PI-Cut-Choo and Friends: Compact Blind Signatures via Parallel Instance Cut-and-Choose and More

Rutchathon Chairattana-Apirom[2], Lucjan Hanzlik[1], Julian Loss[1], Anna Lysyanskaya[2], and Benedikt Wagner[1]

[1] CISPA Helmholtz Center for Information Security
Saarbrücken, Germany
{hanzlik,loss,benedikt.wagner}@cispa.de
[2] Brown University
Providence RI 02906, USA
rutchathon_chairattana-apirom@alumni.brown.edu,anna_lysyanskaya@brown.edu

Abstract. Blind signature schemes are one of the best-studied tools for privacy-preserving authentication. Unfortunately, known constructions of provably secure blind signatures either rely on non-standard hardness assumptions, or require parameters that grow linearly with the number of concurrently issued signatures, or involve prohibitively inefficient general techniques such as general secure two-party computation.

Recently, Katz, Loss and Rosenberg (ASIACRYPT'21) gave a technique that, for the security parameter n, transforms blind signature schemes secure for $O(\log n)$ concurrent executions of the blind signing protocol into ones that are secure for any $\mathsf{poly}(n)$ concurrent executions.

This transform has two drawbacks that we eliminate in this paper: 1) the communication complexity of the resulting blind signing protocol grows linearly with the number of signing interactions; 2) the resulting schemes inherit a very loose security bound from the underlying scheme and, as a result, require impractical parameter sizes.

In this work, we give an improved transform for obtaining a secure blind signing protocol tolerating any $\mathsf{poly}(n)$ concurrent executions from one that is secure for $O(\log n)$ concurrent executions. While preserving the advantages of the original transform, the communication complexity of our new transform only grows logarithmically with the number of interactions. Under the CDH and RSA assumptions, we improve on this generic transform in terms of concrete efficiency and give (1) a BLS-based blind signature scheme over a standard-sized group where signatures are of size roughly 3 KB and communication per signature is roughly 120 KB; and (2) an Okamoto-Guillou-Quisquater-based blind signature scheme with signatures and communication of roughly 9 KB and 8 KB, respectively.

Keywords. Blind Signatures, Standard Assumptions, Random Oracle Model, Cut-and-Choose.

1 Introduction

In 1982, David Chaum introduced blind signature schemes in the context of electronic cash [9]. A blind signature scheme is a cryptographic primitive in which a signer can interactively sign a message held by a user. Informally, a blind signature scheme must satisfy two security requirements [23,30]. *Blindness:* the signer should not be able to see what message is being signed. *Unforgeability:* The user should only be able to obtain valid signatures by interacting with the signer. Classical applications of blind signature schemes include e-cash [9,28], anonymous credentials [6,7] and e-voting [19]. Recently, blind signatures have also been used to add privacy features to blockchain-based systems [22]. Despite this variety of promising applications, the current state-of-the art is unsatisfactory. This is because even in the random oracle model, schemes with reasonable efficiency are either based on non-standard assumptions [4,2,11] or have parameters that grow linearly in the number of concurrent signing interactions [30,20,3,25]. The main goal of this work is to construct blind signature schemes from well-established assumptions with concurrent security and practically efficient parameter sizes.

State-of-the-Art. Juels, Luby and Ostrovsky showed that blind signature schemes can be built generically from any secure signature scheme using secure two-party computation [23]. Their construction was only shown secure when signatures were issued *sequentially.* However, typically one aims for the stronger notion of concurrent security. Fischlin [10] achieved this by giving universally composable blind signatures from commitment schemes and UC zero-knowledge proofs; but it is not clear how to instantiate these generic constructions efficiently. While it is tempting to instantiate these schemes with efficient signature schemes in the random oracle model, the security implications of such an instantiation are unclear. This is because such an instantiation would imply the use of the random oracle as a circuit, which constitutes a non-standard use of the random oracle model. We refer to the recent work of [1] which discusses these issues in more detail.

In the standard model, a variety of blind signature schemes have been proposed. These schemes are either inefficient as they rely on complexity leveraging [14] or rely on strong q-type or non-interactive assumptions [27,15,11,16].

Unfortunately, even in the random oracle model, the situation does not improve much. While there are simple constructions [4,2,30,20,21], they either require similar non-standard assumptions as their standard model counterparts [4,2] or support only a very small number of signatures per public key [30,20,21,3].

As a first step to overcome these limitations, Katz, Loss, and Rosenberg (KLR) [25] showed how to use a cut-and-choose technique to boost the security of these blind signature schemes in the random oracle model. Their approach is based on an early work by Pointcheval [29]. The resulting schemes support polynomially many concurrent signature interactions and are based on standard assumptions. However, the communication between the signer and the user still grows linearly with the number of signature interactions, which renders the scheme impractical.

We note that relying on the algebraic or generic group model [12,32] yields better composition and efficiency, as recent works [13,24,33] show. However, these models are best avoided as they are non-standard.

Our Goal. In this work, we advance the state of the art by giving the first blind signature schemes in the random oracle model that do not suffer from any of the above drawbacks. Our main research question can be summarized as follows:

Are there practical and concurrently secure blind signatures from well-established hardness assumptions which support polynomially many signatures?

1.1 Starting Point: The Basic Boosting Transform

We answer this question in the affirmative. We propose several new techniques which reduce the size and communication complexity of blind signatures in the random-oracle model.

Before we explain our techniques, we briefly recall the KLR transform [25], which will serve as our starting point. The KLR transform can be applied to a blind signature scheme BS in which the user sends a single message and which supports a logarithmic number of signing sessions. The transformed scheme CCBS supports polynomially many signing sessions and achieves the same notion of blindness as BS. We briefly recall the main ideas of CCBS before explaining our improved version:

- In the N^{th} signing interaction, the Signer and the User initiate N sessions of the underlying scheme BS. In the i^{th} session, a commitment μ_i of the actual message is signed.
- The User commits to its randomness ρ_i for the i^{th} session using a commitment $\mathsf{com}_i = \mathsf{H}(\rho_i)$, where H is a hash function (modeled as a random oracle). It sends com_i together with its (only) message in the i^{th} session of BS.
- The Signer picks a session $J \in [N]$ uniformly at random and has the User open the randomness to all commitments $\mathsf{com}_i, i \in [N] \setminus \{J\}$.
- If the User cannot open one of these commitments, the Signer aborts. Otherwise, the Signer and User complete the J^{th} session as in BS.

The proof of one-more unforgeability for CCBS is by reduction to the one-more unforgeability of BS. The reduction's goal is to turn a one-more forgery against CCBS into a one-more forgery against BS. To do so, the reduction must answer all signing queries of the User without knowing the secret key sk of the Signer in BS. It is further restricted by the fact that it may invoke the Signing oracle in the underlying security game for BS only logarithmically many times.

To bypass these restrictions, the reduction heavily relies on its capability of observing the inputs to the random oracle and programming it accordingly. Suppose that the the User behaves honestly in Session J, i.e., it uses the randomness in com_J to compute its message in the J^{th} session of BS. Then the reduction can extract the random coins from the commitments and use random oracle

programming to complete this session without knowing sk. If, on the other hand, the User cheats, then the reduction can not use this technique and must ask the Signing oracle in BS for help.

KLR's key observation is that the probability of such a (successful) cheat is at most $1/N$ in the N^{th} signing session. Thus, the expected number of successful cheats in p interactions is at most $\sum_{N \leq p+1} 1/N < \ln(p+1)$. Using the Chernoff bound, one can show that with overwhelming probability, the number of successful cheats is reasonably close to this expectation. Hence, the signing oracle in the underlying OMUF game of BS needs to be invoked only a logarithmic number of times.

Limitations. Although CCBS exponentially increases the security of the underlying blind signature scheme BS, this comes at a steep price in terms of efficiency: the communication in the resulting scheme grows linearly with the number N of issued signatures. This arguably renders CCBS impractical. In addition, the number of times that the reduction from one-more unforgeability of BS requires invoking the underlying signing oracle behaves as $\ln(1/\epsilon)$. Here, ϵ is the advantage of the adversary in breaking one-more unforgeability of CCBS. For small sizes of ϵ (say, 2^{-128}), this leads to impractical parameter sizes for BS. As an example, if CCBS is applied to the Schnorr blind signature scheme, our calculations show that the resulting scheme will require groups with a 12000 bit representation.

1.2 Our Contribution: Improved Boosting Transforms

As our first contribution, we present a new generic transform to boost the security of blind signature schemes fitting the linear function family framework of Hauck, Kiltz and Loss (HKL) [20]. This is based on three insights, as follows. (1) In the N^{th} signing session, the User can derive the random coins for the i^{th} instance via $\rho_i := \mathsf{PRF}(k, i)$, where PRF denotes a *puncturable pseudorandom function* [31][3]. The User can now commit to all its randomness as in CCBS. To open the commitments $\mathsf{com}_i, i \in [N] \setminus \{J\}$, the User provides the punctured key k_J. From this key, the Signer can deterministically recompute all the commitments, save for com_J. (2) We use a randomness homomorphic commitment scheme to construct the μ_i as rerandomizations of one initial commitment μ_0 that is sent to the signer. The rerandomization is also determined by PRF, which implies that k_J also reveals μ_i for $i \neq J$ without revealing μ_J. (3) To compress the N messages from the Signer to the User, we use the homomorphic properties of HKL blind signatures and derive N first messages of the underlying blind signature from $\log N$ randomly chosen ones. These insights allow us to lower the communication complexity of the resulting blind signature scheme from linear to logarithmic in the number N of signing sessions[4].

Our results have better blindness guarantees than schemes from the KLR transform. A KLR-transformed blind signature scheme has the same blindness

[3] We instantiate PRF efficiently using random oracles [18].

[4] In a different context, namely secure multi-party computation, the combination of puncturable pseudorandom functions and cut-and-choose has been used before

as its underlying scheme; for many of the schemes underlying it, only so-called *honest signer* blindness was known [20], where the Signer's public key is generated honestly. A much more desirable notion is *malicious signer* blindness, in which the Signer is free to pick his public key adversarially. We show how to achieve this notion using a three step approach. First, we show that the schemes in [20] satisfy a slightly stronger (artificial) notion of blindness without any modification. In this intermediate notion (called *semi-honest signer* blindness), the Signer provides the random coins to generate the public key to the experiment. Next, we show that our improved boosting transform preserves any notion of blindness, including the new one. We then show that by having the signer prove knowledge of the random coins we can transform any scheme that satisfies the intermediate notion into a scheme that satisfies malicious signer blindness.

Practical Schemes from CDH and RSA. Even though our generic transform is an exponential improvement over the state-of-the-art, it still results in schemes that require mega bytes of communication when the number of signatures becomes large (say 2^{30}). On top of this, our generic transform would require large (to the point of being currently impractical) group sizes. To overcome these limitations, we give concrete, 128-bit secure, practical blind signature schemes that satisfy concurrent one-more unforgeability under the CDH and RSA assumptions. We summarize the parameter sizes in Table 1.

