb}}}(\overline{\mathsf{crs}})$, we denote by $w_i \leftarrow \mathsf{Ext}(\overline{\mathsf{crs}}, \mathsf{td}, x_i, \pi_i)$ the extracted witness from the i-th proof. We define the following events.

- Ver is the event that all statement-proof pairs verify correctly, i.e. for all $i \in [Q_S]$ it holds that $\Pi.\mathsf{Verify}^{\mathsf{H}_{\mathsf{bb}}}(\overline{\mathsf{crs}}, x_i, \pi_i) = 1$.
- Fail_i is the event that the extractor fails for input $(\overline{\mathsf{crs}}, \mathsf{td}, x_i, \pi_i)$ on a valid proof, i.e. we have $w_i = \bot$ but $\Pi.\mathsf{Verify}^{\mathsf{H}_{\mathsf{bb}}}(\overline{\mathsf{crs}}, x_i, \pi_i) = 1$.
- Fail is the event that the extractor fails for some valid proof (x_i, π_i), i.e. there exists some $i \in [Q_S]$ such that the event Fail_i occurs.

With this notation, let us assume that $\Pr[\mathsf{Ver}] \geq \mu(\lambda)$ for some $\mu(\lambda)$. Under the DLOG assumption in $\widehat{\mathbb{G}}$, we show in Lemma 4 that $\Pr[\mathsf{Fail}_i] = \mathrm{negl}(\lambda)$. Thus:

$$\Pr[\mathsf{Fail}] = \Pr[\exists i \in [Q_S], \mathsf{Fail}_i] \leq \sum_{i \in [Q_S]} \Pr[\mathsf{Fail}_i] = \mathrm{negl}(\lambda),$$

Thus, we have as desired:

$$\begin{aligned} &\Pr[\mathsf{Ver} \wedge \forall i \in [Q_S], (x_i, w_i) \in \mathsf{R}_{\mathsf{bb}}] = \Pr[\mathsf{Ver} \wedge \neg \mathsf{Fail}] \\ &= \Pr[\mathsf{Ver}] - \Pr[\mathsf{Ver} \wedge \mathsf{Fail}] \geq \mu(\lambda) - \Pr[\mathsf{Fail}] \\ &\geq \mu(\lambda) - \mathrm{negl}(\lambda). \end{aligned}$$

It remains to show Lemma 4. Recall that for all PPT adversaries $\mathcal{A}_{\mathsf{dl}}$, it holds that

$$\Pr[w^* \leftarrow \mathcal{A}_{\mathsf{dl}}(\mathsf{crs}) : (\mathsf{crs}, w^*) \in \mathsf{R}_{\mathsf{dlog}}] = \mathrm{negl}(\lambda) \tag{3}$$

under the DLOG assumption, where the probability is over the randomness of crs and the random coins of $\mathcal{A}_{\mathsf{dl}}$. Recall that the simulated $\overline{\mathsf{crs}}$ is distributed identically to a random crs, and as the ElGamal trapdoor td is never provided to $\mathcal{A}$, $\mathcal{A}$'s output is identically distributed on input crs and $\overline{\mathsf{crs}}$. Thus, we only need to analyze the probability of the event Fail_i for a random crs (without known trapdoor td). This is important because td itself provides a non-trivial DLOG relation, but this is the hard problem we want to solve using $\mathcal{A}$.

Lemma 4. *For some fixed $i \in [Q_S]$, we have $\Pr[\mathsf{Fail}_i] = \mathrm{negl}(\lambda)$ under the DLOG assumption.*

Proof. Assume that $\Pr[\mathsf{Fail}_i]$ is non-negligible. We construct an adversary $\mathcal{A}_{\mathsf{dl}}$ that on input a random crs finds a DLOG relation w^* in crs with non-negligible probability in polynomial time. When $\mathcal{A}$ on input crs outputs statement-proof pairs $\{(x'_j, \pi'_j)\}_{j \in [Q_S]}$ for NIZK Π, the i-th pair (x'_i, π'_i) contains a range proof π_0 for Π_{rp} (resp. a Fiat-Shamir proof π_1 for Σ_{ped}), and their statements x_0 (resp. x_1) can be recomputed given (x'_i, π'_i) as in $\Pi.\mathsf{Verify}$. In the analysis, we are interested in these proof-statement pairs and wish to extract the witness for both statements x_0 and x_1 simultaneously. We proceed as follows.

First, we define two wrapper algorithms $\mathcal{B}_0, \mathcal{B}_1$ of $\mathcal{A}$ that output the i-th statement-proof pair $(x_0, \pi_0), (x_1, \pi_1)$ output by $\mathcal{A}$ for $\Pi_{\mathsf{rp}}, \Sigma_{\mathsf{ped}}$, respectively. $\mathcal{B}_0$ and $\mathcal{B}_1$ are defined to run $\mathcal{A}$ with identical random $\mathsf{coin}_{\mathcal{A}}$ and random oracle outputs to ensure the statements x_0 and x_1 are consistent, i.e., contain the same commitments $(E_j)_j$. These algorithms will later be used to define our DLOG solver $\mathcal{A}_{\mathsf{dl}}$.

Description of Wrapper Algorithm $\mathcal{B}_b$. We denote by $\mathsf{coin}_{\mathcal{A}}$ the random coin of $\mathcal{A}$, and by $\vec{h}_0 = (\widehat{\beta}_{0,j})_{j \in [Q_0]} \in \mathbb{Z}_p^{Q_0}, \vec{h}_1 = (\widehat{\beta}_{1,j})_{j \in [Q_1]} \in \mathbb{Z}_p^{Q_1}$ the outputs of $\mathsf{H}_0, \mathsf{H}_1$, respectively. Note that for fixed $(\mathsf{crs}, \mathsf{coin}_{\mathcal{A}}, \vec{h}_0, \vec{h}_1)$, calling $\mathcal{A}$ is deterministic and the statement-proof pairs $\{(x'_j, \pi'_j)\}_{j \in [Q_S]} = \mathcal{A}^{\mathsf{H}_{\mathsf{bb}}}(\mathsf{crs}; \mathsf{coin}_{\mathcal{A}})$ are uniquely defined, where H_{bb} queries are answered via $\vec{h}_0$ and $\vec{h}_1$. We then define $\mathcal{B}_b$ as an algorithm that has oracle access to H_b (and a variant with fixed H_b outputs $\vec{h}_b$) as follows:

$\mathcal{B}_b^{\mathsf{H}_b}(\mathsf{crs}; \mathsf{coin}_b)$: On input $\mathsf{crs} = (\widehat{g}, \widehat{\mathsf{pp}}, (\widehat{g}_i)_{i \in [\ell_{\mathsf{rp}}]})$ and $\mathsf{coin}_b = (\mathsf{coin}_{\mathcal{A}}, \vec{h}_{1-b})$, $\mathcal{B}_b$ runs $\mathcal{A}$ on input crs with fixed randomness $\mathsf{coin}_{\mathcal{A}}$, where for the j-th query to H_{1-b}, it outputs the j-th value $\widehat{\beta}_{1-b,j}$ in $\vec{h}_{1-b}$, and it simulates H_b with the provided oracle. After obtaining $\{(x'_j, \pi'_j)\}_{j \in [Q_S]} = \mathcal{A}(\mathsf{crs}, \mathsf{coin}_{\mathcal{A}})$ from $\mathcal{A}$, it checks if $\Pi.\mathsf{Verify}^{\mathsf{H}_{\mathsf{bb}}}(\mathsf{crs}, x'_i, \pi'_i) = 1$, i.e., the i-th proof verifies correctly. Then, parses $x'_i = (c, u_1, g_1)$ and $\pi'_i = (\pi_0, \pi_1, (E_j, R_j)_{j \in [2\ell]})$, and sets $x_0 = (B, (E_j)_{j \in [2\ell]})$ and $x_1 = (c, u_1, g_1, \widehat{g}, \widehat{\mathsf{pp}}, (E_j, R_j)_{j \in [2\ell]}, B)$. If any check fails, outputs $(\bot, \bot)$. Otherwise, for

$b = 0$, outputs (x_0, π_0). For $b = 1$, parses $\pi_1 = (\alpha, \beta, \gamma)$, and looks for the index I such that $\beta = \widehat{\beta}_{1,I} = \mathsf{H}_1(x_1, \alpha)$, and finally outputs (I, Λ) with $\Lambda = (x_1, \alpha, \beta, \gamma)$. Note that without loss of generality, the index I is well defined, as guessing β correctly without querying H_1 on input (x_1, α) happens with probability at most $1/p = \mathrm{negl}(\lambda)$.

$\mathcal{B}_b(\mathsf{crs}, \vec{h}_b; \mathsf{coin}_b)$: runs $\mathcal{B}_b^{\mathsf{H}_b}(\mathsf{crs}; \mathsf{coin}_b)$, where the j-th query to H_b is answered with the j-th value $\widehat{\beta}_{b,j}$ of $\vec{h}_b$. Note that $\mathcal{B}_b$ is deterministic on input $(\mathsf{crs}, \vec{h}_b; \mathsf{coin}_b)$.

Description of Forking Algorithm $\mathsf{F}_{\mathcal{B}_1}$. We now define a variant $\mathsf{F}_{\mathcal{B}_1}$ of the standard forking algorithm that rewinds $\mathcal{B}_1$ until a related transcript is found. Concretely, $\mathsf{F}_{\mathcal{B}_1}$ takes as input $(\mathsf{crs}, \mathsf{coin}_1, \vec{h}_1)$, and invokes $\mathcal{B}_1$ internally as depicted in algorithm 2. Note that the standard forking algorithm chooses coin_1 and initial hash values $\vec{h}_1$ at random, whereas $\mathsf{F}_{\mathcal{B}_1}$ receives some fixed initial choice of coin_1 and $\vec{h}_1$. In $\mathcal{A}_{\mathsf{dl}}$, we will initialize the input of $\mathsf{F}_{\mathcal{B}_1}$ with the choices made by the extractor Ext_0 of Π_{rp}.

We expect that after at most $T = \mathsf{poly}(\lambda)$ calls to $\mathcal{B}_1$, the forking algorithm is successful with non-negligible probability, i.e., it outputs some non-$\perp$. Note that for inputs drawn independently and uniformly at random, the classical analysis of the forking algorithm yields the desired result. But here, the inputs are conditioned on the event that the extractor Ext_0 is successful on input (x_0, π_0), and we cannot apply the classical result directly. Thus, we analyze the success probability of $\mathsf{F}_{\mathcal{B}_1}$ later. Nevertheless, if $\mathsf{F}_{\mathcal{B}_1}$ is successful, it outputs related transcripts for the statement x_1, which is fixed via the initial run of $\mathcal{A}$ in $\mathcal{B}_1$.

Algorithm 2 Description of the forking algorithm $\mathsf{F}_{\mathcal{B}_1}(\mathsf{crs}, \mathsf{coin}_1, \vec{h}_1)$

$(I, \Lambda) \leftarrow \mathcal{B}_1(\mathsf{crs}, \vec{h}_1; \mathsf{coin}_1)$
if $\Lambda = \perp$ **then**
  return $\perp$ ▷ Return fail.
for $c \in [T]$ **do**
  $\vec{h}_{1,\geq I}^{(c)} \leftarrow \mathbb{Z}_p^{Q_1 - I + 1}$
  $\vec{h}_1^{(c)} := \vec{h}_{1,<I} \| \vec{h}_{1,\geq I}^{(c)}$
  $(I^{(c)}, \Lambda^{(c)}) \leftarrow \mathcal{B}_1(\mathsf{crs}, \vec{h}_1^{(c)}; \mathsf{coin}_1)$
  if $I^{(c)} = I$ **then** ▷ Found related transcript.
    return $(\Lambda, \Lambda^{(c)})$
return $\perp$ ▷ Return fail.