Scheme	Nr. of Signatures	$\lvert\mathsf{pk}\rvert$	$\lvert\sigma\rvert$	a	b	Max
$\mathsf{BS}_{\mathsf{RSA}}$ (Section 5)	2^{20}	18.37	7.91	0.02	7.11	7.51
$\mathsf{BS}_{\mathsf{RSA}}$ (Section 5)	2^{30}	18.74	8.66	0.02	7.48	8.08
$\mathsf{PIKA}_{\mathsf{CDH}}$ (Section 4)	2^{20}	3.68	3.16	3.05	26.50	87.50
$\mathsf{PIKA}_{\mathsf{CDH}}$ (Section 4)	2^{30}	3.90	3.16	3.05	26.73	118.20

Table 1. Concrete efficiency of our schemes supporting a given number of signatures and 128 bit security. Here, communication complexity is given as $a \cdot \log(N) + b$, where N is the number of issued signatures so far. Column Max shows the communication complexity for the maximum N. All sizes are in KiloBytes.

Our scheme from CDH is statistically malicious signer blind and builds on Boldyreva's blind version of the BLS signature scheme [4] (which is secure under a one-more version of CDH). We observe that by running our boosting transform for several independent keys in parallel, we can ensure that with overwhelming probability, there will be at least one key for which the User is never able to cheat the Signer. We can leverage this into a reduction that embeds the challenge key pk randomly into one of these keys. Then, with high probability, no cheat ever occurs for pk and the reduction can carry out the simulation without having to ever invoke the signing oracle from the underlying one-more unforgeability experiment. This makes it possible to run the scheme with a standard sized group and assuming no more than hardness of the CDH problem. To reduce the size of our resulting signatures, we can use the aggregatability of the BLS scheme.

Overall, our scheme from CDH supports 2^{30} signatures at a size of 3KB and 120KB communication per signature.

Our scheme from RSA does not use parallel repetitions to reduce parameter sizes. Instead, we use the trapdoor provided by the RSA system to improve communication complexity of the generic transform. In this way, the Signer can send a single seed from which the User can deterministically derive several values. The Signer, who needs to know the preimages of these values, can then simply use its trapdoor to learn these preimages and proceed with the remainder of the signing protocol. Overall, our scheme from RSA is statistically semi-honest signer blind and supports 2^{30} signatures at a more balanced size of 9KB per signature and 8KB communication per signature. To upgrade it to malicious signer blindness we can either rely on generic proof systems, or on more efficient ones based on quadratic residuosity [17] or discrete logarithms [8].[5] We emphasize, however, that using proofs from general complexity assumptions may be sufficiently efficient in our context, as the proofs only have to be generated and verified once upon registering the Signer's public key. Therefore, they do not affect the complexity of the signing protocol or the size of our signatures.

2 Preliminaries

The security parameter is $n \in \mathbb{N}$. All algorithms get 1^n implicitly as input. For a finite set S, we write $x \leftarrow\!\!\$\, S$ if x is sampled uniformly at random from S. For a distribution $\mathcal{D}$, we write $x \leftarrow \mathcal{D}$ if x is sampled according to $\mathcal{D}$. For a (probabilistic) algorithm $\mathcal{A}$, we write $y \leftarrow \mathcal{A}(x)$, if y is output from $\mathcal{A}$ on input x with uniformly sampled random coins. We write $y = \mathcal{A}(x;\rho)$ to make the random coins ρ explicit, and $y \in \mathcal{A}(x)$ means that y is a possible output of $\mathcal{A}(x)$. An algorithm is said to be PPT if its running time can be bounded by a polynomial in its input size. We say that a function $f : \mathbb{N} \to \mathbb{R}_+$ is negligible in its input n, if $f \in n^{-\omega(1)}$. For a security game $\mathbf{G}$, we write $\mathbf{G} \Rightarrow b$ to indicate that $\mathbf{G}$ outputs b. We denote the first K natural numbers by $[K] := \{1, \ldots, K\}$, Euler's totient function by φ and the group of units in $\mathbb{Z}_N$ by $\mathbb{Z}_N^*$.

Next, we introduce the cryptographic primitives that we need. We make use of the well-known computational assumptions CDH and RSA. For the definition of puncturable pseudorandom functions, we follow [31].

Definition 1 (Puncturable Pseudorandom Function). *A puncturable pseudorandom function (PPRF) is defined to be a triple of PPT algorithms* $\mathsf{PRF} = (\mathsf{Gen}, \mathsf{Puncture}, \mathsf{Eval})$ *with the following syntax:*

- $\mathsf{Gen}(1^n, 1^{d(n)})$ *takes as input the security parameter* 1^n*, an input length* $1^{d(n)}$ *and outputs a key* k*.*
- $\mathsf{Puncture}(k, X)$ *takes as input a key* k *and a polynomial size set* $\emptyset \neq X \subseteq \mathcal{D} = \{0,1\}^{d(n)}$ *and outputs a punctured key* k_X*.*

[5] If we rely on these proof systems, our scheme can be proven secure assuming that both the RSA assumption and either of these assumptions hold.

– $\mathsf{Eval}(k, x)$ *is deterministic, takes a key* k *and an element* $x \in \mathcal{D}$ *as input and outputs an element* $r \in \mathcal{R} = \{0,1\}^{r(n)}$.

Further, the following security and completeness properties should hold:

– ***Completeness of Puncturing.*** *For any* $d(n) = \mathsf{poly}(n), X \subseteq \{0,1\}^{d(n)}$, *any* $k \in \mathsf{Gen}(1^n, 1^{d(n)})$, *any* $k_X \in \mathsf{Puncture}(k, X)$ *and any* $x' \notin X$ *we have* $\mathsf{Eval}(k, x') = \mathsf{Eval}(k_X, x')$.
– ***Pseudorandomness.*** *For any* $d(n) = \mathsf{poly}(n)$ *and any PPT algorithm* $\mathcal{A}$ *the following is negligible:*

$$\left|\Pr\left[\mathcal{A}(St, k_X, (r_x)_{x\in X}) = 1 \;\middle|\; \begin{array}{l}(X, St) \leftarrow \mathcal{A}(1^n), k \leftarrow \mathsf{Gen}(1^n, 1^{d(n)}),\\ k_X \leftarrow \mathsf{Puncture}(k, X),\\ r_x := \mathsf{Eval}(k, x) \text{ for } x \in X\end{array}\right]\right.$$
$$\left.-\Pr\left[\mathcal{A}(St, k_X, (r_x)_{x\in X}) = 1 \;\middle|\; \begin{array}{l}(X, St) \leftarrow \mathcal{A}(1^n), k \leftarrow \mathsf{Gen}(1^n, 1^{d(n)}),\\ k_X \leftarrow \mathsf{Puncture}(k, X),\\ r_x \leftarrow\!\!\$ \{0,1\}^{r(n)} \text{ for } x \in X\end{array}\right]\right|.$$

We define a special type of perfectly hiding commitment scheme in which the randomness can be rerandomized publicly. Such commitment schemes can be easily constructed from standard assumptions. For that, we refer to the full version.

Definition 2 (Randomness Homomorphic Commitment Scheme). *A randomness homomorphic commitment scheme is a tuple of PPT algorithms* $\mathsf{CMT} = (\mathsf{Gen}, \mathsf{Com}, \mathsf{Translate})$ *with the following syntax:*

– $\mathsf{Gen}(1^n)$ *takes as input the security parameter* 1^n *and outputs a commitment key* ck. *We assume that* ck *implicitly defines a message space* $\mathcal{M}_{\mathsf{ck}}$ *and a randomness space* $\mathcal{R}_{\mathsf{ck}}$. *Further, we assume that* $\mathcal{R}_{\mathsf{ck}}$ *is a group with respect to an efficiently computable group operation* $+$.
– $\mathsf{Com}(\mathsf{ck}, x; r)$ *takes as input a key* ck, *an element* $x \in \mathcal{M}_{\mathsf{ck}}$, *a randomness* $r \in \mathcal{R}_{\mathsf{ck}}$ *and outputs a commitment* $\mu \in \{0,1\}^*$.
– $\mathsf{Translate}(\mathsf{ck}, \mu, r)$ *is deterministic, takes a key* ck, *a commitment* $\mu \in \{0,1\}^*$, *and a randomness* $r \in \mathcal{R}_{\mathsf{ck}}$ *as input and outputs a commitment* μ'.

Further, the following security and completeness properties should hold:

– ***Completeness of Translation.*** *For any* $\mathsf{ck} \in \mathsf{Gen}(1^n)$, *and* $x \in \mathcal{M}_{\mathsf{ck}}$ *and any* $r, r' \in \mathcal{R}_{\mathsf{ck}}$, *we have*

$$\mathsf{Translate}(\mathsf{ck}, \mathsf{Com}(\mathsf{ck}, x; r), r') = \mathsf{Com}(\mathsf{ck}, x; r + r').$$

– ***Perfectly Hiding.*** *For any key* ck *and any* $x_0, x_1 \in \mathcal{M}_{\mathsf{ck}}$, *the following distributions are identical:*

$$\{(\mathsf{ck}, x_0, x_1, \mu) \mid r \leftarrow\!\!\$ \mathcal{R}_{\mathsf{ck}}, \mu := \mathsf{Com}(\mathsf{ck}, x_0; r)\} \text{ and}$$
$$\{(\mathsf{ck}, x_0, x_1, \mu) \mid r \leftarrow\!\!\$ \mathcal{R}_{\mathsf{ck}}, \mu := \mathsf{Com}(\mathsf{ck}, x_1; r)\}.$$

– ***Computationally Binding.*** *For any PPT algorithm $\mathcal{A}$, the following is negligible:*

$$\Pr\left[\mathsf{Com}(\mathsf{ck}, x_0; r_0) = \mathsf{Com}(\mathsf{ck}, x_1; r_1) \wedge x_0 \neq x_1 \;\middle|\; \begin{array}{l}\mathsf{ck} \leftarrow \mathsf{Gen}(1^n),\\ (x_0, r_0, x_1, r_1) \leftarrow \mathcal{A}(\mathsf{ck})\end{array}\right].$$

Next, we define the primitive of interest, namely blind signature scheme.