Description of the DLOG *Adversary* $\mathcal{A}_{\mathsf{dl}}$. We are now ready to define $\mathcal{A}_{\mathsf{dl}}$. Roughly, $\mathcal{A}_{\mathsf{dl}}$ samples inputs (incl. randomness) for both wrapping algorithms $(\mathcal{B}_0, \mathcal{B}_1)$ such that they output proofs (π_0, π_1) for statements (x_0, x_1) with shared commitments $(E_j)_{j \in [2\ell]}$. Then, it invokes the extractor Ext_0 of Π_{rp} on (x_0, π_0) to extract a witness w_0 for the i-th proof output by $\mathcal{A}$. Next, it rewinds $\mathcal{B}_0$ with matching initial inputs via the forking algorithm $\mathsf{F}_{\mathcal{B}_1}$ to obtain a related transcript for (x_1, π_1), and extracts witness w_1 via 2-special soundness. If both extractions were successful, $\mathcal{A}_{\mathsf{dl}}$ can compute a DLOG relation in crs conditioned on event Fail_i, as x_0 and x_1 share the commitment $(E_j)_{j \in [2\ell]}$ and the parameter B by construction. We denote by Ext_0 the extractor of Π_{rp} (cf. Definition 20) and by Ext_1 the extractor of Σ_{ped} (cf. Definition 16). We now describe $\mathcal{A}_{\mathsf{dl}}$.

$\mathcal{A}_{\mathsf{dl}}(\mathsf{crs})$: On input crs, $\mathcal{A}_{\mathsf{dl}}$ prepares a list $\vec{h}_b \leftarrow \mathbb{Z}_p^{Q_b}$ of initial responses for H_b queries, where $b \in \{0, 1\}$. Then, she draws some random $\mathsf{coin}_{\mathcal{A}}$ for $\mathcal{A}$, and initializes for $b \in \{0, 1\}$ the randomness of $\mathcal{B}_b$ via $\mathsf{coin}_b = (\mathsf{coin}_{\mathcal{A}}, \vec{h}_{1-b})$. Then, $\mathcal{A}_{\mathsf{dl}}$ runs $(x_0, \pi_0) \leftarrow \mathcal{B}_0(\mathsf{crs}, \vec{h}_0; \mathsf{coin}_0)$, extracts a witness $w_0 \leftarrow \mathsf{Ext}_0(\mathsf{crs}, x_0, \pi_0, \mathsf{coin}_0, \vec{h}_0)$, and checks $(x_0, w_0) \in \mathsf{R}_{\mathsf{lax}}$. Next, she runs $R \leftarrow \mathsf{F}_{\mathcal{B}_1}(\mathsf{crs}, \mathsf{coin}_1, \vec{h}_1)$, checks $R \neq \perp$, and parses $R = \{(x_1, \pi_1), (x_1, \pi_2)\}$ with $\pi_1 = (\alpha, \beta, \gamma), \pi_2 = (\alpha, \beta', \gamma')$ and $\beta \neq \beta$. Then, she extracts $w_1 \leftarrow \mathsf{Ext}_1(x_1, \pi_1, \pi_2)$. Note that by construction, if $R \neq \perp$, we have $(x_1, w_1) \in \mathsf{R}_{\mathsf{ped}}$ under 2-special soundness of Σ_{ped} (see Lemma 6 for more details). Next, it parses the statements $x_0 = (B', (E'_j)_{j \in [2\ell]})$ and $x_1 = (c, u_1, g_1, \widehat{g}, \widehat{\mathsf{pp}}, (E_j, R_j)_{j \in [2\ell]}, B)$. She outputs $\perp$ if any check fails. As the initial run of $\mathcal{A}$ in $\mathcal{B}_0(\mathsf{crs}, \vec{h}_0; \mathsf{coin}_0)$ and $\mathsf{F}_{\mathcal{B}_1}(\mathsf{crs}, \mathsf{coin}_1, \vec{h}_1)$ are identical, we have $B = B'$ and $E'_j = E_j$ for all $j \in [2\ell]$. Next, $\mathcal{A}_{\mathsf{dl}}$ computes a DLOG relation w^* as follows:

First, if $(\mathsf{crs}, w_0) \in \mathsf{R}_{\mathsf{dlog}}$, sets $w^* = w_0$, else parses $w_0 = ((e'_j, r'_j)_{j\in[2\ell]})$ and $w_1 = (m, s, (e_j, r_j)_{j\in[2\ell]}, t_m, t_s)$. Next, if there is some $j \in [2\ell]$ with $e'_j \neq e_j$, the binding property of the Pedersen commitment E_j is broken, and $w^* \leftarrow (e_j - e'_j)/(r'_j - r_j) \bmod p$ yields a non-trivial DLOG relation in crs, as $\widehat{\mathsf{pp}} = \widehat{g}^{w^*}$. Further, if $m \neq \sum_{i=1}^{\ell} e_i B^{i-1}$, then $w^* \leftarrow (m - \sum_{i=1}^{\ell} m_i B^{i-1})/(\sum_{i=1}^{\ell} r_i B^{i-1} - t_m)$ yields a DLOG relation due to the following:

$$\begin{aligned}
&\prod_{i\in[\ell]} E_i^{B^{i-1}} = \widehat{g}^m \cdot \widehat{\mathsf{pp}}^{t_m} \\
\Longrightarrow &\prod_{i\in[\ell]} (\widehat{g}^{e_i} \widehat{\mathsf{pp}}^{r_i})^{B^{i-1}} = \widehat{g}^m \cdot \widehat{\mathsf{pp}}^{t_m} \\
\Longrightarrow &\widehat{g}^{\sum_{i\in[\ell]} e_i B^{i-1}} \widehat{\mathsf{pp}}^{\sum_{i\in[\ell]} r_i B^{i-1}} = \widehat{g}^m \cdot \widehat{\mathsf{pp}}^{t_m} \\
\Longrightarrow &\widehat{g}^{\sum_{i\in[\ell]} e_i B^{i-1} - m} = \widehat{\mathsf{pp}}^{t_m - \sum_{i\in[\ell]} r_i B^{i-1}} \\
\Longrightarrow &\widehat{g}^{(\sum_{i\in[\ell]} e_i B^{i-1} - m)/(t_m - \sum_{i\in[\ell]} r_i B^{i-1})} = \widehat{\mathsf{pp}},
\end{aligned}$$

where the first equality is due to $(x_1, w_1) \in \mathsf{R}_{\mathsf{ped}}$. A similar calculation shows that $\mathcal{A}_{\mathsf{dl}}$ can compute a DLOG relation w^* if $s \neq \sum_{i=1}^{\ell} e_{i+\ell} B^{i-1}$.
In summary, $\mathcal{A}_{\mathsf{dl}}$ succeeds extracting a DLOG relation w^* if the extraction of both $(x_b, \pi_b)_{b\in\{0,1\}}$ succeeds and the extracted witness reconstructs to a witness *not* in R_{bb}. Otherwise, if either extraction fails or the extracted witness reconstructs to a witness in R_{bb}, she outputs $\perp$.

Analysis of the Success Probability of $\mathcal{A}_{\mathsf{dl}}$. We finally analyze the probability that $\mathcal{A}_{\mathsf{dl}}$ outputs a DLOG relation in crs conditioned on event Fail_i. If the probability is non-negligible, we conclude $\Pr[\mathsf{Fail}_i] = \mathrm{negl}(\lambda)$ as desired under the hardness of DLOG. Below, for simplicity, we omit the subscript and use Fail for Fail_i.

First, notice that conditioned on the event Fail, $\mathcal{A}_{\mathsf{dl}}$ cannot extract a witness that reconstructs to a witness in R_{bb}. Therefore, $\mathcal{A}_{\mathsf{dl}}$ outputs $\perp$ if and only if extraction fails, and on the other hand, when it outputs a non-$\perp$, then this always results in a DLOG relation in crs as desired. Thus, we only need to prove that $\mathcal{A}_{\mathsf{dl}}$ outputs a non-$\perp$ with non-negligible probability — or equivalently, succeeds extracting from both (x_0, π_0) and (x_1, π_1) with non-negligible probability in polynomial time — to conclude the proof.

We first prove the following lemma which states that if event Fail occurs then $\mathcal{A}_{\mathsf{dl}}$ succeeds in extracting a witness from (x_0, π_0) with non-negligible probability. We later analyze the probability that $\mathcal{A}_{\mathsf{dl}}$ further succeeds in extracting a witness from (x_1, π_1).

Lemma 5. *We have $\Pr[(x_0, w_0) \in \mathsf{R}_{\mathsf{lax}} \wedge \mathsf{Fail}] \geq \varepsilon$ under adaptive knowledge soundness of Π_{rp}, where $\varepsilon = (\Pr[\mathsf{Fail}] - \mathrm{negl}(\lambda))/p_{\mathsf{P}}(\lambda, Q_0)$ and p_{P} is the polynomial in definition 20. Here, the probability is taken over the randomness of crs and those used by $\mathcal{A}_{\mathsf{dl}}$.*

Proof. The statement follows from adaptive knowledge soundness of Π_{rp} if we restrict $\mathcal{B}_0$ to only output proofs if extraction of (x'_i, π'_i) fails. As checking this condition requires knowledge of the trapdoor td of crs, we analyze the probability for some $\overline{\mathsf{crs}}$ with known td. The statement follows as crs and $\overline{\mathsf{crs}}$ are identically distributed.

In more detail, we define a wrapper algorithm $\mathcal{B}$ of $\mathcal{B}_0$. $\mathcal{B}$ samples $(\overline{\mathsf{crs}}, \mathsf{td}) \leftarrow \mathsf{SimCRS}(1^\lambda)$ and initializes coin_b and $\vec{h}_0$ as in $\mathcal{A}_{\mathsf{dl}}$. Then, it runs $(x_{\mathcal{B}}, \pi_{\mathcal{B}}) \leftarrow \mathcal{B}_0(\overline{\mathsf{crs}}, \vec{h}_0; \mathsf{coin}_0)$, and outputs $(x_{\mathcal{B}}, \pi_{\mathcal{B}})$ if $\perp = \mathsf{Ext}(\overline{\mathsf{crs}}, \mathsf{td}, x'_i, \pi'_i)$, where (x'_i, π'_i) is the i-th statement-proof pair output by $\mathcal{A}$ in $\mathcal{B}_0$. Note that by definition of $\mathcal{B}_0$, if $(x_{\mathcal{B}}, \pi_{\mathcal{B}}) \neq (\perp, \perp)$, then we also have $\Pi_{\mathsf{rp}}.\mathsf{Verify}^{\mathsf{H}_0}(\overline{\mathsf{crs}}, x_{\mathcal{B}}, \pi_{\mathcal{B}}) = 1$. Thus, under adaptive knowledge soundness of Π_{rp}, we have

$$\Pr[(x_{\mathcal{B}}, w_{\mathcal{B}}) \in \mathsf{R}_{\mathsf{lax}} \wedge \perp = \mathsf{Ext}(\overline{\mathsf{crs}}, \mathsf{td}, x'_i, \pi'_i)] \geq \varepsilon$$

for $w_{\mathcal{B}} \leftarrow \mathsf{Ext}_0(\overline{\mathsf{crs}}, x_{\mathcal{B}}, \pi_{\mathcal{B}}, \mathsf{coin}_0, \vec{h}_0)$. As crs and $\overline{\mathsf{crs}}$ are identically distributed, the pair (x_0, w_0) in $\mathcal{A}_{\mathsf{dl}}$ is identically distributed to the output of $\mathcal{B}$ conditioned on the event Fail. Thus, a quick calculation yields as desired

$$\Pr[(x_0, w_0) \in \mathsf{R}_{\mathsf{lax}} \wedge \mathsf{Fail}] \geq \Pr[(x_{\mathcal{B}}, w_{\mathcal{B}}) \in \mathsf{R}_{\mathsf{lax}} \wedge \perp = \mathsf{Ext}(\overline{\mathsf{crs}}, \mathsf{td}, x'_i, \pi'_i)] \geq \varepsilon.$$

This completes the proof.

It remains to analyze the probability that $\mathcal{A}_{\mathsf{dl}}$ extracts $(x_1, w_1) \in \mathsf{R}_{\mathsf{ped}}$ with non-negligible probability, conditioned on $(x_0, w_0) \in \mathsf{R}_{\mathsf{lax}}$ and Fail. We stress that the two extractions are not independent, as the same random $\mathsf{coin}_{\mathcal{A}}$ and initial hash values $\vec{h}_0, \vec{h}_1$ are used for both extractions, so a non-black-box analysis is required. In Lemma 6, we show that the events $(x_0, w_0) \in \mathsf{R}_{\mathsf{lax}}, (x_1, w_1) \in \mathsf{R}_{\mathsf{ped}}$ and Fail occur after a polynomial number of forking steps in $\mathsf{F}_{\mathcal{B}_1}$ with non-negligible probability. Combining this with lemma 5, we conclude that $\mathcal{A}_{\mathsf{dl}}$ succeeds extracting from both (x_0, π_0) and (x_1, π_1) with non-negligible probability in polynomial time.

Lemma 6. *For $T = 4Q_1/\varepsilon$, we have $\Pr[(x_0, w_0) \in \mathsf{R}_{\mathsf{lax}} \wedge (x_1, w_1) \in \mathsf{R}_{\mathsf{ped}} \wedge \mathsf{Fail}] \geq \frac{\varepsilon}{8} - \mathrm{negl}(\lambda)$, and the runtime of $\mathsf{F}_{\mathcal{B}_1}$ is at most $(4Q_1/\varepsilon) \cdot \mathsf{Time}(\mathcal{A}) = \mathsf{poly}(\lambda)$. Here, the probability is taken over the randomness of crs and those used by $\mathcal{A}_{\mathsf{dl}}$.*

Proof. Denote by E the event that $(x_0, w_0) \in \mathsf{R}_{\mathsf{lax}}$ and the event Fail occurs, i.e. the extraction of Π_{rp} succeeds but online-extraction of Π failed for the i-th proof. Note that $\Pr[\mathsf{E}] \geq \varepsilon$ is non-negligible (cf. Lemma 5). We show that even though the forking algorithm uses the same initial randomness, it also holds that $\Pr[\mathsf{E} \wedge R \neq \bot]$ with non-negligible probability, where $R \leftarrow \mathsf{F}_{\mathcal{B}_1}(\mathsf{crs}, \mathsf{coin}_1, \vec{h}_1)$ is the output of $\mathsf{F}_{\mathcal{B}_1}$ in $\mathcal{A}_{\mathsf{dl}}$. This directly yields $(x_1, w_1) \in \mathsf{R}_{\mathsf{ped}}$ as follows:

$\mathsf{F}_{\mathcal{B}_1}$ runs $\mathcal{B}_1$ with identical randomness until the I-th H_1 query and it outputs statement-transcript pairs (x_1, π_1) and (x_1', π_2) for Σ_{ped}, each associated to the I-th H_1 query. Thus, we have that $\alpha = \alpha'$ and $x_1 = x_1'$ for $\pi_1 = (\alpha, \beta, \gamma), \pi_2 = (\alpha', \beta', \gamma')$. Also, we have that $\beta' \neq \beta$ except with negligible probability, as hash outputs are sampled uniformly and independently at random, in which case the extractor Ext_1 succeeds, i.e. $(x_1, w_1) \in \mathsf{R}_{\mathsf{ped}}$.

We are now ready to analyze the probability that $\Pr[\mathsf{E} \wedge R \neq \bot]$. The argument follows the high level structure of the proof of the forking lemma in [73]. First, denote by E_k the event that π_1 is associated to the k-th H_1 query, i.e. the k-th entry $\widehat{\beta}_{1,k}$ of $\vec{h}_1$ is equal to β. Next, we define the set P as

$$P = \left\{ k \;\middle|\; \Pr[\mathsf{E}_k \mid \mathsf{E}] \geq \frac{1}{2Q_1} \right\}.$$

Note that for any $k \in P$, we have $\Pr[\mathsf{E}_k] \geq \frac{\varepsilon}{2Q_1}$. Further, for the event $\mathsf{E}_1^{\mathsf{good}} = \bigvee_{k \in P} \mathsf{E}_k$, we have

$$\Pr\left[\mathsf{E}_1^{\mathsf{good}} \;\middle|\; \mathsf{E}\right] = \sum_{k \in P} \Pr\left[\mathsf{E}_k \mid \mathsf{E}\right] \geq \frac{Q_1}{2Q_1} = \frac{1}{2}. \tag{4}$$

We define the set $X_k = (R_{\mathcal{A}} \times R_{\mathsf{Ext}_0} \times \mathbb{Z}_p^{Q_0}) \times \mathbb{Z}_p^{k-1}$ and $Y_k = \mathbb{Z}_p^{Q_1 - k + 1}$, where $R_{\mathcal{A}}$ (resp. R_{Ext_0}) denotes the randomness space of $\mathcal{A}$ (resp. Ext_0). Note that for fixed crs, the tuple $(x, y) \in X_k \times Y_k$ can be parsed to define all inputs of $\mathcal{A}_{\mathsf{dl}}$, including randomness except the random choices in $\mathsf{F}_{\mathcal{B}_1}$. In more detail, parse $x = (\mathsf{coin}_{\mathcal{A}}, \mathsf{coin}_{\mathsf{Ext}_0}, \vec{h}_0, \vec{h}_{1,<k})$ and $y = \vec{h}_{1,\geq k}$. Set $\vec{h}_1 = (\vec{h}_{1,<k} \| \vec{h}_{1,\geq k})$. Note that given $\mathsf{coin}_{\mathsf{Ext}_0}$, $\mathcal{A}_{\mathsf{dl}}$ runs $w_0 = \mathsf{Ext}_0(\mathsf{crs}, x_0, \pi_0, \mathsf{coin}_0, \vec{h}_0; \mathsf{coin}_{\mathsf{Ext}_0})$ with randomness $\mathsf{coin}_{\mathsf{Ext}_0}$. Then, the execution of $\mathcal{A}_{\mathsf{dl}}$ on input crs and randomness (x, y) is deterministic up to the run of $\mathsf{F}_{\mathcal{B}_1}$.

Further, note that given $(x, y) \in X_k \times Y_k$, it can be determined whether E_k occurred. We define $A_k \subseteq X_k \times Y_k$ as the set of such inputs, i.e. $(x, y) \in A_k$ iff (x, y) triggers E_k. Then, the splitting lemma (cf. Lemma 1) with $\alpha = \frac{\varepsilon}{4Q_1}$ yields that that for the set $B_k \subseteq X_k \times Y_k$ defined

$$B_k = \left\{ (x, y) \in X_k \times Y_k \;\middle|\; \Pr_{y' \leftarrow Y_k}\left[(x, y') \in A_k\right] \geq \frac{\varepsilon}{4Q_1} \right\}, \tag{5}$$

we have

$$\Pr_{(x,y) \leftarrow X_k \times Y_k}\left[(x, y) \in B_k \mid (x, y) \in A_k\right] \geq \frac{1}{2}. \tag{6}$$

We are now ready to evaluate the probability of $(x_0, w_0) \in \mathsf{R}_{\mathsf{rp}}, (x_1, w_1) \in \mathsf{R}_{\mathsf{ped}}$ and Fail, when $\mathcal{A}_{\mathsf{dl}}$ is run with initial randomness (x, y). As shown above, this event occurs if both E and $R \neq 0$, i.e. the the run of $\mathsf{F}_{\mathcal{B}_1}$ outputs some non-$\bot$.

With probability ε, we have that E occurs. As in that case, the i-th proof output by $\mathcal{A}$ verifies, we further have that the initial run of $\mathcal{B}_1$ in $\mathsf{F}_{\mathcal{B}_1}$ produces some $(I, \Lambda) \neq \bot$. Then, the probability

that $\mathsf{E}_1^{\mathsf{good}}$ occurs is at least $\Pr\left[\mathsf{E}_1^{\mathsf{good}} \;\middle|\; \mathsf{E}\right] \geq \frac{1}{2}$ due to eq. (4). In that case, we have $(x, y) \in A_I$ by definition. Then, from eq. (6) we have that $(x, y) \in B_k$ with probability at least $\frac{1}{2}$. Thus, eq. (5) yields that the probability that $\mathsf{F}_{\mathcal{B}_1}$ resamples some $y' \in Y_I$ with $(x, y') \in A_I$ is at least $\frac{\varepsilon}{4Q_1}$ conditioned on $(x, y) \in B_k$.

In words, in the c-th iteration in $\mathsf{F}_{\mathcal{B}_1}$, the sampled $\vec{h}_{1,\geq I}^{(c)} \leftarrow \mathbb{Z}_p^{Q_1-I+1}$ triggers E_I, as $Y_I = \mathbb{Z}_p^{Q_1-I+1}$. This means that $\mathcal{A}_{\mathsf{dl}}$ computes some π_1 that is associated to I-th H_1 query on input $(x, \vec{h}_{1,\geq I}^{(c)})$. Equivalently, we have $I^{(c)} = I$ on original input (x, y) and thus $R \neq \bot$.

Note that all $\vec{h}_{1,\geq I}^{(c)} \leftarrow \mathbb{Z}_p^{Q_1-I+1}$ are sampled independently and uniformly at random in $\mathsf{F}_{\mathcal{B}_1}$ for $c \in [T]$, where $T = 4Q_1/\varepsilon$. Thus, conditioned on $\mathcal{A}_{\mathsf{dl}}$ sampling some $(x, y) \in B_k$, the probability that $R \neq \bot$ is at least

$$1 - \left(1 - \frac{\varepsilon}{4Q_1}\right)^{\frac{4Q_1}{\varepsilon}} \geq 1 - \frac{1}{e} \geq \frac{1}{2}$$

Collecting all the bounds, we conclude that $R \neq \bot$ with probability at least $\frac{\varepsilon}{8}$. Further, the runtime of $\mathsf{F}_{\mathcal{B}_1}$ is at most $(4Q_1/\varepsilon) \cdot \mathsf{Time}(\mathcal{A})$.

6.4 Optimizations and Efficiency

We now analyze the efficiency of the blind signature $\mathsf{BS}_{\mathsf{BB}}[\Pi]$.

Efficiency of Π. First, we optimize the extractor Ext of Π which allows us to choose better parameters. Note that the runtime of Ext scales linearly with the size of the exponents $e_i \in [0, B-1]$, as it brute-forces the discrete logarithm of 2ℓ group elements $F_i = \widehat{g}^{e_i}$. At the same time, the proof size scales linearly with $\ell = \lceil \log_B(\ell) \rceil$. Thus, for practical efficiency, we would like to set B as large as possible, while keeping the extractor efficient for the security reduction. If we use a more efficient algorithm to compute discrete logarithms of elements (with exponents in interval $[0, B-1]$), we can choose a larger bound B without any loss in runtime of the extractor. A good choice is Pollard's kangaroo algorithm [74] which has runtime $\mathcal{O}(\sqrt{B})$. This allows us to increase the bit size of B by a factor 2 for the same level of security.

Second, we can omit α_1 in π_1, as the identities in $\Sigma_{\mathsf{ped}}.\mathsf{Verify}$ can be recomputed and then verified via β_1 due to collision resistance.

With these optimizations, the proof π_1 contains $5 + 4\ell$ elements in $\mathbb{Z}_p$. For $n = \log_2(B)$, the batched range proof π_0 contains $2\lceil \log_2(2n\ell + \ell + 4) \rceil + 1$ elements in $\widehat{\mathbb{G}}$ and 7 elements in $\mathbb{Z}_p$. As a proof π of Π consists of $\pi = (\pi_0, \pi_1, (E_i, R_i)_{i \in [2\ell]})$, the total proof size is $12 + 4\ell$ elements in $\mathbb{Z}_p$ and $2\lceil \log_2(2n\ell + \ell + 4) \rceil + 4\ell + 1$ elements in $\widehat{\mathbb{G}}$.