Definition 3 (Blind Signature Scheme). *A blind signature scheme* $\mathsf{BS} = (\mathsf{Gen}, \mathsf{S}, \mathsf{U}, \mathsf{Ver})$ *is a quadruple of PPT algorithms, where*

– $\mathsf{Gen}(1^n)$ *takes as input the security parameter* 1^n *and outputs a pair of keys* $(\mathsf{pk}, \mathsf{sk})$*. We assume that the public key* pk *defines a message space* $\mathcal{M} = \mathcal{M}_{\mathsf{pk}}$ *implicitly.*
– S *and* U *are interactive algorithms, where* S *takes as input a secret key* sk *and* U *takes as input a key* pk *and a message* $\mathsf{m} \in \mathcal{M}$*. After the execution,* U *returns a signature* σ *and we write* $(\bot, \sigma) \leftarrow \langle \mathsf{S}(\mathsf{sk}), \mathsf{U}(\mathsf{pk}, \mathsf{m})\rangle$*.*
– $\mathsf{Ver}(\mathsf{pk}, \mathsf{m}, \sigma)$ *is deterministic and takes as input public key* pk*, message* $\mathsf{m} \in \mathcal{M}$*, and a signature* σ*, and returns* $b \in \{0, 1\}$*.*

We say that BS *is complete if for all* $(\mathsf{pk}, \mathsf{sk}) \in \mathsf{Gen}(1^n)$ *and all* $\mathsf{m} \in \mathcal{M}_{\mathsf{pk}}$ *it holds that*

$$\Pr\left[\mathsf{Ver}(\mathsf{pk}, \mathsf{m}, \sigma) = 1 \mid (\bot, \sigma) \leftarrow \langle \mathsf{S}(\mathsf{sk}), \mathsf{U}(\mathsf{pk}, \mathsf{m})\rangle\right] = 1.$$

Definition 4 (One-More Unforgeability). *Let* $\mathsf{BS} = (\mathsf{Gen}, \mathsf{S}, \mathsf{U}, \mathsf{Ver})$ *be a blind signature scheme and* $\ell\colon \mathbb{N} \to \mathbb{N}$*. For an adversary* $\mathcal{A}$*, we consider the following game* ℓ-$\mathbf{OMUF}^{\mathcal{A}}_{\mathsf{BS}}(n)$*:*

1. *Sample keys* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$*.*
2. *Let* O *be an interactive oracle simulating* $\mathsf{S}(\mathsf{sk})$*. Run*

$$((\mathsf{m}_1, \sigma_1), \ldots, (\mathsf{m}_k, \sigma_k)) \leftarrow \mathcal{A}^{O}(\mathsf{pk}),$$

   where $\mathcal{A}$ *can query* O *in an arbitrarily interleaved way and complete at most* $\ell = \ell(n)$ *of the interactions with* O*.*
3. *Output* 1 *if and only if all* $\mathsf{m}_i, i \in [k]$ *are distinct,* $\mathcal{A}$ *completed at most* $k - 1$ *interactions with* O *and for each* $i \in [k]$ *it holds that* $\mathsf{Ver}(\mathsf{pk}, \mathsf{m}_i, \sigma_i) = 1$*.*

We say that BS *is* ℓ*-one-more unforgeable (*ℓ-$\mathsf{OMUF}*), if for every PPT algorithm* $\mathcal{A}$ *the following advantage is negligible:*

$$\Pr\left[\ell\text{-}\mathbf{OMUF}^{\mathcal{A}}_{\mathsf{BS}}(n) \Rightarrow 1\right].$$

Further, we say that BS *is one-more unforgeable (*OMUF*), if it is* ℓ-OMUF *for all polynomial* ℓ*.*

We note that from a practical perspective, it is sufficient to focus on ℓ-OMUF for some large but a priori bounded ℓ (e.g. $\ell = 2^{30}$), while full OMUF is more of theoretical interest.

Definition 5 (Blindness). *Consider a blind signature scheme* $\mathsf{BS} = (\mathsf{Gen}, \mathsf{S}, \mathsf{U}, \mathsf{Ver})$*. For an adversary* $\mathcal{A}$ *and bit* $b \in \{0,1\}$*, consider the following game* $\mathbf{BLIND}^{\mathcal{A}}_{b,\mathsf{BS}}(n)$*:*

1. *Sample* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$ *and run* $(\mathsf{m}_0, \mathsf{m}_1, St) \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{sk})$*.*
2. *Let* O_0 *be an interactive oracle simulating* $\mathsf{U}(\mathsf{pk}, \mathsf{m}_b)$ *and* O_1 *be an interactive oracle simulating* $\mathsf{U}(\mathsf{pk}, \mathsf{m}_{1-b})$*. Run* $\mathcal{A}$ *on input* St *with arbitrary interleaved one-time access to each of these oracles, i.e.* $St' \leftarrow \mathcal{A}^{O_0, O_1}(St)$*.*
3. *Let* σ_b, σ_{1-b} *be the local outputs of* O_0, O_1*, respectively. If* $\sigma_0 = \ \perp$ *or* $\sigma_1 = \ \perp$*, then run* $b' \leftarrow \mathcal{A}(St', \perp, \perp)$*. Else, obtain a bit* b' *from* $\mathcal{A}$ *on input* σ_0, σ_1*, i.e. run* $b' \leftarrow \mathcal{A}(St', \sigma_0, \sigma_1)$*.*
4. *Output* b'*.*

We say that BS *satisfies honest signer blindness, if for every PPT algorithm* $\mathcal{A}$ *the following advantage is negligible:*

$$\left| \Pr\left[\mathbf{BLIND}^{\mathcal{A}}_{0,\mathsf{BS}}(n) \Rightarrow 1\right] - \Pr\left[\mathbf{BLIND}^{\mathcal{A}}_{1,\mathsf{BS}}(n) \Rightarrow 1\right] \right|.$$

We also consider semi-honest and malicious signer blindness, where we modify the game in the following way:

- *For semi-honest signer blindness,* $(\mathsf{pk}, \mathsf{sk})$ *is not sampled by the game, but* $\mathcal{A}$ *outputs random coins* ρ *in addition to* $\mathsf{m}_0, \mathsf{m}_1$*. Then, the game defines* $(\mathsf{pk}, \mathsf{sk})$ *via* $(\mathsf{pk}, \mathsf{sk}) := \mathsf{Gen}(1^n; \rho)$*.*
- *For malicious signer blindness,* $(\mathsf{pk}, \mathsf{sk})$ *is not sampled by the game, but* $\mathcal{A}$ *outputs* pk *in addition to* $\mathsf{m}_0, \mathsf{m}_1$*.*

Semi-honest signer blindness is a non-standard notion and lies inbetween honest and malicious signer blindness. We claim that any semi-honest signer blind scheme can be transformed into a malicious signer blind scheme while preserving one-more unforgeability. The high-level idea is to append a non-interactive zero-knowledge proof-of-knowledge to the public key. This proof shows that the signer knows corresponding random coins that generate the key. The rest of the scheme does not change, and thus the transformation is very efficient. For details, we refer to the full version.

We will now introduce linear function families, following [20].

Definition 6 (Linear Function Family). *A linear function family* LF *is a given by a tuple of algorithms* $\mathsf{LF} = (\mathsf{PGen}, \mathsf{F}, \Psi)$ *with the following properties:*

- $\mathsf{PGen}(1^n)$ *returns system parameters* par *which define abelian groups* $\mathcal{S}, \mathcal{D}, \mathcal{R}$ *with* $|\mathcal{S}|, |\mathcal{R}| \geq 2^n$ *and there exists scalar multiplication* $\cdot : \mathcal{S} \times \mathcal{D} \to \mathcal{D}$ *with* $s \cdot (x + x') = s \cdot x + s \cdot x'$ *for all* $s \in \mathcal{S}$ *and* $x, x' \in \mathcal{D}$*. The same applies for* $\mathcal{R}$*. Note that it is not necessarily true that* $(s + s') \cdot x = s \cdot x + s' \cdot x$*.*
- $\mathsf{F}_{\mathsf{par}}(x)$ *is deterministic, takes as input an element* $x \in \mathcal{D}$*, and returns an element in* $y \in \mathcal{R}$*. We require that:*
  - *For all* $s \in \mathcal{S}, x, y \in \mathcal{D}$, $\mathsf{F}_{\mathsf{par}}(s \cdot x + y) = s \cdot \mathsf{F}_{\mathsf{par}}(x) + \mathsf{F}_{\mathsf{par}}(y)$*.*
  - $\mathsf{F}_{\mathsf{par}}$ *has a pseudo torsion-free element in the kernel, i.e. there exists* $z^* \in \mathcal{D}$ *such that* $\mathsf{F}_{\mathsf{par}}(z^*) = 0$ *and for all distinct* $s, s' \in \mathcal{S}, s \cdot z^* \neq s' \cdot z^*$*.*

- $\mathsf{F}_{\mathsf{par}}$ *is smooth, i.e. if* $x \leftarrow \mathcal{D}$ *is sampled uniformly,* $\mathsf{F}_{\mathsf{par}}(x)$ *has uniform distribution in* $\mathcal{R}$*.*

– $\Psi_{\mathsf{par}}(y, s, s')$ *is deterministic, takes as inputs* $y \in \mathcal{R}$*, and* $s, s' \in \mathcal{S}$*, and returns a value* $x \in \mathcal{D}$*. The function satisfies for all* y *in the range of* $\mathsf{F}_{\mathsf{par}}$ *and* $s, s' \in \mathcal{S}$*,*

$$(s + s') \cdot y = s \cdot y + s' \cdot y + \mathsf{F}_{\mathsf{par}}(\Psi_{\mathsf{par}}(y, s, s')).$$

Intuitively, the function Ψ_{par} *corrects for the fact that the group operation in* $\mathcal{S}$ *may not distribute over* $\mathcal{R}$*. When it is clear from the context, we will omit the subscript* par*.*

As in [25], we define preimage resistance for a linear function family. For the related notion of collision resistence, we refer to the full version and [25].

Definition 7 (Preimage Resistance). *A linear function family* LF *is preimage resistant if for any adversary* $\mathcal{A}$*, the following advantage is negligible:*

$$\Pr\left[\mathsf{F}(x) = \mathsf{F}(x') \mid x \leftarrow \mathcal{D}, x' \leftarrow \mathcal{A}(\mathsf{par}, \mathsf{F}(x))\right].$$

3 An Improved Boosting Transform

Hauck, Kiltz, and Loss [20] introduced a generic construction of a three-move blind signature scheme $\mathsf{BS}[\mathsf{LF}]$ from any linear function family LF and a hash function H modeled as a random-oracle. The main result of [20] is that the linear blind signature scheme $\mathsf{BS}[\mathsf{LF}]$ is ℓ-one-more unforgeable for $\ell = \mathcal{O}(\log n)$. Building on that, Katz, Loss, and Rosenberg [25] presented a boosting transform $\mathsf{CCBS}[\mathsf{LF}]$ that turns this logarithmic security into polynomial security. In this section, we introduce an improved boosting transform $\mathsf{CCCBS}[\mathsf{LF}]$ that eliminates the drawback of linearly growing communication complexity.

3.1 Overview

We recall the main idea of the boosting transform [25] that turns a linear blind signature scheme $\mathsf{BS}[\mathsf{LF}]$ into a boosted blind signature scheme $\mathsf{CCBS}[\mathsf{LF}]$.

In the scheme $\mathsf{CCBS}[\mathsf{LF}]$, at the onset of the N^{th} interaction, the signer sends the current value of the counter N to the user. Then, user and signer proceed as follows.

1. The user chooses N random strings $\mathsf{ur}_j, j \in [N]$ and N random strings $\varphi_j, j \in [N]$. It prepares N commitments $\mu_j = \mathsf{H}(\mathsf{m}, \varphi_j)$, where H is a random oracle and m is the message to be signed. It also prepares commitments $\mathsf{com}_j = \mathsf{H}(\mathsf{ur}_j, \mu_j)$. Then it sends the commitments com_j to the signer.
2. The user and the signer run N independent sessions of the underlying linear blind signature scheme $\mathsf{BS}[\mathsf{LF}]$, where the user inputs μ_j, ur_j in the j^{th} session. Recall that the scheme $\mathsf{BS}[\mathsf{LF}]$ contains three messages R, c, s.

3. Before the signer sends the last message s_j of the underlying scheme, it chooses a cut-and-choose index $J \in [N]$ at random and asks the user to open all commitments com_j with $j \neq J$.
4. Once the signer knows the values μ_j and randomness ur_j, it runs the user algorithm U to check if the user behaved honestly so far, at least for the sessions $j \neq J$. If there is some session for which this check fails, the signer aborts.
5. The signer sends only s_J to the user. That is, signer and user only complete the J^{th} session. The final signature consists of a signature on μ_J from the underlying scheme $\mathsf{BS}[\mathsf{LF}]$ as well as the randomness φ_J which binds m to μ_J.

We highlight that the communication now grows linearly with the number of issued signatures.

a) In the second message, the user sends N commitments com_j.
b) In the third message, the signer sends N commitments R_j.
c) In the fourth message, the user sends N challenges c_j.
d) In the sixth message, the user opens $N-1$ of the commitments com_j.