Efficiency of $\mathsf{BS}_{\mathsf{BB}}[\Pi]$. When $\mathsf{BS}_{\mathsf{BB}}[\Pi]$ is instantiated with Π for $B = \mathsf{poly}(\lambda)$, the user sends 1 element in $\mathbb{G}_1$, $2\lceil \log_2(2n\ell + \ell + 4) \rceil + 4\ell + 1$ in $\widehat{\mathbb{G}}$, and $10 + 2\ell$ elements in $\mathbb{Z}_p$ to the signer. The signer sends 2 elements in $\mathbb{G}_1$, and the final signature contains 2 elements in $\mathbb{G}_1$. We set $B = 2^{64}$ in order to have an extractor that performs roughly $\ell \cdot 2^{32}$ group operations, where $\ell = \lceil \log_B p \rceil = 4$. The total communication is 2.2 KB and signatures are of size 96 Byte for $\lambda = 128$.

7 Frameworks for Partially Blind Signatures

In this section, we present variants of our constructions in sections 3 and 5 that achieve partial blindness.

7.1 Partial Blindness of the Optimized Fischlin Transform

We show how to adapt $\mathsf{BS}_{\mathsf{Rnd}}$ for partial blindness. Roughly, instead of signing only the commitment c, the signer signs the vector $(c, \mathsf{H}_{\mathsf{T}}(t))$ instead, where t is the common message and H_{T} is a random oracle.

The partially blind signature $\mathsf{PBS}_{\mathsf{Rnd}}$ is based on building blocks $(\mathsf{C}, \mathsf{S}, \Sigma)$ that are $\mathsf{BS}_{\mathsf{Rnd}}$-suitable (cf. definition 22), except the message space $\mathcal{S}_{\mathsf{msg}}$ of the signature scheme S contains all tuples $(c, \bar{t})$, where $\bar{t} = \mathsf{H}_{\mathsf{T}}(t)$.

Definition 23 ($\mathsf{PBS}_{\mathsf{Rnd}}$-Suitable $(\mathsf{C},\mathsf{S},\Sigma)$). *The tuple of schemes $(\mathsf{C},\mathsf{S},\Sigma)$ are called* $\mathsf{PBS}_{\mathsf{Rnd}}$-suitable*, if it holds that*

- C *is the same as a commitment scheme that is* $\mathsf{BS}_{\mathsf{Rnd}}$*-suitable.*
- S *is the same as a signature scheme that is* $\mathsf{BS}_{\mathsf{Rnd}}$*-suitable, except that the message space $\mathcal{S}_{\mathsf{msg}}$ encompasses $\mathcal{C}_{\mathsf{com}} \times \mathcal{TH}$, where $\mathcal{TH}$ is a set with $1/|\mathcal{TH}| = \mathrm{negl}(\lambda)$.*
- Σ *is the same as a Σ-protocol that is* $\mathsf{BS}_{\mathsf{Rnd}}$*-suitable except for the relation*

$$\mathsf{R}_{\mathsf{p\text{-}rnd}} \coloneqq \{x = (\mathsf{pp},\mathsf{vk},\overline{m},\bar{t}), w = (\mu, c, r) \mid \mathsf{C.Commit}(\mathsf{pp},\overline{m};r) = c \wedge \mathsf{S.Verify}(\mathsf{vk},\mu,(c,\bar{t})) = 1\}.$$

Construction We present the partially blind signature $\mathsf{PBS}_{\mathsf{Rnd}}$ below.

Overview. Let $(\mathsf{C},\mathsf{S},\Sigma)$ be $\mathsf{PBS}_{\mathsf{Rnd}}$-suitable. Again, let $\mathsf{H}_{\mathsf{par}},\mathsf{H}_{\mathsf{M}},\mathsf{H}_{\beta}$ be a random oracles from $\{0,1\}^*$ into $\{0,1\}^{\ell_{\mathsf{C}}},\mathcal{C}_{\mathsf{msg}},\mathcal{CH}$, respectively. Further, let H_{T} be a random oracle from $\{0,1\}^*$ into $\mathcal{TH}$. We now present the framework $\mathsf{PBS}_{\mathsf{Rnd}}[\mathsf{C},\mathsf{S},\Sigma]$ (or $\mathsf{PBS}_{\mathsf{Rnd}}$ for short) for partially blind signatures based on $\mathsf{BS}_{\mathsf{Rnd}}$.

Key generation is the same as before, i.e. it outputs $(\mathsf{bvk},\mathsf{bsk}) \leftarrow \mathsf{S.KeyGen}(1^\lambda)$, and implicitly defines a public parameter pp for C via $\mathsf{pp} = \mathsf{H}_{\mathsf{par}}(0)$. For message m and common message t, the user commits to $\overline{m} \leftarrow \mathsf{H}_{\mathsf{M}}(m)$ with randomness r via C and sends the commitment c to the signer. As before, the signer rerandomizes c with some random Δr, but signs $(c',\bar{t})$ instead of c' via S, where $\bar{t} = \mathsf{H}_{\mathsf{T}}(t)$, and sends the pair $(\mu,\Delta r)$ to the user. The derived signature is a proof π for relation $\mathsf{R}_{\mathsf{p\text{-}rnd}}$ generated by Σ using the Fiat-Shamir transform. Note that the user can recompute c' and its randomness via Δr and r, as before.

Construction. For completeness, we provide the full description below, where we assume pp is provided to all of the algorithms for readability. The changes are highlighted with a box.

- $\mathsf{PBS}_{\mathsf{Rnd}}.\mathsf{KeyGen}(1^\lambda)$: samples $(\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{S.KeyGen}(1^\lambda)$ and outputs verification key $\mathsf{bvk} = \mathsf{vk}$ and signing key $\mathsf{bsk} = \mathsf{sk}$.
- $\mathsf{PBS}_{\mathsf{Rnd}}.\mathsf{User}(\mathsf{bvk},t,m)$: sets $\overline{m} \leftarrow \mathsf{H}_{\mathsf{M}}(m)$ and outputs the commitment $c \in \mathcal{C}_{\mathsf{com}}$ generated via $(c,r) \leftarrow \mathsf{C.Commit}(\mathsf{pp},\overline{m})$ as the first message and stores the randomness $\mathsf{st} = r \in \mathcal{C}_{\mathsf{rnd}}$.
- $\mathsf{PBS}_{\mathsf{Rnd}}.\mathsf{Signer}(\mathsf{bsk},t,c)$: checks if $c \in \mathcal{C}_{\mathsf{com}}$, samples a rerandomization randomness $\Delta r \leftarrow \mathcal{C}_{\mathsf{rnd}}$, rerandomizes the commitment c via $c' \leftarrow \mathsf{C.RerandCom}(\mathsf{pp},c,\Delta r)$, signs $\mu \leftarrow \mathsf{S.Sign}(\mathsf{sk},\boxed{(c',\bar{t})})$ for $\bar{t} \leftarrow \mathsf{H}_{\mathsf{T}}(t)$, and finally outputs the second message $\rho = (\mu,\Delta r)$.
- $\mathsf{PBS}_{\mathsf{Rnd}}.\mathsf{Derive}(\mathsf{st},t,\rho)$: parses $\mathsf{st} = r$, $\rho = (\mu,\Delta r)$ and checks $\Delta r \in \mathcal{C}_{\mathsf{rnd}}$. It then computes the randomized commitment $c'' = \mathsf{C.RerandCom}(\mathsf{pp},c,\Delta r)$ and randomized randomness $r' \leftarrow \mathsf{C.RerandRand}(\mathsf{pp},c,\overline{m},r,\Delta r)$, and checks $\mathsf{S.Verify}(\mathsf{vk},\boxed{(c'',\bar{t})},\mu) = 1$ and $c'' = \mathsf{C.Commit}(\mathsf{pp},\overline{m};r')$. It then outputs a signature $\sigma = \pi$ for common message $\bar{t}$, where $(\alpha,\mathsf{st}') \leftarrow \Sigma.\mathsf{Init}(x,w)$, $\beta \leftarrow \mathsf{H}_{\beta}(x,\alpha), \gamma \leftarrow \Sigma.\mathsf{Resp}(x,\mathsf{st}',\beta)$, $\pi = (\alpha,\beta,\gamma)$ with $x = (\mathsf{pp},\mathsf{vk},\overline{m},\boxed{\bar{t}})$, $w = (\mu,c',r')$.
- $\mathsf{PBS}_{\mathsf{Rnd}}.\mathsf{Verify}(\mathsf{bvk},t,m,\sigma)$: parses $\sigma = \pi$ and $\pi = (\alpha,\beta,\gamma)$, and sets $\overline{m} = \mathsf{H}_{\mathsf{M}}(m)$, $\bar{t} \leftarrow \mathsf{H}_{\mathsf{T}}(t)$, and $x = (\mathsf{pp},\mathsf{vk},\overline{m},\boxed{\bar{t}})$, and outputs 1 if $\beta = \mathsf{H}_{\beta}(x,\alpha)$, $\Sigma.\mathsf{Verify}(x,\alpha,\beta,\gamma) = 1$, and otherwise outputs 0.

Correctness and Security We have the following theorem. As the proof is similar to the security proof of $\mathsf{BS}_{\mathsf{Rnd}}$, we only provide a sketch.

Theorem 21. *The partially blind signature* $\mathsf{PBS}_{\mathsf{Rnd}}$ *is (i) correct, (ii) partially blind under malicious keys under the hiding and rerandomization properties of* C *and the high min-entropy and HVZK properties of* Σ*, and (iii) one-more unforgeable under the binding and rerandomizability properties of* C*,* $\mathtt{euf\text{-}cma}$ *security of* S *and the 2-special soundness and f-unique extraction properties of* Σ*.*

Proof (Sketch). Compared to $\mathsf{BS}_{\mathsf{Rnd}}$, the only change is that the signer signs the message $(c',\bar{t})$ instead of c'. Note that the relation $\mathsf{R}_{\mathsf{p\text{-}rnd}}$ of Σ is adapted appropriately. Thus, correctness and zero-knowledge follow as before. One-more unforgeability is almost identical to before.

Let $\mathcal{A}$ be a PPT adversary that performs Q_S signing queries for some (adaptively chosen) common message t^*. We define the game $\mathcal{G}$ as the real game with $\mathcal{A}$, except the challenger aborts if

there is a collision in H_M or H_T or there is some (x_i, α_i) in the forgeries of $\mathcal{A}$ that was never queried to H_β. As before, we can show that the challenger never aborts except with negligible probability.

Then, the challenger interacts with $\mathcal{A}$ as in the real game, and obtains forgeries $(\pi_i = (\alpha_i, \beta_i, \gamma_i))_{i\in[Q_S+1]}$ from $\mathcal{A}$, for messages m_i and common message t^*. Denote by $x_i = (\mathsf{pp}, \mathsf{vk}, \overline{m}_i, \bar{t}^*)$ the corresponding statements.

Then, we rewind the adversary $\mathcal{A}$ as in theorem 3 to obtain the witnesses $w_i = (\mu_i, c_i, r)$. Under 2-special soundness of Σ, we have $(x_i, w_i) \in \mathsf{R}_\mathsf{rnd}$. That is, we have for all $i \in [Q_S + 1]$ that $c_i = \mathsf{C.Commit}(\mathsf{pp}, \overline{m}_i; r)$ and $\mathsf{S.Verify}(\mathsf{vk}, \mu_i, (c_i, \bar{t}^*)) = 1$, where $\overline{m}_i = \mathsf{H}_\mathsf{M}(m_i)$ and $\bar{t}^* = \mathsf{H}_\mathsf{T}(t^*)$. Recall that there are no collisions in H_M in $\mathcal{G}$, so we have $\overline{m}_i \neq \overline{m}_j$ for all distinct $i, j \in [Q_S + 1]$. Thus, under the binding property of C, there cannot exist two distinct indices $i, j \in [Q_S + 1]$ such that $c_i = c_j$, as both c_i and c_j open to distinct messages $\overline{m}_i \neq \overline{m}_j$.

Consequently, as at most Q_S signing sessions under common message t^* were performed, there must be at least one commitment c_{i^*} that was never signed in the initial run under common message t^*. As before, this initial run fixes c_{i^*} due to f-unique extraction of Σ, so we are guaranteed to never sign the tuple $(c_{i^*}, \mathsf{H}_\mathsf{T}(t^*))$ during rewinding under rerandomizability of C. But as μ_i is a valid signature for $(c_{i^*}, \mathsf{H}_\mathsf{T}(t^*))$, we break `euf-cma` security of S. Note μ_i is a valid forgery, even if c_{i^*} was signed under some other common message $t \neq t^*$, as there are no collisions in H_T in $\mathcal{G}$ and we have $(c_{i^*}, \mathsf{H}_\mathsf{T}(t)) \neq (c_{i^*}, \mathsf{H}_\mathsf{T}(t^*))$.

Instantiation We can set $\mathcal{TH} = \mathbb{G}_1$. Then, H_T maps into $\mathbb{G}_1$ and we can instantiate the scheme using S_KPW signatures and C_Ped commitments as in section 4. Note that the size of S_KPW signatures is independent of the number of signed group elements. Also, it is simple to adapt the Σ-protocol Σ_rnd accordingly. Consequently, the total communication and signature size remains 303 and 447 Byte for $\lambda = 128$, respectively.

7.2 Partial Blindness of Blind Signature based on Boneh-Boyen

We now show how to adapt the framework BS_BB to obtain a partially blind signature PBS_BB. Again, we focus on the asymmetric setting. Our idea is to construct part of the verification key $\mathsf{vk} = (u_1, u_2, h_1, h_2, v)$ (cf. section 5.1) using a hash of the common message t. However, since $(u_1, u_2, h_1, h_2) = (g_1^\alpha, g_2^\alpha, g_1^\gamma, g_2^\gamma)$ formed a valid DDH tuple, it is not clear how to do this. To this end, we adapt the Boneh-Boyen signature scheme S_BB such that u_2, h_2 no longer needs to be a part of the verification key. Instead, we include one additional element in the signature to compensate for this modification. Looking ahead, since the verification key is now $\mathsf{vk} = (u_1, h_1, v)$ where h_1 is a random group element, we are able to create this term by a hash of the common message t.[17] The resulting scheme $\overline{S}_\mathsf{BB}$ is defined below. We later show (implicitly) in the security proof of PBS_BB that the variant $\overline{S}_\mathsf{BB}$ is selectively secure under CDH. We omit the subscript and simply use (u, h) below.

- $\overline{S}_\mathsf{BB}.\mathsf{KeyGen}(1^\lambda)$: samples $\alpha, \beta, \gamma \in \mathbb{Z}_p$, and sets $u = g_1^\alpha, h = g_1^\gamma, v = e(g_1, g_2)^{\alpha\beta}$, and outputs $\mathsf{vk} = (u, h, v)$ and $\mathsf{sk} = g_1^{\alpha\beta}$.
- $\overline{S}_\mathsf{BB}.\mathsf{Sign}(\mathsf{sk}, m)$: samples $r \in \mathbb{Z}_p$ and outputs $(\sigma_0, \sigma_1, \sigma_2) = (\mathsf{sk} \cdot (u^m h)^r, g_1^r, g_2^r)$.
- $\overline{S}_\mathsf{BB}.\mathsf{Verify}(\mathsf{vk}, m, (\sigma_0, \sigma_1, \sigma_2))$: verify that $e(\sigma_0, g_2) = v \cdot e(u^m h, \sigma_2)$ and $e(\sigma_1, g_2) = e(g_1, \sigma_2)$.

Construction. Now, we construct PBS_BB and detail the changes required to the framework BS_BB (cf. section 5). Again, let Π be an online-extractable NIZK proof system, with random oracle $\mathsf{H}_\mathsf{zk} : \{0,1\}^* \mapsto \{0,1\}^{\ell_\mathsf{zk}}$ and common reference string length ℓ_crs for the relation

$$\mathsf{R}_\mathsf{bb} := \{x = (c, u, g_1), w = (\overline{m}, s) \mid c = u^{\overline{m}} \cdot g_1^s\},$$

and let $\mathsf{H}_\mathsf{M}, \mathsf{H}_\mathsf{crs}$ be a random oracles mapping into $\mathbb{Z}_p, \{0,1\}^{\ell_\mathsf{crs}}$ respectively. Further, we rely on a random oracle $\mathsf{H}_{\mathbb{G}_1}$ mapping into $\mathbb{G}_1$. The framework $\mathsf{PBS}_\mathsf{BB}[\Pi]$, or PBS_BB for short, is as BS_BB

[17] Note in the symmetric setting, we can use standard the Boneh-Boyen signature scheme S_BB and let a random oracle output h_1, as there is no h_2. The signature size in the symmetric setting remains 2 group elements.

except that the underlying signature is replaced with $\overline{S}_{\mathsf{BB}}$ and the value h in the verification key is sampled via $\mathsf{H}_{\mathbb{G}_1}$.

Construction. Below, we detail the construction and highlight the changes with respect to $\mathsf{BS}_{\mathsf{BB}}$ via a box. We assume that crs is provided to all of the algorithms for readability.

- $\mathsf{PBS}_{\mathsf{BB}}.\mathsf{KeyGen}(1^\lambda)$: samples $\alpha, \beta \in \mathbb{Z}_p$, and sets $u = g_1^\alpha, v = e(g_1, g_2)^{\alpha\beta}$, and outputs $\mathsf{bvk} = (u, v)$ and $\mathsf{bsk} = g_1^{\alpha\beta}$.
- $\mathsf{PBS}_{\mathsf{BB}}.\mathsf{User}(\mathsf{bvk}, t, m)$: sets $\overline{m} \leftarrow \mathsf{H}_{\mathsf{M}}(m)$ and computes a Pedersen commitment $c = u^{\overline{m}} g_1^s$ and a proof $\pi \leftarrow \Pi.\mathsf{Prove}^{\mathsf{H}_{\mathsf{zk}}}(\mathsf{crs}, x, w)$, where $s \leftarrow \mathbb{Z}_p$, $x = (c, u, g_1)$, and $w = (\overline{m}, s)$. It outputs the first message $\rho_1 = (c, \pi)$ and stores the randomness $\mathsf{st} = s$.
- $\mathsf{PBS}_{\mathsf{BB}}.\mathsf{Signer}(\mathsf{bsk}, t, \rho_1)$: parses $\rho_1 = (c, \pi)$ and checks $\Pi.\mathsf{Verify}^{\mathsf{H}_{\mathsf{zk}}}(\mathsf{crs}, x, \pi) = 1$. It then sets $\boxed{h_t \leftarrow \mathsf{H}_{\mathbb{G}_1}(t)}$ and outputs the second meesage $\rho_2 = (\rho_{2,0}, \rho_{2,1}, \rho_{2,2}) \leftarrow (\mathsf{sk} \cdot (c \cdot \boxed{h_t})^r, g_1^r, \boxed{g_2^r})$, where $r \leftarrow \mathbb{Z}_p$.
- $\mathsf{PBS}_{\mathsf{BB}}.\mathsf{Derive}(\mathsf{st}, t, \rho_2)$: parses $\mathsf{st} = s$ and $\rho_2 = (\rho_{2,0}, \rho_{2,1}, \rho_{2,2})$, checks $e(\rho_{2,0}, g_2) = v \cdot e(c \cdot \boxed{h_t}, \rho_{2,2})$ and $\boxed{e(\rho_{2,1}, g_2) = e(g_1, \rho_{2,2})}$, and outputs the signature $\sigma = (\sigma_0, \sigma_1, \sigma_2) = (\rho_{2,0}/\rho_{2,1}^s \cdot (u^{\overline{m}} \boxed{h_t})^{r'}, \rho_{2,1} \cdot g_1^{r'}, \boxed{\rho_{2,2} \cdot g_2^{r'}})$ for $r' \leftarrow \mathbb{Z}_p$.
- $\mathsf{BS}_{\mathsf{BB}}.\mathsf{Verify}(\mathsf{bvk}, t, m, \sigma)$: checks $e(\sigma_0, g_2) = v \cdot e(u^{\overline{m}} \boxed{h_t}, \sigma_2)$ and $\boxed{e(\sigma_1, g_2) = e(g_1, \sigma_2)}$, where $\overline{m}$ and h_t as above.

Correctness and Security We can show that $\mathsf{PBS}_{\mathsf{BB}}$ is correct, blind, and one-more unforgeable.

Theorem 22. *The scheme $\mathsf{PBS}_{\mathsf{BB}}$ is correct, blind under malicious keys under the zero-knowledge property of Π, and one-more unforgeable under the CDH assumption and the online-extractability of Π.*

Proof. The proof of correctness and blindness follow almost as in theorems 10 and 11, and we omit details. In the following, we show that $\mathsf{PBS}_{\mathsf{BB}}$ is one-more unforgeable. Roughly, the reduction punctures the verification key (including h_t for all common messages t) depending on whether (i) t is the (adaptively chosen) common message t^* from the forgeries of $\mathcal{A}$ or (ii) t is some other common message. In case of (i), the reduction punctures h_t such that it can sign all but some random message $\overline{m}_t \leftarrow \mathbb{Z}_p$. As the message $\overline{m}_t$ is hidden from $\mathcal{A}$, she does not provide a signing request for (a commitment of) $\overline{m}_t$ with high probability, and the reduction can answer all signing queries for $t \neq t^*$. In case of (ii), for the common message t^* of the $\mathcal{A}$'s forgeries, the reduction punctures h_{t^*} on $\overline{m}^* \leftarrow \mathbb{Z}_p$. As in theorem 12, we show that by programming the random oracle H_{M}, we can argue that $\mathcal{A}$ never asks for a signature on (a commitment of) $\overline{m}^*$ but provides a forgery for message m_{i^*} such that $\overline{m}^* = \mathsf{H}_{\mathsf{M}}(m_{i^*})$ with noticeable probability, in which case we can solve CDH. We formalize the above intuition below.

Let $\mathcal{A}$ be a PPT adversary against the one-more unforgeability of $\mathsf{PBS}_{\mathsf{BB}}$. Let Ext and $\mathsf{Sim}_{\mathsf{crs}}$ be the extractor and simulator of Π, respectively (cf. definition 21). Without loss of generality, let Q_S be the number of signing queries per common message, and Q_T the total number of common messages. We denote by Q_M the number of H_{M} queries, by Q_H the number of H_{zk} queries, and by Q_G the number of $\mathsf{H}_{\mathbb{G}_1}$ queries. We assume without loss of generality that $\mathcal{A}$'s queries to the random oracles are unique. Moreover, we assume $\mathcal{A}$ queries the common message t to $\mathsf{H}_{\mathbb{G}_1}$ before submitting a signing query with t. We denote by q_j (resp. q'_j) the j-th query to H_{M} for $j \in [Q_M]$ (resp., $\mathsf{H}_{\mathbb{G}_1}$ for $j \in [Q_G]$). After Q_S signing queries per common message, $\mathcal{A}$ outputs $Q_S + 1$ forgeries $\{(m_i, \sigma_i)\}_{i \in [Q_S+1]}$ for some common message t^*. We write $\sigma_i = (\sigma_{i,0}, \sigma_{i,1}, \sigma_{i,2})$, and denote by $\rho_{1,i} = (c_i, \pi_i)$ the signing queries issued by $\mathcal{A}$. We define the following hybrids and denote by $\mathsf{Adv}^{\mathrm{H}_i}_{\mathcal{A}}(\lambda)$ the advantage of $\mathcal{A}$ in Hybrid i. (Note that until Hybrid 3, the steps are identical to the steps in theorem 12).