Our goal is to eliminate these linear dependencies on N and improve them by an at most logarithmic dependency.

First, we eliminate the linear dependency a) by replacing the commitments $\mathsf{com}_j = \mathsf{H}(\mathsf{ur}_j, \mu_j)$ by a single commitment com_r, which commits to (salted) hashes of all ur_j, μ_j at once. By sending all ur_j for $j \neq J$ and the hash of ur_J, the user can still open this commitment without revealing ur_J.

Next, we focus on d). Here, we let the user generate the randomness $(\mathsf{ur}_j, \varphi_j)$ used for each session using the puncturable pseudorandom function PRF. We replace the unstructured commitment with a randomness homomorphic commitment scheme. This allows us to let the user derive the commitments μ_i as rerandomizations $\mathsf{Com}(\mathsf{m}, \varphi_0 + \varphi_j)$ of one single commitment $\mu_0 = \mathsf{Com}(\mathsf{m}, \varphi_0)$ with randomness φ_j. The user sends commitment μ_0 together with com_r. Now, the user can open the commitment com_r by sending only a punctured key k_J. Intuitively, this preserves blindness, as the punctured key does not reveal anything about the randomness ur_J, φ_J. Using similar tricks, we eliminate c).

To tackle b), we compute the N values R_i of the underlying linear scheme $\mathsf{BS}[\mathsf{LF}]$ as subset sums of a logarithmic number of such values. Then, only these basis values have to be sent.

We end up with a scheme with logarithmic communication complexity, for which the ideas that underlie the original boosting transform still apply.

3.2 Blind Signatures from Linear Function Families

We briefly recall the blind signature scheme $\mathsf{BS}[\mathsf{LF}]$ from a linear function family LF. For more details, we refer the reader to [20] or the full version. For key generation of the blind signature scheme $\mathsf{BS}[\mathsf{LF}]$, parameters $\mathsf{par} \leftarrow \mathsf{LF.PGen}(1^n)$ are generated. Then, a secret key and public key are sampled via $\mathsf{sk} \leftarrow\!\!\$\, \mathcal{D}$ and

$\mathsf{pk} := \mathsf{F}(\mathsf{sk})$, assuming pk implicitly contains par. We present the signature issuing protocol formally in Figure 1. Signatures $\sigma = (c', s')$ for a message m are verified by checking if $c' = \mathsf{H}(\mathsf{m}, \mathsf{F}(s') - c' \cdot \mathsf{pk})$ holds.

$\underline{\mathsf{S}(\mathsf{sk})}$		$\underline{\mathsf{U}(\mathsf{pk}, \mathsf{m})}$
$r \leftarrow\!\!\$\ \mathcal{D}, R := \mathsf{F}(r)$	$\xrightarrow{R}$	$\alpha \leftarrow\!\!\$\ \mathcal{D}, \beta \leftarrow\!\!\$\ \mathcal{S}, R' := R + \mathsf{F}(\alpha) + \beta \cdot \mathsf{pk}$
$s := r + c \cdot \mathsf{sk}$	$\xleftarrow{c}$	$c' := \mathsf{H}(\mathsf{m}, R'), c := c' + \beta$
	$\xrightarrow{s}$	**if** $\mathsf{F}(s) \neq R + c \cdot \mathsf{pk}$: **abort**
		$s' := s + \alpha + \Psi(\mathsf{pk}, c, -c')$
		return $\sigma := (c', s')$

Fig. 1. The signature issuing protocol of the linear blind signature scheme $\mathsf{BS}[\mathsf{LF}]$ for a linear function family LF and a random oracle $\mathsf{H} : \{0,1\}^* \to \mathcal{S}$ [20].

3.3 Construction

In this section, we define our Compact Cut-and-Choose blind signature scheme for a linear function family LF, abbreviated as $\mathsf{CCCBS}[\mathsf{LF}]$. To this end, let $\mathsf{LF} = (\mathsf{PGen}, \mathsf{F}, \Psi)$ be a linear function family, CMT be a randomness homomorphic commitment scheme, and PRF be a puncturable pseudorandom function. For efficient instantiations of CMT and PRF, we refer to the full version. Further, let $\mathsf{H}\colon \{0,1\}^* \to \mathcal{S}, \mathsf{H}_\mathsf{r}\colon \{0,1\}^* \to \{0,1\}^n, \mathsf{H}_x\colon \{0,1\}^* \to \mathcal{D} \times \mathcal{S} \times \mathcal{R}_\mathsf{ck} \times \{0,1\}^{n_\mathsf{PRF}}, \mathsf{H}_c\colon \{0,1\}^* \to \{0,1\}^n$ be random oracles, where $n_\mathsf{PRF} = \Theta(n)$ is a security parameter used for the pseudorandom function.

Key Generation. Algorithm $\mathsf{CCCBS}[\mathsf{LF}].\mathsf{Gen}(1^n)$ is as follows:

1. Sample $\mathsf{ck} \leftarrow \mathsf{CMT}.\mathsf{Gen}(1^n)$ and $\mathsf{par} \leftarrow \mathsf{LF}.\mathsf{PGen}(1^n)$.
2. Sample $\mathsf{sk}' \leftarrow\!\!\$\ \mathcal{D}$, and let $\mathsf{sk} := \mathsf{sk}'$, $\mathsf{pk} = (\mathsf{par}, \mathsf{ck}, \mathsf{pk}' := \mathsf{F}(\mathsf{sk}'))$.
3. Return the public key pk and the secret key sk.

Signature Issuing. The signer and user algorithms S, U are given in Figures 2 and 3, where the S keeps a state (N, ctr) which is initialized as $N := 2 = 2^2 - 2, \mathsf{ctr} := 0$. In each interaction, S atomically increments ctr and, if $\mathsf{ctr} = N$, sets $N := 2N + 2, \mathsf{ctr} := 0$.

Verification. Algorithm $\mathsf{CCCBS}[\mathsf{LF}].\mathsf{Ver}(\mathsf{pk}, \mathsf{m}, \sigma = (c, s, \varphi))$ returns the output of $\mathsf{BS}[\mathsf{LF}].\mathsf{Ver}(\mathsf{pk}', \mathsf{Com}(\mathsf{ck}, \mathsf{m}; \varphi), (c, s))$.

$\mathsf{Check}(\mathsf{pk}, N, \mu_0, \mathsf{com}_\mathsf{r}, \{R_i\}_i, \mathsf{com}_c, J, k_J, c_J, h_J)$

1 : **for** $j \in [N] \setminus \{J\}$:
2 : $\quad \mathsf{prer}_j := \mathsf{PRF.Eval}(k_J, j),\ \mathsf{r}_j := \mathsf{H}_x(\mathsf{prer}_j)$
3 : $\quad$ **parse** $\mathsf{r}_j = (\alpha_j, \beta_j, \varphi_j, \gamma_j) \in \mathcal{D} \times \mathcal{S} \times \mathcal{R}_\mathsf{ck} \times \{0,1\}^{n_\mathsf{PRF}}$
4 : $\quad \tilde{R}_j := \sum_{i \in S_j} R_j, \quad \mu_j := \mathsf{Translate}(\mathsf{ck}, \mu_0, \varphi_j)$
5 : $\quad c_j := \mathsf{H}(\mu_j, \tilde{R}_j + \mathsf{F}(\alpha_j) + \beta_j \cdot \mathsf{pk}') + \beta_j$
6 : **if** $\mathsf{com}_\mathsf{r} \neq \mathsf{H}_\mathsf{r}(\mathsf{H}_\mathsf{r}(\mathsf{r}_1), \ldots, \mathsf{H}_\mathsf{r}(\mathsf{r}_{J-1}), h_J, \mathsf{H}_\mathsf{r}(\mathsf{r}_{J+1}), \ldots, \mathsf{H}_\mathsf{r}(\mathsf{r}_N))$: **return** 0
7 : **if** $\mathsf{com}_c \neq \mathsf{H}_c(c_1, \ldots, c_N)$: **return** 0
8 : **return** 1

Fig. 2. The algorithm Check used in the issuing protocol of CCCBS[LF], where $\mathsf{H} : \{0,1\}^* \to \mathcal{S}, \mathsf{H}_\mathsf{r}, \mathsf{H}_c : \{0,1\}^* \to \{0,1\}^n$, and $\mathsf{H}_x : \{0,1\}^* \to \mathcal{D} \times \mathcal{S} \times \mathcal{R}_\mathsf{ck} \times \{0,1\}^{n_\mathsf{PRF}}$ are random oracles. The set S_j is defined as $\{i \in [l] : i^{th}\text{-bit of } j \text{ is } 1\}$.

3.4 Security Analysis

Completeness of CCCBS[LF] follows by inspection. We show blindness and one-more unforgeability.

Theorem 1. *Let* PRF *be a puncturable pseudorandom function,* LF *be a linear function family, and* CMT *be a randomness homomorphic commitment scheme. Let* $\mathsf{H}_\mathsf{r} : \{0,1\}^* \to \{0,1\}^n, \mathsf{H}_x : \{0,1\}^* \to \mathcal{D} \times \mathcal{S} \times \mathcal{R}_\mathsf{ck} \times \{0,1\}^{n_\mathsf{PRF}}$ *be random oracles. If* BS[LF] *satisfies honest, semi-honest, or malicious signer blindness, then* CCCBS[LF] *satisfies honest, semi-honest, or malicious signer blindness, respectively.*

Concretely, for any adversary that uses N^L *and* N^R *as the counters in its executions with the user, runs in time* t*, has advantage* ϵ *in the blindness game and makes at most* $Q_{\mathsf{H}_x}, Q_{\mathsf{H}_\mathsf{r}}$ *queries to* $\mathsf{H}_x, \mathsf{H}_\mathsf{r}$ *respectively, there exists an adversary against blindness of* BS[LF] *running in time* t *with advantage* $\epsilon_{\mathsf{BS[LF]}}$ *such that*

$$\epsilon \leq N^L N^R \left(\frac{4(Q_{\mathsf{H}_x} + Q_{\mathsf{H}_\mathsf{r}})}{2^{n_\mathsf{PRF}}} + 4\epsilon_\mathsf{PRF} + \epsilon_{\mathsf{BS[LF]}} \right),$$

where ϵ_PRF *is the advantage of an adversary against the security of* PRF *with input length* $\max\{\log(N^L), \log(N^R)\}$ *puncturing at one point.*

We give a intuition of the proof and postpone details to the full version. The strategy is to apply a sequence of changes to the user oracles, such that final game is independent of bit b. In a first step, we guess the cut-and-choose index J. Then, we compute the commitment μ_J directly instead of deriving it from the commitment μ_0. Next, we use the security of PRF and generate r_J for session J at random instead of using the key k. Now, we observe that the randomness