- Hybrid 0 is identical to the real game.
- Hybrid 1 is the same as Hybrid 0, except it samples $(\overline{\mathsf{crs}}, \tau) \leftarrow \mathsf{Sim}_{\mathsf{crs}}(1^\lambda)$ and programs $\overline{\mathsf{crs}}$ into the random oracle $\mathsf{H}_{\mathsf{crs}}$ via $\mathsf{H}_{\mathsf{crs}}(0) \leftarrow \overline{\mathsf{crs}}$. We can construct an adversary $\mathcal{B}_{\mathsf{crs}}$ against the CRS indistinguishability from Π such that $\mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\mathrm{H}_0}_{\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathrm{crs}}_{\mathcal{B}_{\mathsf{crs}}}(\lambda)$.

– Hybrid 2 is the same as Hybrid 1, except the witnesses $(\overline{m}_i, s_i)$ are extracted from all the proofs π_i for all $i \in [Q_S \cdot Q_T]$ using Ext. Specifically, when $\mathcal{A}$ provides the signing query (c_i, π_i), the challenger runs $w_i \leftarrow \mathsf{Ext}(\mathsf{crs}, \mathsf{td}, x_i, \pi_i)$, where $x_i = (c_i, u, g_1)$. It parses $w_i = (\overline{m}_i, s_i)$ and aborts if $u^{\overline{m}_i} \cdot g_1^{s_i} \neq c_i$. Then, proceeds as in Hybrid 1.
  We have $\mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda) \geq \frac{\mathsf{Adv}^{\mathrm{H}_1}_{\mathcal{A}}(\lambda) - \mathrm{negl}(\lambda)}{p_{\mathsf{P}}(\lambda, Q_H)}$ under the online extractability of Π. Note that the challenger has an additional runtime overhead $p_{\mathsf{T}}(\lambda, Q_H) \cdot \mathsf{Time}(\mathcal{A})$, and that p_{P} and p_{T} are polynomials as defined in definition 21. We note that Ext does not need to be invoked at the end of the game as required by the definition of online-extractability. See theorem 12 for more detail.
– Hybrid 3 is the same as Hybrid 2, except it aborts if there is a collision in H_{M} or if there is some message m_i in $\mathcal{A}$'s output that was never queried to H_{M}.
  It holds that $\mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\mathrm{H}_2}_{\mathcal{A}}(\lambda) - \frac{Q_M^2+1}{p}$.
– Hybrid 4 is the same as Hybrid 3, except it aborts if there is a collision in $\mathsf{H}_{\mathbb{G}_1}$ or if the common message t^* of $\mathcal{A}$'s output was never queried to $\mathsf{H}_{\mathbb{G}_1}$.
  A given signature verifies with respect to some random h_t at most with probability $1/p$ and a collision happens with probability at most Q_G^2/p. Thus, it holds that $\mathsf{Adv}^{\mathrm{H}_4}_{\mathcal{A}}(\lambda) \geq \mathsf{Adv}^{\mathrm{H}_3}_{\mathcal{A}}(\lambda) - \frac{Q_G^2+1}{p}$.
– Hybrid 5 is the same as Hybrid 4, except it first samples $\overline{m}^* \in \mathbb{Z}_p$, $k^* \leftarrow [Q_G]$, and $\delta_{k^*} \leftarrow \mathbb{G}_1$. For the k^*-th query q'_{k^*} to $\mathsf{H}_{\mathbb{G}_1}$, it sets $\mathsf{H}_{\mathbb{G}_1}(q'_{k^*}) \leftarrow h_t^* := u^{-\overline{m}^*} \cdot g_1^{\delta_{k^*}}$. Hybrid 5 aborts if the forgeries output by $\mathcal{A}$ is not on a common message t^* such that $h_t^* = \mathsf{H}_{\mathbb{G}_1}(t^*)$.
  Due to the modification we made in Hybrid 4, the common message t^* is guaranteed to be queried to $\mathsf{H}_{\mathbb{G}_1}$. Since k^* is information-theoretically hidden to $\mathcal{A}$ and the distribution of h_t^* is uniform over $\mathbb{G}_1$ as in the previous hybrid, we have $\mathsf{Adv}^{\mathrm{H}_5}_{\mathcal{A}}(\lambda) \geq 1/Q_G \cdot \mathsf{Adv}^{\mathrm{H}_4}_{\mathcal{A}}(\lambda)$.
– Hybrid 6 is the same as Hybrid 5, except it guesses a query index j^* of H_{M} for which it outputs $\overline{m}^* \in \mathbb{Z}_p$, and aborts if either the challenger extracts $\overline{m}^*$ in some signing query with common message t^* or if $\mathcal{A}$ provides no forgery for some message that hashes to $\overline{m}^*$. More concrete, Hybrid 6 further samples $j^* \leftarrow [Q_M]$ at the outset of the game. For the j^*-th query to H_{M}, it sets $\mathsf{H}_{\mathsf{M}}(q_{j^*}) \leftarrow \overline{m}^*$. At the i-th signing query, the signer aborts if the extracted witness is $(\overline{m}^*, s_i)$ and the common message is t^*, else proceeds as usual. Also, Hybrid 6 aborts if $m^* \notin \{\mathsf{H}_{\mathsf{M}}(m_i)\}_{i \in [Q_S+1]}$, where m_i are the messages from the forgeries of the final output of $\mathcal{A}$. Conditioned on no collision in H_{M}, there are only Q_S signing queries but $Q_S + 1$ values $\overline{m}_i \leftarrow \mathsf{H}_{\mathsf{M}}(m_i)$ from the forgeries, and with probability $1/Q_M$ the challenger guesses the right H_{M} query. Following the same calculation as in theorem 12, we have $\mathsf{Adv}^{\mathrm{H}_6}_{\mathcal{A}}(\lambda) \geq \frac{1}{Q_M} \mathsf{Adv}^{\mathrm{H}_5}_{\mathcal{A}}(\lambda)$.
– Hybrid 7 is the same as Hybrid 6, except it sets up a punctured verification key and simulates signing without knowing the full sk. Specifically, it sets $A_1 = g_1^{\alpha}, A_2 = g_2^{\alpha}, B = g_1^{\beta}, B_2 = g_2^{\beta}, u = A_1, v = e(A_1, B_2)$, for $\alpha, \beta \leftarrow \mathbb{Z}_p$. Then, sends $\mathsf{bvk} = (u, v)$ to $\mathcal{A}$. Also, Hybrid 7 initially samples $\overline{m}_{t,k} \leftarrow \mathbb{Z}_p$ for $k \in [Q_G] \backslash \{k^*\}$, and answers the k-th $\mathsf{H}_{\mathbb{G}_1}$ query with $h_{t,k} = u^{-\overline{m}_{t,k}} \cdot g_1^{\delta_k}$ if $k \neq k^*$. It answers the k^*-th query as in the previous Hybrid 5.
  For the i-th signing query (c_i, π_i) with common message t_i and extracted witness $w_i = (\overline{m}_i, s_i)$, parses $\mathsf{H}_{\mathbb{G}_1}(t_i) = h_{t,k} = u^{-\overline{m}_{t,k}} \cdot g_1^{\delta_k}$ for an appropriate $k \in [Q_G]$, and aborts if $\overline{m}_i = \overline{m}_{t,k}$ and $k \neq k^*$. Note that such a k uniquely exists due to Hybrid 4. Moreover, due to the modification we made in Hybrid 6, it always abort when $(\overline{m}_i, t_i) = (\overline{m}^*, t^*)$ or equivalently $(\overline{m}_i, k) = (\overline{m}^*, k^*)$. If it didn't abort, it samples some $\tilde{r_i} \leftarrow \mathbb{Z}_p$, sets $\rho_{2,1} = g_1^{\tilde{r_i}} \cdot B_1^{-1/(\overline{m}_i - \overline{m}_{t,k})}, \rho_{2,2} = g_2^{\tilde{r_i}} \cdot B_2^{-1/(\overline{m}_i - \overline{m}_{t,k})}$ and $\rho_{2,0} = A_1^{(\overline{m}_i - \overline{m}_{t,k})\tilde{r}} \cdot g_1^{(s_i+\delta_k)\tilde{r_i}} \cdot B_1^{-(s_i+\delta_k)/(\overline{m}_i - \overline{m}_{t,k})}$.
  It is not hard to check that Hybrid 7 and Hybrid 6 are identically distributed as long as $\mathcal{A}$ doesn't query a common message $t_i \neq t^*$ such that $\overline{m}_i = \overline{m}_{t,k}$ for $k \neq k^*$. Since $\overline{m}_{t,k}$ is information-theoretically hidden from $\mathcal{A}$, we conclude via a union bound that the abort probability is at most $(Q_S Q_T)/p$. Thus, $\mathsf{Adv}^{\mathrm{H}_7}_{\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathrm{H}_6}_{\mathcal{A}}(\lambda) - (Q_S Q_T)/p$.

We can now construct an adversary $\mathcal{B}_{\mathsf{CDH}}$ to solve CDH with $\mathsf{Adv}^{\mathrm{CDH}}_{\mathcal{A}_{\mathsf{CDH}}}(\lambda) = \mathsf{Adv}^{\mathrm{H}_7}_{\mathcal{A}}(\lambda)$. First, $\mathcal{B}_{\mathsf{CDH}}$ receives a CDH-tuple $(g_1, g_2, A_1, A_2, B_1, B_2)$, with which it simulates Hybrid 7 to $\mathcal{A}$. After $\mathcal{A}$ outputs the forgeries $\{(m_i, \sigma_i)\}_{i \in [Q_S+1]}$, $\mathcal{B}_{\mathsf{CDH}}$ outputs $\sigma_{i^*,0} \cdot \sigma_{i^*,1}^{-\delta_{k^*}}$ to its challenger, where i^* such that $\mathsf{H}_{\mathsf{M}}(m_{i^*}) = \overline{m}^*$.

Due to the abort conditions in Hybrid 5 and Hybrid 6, the adversary is guaranteed to output an appropriate forgery that verifies correctly with probability at least $\mathsf{Adv}^{\mathrm{H}_7}_{\mathcal{A}}(\lambda)$. In that case, we

show that $\sigma_{i^*,0}^{\delta_{k^*}}/\sigma_{i^*,1}$ is indeed of the form $g_1^{\alpha\beta}$, where α, β is the discrete logarithm of A_1, B_1, respectively. For readability, we write $(\sigma_0, \sigma_1, \sigma_2) = (\sigma_{i^*,0}, \sigma_{i^*,1}, \sigma_{i^*,2})$ in the following (with slight abuse of notation).

As the i^*-th forgery verifies, we have $e(\sigma_0, g_2) = v \cdot e(u^{\overline{m}^*} \cdot h_t^*, \sigma_2)$ and $e(\sigma_1, g_2) = e(g_1, \sigma_2)$. The latter guarantees that the discrete logarithms of σ_1 and σ_2 are identical, i.e. we have $\sigma_1 = g_1^\rho$ and $\sigma_2 = g_2^\rho$ for some appropriate ρ. Note that we have $v = e(g_1, g_2)^{\alpha\beta}$, $u = A_1$ and $h_t^* = u^{-\overline{m}^*} \cdot g_1^{\delta_{k^*}}$ due to the changes in Hybrid 5 and Hybrid 7. Consequently, we have

$$
\begin{aligned}
& e(\sigma_0, g_2) = v \cdot e(u^{\overline{m}^*} \cdot h_t^*, \sigma_2) \\
\implies & e(\sigma_0, g_2) = e(g_1, g_2)^{\alpha\beta} \cdot e(u^{\overline{m}^*} \cdot u^{-\overline{m}^*} \cdot g_1^{\delta_{k^*}}, \sigma_2) \\
\implies & e(\sigma_0, g_2) = e(g_1, g_2)^{\alpha\beta} \cdot e(g_1^{\delta_{k^*}}, \sigma_2) \\
\implies & e(\sigma_0, g_2) = e(g_1, g_2)^{\alpha\beta} \cdot e(g_1, g_2)^{\delta_{k^*} \cdot \rho} \\
\implies & e(\sigma_0 \cdot g_1^{-\delta_{k^*} \cdot \rho}, g_2) = e(g_1, g_2)^{\alpha\beta} \\
\implies & e(\sigma_0 \cdot \sigma_1^{-\delta_{k^*}}, g_2) = e(g_1^{\alpha\beta}, g_2) \\
\implies & \sigma_0 \cdot \sigma_1^{-\delta_{k^*}} = g_1^{\alpha\beta}
\end{aligned}
$$

The statement follows by collecting all the above bounds.

Instantiation. We can use the online extractable NIZK Π from section 6. Both communication and signature size increase by one element in $\mathbb{G}_2$.

8 Acknowledgments

This work was partially supported by JST CREST Grant Number JPMJCR22M1, Japan, JST AIP Acceleration Research JPMJCR22U5, Japan, JSPS KAKENHI Grant Numbers JP18K18055 and JP19H01109. Also, we would like to thank Michael Klooß for helpful discussions on the soundness of Bulletproofs and Geoffroy Couteau for helpful discussions in early stages of this work.

References

1. Abe, M.: A secure three-move blind signature scheme for polynomially many signatures. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 136–151. Springer, Heidelberg (May 2001). https://doi.org/10.1007/3-540-44987-6_9
2. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K., Ohkubo, M.: Structure-preserving signatures and commitments to group elements. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 209–236. Springer, Heidelberg (Aug 2010). https://doi.org/10.1007/978-3-642-14623-7_12
3. Abe, M., Groth, J., Haralambiev, K., Ohkubo, M.: Optimal structure-preserving signatures in asymmetric bilinear groups. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 649–666. Springer, Heidelberg (Aug 2011). https://doi.org/10.1007/978-3-642-22792-9_37
4. Abe, M., Jutla, C.S., Ohkubo, M., Roy, A.: Improved (almost) tightly-secure simulation-sound QA-NIZK with applications. In: Peyrin, T., Galbraith, S. (eds.) ASIACRYPT 2018, Part I. LNCS, vol. 11272, pp. 627–656. Springer, Heidelberg (Dec 2018). https://doi.org/10.1007/978-3-030-03326-2_21
5. Abe, M., Okamoto, T.: Provably secure partially blind signatures. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 271–286. Springer, Heidelberg (Aug 2000). https://doi.org/10.1007/3-540-44598-6_17
6. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (May / Jun 2010). https://doi.org/10.1007/978-3-642-13190-5_28
7. Agrawal, S., Kirshanova, E., Stehlé, D., Yadav, A.: Practical, round-optimal lattice-based blind signatures. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) ACM CCS 2022. pp. 39–53. ACM Press (Nov 2022). https://doi.org/10.1145/3548606.3560650
8. Attema, T., Cramer, R.: Compressed Σ-protocol theory and practical application to plug & play secure algorithmics. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part III. LNCS, vol. 12172, pp. 513–543. Springer, Heidelberg (Aug 2020). https://doi.org/10.1007/978-3-030-56877-1_18

9. Attema, T., Fehr, S., Klooß, M.: Fiat-shamir transformation of multi-round interactive proofs. In: Kiltz, E., Vaikuntanathan, V. (eds.) Theory of Cryptography - 20th International Conference, TCC 2022, Chicago, IL, USA, November 7-10, 2022, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13747, pp. 113–142. Springer (2022). https://doi.org/10.1007/978-3-031-22318-1_5, https://doi.org/10.1007/978-3-031-22318-1_5
10. Baldimtsi, F., Lysyanskaya, A.: Anonymous credentials light. In: Sadeghi, A.R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013. pp. 1087–1098. ACM Press (Nov 2013). https://doi.org/10.1145/2508859.2516687
11. Barreto, P.S., Lynn, B., Scott, M.: Constructing elliptic curves with prescribed embedding degrees. In: Security in Communication Networks: Third International Conference, SCN 2002 Amalfi, Italy, September 11–13, 2002 Revised Papers 3. pp. 257–267. Springer (2003)
12. Bellare, M., Boldyreva, A., Micali, S.: Public-key encryption in a multi-user setting: Security proofs and improvements. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 259–274. Springer, Heidelberg (May 2000). https://doi.org/10.1007/3-540-45539-6_18
13. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The one-more-rsa-inversion problems and the security of chaum's blind signature scheme. J. Cryptol. **16**(3), 185–215 (2003). https://doi.org/10.1007/s00145-002-0120-1, https://doi.org/10.1007/s00145-002-0120-1
14. Bellare, M., Neven, G.: Multi-signatures in the plain public-key model and a general forking lemma. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM CCS 2006. pp. 390–399. ACM Press (Oct / Nov 2006). https://doi.org/10.1145/1180405.1180453
15. Benhamouda, F., Lepoint, T., Loss, J., Orrù, M., Raykova, M.: On the (in)security of ROS. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 33–53. Springer, Heidelberg (Oct 2021). https://doi.org/10.1007/978-3-030-77870-5_2
16. Bernhard, D., Fischlin, M., Warinschi, B.: Adaptive proofs of knowledge in the random oracle model. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 629–649. Springer, Heidelberg (Mar / Apr 2015). https://doi.org/10.1007/978-3-662-46447-2_28
17. Beullens, W., Lyubashevsky, V., Nguyen, N.K., Seiler, G.: Lattice-based blind signatures: Short, efficient, and round-optimal. Cryptology ePrint Archive, Paper 2023/077 (2023), https://eprint.iacr.org/2023/077, https://eprint.iacr.org/2023/077
18. Blazy, O., Fuchsbauer, G., Pointcheval, D., Vergnaud, D.: Short blind signatures. Journal of computer security **21**(5), 627–661 (2013)
19. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In: Desmedt, Y. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (Jan 2003). https://doi.org/10.1007/3-540-36288-6_3
20. Boneh, D., Boyen, X.: Efficient selective-ID secure identity based encryption without random oracles. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (May 2004). https://doi.org/10.1007/978-3-540-24676-3_14
21. Boneh, D., Boyen, X.: Short signatures without random oracles. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (May 2004). https://doi.org/10.1007/978-3-540-24676-3_4
22. Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. Journal of Cryptology **21**(2), 149–177 (Apr 2008). https://doi.org/10.1007/s00145-007-9005-7
23. Bowe, S.: Bls12-381: New zk-snark elliptic curve construction. https://electriccoin.co/blog/new-snark-curve/ (2017), accessed: 2023-02-02
24. Brands, S.: Untraceable off-line cash in wallets with observers (extended abstract). In: Stinson, D.R. (ed.) CRYPTO'93. LNCS, vol. 773, pp. 302–318. Springer, Heidelberg (Aug 1994). https://doi.org/10.1007/3-540-48329-2_26
25. Brickell, E.F., Camenisch, J., Chen, L.: Direct anonymous attestation. In: Atluri, V., Pfitzmann, B., McDaniel, P. (eds.) ACM CCS 2004. pp. 132–145. ACM Press (Oct 2004). https://doi.org/10.1145/1030083.1030103
26. Brier, E., Coron, J.S., Icart, T., Madore, D., Randriam, H., Tibouchi, M.: Efficient indifferentiable hashing into ordinary elliptic curves. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 237–254. Springer, Heidelberg (Aug 2010). https://doi.org/10.1007/978-3-642-14623-7_13
27. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: 2018 IEEE Symposium on Security and Privacy. pp. 315–334. IEEE Computer Society Press (May 2018). https://doi.org/10.1109/SP.2018.00020
28. Buser, M., Dowsley, R., Esgin, M.F., Gritti, C., Kermanshahi, S.K., Kuchta, V., LeGrow, J.T., Liu, J.K., Phan, R.C.W., Sakzad, A., Steinfeld, R., Yu, J.: A survey on exotic signatures for post-quantum blockchain: Challenges & research directions. ACM Comput. Surv. (2022). https://doi.org/10.1145/3572771, https://doi.org/10.1145/3572771, just accepted

29. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 93–118. Springer, Heidelberg (May 2001). https://doi.org/10.1007/3-540-44987-6_7
30. Chairattana-Apirom, R., Hanzlik, L., Loss, J., Lysyanskaya, A., Wagner, B.: PI-cut-choo and friends: Compact blind signatures via parallel instance cut-and-choose and more. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part III. LNCS, vol. 13509, pp. 3–31. Springer, Heidelberg (Aug 2022). https://doi.org/10.1007/978-3-031-15982-4_1
31. Chaum, D.: Blind signatures for untraceable payments. In: Chaum, D., Rivest, R.L., Sherman, A.T. (eds.) CRYPTO'82. pp. 199–203. Plenum Press, New York, USA (1982)
32. Chaum, D.: Elections with unconditionally-secret ballots and disruption equivalent to breaking RSA. In: Günther, C.G. (ed.) EUROCRYPT'88. LNCS, vol. 330, pp. 177–182. Springer, Heidelberg (May 1988). https://doi.org/10.1007/3-540-45961-8_15
33. Chaum, D., Fiat, A., Naor, M.: Untraceable electronic cash. In: Goldwasser, S. (ed.) CRYPTO'88. LNCS, vol. 403, pp. 319–327. Springer, Heidelberg (Aug 1990). https://doi.org/10.1007/0-387-34799-2_25
34. del Pino, R., Katsumata, S.: A new framework for more efficient round-optimal lattice-based (partially) blind signature via trapdoor sampling. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part II. LNCS, vol. 13508, pp. 306–336. Springer, Heidelberg (Aug 2022). https://doi.org/10.1007/978-3-031-15979-4_11
35. ElGamal, T.: A public key cryptosystem and a signature scheme based on discrete logarithms. In: Blakley, G.R., Chaum, D. (eds.) CRYPTO'84. LNCS, vol. 196, pp. 10–18. Springer, Heidelberg (Aug 1984)
36. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO'86. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (Aug 1987). https://doi.org/10.1007/3-540-47721-7_12
37. Fischlin, M.: Communication-efficient non-interactive proofs of knowledge with online extractors. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 152–168. Springer, Heidelberg (Aug 2005). https://doi.org/10.1007/11535218_10
38. Fischlin, M.: Round-optimal composable blind signatures in the common reference string model. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 60–77. Springer, Heidelberg (Aug 2006). https://doi.org/10.1007/11818175_4
39. Fischlin, M., Schröder, D.: On the impossibility of three-move blind signature schemes. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 197–215. Springer, Heidelberg (May / Jun 2010). https://doi.org/10.1007/978-3-642-13190-5_10
40. Fuchsbauer, G., Hanser, C., Kamath, C., Slamanig, D.: Practical round-optimal blind signatures in the standard model from weaker assumptions. In: Zikas, V., De Prisco, R. (eds.) SCN 16. LNCS, vol. 9841, pp. 391–408. Springer, Heidelberg (Aug / Sep 2016). https://doi.org/10.1007/978-3-319-44618-9_21
41. Fuchsbauer, G., Hanser, C., Slamanig, D.: Practical round-optimal blind signatures in the standard model. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 233–253. Springer, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-48000-7_12
42. Fuchsbauer, G., Plouviez, A., Seurin, Y.: Blind schnorr signatures and signed ElGamal encryption in the algebraic group model. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020, Part II. LNCS, vol. 12106, pp. 63–95. Springer, Heidelberg (May 2020). https://doi.org/10.1007/978-3-030-45724-2_3
43. Fuchsbauer, G., Wolf, M.: (concurrently secure) blind schnorr from schnorr. IACR Cryptol. ePrint Arch. p. 1676 (2022), https://eprint.iacr.org/2022/1676
44. Fujioka, A., Okamoto, T., Ohta, K.: A practical secret voting scheme for large scale elections. In: AUSCRYPT. pp. 244–251. Springer (1992)
45. Ganesh, C., Orlandi, C., Pancholi, M., Takahashi, A., Tschudi, D.: Fiat-shamir bulletproofs are non-malleable (in the algebraic group model). In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 397–426. Springer, Heidelberg (May / Jun 2022). https://doi.org/10.1007/978-3-031-07085-3_14
46. Garg, S., Gupta, D.: Efficient round optimal blind signatures. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 477–495. Springer, Heidelberg (May 2014). https://doi.org/10.1007/978-3-642-55220-5_27
47. Garg, S., Rao, V., Sahai, A., Schröder, D., Unruh, D.: Round optimal blind signatures. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 630–648. Springer, Heidelberg (Aug 2011). https://doi.org/10.1007/978-3-642-22792-9_36
48. Ghadafi, E.: Efficient round-optimal blind signatures in the standard model. In: Kiayias, A. (ed.) FC 2017. LNCS, vol. 10322, pp. 455–473. Springer, Heidelberg (Apr 2017)
49. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM CCS 2006. pp. 89–98. ACM Press (Oct / Nov 2006). https://doi.org/10.1145/1180405.1180418, available as Cryptology ePrint Archive Report 2006/309