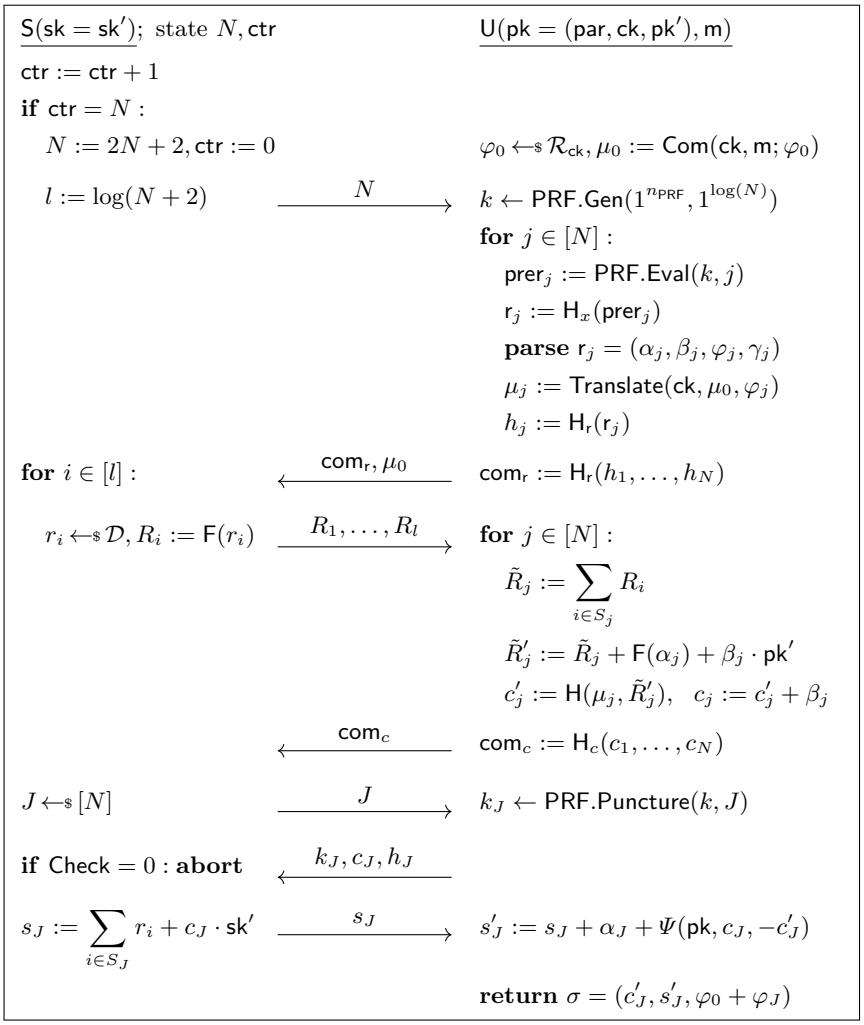

Fig. 3. The signature issuing protocol of the blind signature scheme CCCBS[LF], where $\mathsf{H}\colon \{0,1\}^* \to \mathcal{S}, \mathsf{H}_\mathsf{r}, \mathsf{H}_c\colon \{0,1\}^* \to \{0,1\}^n, \mathsf{H}_x\colon \{0,1\}^* \to \mathcal{D} \times \mathcal{S} \times \mathcal{R}_\mathsf{ck} \times \{0,1\}^{n_\mathsf{PRF}}$ are random oracles. The algorithm Check is defined in Figure 2. The set S_j is defined as $\{i \in [l] : i^{th}\text{-bit of } j \text{ is } 1\}$. The states ctr and N are incremented atomically.

φ_0 is hidden in the final signature, and we can switch μ_0 to a commitment of a random message. Finally, we see that the only dependency on the message is in session J and we can reduce from the blindness of BS[LF].

Theorem 2. *Let* PRF *be a puncturable pseudorandom function,* LF *be a linear function family, and* CMT *be a randomness homomorphic commitment scheme. Let* $\mathsf{H}\colon \{0,1\}^* \to \mathcal{S}, \mathsf{H}_\mathsf{r}, \mathsf{H}_c\colon \{0,1\}^* \to \{0,1\}^n$ *be random oracles. If* BS[LF] *satis-*

fies ℓ-one-more unforgeability for any $\ell = O(\log(n))$, then $\mathsf{CCCBS}[\mathsf{LF}]$ *satisfies ℓ-one-more unforgeability for any $\ell = \mathsf{poly}(n)$.*

Concretely, suppose there exists an adversary with advantage ϵ against the ℓ-one-more unforgeability of $\mathsf{CCCBS}[\mathsf{LF}]$*, that runs in time t, starts at most p interactions with his signer oracle, and makes at most $Q_{\mathsf{H}}, Q_{\mathsf{H_r}}, Q_{\mathsf{H}_c}$ queries to $\mathsf{H}, \mathsf{H_r}, \mathsf{H}_c$ respectively. Then, there exists an adversary against the λ-one-more unforegability* $\mathsf{BS}[\mathsf{LF}]$*, where $\lambda = 3\lceil \log p \rceil + \log(2/\epsilon)$, that runs in time t, starts at most p interactions with his signer oracle, makes at most Q_{H} queries to* H*, and has advantage $\epsilon_{\mathsf{BS}[\mathsf{LF}]}$, such that*

$$\epsilon \leq 2\left(\epsilon_{\mathsf{BS}[\mathsf{LF}]} + p \cdot \epsilon_{\mathsf{LF}} + \epsilon_{\mathsf{CMT}} + \frac{Q_{\mathsf{H_r}}^2 + Q_{\mathsf{H}_c}^2 + pQ_{\mathsf{H_r}} + pQ_{\mathsf{H}_c}}{2^n} + \frac{p^2(p^2 + Q_{\mathsf{H}})}{|\mathcal{R}|}\right),$$

where ϵ_{LF} is the advantage of an adversary with running time t against the preimage resistance of LF *and ϵ_{CMT} is the advantage of an adversary with running time t against the binding property of* CMT*.*

The proof is very similar to the proof for the original boosting transform [25] and can be found in the full version.

Remark 1. As an asymptotic result, we are satisfied with our improved boosting transform with logarithmic communication complexity. However, similar to the original boosting transform, we rely on the very loose security bound of the underlying linear blind signature scheme $\mathsf{BS}[\mathsf{LF}]$. For concrete efficiency, this is prohibitive, as we require that $\mathsf{BS}[\mathsf{LF}]$ supports a non-trivial number λ of signatures. Also, the logarithmic term of the communication complexity depends on computational assumptions. Thus, the loose bound will also have a negative impact on communication complexity.

To highlight this, we computed the parameter sizes for the instantiations of the boosting transform based on the discrete logarithm problem. Our calculations show that in order to support 2^{30} signatures, the scheme requires a 12035 bit group. It is apparent that this group size is impractical, and no standardized elliptic curve groups of this size exist. We remark that Katz et al. [25] also provide a parameter estimate, but this holds only for a very specific choice of signing queries, random oracle queries and advantage. A detailed explanation of our calculations can be found in the full version.

In the following, we will see how to augment the ideas of this section to construct schemes which eliminate aforementioned drawbacks and come with practical concrete parameters.

4 A Concrete Scheme based on CDH

Here, we construct a concrete blind signature scheme $\mathsf{PIKA}_{\mathsf{CDH}}$ based on the CDH assumption. While the construction in the previous section was generic, we aim for a scheme with concrete efficiency in this section.

4.1 Overview

As discussed in Remark 1, our improved boosting transform inherits the loose security bound of the underlying linear blind signature scheme. To see how we can circumvent this, let us first recall the reduction idea of the boosting transform. The main challenge is that the underlying scheme $\mathsf{BS}[\mathsf{LF}]$ allows for a logarithmic number of signing interactions, while the reduction has to simulate an arbitrary polynomial number of signing interactions for the adversary. This is solved as follows. First, note that whenever the adversary honestly commits to ur_j, μ_j, the reduction can extract these values from the commitments com_r by observing the random oracle queries. Then, an important property of linear blind signature schemes comes into play: If one knows the randomness and the message that is input into the user algorithm $\mathsf{BS}[\mathsf{LF}].\mathsf{U}$ and controls the random oracle, one can simulate the signer algorithm without knowing the secret key. Thus, the reduction only needs to access the signer oracle of $\mathsf{BS}[\mathsf{LF}]$ if the adversary *cheats* (i.e., it malforms the commitment for the J^{th} session in the first step and is not caught). Fortunately, the probability of such a (successful) cheat is at most $1/N$ in the N^{th} signing session. Thus, the expected number of successful cheats in p interactions is at most logarithmic in p. Using the Chernoff bound, one can show that with overwhelming probability, the number of successful cheats is reasonably close to this expectation.

We observe that by letting the cut-and-choose parameter grow slightly faster than before and scaling appropriately, the expected number of successful cheats can be bounded to be less than 1. Unfortunately, we can not just use the Chernoff bound, if we want to argue that this also holds with overwhelming probability. We can, however, use the Chernoff bound to show that exceeding a single cheat happens with some constant probability less than 1. Then, we play our next card, which is parallel repetition. Namely, we run K independent instances of our scheme so far, where each instance is relative to a separate key pair. We show that with high probability, in one randomly chosen instance, there is *no cheat at all.* Using this observation, we can give a reduction from the key-only security of the underlying blind signature scheme to finish our proof.

We do not apply this overall strategy to a linear blind signature scheme, but instead to the BLS blind signature scheme [4]. We notice that the approach also works for this scheme and observe additional benefits: First, the BLS scheme allows to aggregate signatures. Hence, it is easy to merge the resulting signatures from the K instances for a significant efficiency improvement. Second, the scheme has two rounds and thus the logarithmic term in the communication complexity is independent of computational assumptions (cf. Remark 1). We emphasize that the original BLS blind signature scheme is secure under a one-more variant of the CDH assumption. Fortunately, we only need key-only security here, which is implied by CDH. Also, the concrete security loss of our scheme is as for the standard BLS digital signature scheme [5], which means that it can be used over the same groups as BLS.

Finally, we introduce further minor optimizations such as making the signer commit to its cut-and-choose indices in its message. In this way, the reduction in

the blindness proof can extract these indices rather than guessing them. This leads to more efficient statistical security parameters [6].

4.2 Construction

Let $\mathsf{PGGen}(1^n)$ be a bilinear group generation algorithm that outputs a cyclic group $\mathbb{G}$ of prime order p with generator g, and a pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ into some target group $\mathbb{G}_T$. We assume that these system parameters are known to all algorithms. Note that their correctness can be verified efficiently. Our scheme makes use of a randomness homomorphic commitment scheme CMT with randomness space $\mathcal{R}_{\mathsf{ck}}$ and a puncturable pseudorandom function PRF. We can instantiate PRF using random oracles and CMT tightly based on the DLOG assumption. We also need random oracles $\mathsf{H}\colon \{0,1\}^* \to \mathbb{Z}_p, \mathsf{H}'\colon \{0,1\}^* \to \{0,1\}^n$ and $\mathsf{H}_\mathsf{r}, \mathsf{H}_c\colon \{0,1\}^* \to \{0,1\}^n, \mathsf{H}_x\colon \{0,1\}^* \to \mathbb{Z}_p \times \mathcal{R}_{\mathsf{ck}} \times \{0,1\}^{n_{\mathsf{PRF}}}$, where n_{PRF} is a security parameter used for PRF.

Our scheme makes use of a parameter $K \in \mathbb{N}$, which defines how many instances of the underlying boosting transform are executed in parallel, and a function $f\colon \mathbb{N} \to \mathbb{N}$, which determines how fast the cut-and-choose parameter N grows. We give a detailed explanation and Python scripts computing all parameters in the full version of our paper.

Key Generation. To generate keys algorithm $\mathsf{PIKA}_{\mathsf{CDH}}.\mathsf{Gen}(1^n)$ does the following:

1. For each instance $i \in [K]$, sample $\mathsf{sk}_i \leftarrow\!\!\$\, \mathbb{Z}_p$ and set $\mathsf{pk}_i := g^{\mathsf{sk}_i}$.
2. Sample a commitment key $\mathsf{ck} \leftarrow \mathsf{CMT}.\mathsf{Gen}(1^n)$.
3. Return public key $\mathsf{pk} := (\mathsf{pk}_1, \ldots, \mathsf{pk}_K, \mathsf{ck})$ and secret key $\mathsf{sk} := (\mathsf{sk}_1, \ldots, \mathsf{sk}_K)$.

Signature Issuing. The algorithms S, U and their interaction are formally given in Figures 4 and 5. Here, S keeps a state ctr, which is inititalized as $\mathsf{ctr} := 1$ and incremented in every interaction.

Verification. The resulting signature $\sigma = (\bar{\sigma}, \varphi_1, \ldots, \varphi_K)$ for a message m is verified by algorithm $\mathsf{PIKA}_{\mathsf{CDH}}.\mathsf{Ver}(\mathsf{pk}, \mathsf{m}, \sigma)$ as follows:

1. For each instance $i \in [K]$, compute the commitment $\mu_i := \mathsf{Com}(\mathsf{ck}, \mathsf{m}; \varphi_i)$.
2. Return 1 if and only if

$$e\left(\bar{\sigma}, g\right) = \prod_{i=1}^{K} e\left(\mathsf{H}(\mathsf{pk}_i, \mu_i), \mathsf{pk}_i\right).$$

[6] Note that without this optimization, the security loss would be exponential in K.

$\mathsf{Check}(\mathsf{pk}, N, \mu_0, \mathsf{com}_\mathsf{r}, \mathsf{com}_c, \mathsf{seed}_\mathbf{J}, k_\mathbf{J}, \{c_{i,\mathbf{J}_i}\}_i, \{\eta_i\}_i)$

$\mathbf{J} = (\mathsf{H}'(\mathsf{seed}_\mathbf{J}, 1), \ldots, \mathsf{H}'(\mathsf{seed}_\mathbf{J}, K)) \in [N]^K$
for $i \in [K]$:
  for $j \in [N] \setminus \{\mathbf{J}_i\}$:
    $\mathsf{prer}_{i,j} := \mathsf{PRF.Eval}(k_\mathbf{J}, (i,j)),\ \mathsf{r}_{i,j} := \mathsf{H}_x(\mathsf{prer}_{i,j})$
    parse $\mathsf{r}_{i,j} = (\alpha_{i,j}, \varphi_{i,j}, \gamma_{i,j}) \in \mathbb{Z}_p \times \mathcal{R}_\mathsf{ck} \times \{0,1\}^n$
    $\mu_{i,j} := \mathsf{Translate}(\mathsf{ck}, \mu_0, \varphi_{i,j})$
    $c_{i,j} := \mathsf{H}(\mathsf{pk}_i, \mu_{i,j}) \cdot g^{\alpha_{i,j}}$
  $\mathsf{com}_{\mathsf{r},i} := \mathsf{H}_\mathsf{r}(\mathsf{H}_\mathsf{r}(\mathsf{r}_{i,1}), \ldots, \mathsf{H}_\mathsf{r}(\mathsf{r}_{i,\mathbf{J}_i-1}), \eta_i, \mathsf{H}_\mathsf{r}(\mathsf{r}_{i,\mathbf{J}_i+1}), \ldots, \mathsf{H}_\mathsf{r}(\mathsf{r}_{i,N}))$
if $\mathsf{com}_\mathsf{r} \neq \mathsf{H}_\mathsf{r}(\mathsf{com}_{\mathsf{r},1}, \ldots, \mathsf{com}_{\mathsf{r},K})$: **return** 0
if $\mathsf{com}_c \neq \mathsf{H}_c(c_{1,1}, \ldots, c_{K,N})$: **return** 0
return 1

Fig. 4. The algorithm Check used in the issuing protocol of blind signature scheme $\mathsf{PIKA}_\mathsf{CDH}$, where $\mathsf{H}\colon \{0,1\}^* \to \mathbb{G}, \mathsf{H}'\colon \{0,1\}^* \to \{0,1\}^n$ and $\mathsf{H}_\mathsf{r}, \mathsf{H}_c\colon \{0,1\}^* \to \{0,1\}^n, \mathsf{H}_x\colon \{0,1\}^* \to \mathbb{Z}_p \times \mathcal{R}_\mathsf{ck} \times \{0,1\}^{n_\mathsf{PRF}}$ are random oracles.

4.3 Security Analysis

Completeness of the scheme follows by inspection. We show blindness and one-more unforgeability. For one-more unforgeability, we show $q_{\max}$-OMUF, where $q_{\max}$ is a parameter that can be set freely (e.g. $q_{\max} = 2^{30}$) and has influence the function f. We note that making f grow quadratically, one could show full OMUF using a similar proof.

Theorem 3. *Let* PRF *be a puncturable pseudorandom function and* CMT *be a randomness homomorphic commitment scheme. Let* $\mathsf{H}'\colon \{0,1\}^* \to \{0,1\}^n$ *and* $\mathsf{H}_\mathsf{r}\colon \{0,1\}^* \to \{0,1\}^n, \mathsf{H}_x\colon \{0,1\}^* \to \mathbb{Z}_p \times \mathcal{R}_\mathsf{ck} \times \{0,1\}^{n_\mathsf{PRF}}$ *be random oracles. Then* $\mathsf{PIKA}_\mathsf{CDH}$ *satisfies malicious signer blindness.*

In particular, for any adversary who uses N^L *and* N^R *as the counters in its executions with the user and queries* $\mathsf{H}', \mathsf{H}_\mathsf{r}, \mathsf{H}_x$ *at most* $Q_{\mathsf{H}'}, Q_{\mathsf{H}_\mathsf{r}}, Q_{\mathsf{H}_x}$ *times, respectively, the malicious signer blindness advantage can be bounded by*

$$4\epsilon_\mathsf{PRF} + \frac{Q_{\mathsf{H}'}^2}{2^{n-1}} + \frac{Q_{\mathsf{H}'}}{2^{n-2}} + \frac{KQ_{\mathsf{H}_x}}{2^{n_\mathsf{PRF}-2}} + \frac{KQ_{\mathsf{H}_\mathsf{r}}}{2^{n_\mathsf{PRF}-2}},$$

where ϵ_PRF *is the advantage of an adversary against the security of* PRF *with input length* $\max\{\log(N^L), \log(N^R)\}$ *when puncturing at* K *points.*

Due to space limitation, we postpone the proof to the full version.

Theorem 4. *Let* CMT *be a randomness homomorphic commitment scheme and* PRF *be a puncturable pseudorandom function. Let* $\mathsf{PGGen}(1^n)$ *be a bilinear group generation algorithm. Further, let* $\mathsf{H}\colon \{0,1\}^* \to \mathbb{Z}_p, \mathsf{H}'\colon \{0,1\}^* \to \{0,1\}^n$ *and*

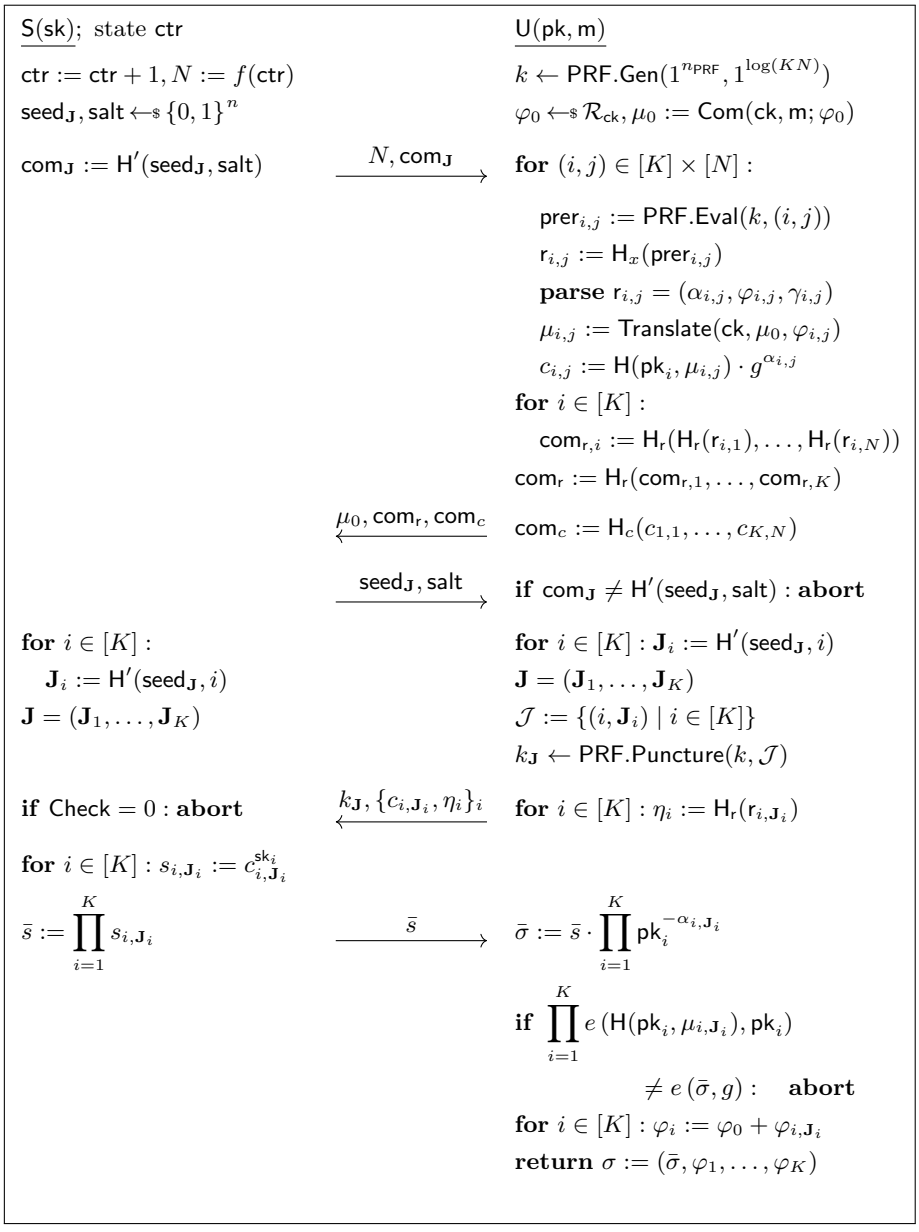

Fig. 5. The signature issuing protocol of the blind signature scheme $\mathsf{PIKA}_{\mathsf{CDH}}$, where $\mathsf{H}\colon \{0,1\}^* \to \mathbb{Z}_p, \mathsf{H}'\colon \{0,1\}^* \to \{0,1\}^n$ and $\mathsf{H}_\mathsf{r}, \mathsf{H}_c\colon \{0,1\}^* \to \{0,1\}^n, \mathsf{H}_x\colon \{0,1\}^* \to \mathbb{Z}_p \times \mathcal{R}_\mathsf{ck} \times \{0,1\}^{n_\mathsf{PRF}}$ are random oracles. The algorithm Check is defined in Figure 4. The state ctr of S is incremented atomically.