50. Groth, J., Ostrovsky, R., Sahai, A.: New techniques for non-interactive zero-knowledge. Journal of the ACM (JACM) **59**(3), 1–35 (2012)
51. Hanzlik, L., Loss, J., Wagner, B.: Rai-choo! evolving blind signatures to the next level. To Appear at EUROCRYPT (2023)
52. Hauck, E., Kiltz, E., Loss, J.: A modular treatment of blind signatures from identification schemes. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part III. LNCS, vol. 11478, pp. 345–375. Springer, Heidelberg (May 2019). https://doi.org/10.1007/978-3-030-17659-4_12
53. Hendrickson, S., Iyengar, J., Pauly, T., Valdez, S., Wood, C.A.: Private access tokens. internet-draft draft-private-access-tokens-01 (April 2022), https://datatracker.ietf.org/doc/draft-private-access-tokens/, work in Progress
54. Juels, A., Luby, M., Ostrovsky, R.: Security of blind digital signatures (extended abstract). In: Kaliski Jr., B.S. (ed.) CRYPTO'97. LNCS, vol. 1294, pp. 150–164. Springer, Heidelberg (Aug 1997). https://doi.org/10.1007/BFb0052233
55. Jutla, C.S., Roy, A.: Improved structure preserving signatures under standard bilinear assumptions. In: Fehr, S. (ed.) PKC 2017, Part II. LNCS, vol. 10175, pp. 183–209. Springer, Heidelberg (Mar 2017). https://doi.org/10.1007/978-3-662-54388-7_7
56. Kastner, J., Loss, J., Xu, J.: The Abe-Okamoto partially blind signature scheme revisited. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part IV. Lecture Notes in Computer Science, vol. 13794, pp. 279–309. Springer (2022). https://doi.org/10.1007/978-3-031-22972-5_10, https://doi.org/10.1007/978-3-031-22972-5_10
57. Kastner, J., Loss, J., Xu, J.: On pairing-free blind signature schemes in the algebraic group model. In: Hanaoka, G., Shikata, J., Watanabe, Y. (eds.) Public-Key Cryptography - PKC 2022 - 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8-11, 2022, Proceedings, Part II. Lecture Notes in Computer Science, vol. 13178, pp. 468–497. Springer (2022). https://doi.org/10.1007/978-3-030-97131-1_16, https://doi.org/10.1007/978-3-030-97131-1_16
58. Katsumata, S., Nishimaki, R., Yamada, S., Yamakawa, T.: Round-optimal blind signatures in the plain model from classical and quantum standard assumptions. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 404–434. Springer, Heidelberg (Oct 2021). https://doi.org/10.1007/978-3-030-77870-5_15
59. Katz, J.: Digital signatures: Background and definitions. In: Digital Signatures. Springer (2010)
60. Katz, J., Loss, J., Rosenberg, M.: Boosting the security of blind signature schemes. In: Tibouchi, M., Wang, H. (eds.) ASIACRYPT 2021, Part IV. LNCS, vol. 13093, pp. 468–492. Springer, Heidelberg (Dec 2021). https://doi.org/10.1007/978-3-030-92068-5_16
61. Khalili, M., Slamanig, D., Dakhilalian, M.: Structure-preserving signatures on equivalence classes from standard assumptions. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019, Part III. LNCS, vol. 11923, pp. 63–93. Springer, Heidelberg (Dec 2019). https://doi.org/10.1007/978-3-030-34618-8_3
62. Kiltz, E., Pan, J., Wee, H.: Structure-preserving signatures from standard assumptions, revisited. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 275–295. Springer, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-48000-7_14
63. Kiltz, E., Wee, H.: Quasi-adaptive NIZK for linear subspaces revisited. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part II. LNCS, vol. 9057, pp. 101–128. Springer, Heidelberg (Apr 2015). https://doi.org/10.1007/978-3-662-46803-6_4
64. Lindell, Y.: Lower bounds and impossibility results for concurrent self composition. Journal of Cryptology **21**(2), 200–249 (Apr 2008). https://doi.org/10.1007/s00145-007-9015-5
65. Meiklejohn, S., Shacham, H., Freeman, D.M.: Limitations on transformations from composite-order to prime-order groups: The case of round-optimal blind signatures. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 519–538. Springer, Heidelberg (Dec 2010). https://doi.org/10.1007/978-3-642-17373-8_30
66. Micali, S., Reyzin, L.: Improving the exact security of digital signature schemes. Journal of Cryptology **15**(1), 1–18 (Jan 2002). https://doi.org/10.1007/s00145-001-0005-8
67. Nishimaki, R.: Equipping public-key cryptographic primitives with watermarking (or: A hole is to watermark). In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part I. LNCS, vol. 12550, pp. 179–209. Springer, Heidelberg (Nov 2020). https://doi.org/10.1007/978-3-030-64375-1_7
68. Okamoto, T.: Provably secure and practical identification schemes and corresponding signature schemes. In: Brickell, E.F. (ed.) CRYPTO'92. LNCS, vol. 740, pp. 31–53. Springer, Heidelberg (Aug 1993). https://doi.org/10.1007/3-540-48071-4_3
69. Okamoto, T., Ohta, K.: Universal electronic cash. In: Feigenbaum, J. (ed.) CRYPTO'91. LNCS, vol. 576, pp. 324–337. Springer, Heidelberg (Aug 1992). https://doi.org/10.1007/3-540-46766-1_27
70. Pass, R.: Limits of provable security from standard assumptions. In: Fortnow, L., Vadhan, S.P. (eds.) 43rd ACM STOC. pp. 109–118. ACM Press (Jun 2011). https://doi.org/10.1145/1993636.1993652

71. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) CRYPTO'91. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (Aug 1992). https://doi.org/10.1007/3-540-46766-1_9
72. Pointcheval, D.: Strengthened security for blind signatures. In: Nyberg, K. (ed.) EUROCRYPT'98. LNCS, vol. 1403, pp. 391–405. Springer, Heidelberg (May / Jun 1998). https://doi.org/10.1007/BFb0054141
73. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. Journal of Cryptology **13**(3), 361–396 (Jun 2000). https://doi.org/10.1007/s001450010003
74. Pollard, J.M.: Monte carlo methods for index computation (mod p). Mathematics of computation **32**(143), 918–924 (1978)
75. Sahai, A., Waters, B.R.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (May 2005). https://doi.org/10.1007/11426639_27
76. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO'89. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (Aug 1990). https://doi.org/10.1007/0-387-34805-0_22
77. Seo, J.H., Cheon, J.H.: Beyond the limitation of prime-order bilinear groups, and round optimal blind signatures. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 133–150. Springer, Heidelberg (Mar 2012). https://doi.org/10.1007/978-3-642-28914-9_8
78. Shoup, V., Gennaro, R.: Securing threshold cryptosystems against chosen ciphertext attack. In: Nyberg, K. (ed.) EUROCRYPT'98. LNCS, vol. 1403, pp. 1–16. Springer, Heidelberg (May / Jun 1998). https://doi.org/10.1007/BFb0054113
79. Tessaro, S., Zhu, C.: Short pairing-free blind signatures with exponential security. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part II. LNCS, vol. 13276, pp. 782–811. Springer, Heidelberg (May / Jun 2022). https://doi.org/10.1007/978-3-031-07085-3_27
80. Unruh, D.: Quantum proofs of knowledge. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 135–152. Springer, Heidelberg (Apr 2012). https://doi.org/10.1007/978-3-642-29011-4_10
81. Vpn by Google one, explained. https://one.google.com/about/vpn/howitworks (2022), accessed: 2023-02-02
82. Waters, B.R.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (May 2005). https://doi.org/10.1007/11426639_7
83. Yi, X., Lam, K.Y.: A new blind ECDSA scheme for bitcoin transaction anonymity. In: Galbraith, S.D., Russello, G., Susilo, W., Gollmann, D., Kirda, E., Liang, Z. (eds.) ASIACCS 19. pp. 613–620. ACM Press (Jul 2019). https://doi.org/10.1145/3321705.3329816

A Related Work

Since blind signatures were introduced in [31], they have become a vast area of research. Here, we give a short overview of other research directions on blind signatures which are not the focus of this work.

Restricted Concurrency. In the ROM, there are efficient 3-move blind signatures [76, 68, 5]. Notably, these schemes do not rely on pairings. In their seminal work, Pointcheval and Stern [73] show that [68] is secure for at most polylog-many signing sessions. Their analysis was generalized and refined in [52]. A similar result was shown for [5] in [56]. A recent attack [15] shows that the polylog upper bound on concurrent signing sessions is tight.

Generic Group Model and ROM. In pairing-free generic groups and the ROM, there are 3-move blind signatures [1, 57, 42, 79] that avoid the attack given in [15]. These constructions are practical, and notably [79, 57] provide full concurrent security. At least for now, the security argument relies on generic groups *and* random oracles. Also, it seems that this approach is inherently not round-optimal.

In the ROM, [10] shows sequential security of [1] without generic groups, but the signer needs to ensure that at most one signing session is open at all times.

Boosting Transforms. A recent line of work [60, 30] based on [72] provides generic boosting transformations for blind signatures with limited concurrent security. In the pairing-free setting, the obtained blind signatures are *not* round-optimal, and computational efficiency scales linearly with number of signed signatures. [51] provides a concrete pairing-based construction in the ROM (cf. Section 1 for more details).

Trusted Setup. In the pairing setting, there are several blind signatures with trusted setup [65, 77, 61] in the standard model. As opposed to [18, 4] (discussed in Section 1), these schemes are less practical and the trusted setup is *structured*, i.e., non-uniform.

Complexity Leveraging. There are round-optimal blind signatures [47, 46] that circumvent impossibility results to construct standard model blind signatures via complexity leveraging. [58] manages to avoid complexity leveraging by relying on both post-quantum assumptions and classical assumptions. The aforementioned schemes are mainly of theoretical interest and not practical.

Interactive assumptions. There are several blind signatures [31, 13, 19, 7] secure under interactive assumptions and in the ROM. Also, there are constructions in the standard model [2, 41, 40, 48] which rely on tailored interactive hardness assumptions. These constructions have good efficiency, but their security is based on strong assumptions. Further, [2] first notices that Fischlin blind signatures can be instantiated by combining sufficiently algebraic signatures and GOS proofs. Their initial construction relies on a signature based on non-interactive assumptions. Interestingly, their approach yields a scheme with with 0.4 KB communication and 2.7 KB signature size under standard assumptions in the ROM, if instantiated with Jutla-Roy signatures [55]. With this approach, the signature size remains an order of magnitude larger compared to $\mathsf{BS}_{\mathsf{Rnd}}$ (cf. Section 3) and $\mathsf{BS}_{\mathsf{BB}}$ (cf. Section 5). Finally, two constructions were proposed that rely on an NIZK for proving relations of a concrete hash function [17, 43], where the security of the underlying signature scheme is assumed for this fixed choice of hash function. This approach leads to efficient signatures, but slow signing times due to the proof.