$\mathsf{H}_\mathsf{r}, \mathsf{H}_c \colon \{0,1\}^* \to \{0,1\}^n$ *be random oracles. Also, assume that there is a* $\vartheta > 0$ *and* f *is such that*

$$f(\mathsf{ctr}) = \lceil 3\vartheta \ln(q_{\max} + 1) \cdot \mathsf{ctr} \rceil .$$

Then $\mathsf{PIKA}_{\mathsf{CDH}}$ *satisfies* $q_{\max}$*-one-more unforgeability, under the* CDH *assumption relative to* PGGen*.*

Specifically, assume the existence of an adversary against the OMUF *security of* $\mathsf{PIKA}_{\mathsf{CDH}}$ *that has advantage* ϵ*, runs in time* t*, makes at most* $Q_{\mathsf{H}_\mathsf{r}}, Q_{\mathsf{H}_c}, Q_{\mathsf{H}'}, Q_{\mathsf{H}}$ *queries to oracles* $\mathsf{H}_\mathsf{r}, \mathsf{H}_c, \mathsf{H}', \mathsf{H}$*, respectively, and starts at most* $q \leq q_{\max}$ *interactions with his signer oracle. Let* $\delta > 0$ *such that* $(1-\delta)\vartheta > 1$*. Then there exists an adversary against the* CDH *problem relative to* PGGen *with advantage* ϵ_{CDH} *and running time* t *and an adversary against the binding property of* CMT *with advantage* ϵ_{CMT} *and running time* t *such that*

$$\epsilon - e^{-\delta K} \leq \epsilon_{\mathsf{CMT}} + \frac{K}{p} + 4qK\epsilon_{\mathsf{CDH}} + \mathsf{stat}$$

where

$$\mathsf{stat} = \frac{Q_{\mathsf{H}_\mathsf{r}}^2}{2^n} + \frac{Q_{\mathsf{H}_c}^2}{2^n} + \frac{qQ_{\mathsf{H}_\mathsf{r}}}{2^n} + \frac{qKQ_{\mathsf{H}_\mathsf{r}}}{2^n} + \frac{qQ_{\mathsf{H}_c}}{2^n} + \frac{qQ_{\mathsf{H}'}}{2^{n-1}}.$$

Proof. Set $\mathsf{BS} := \mathsf{PIKA}_{\mathsf{CDH}}$. Let $\mathcal{A}$ be an adversary against the OMUF security of BS. We prove the statement via a sequence of games.

Game $\mathbf{G}_0$: We start with game $\mathbf{G}_0 := q_{\max}\text{-}\mathbf{OMUF}^{\mathcal{A}}_{\mathsf{BS}}$, which is the one-more unforgeability game. We briefly recall this game. A key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$ is sampled, $\mathcal{A}$ is run with concurrent access to an interactive oracle O simulating the signer $\mathsf{S}(\mathsf{sk})$. Assume that $\mathcal{A}$ completes ℓ interactions with O. Further, $\mathcal{A}$ gets access to random oracles $\mathsf{H}, \mathsf{H}', \mathsf{H}_\mathsf{r}$ and H_c, which are provided by the game in the standard lazy manner. When $\mathcal{A}$ finishes its execution, it outputs tuples $(\mathsf{m}_1, \sigma_1), \ldots, (\mathsf{m}_k, \sigma_k)$ and wins, if all m_i are distinct, $k > \ell$ and all signatures σ_i verify with respect to pk and m_i.

Game $\mathbf{G}_1$: In game $\mathbf{G}_1$, we add an additional abort. The game aborts if in the end $\mathcal{A}$'s output contains two pairs $(\mathsf{m}^{(0)}, \sigma^{(0)}), (\mathsf{m}^{(1)}, \sigma^{(1)})$ such that $\mathsf{m}^{(0)} \neq \mathsf{m}^{(1)}$ but there exists $i^{(0)}, i^{(1)} \in [K]$ such that

$$\mathsf{Com}(\mathsf{ck}, \mathsf{m}^{(0)}; \varphi^{(0)}_{i^{(0)}}) = \mathsf{Com}(\mathsf{ck}, \mathsf{m}^{(1)}; \varphi^{(1)}_{i^{(1)}}).$$

As CMT is computationally binding, a straight-forward reduction with advantage ϵ_{CMT} and running time t shows that

$$|\Pr[\mathbf{G}_0 \Rightarrow 1] - \Pr[\mathbf{G}_1 \Rightarrow 1]| \leq \epsilon_{\mathsf{CMT}}.$$

Game $\mathbf{G}_2$: This game is as $\mathbf{G}_1$, but we rule out collisions for oracles $\mathsf{H}_t, t \in \{\mathsf{r}, c\}$. To be more precise, we change the simulation of oracles $\mathsf{H}_t, t \in \{\mathsf{r}, c\}$ in the

following way. If $\mathcal{A}$ queries $\mathsf{H}_t(x)$ and this value is not yet defined, the game samples an image $y \leftarrow_{\$} \{0,1\}^n$. However, if there exists an $x' \neq x$ with $\mathsf{H}_t(x') = y$, the game returns $\perp$. Otherwise it behaves as before. Note that $\mathcal{A}$ can only distinguish between $\mathbf{G}_1$ and $\mathbf{G}_2$ if such a collision happens, i.e. H_t returns $\perp$. We can apply a union bound over all $Q^2_{\mathsf{H}_t}$ pairs of random oracle queries and obtain

$$|\Pr[\mathbf{G}_1 \Rightarrow 1] - \Pr[\mathbf{G}_2 \Rightarrow 1]| \leq \frac{Q^2_{\mathsf{H}_r}}{2^n} + \frac{Q^2_{\mathsf{H}_c}}{2^n}.$$

Note that the change in $\mathbf{G}_2$ implies that at each point of the execution of the game and for each image $y \in \{0,1\}^n$, there is at most one preimage $\mathsf{H}_t^{-1}(y)$ under H_t. By looking at the random oracle queries of $\mathcal{A}$, the game can extract preimages of given images y, and we know that for each y at most one preimage can be extracted. We will make use of such an extraction in the following games.

Game $\mathbf{G}_3$: We change the way the signer oracle is executed. In particular, when $\mathcal{A}$ sends $\mu_0, \mathsf{com}_\mathsf{r}, \mathsf{com}_c$ as its first message, the game tries to extract values $\bar{\mathsf{com}}_{\mathsf{r},i}$ such that $\mathsf{com}_\mathsf{r} = \mathsf{H}_\mathsf{r}(\bar{\mathsf{com}}_{\mathsf{r},1}, \ldots, \bar{\mathsf{com}}_{\mathsf{r},K})$ by searching through random oracle queries. If the game can not extract such a preimage, we write $\bar{\mathsf{com}}_{\mathsf{r},i} = \perp$ for all $i \in [K]$. Then, the game aborts if it can not extract such a preimage , i.e. $\bar{\mathsf{com}}_{\mathsf{r},i} = \perp$, but later algorithm Check outputs 1. Recall that algorithm Check verifies that

$$\mathsf{com}_\mathsf{r} = \mathsf{H}_\mathsf{r}(\mathsf{com}_{\mathsf{r},1}, \ldots, \mathsf{com}_{\mathsf{r},K}).$$

Thus, for every fixed interaction, we can bound the probability of such an abort by $Q_{\mathsf{H}_\mathsf{r}}/2^n$. Indeed, once com_r is sent by $\mathcal{A}$ and thus fixed, and the game can not extract, we know that there is no bitstring x such that $\mathsf{H}_\mathsf{r}(x) = \mathsf{com}_\mathsf{r}$. Also, if algorithm Check outputs 1, we know that $\mathcal{A}$ was able to find a preimage of com_r after this was fixed. This can happen with probability at most $1/2^n$ for each random oracle query. Using a union bound over all interactions we obtain

$$|\Pr[\mathbf{G}_3 \Rightarrow 1] - \Pr[\mathbf{G}_4 \Rightarrow 1]| \leq \frac{qQ_{\mathsf{H}_\mathsf{r}}}{2^n}.$$

Game $\mathbf{G}_4$: We introduce another abort in the signer oracle. In this game, after the extraction of $(\bar{\mathsf{com}}_{\mathsf{r},1}, \ldots, \bar{\mathsf{com}}_{\mathsf{r},K})$ from com_r we introduced before, the game extracts $(\bar{\mathsf{r}}_{i,1}, \ldots, \bar{\mathsf{r}}_{i,N})$ from $\bar{\mathsf{com}}_{\mathsf{r},i}$ for every $i \in [K]$ for which $\bar{\mathsf{com}}_{\mathsf{r},i} \neq \perp$, such that

$$\bar{\mathsf{com}}_{\mathsf{r},i} = \mathsf{H}_\mathsf{r}(\mathsf{H}_\mathsf{r}(\bar{\mathsf{r}}_{i,1}), \ldots, \mathsf{H}_\mathsf{r}(\bar{\mathsf{r}}_{i,N})).$$

Again, the game does this by looking at the random oracle queries of $\mathcal{A}$ and we write $\bar{\mathsf{r}}_{i,j} = \perp$ if the game can not extract the value $\bar{\mathsf{r}}_{i,j}$. If there is an instance $i \in [K]$ and a session $j \in [N]$ such that $\bar{\mathsf{com}}_{\mathsf{r},i} \neq \perp$ but $\bar{\mathsf{r}}_{i,j} = \perp$ and later in that execution $\mathbf{J}_i \neq j$ but algorithm Check outputs 1, the game aborts.

To analyze the probability of this abort, fix an interaction and an instance $i \in [K]$. Assume that $\bar{\mathsf{com}}_{\mathsf{r},i} \neq \perp$ and there is a session $j \in [N]$ such that $\bar{\mathsf{r}}_{i,j} = \perp$ and later in that interaction $\mathbf{J}_i \neq j$. Then, after $\bar{\mathsf{com}}_{\mathsf{r},i}$ is fixed, we consider two cases. In the first case, the game could not extract $h_1, \ldots, h_N$ such

that $\bar{\mathsf{com}}_{\mathsf{r},i} = \mathsf{H}_{\mathsf{r}}(h_1, \ldots, h_N)$. Clearly, once $\bar{\mathsf{com}}_{\mathsf{r},i}$, the probability that one of the hash queries of $\mathcal{A}$ evaluates to $\bar{\mathsf{com}}_{\mathsf{r},i}$ is at most $1/2^n$. Thus, the probability that Check outputs 1, i.e. $\mathcal{A}$ is able to open $\bar{\mathsf{com}}_{\mathsf{r},i}$ in this case, is at most $Q_{\mathsf{H}_{\mathsf{r}}}/2^n$. Similarly, in the case where the game could extract $h_1, \ldots, h_N$, but could not extract $\bar{\mathsf{r}}_{i,j}$ such that $\mathsf{H}_{\mathsf{r}}(\bar{\mathsf{r}}_{i,j}) = h_j$, the probability that one of $\mathcal{A}$'s hash queries evaluates to h_j is at most $1/2^n$. Thus, the probability that Check outputs 1, i.e. $\mathcal{A}$ is able to open h_j in this case, is at most $Q_{\mathsf{H}_{\mathsf{r}}}/2^n$. Note that here we needed that $j \neq \mathbf{J}_i$, as the definition of Check does not require $\mathcal{A}$ to open $h_{\mathbf{J}_i}$.

Applying a union bound over the interactions and instances we get

$$|\Pr[\mathbf{G}_3 \Rightarrow 1] - \Pr[\mathbf{G}_4 \Rightarrow 1]| \leq \frac{qKQ_{\mathsf{H}_{\mathsf{r}}}}{2^n}.$$

Game $\mathbf{G}_5$: We introduce another abort: Whenever $\mathcal{A}$ sends $\mu_0, \mathsf{com}_{\mathsf{r}}, \mathsf{com}_c$ as its first message, the game behaves as before, but additionally the game extracts values $\bar{c}_{1,1}, \ldots, \bar{c}_{K,N}$ from com_c such that

$$\mathsf{com}_c = \mathsf{H}_c(\bar{c}_{1,1}, \ldots, \bar{c}_{K,N}).$$

If the game can not extract, but later algorithm Check outputs 1, the game aborts. Note that algorithm Check internally checks if

$$\mathsf{com}_c = \mathsf{H}_c(c_{1,1}, \ldots, c_{K,N}).$$

Thus, for each fixed interaction it is possible to argue as in the previous games to bound the probability of such an abort and hence we obtain

$$|\Pr[\mathbf{G}_4 \Rightarrow 1] - \Pr[\mathbf{G}_5 \Rightarrow 1]| \leq \frac{qQ_{\mathsf{H}_c}}{2^n}.$$

Game $\mathbf{G}_6$: In $\mathbf{G}_6$, the signer oracle sends a random $\mathsf{com}_{\mathbf{J}}$ in the beginning of each interaction. Later, before it has to send $\mathsf{seed}_{\mathbf{J}}, \mathsf{salt}$, it samples $\mathsf{salt} \leftarrow\!\!\$\ \{0,1\}^n$ and aborts if $\mathsf{H}'(\mathsf{seed}_{\mathbf{J}}, \mathsf{salt})$ is already defined. If it is not yet defined, it defines it as $\mathsf{H}'(\mathsf{seed}_{\mathbf{J}}, \mathsf{salt}) := \mathsf{com}_{\mathbf{J}}$. The adversary $\mathcal{A}$ can only distinguish between $\mathbf{G}_5$ and $\mathbf{G}_6$ if $\mathsf{H}'(\mathsf{seed}_{\mathbf{J}}, \mathsf{salt})$ is already defined. By a union bound over all $Q_{\mathsf{H}'}$ hash queries and q interactions we obtain

$$|\Pr[\mathbf{G}_5 \Rightarrow 1] - \Pr[\mathbf{G}_6 \Rightarrow 1]| \leq \frac{qQ_{\mathsf{H}'}}{2^n}.$$

Game $\mathbf{G}_7$: In $\mathbf{G}_7$, the game aborts if in some interaction there exists an $i \in [K]$ such that $\mathsf{H}'(\mathsf{seed}_{\mathbf{J}}, i)$ has already been queried before the signing oracle sends $\mathsf{seed}_{\mathbf{J}}$ to $\mathcal{A}$. Clearly, $\mathcal{A}$ obtains no information about $\mathsf{seed}_{\mathbf{J}}$ before the potential abort, see $\mathbf{G}_6$. Further, $\mathsf{seed}_{\mathbf{J}}$ is sampled uniformly at random. A union bound over all $Q_{\mathsf{H}'}$ queries and q interactions shows that

$$|\Pr[\mathbf{G}_6 \Rightarrow 1] - \Pr[\mathbf{G}_7 \Rightarrow 1]| \leq \frac{qQ_{\mathsf{H}'}}{2^n}.$$

Now, fix an interaction in $\mathbf{G}_7$ and assume that Check returns 1 and the game does not abort due to any of the reasons we introduced so far. Note that this means that for all instances $i \in [K]$ the value $\bar{\mathsf{com}}_{\mathsf{r},i}$ could be extracted. Furthermore, this means that if there exists $i \in [K], j_0 \in [N]$ such that $\bar{\mathsf{r}}_{i,j_0} = \perp$ then later $\mathbf{J}_i = j_0$. Also, note that if Check does not abort, then we have $\bar{\mathsf{com}}_{\mathsf{r},i} = \mathsf{com}_{\mathsf{r},i}, \bar{\mathsf{r}}_{i,j} = \mathsf{r}_{i,j}$ and $\bar{c}_{i,j} = c_{i,j}$ for all $(i,j) \in [K] \times [N]$ for which these values are defined. This is because we ruled out collisions for oracles $\mathsf{H}_\mathsf{r}, \mathsf{H}_c$. Now, we define an indicator random variable $\mathsf{cheat}_{i,\mathsf{ctr}}$ for the event that in the ctr^{th} interaction, the signer oracle does not abort and there exists $i \in [K], j \in [N]$ such that $\bar{\mathsf{r}}_{i,j} = \perp$ or $\bar{\mathsf{r}}_{i,j} = (\alpha, \varphi, \gamma)$ such that

$$c_{i,j} \neq \mathsf{H}(\mathsf{pk}_i, \mathsf{Translate}(\mathsf{ck}, \mu_0, \varphi)) \cdot g^{\alpha}.$$

We say that $\mathcal{A}$ successfully cheats in instance $i \in [K]$ and interaction ctr if $\mathsf{cheat}_{i,\mathsf{ctr}} = 1$. We also define the number of interactions in which $\mathcal{A}$ successfully cheats in instance i as $\mathsf{cheat}_i^* := \sum_{\mathsf{ctr}=2}^{q+1} \mathsf{cheat}_{i,\mathsf{ctr}}$.

By the above discussion, we have that $\mathsf{cheat}_{i,\mathsf{ctr}} = 1$ implies that $\mathbf{J}_i = j_0$ and thus

$$\Pr\left[\mathsf{cheat}_{i,\mathsf{ctr}} = 1\right] \leq \frac{1}{N}.$$

Therefore, we can bound the expectation of cheat_i^* using

$$\mathbb{E}\left[\mathsf{cheat}_i^*\right] \leq \frac{1}{3\vartheta \ln(q_{\max}+1)} \sum_{\mathsf{ctr}=2}^{q+1} \frac{1}{\mathsf{ctr}} \leq \frac{\ln(q+1)}{3\vartheta \ln(q_{\max}+1)} \leq \frac{1}{3\vartheta}.$$

Now, if we plug $X := \mathsf{cheat}_i^*$ and $s := 3\mathbb{E}\left[\mathsf{cheat}_i^*\right] + \delta = 1/\vartheta + \delta$ into the Chernoff bound (see the full version), we get that for all $i \in [K]$

$$\Pr\left[\mathsf{cheat}_i^* \geq \frac{1}{\vartheta} + \delta\right] \leq e^{-\delta}.$$

We note that the entire calculation of this probability also holds if we fix the random coins of the adversary.

Game $\mathbf{G}_8$: Game $\mathbf{G}_8$ is defined as $\mathbf{G}_7$, but additionally aborts if for all $i \in [K]$ we have $\mathsf{cheat}_i^* \geq \delta + 1/\vartheta$. In particular, if $\mathbf{G}_8$ does not abort, then there is some instance i for which $\mathcal{A}$ does not successfully cheat at all, which follows from the assumption $(1-\delta)\vartheta > 1$.

We can now bound the distinguishing advantage of $\mathcal{A}$ between $\mathbf{G}_7$ and $\mathbf{G}_8$ as follows. We denote the random coins of $\mathcal{A}$ by $\rho_\mathcal{A}$ and the random coins of the experiment (excluding $\rho_\mathcal{A}$) by ρ. Let bad be the event that for all $i \in [K]$ we have $\mathsf{cheat}_i^* \geq \delta + 1/\vartheta$. We note that the coins ρ that the experiment uses for

the K instances are independent. Thus we have

$$\begin{aligned}\Pr_{\rho,\rho_{\mathcal{A}}}[\mathsf{bad}] &= \sum_{\bar{\rho}_{\mathcal{A}}} \Pr_{\rho_{\mathcal{A}}}[\rho_{\mathcal{A}} = \bar{\rho}_{\mathcal{A}}] \cdot \Pr_{\rho,\rho_{\mathcal{A}}}[\mathsf{bad} \mid \rho_{\mathcal{A}} = \bar{\rho}_{\mathcal{A}}] \\ &= \sum_{\bar{\rho}_{\mathcal{A}}} \Pr_{\rho_{\mathcal{A}}}[\rho_{\mathcal{A}} = \bar{\rho}_{\mathcal{A}}] \cdot \prod_{i\in[K]} \Pr_{\rho,\rho_{\mathcal{A}}}\left[\mathsf{cheat}_i^* \geq \frac{1}{\vartheta} + \delta \;\middle|\; \rho_{\mathcal{A}} = \bar{\rho}_{\mathcal{A}}\right] \\ &\leq \sum_{\bar{\rho}_{\mathcal{A}}} \Pr_{\rho_{\mathcal{A}}}[\rho_{\mathcal{A}} = \bar{\rho}_{\mathcal{A}}] \cdot e^{-\delta K} = e^{-\delta K},\end{aligned}$$

which implies

$$|\Pr[\mathbf{G}_7 \Rightarrow 1] - \Pr[\mathbf{G}_8 \Rightarrow 1]| \leq \Pr_{\rho,\rho_{\mathcal{A}}}[\mathsf{bad}] \leq e^{-\delta K}.$$

Game $\mathbf{G}_9$: In game $\mathbf{G}_9$, we sample a random instance $i^* \leftarrow\!\!\$\, [K]$ at the beginning of the game. In the end, the game aborts if $\mathsf{cheat}_{i^*}^* \geq \delta + 1/\vartheta$. In particular, if this game does not abort, then $\mathcal{A}$ does not successfully cheat in instance i^* at all. As $\mathcal{A}$'s view is independent from i^*, we have

$$\begin{aligned}\Pr[\mathbf{G}_9 \Rightarrow 1] &= \Pr\left[\mathbf{G}_8 \Rightarrow 1 \wedge \mathsf{cheat}_{i^*}^* < \frac{1}{\vartheta} + \delta\right] \\ &= \Pr[\mathbf{G}_8 \Rightarrow 1] \cdot \Pr\left[\mathsf{cheat}_{i^*}^* < \frac{1}{\vartheta} + \delta \;\middle|\; \mathbf{G}_8 \Rightarrow 1\right] \\ &\geq \Pr[\mathbf{G}_8 \Rightarrow 1] \cdot \Pr\left[\mathsf{cheat}_{i^*}^* < \frac{1}{\vartheta} + \delta \;\middle|\; \exists i \in [K] : \mathsf{cheat}_i^* < \frac{1}{\vartheta} + \delta\right] \\ &\geq \Pr[\mathbf{G}_8 \Rightarrow 1] \cdot \frac{1}{K},\end{aligned}$$

where the first inequality follows from the fact that the event $\mathbf{G}_8 \Rightarrow 1$ implies the event $\exists i \in [K] : \mathsf{cheat}_i^* < \delta + 1/\vartheta$.

We note that from now on, our proof follows the proof strategy of the BLS signature scheme [5].

Game $\mathbf{G}_{10}$: In game $\mathbf{G}_{10}$, we introduce an initially empty set $\mathcal{L}$ and a new abort. We highlight that we treat $\mathcal{L}$ as a set and therefore every bitstring is in $\mathcal{L}$ only once. Recall that when $\mathcal{A}$ sends $\mu_0, \mathsf{com}_\mathsf{r}, \mathsf{com}_c$ to the signer oracle, the game tries to extract values $\bar{\mathsf{r}}_{i,j}$ for $(i,j) \in [K] \times [N]$. Then the game samples $\mathsf{seed}_\mathbf{J}$ and computes $\mathbf{J}$ accordingly. In particular, due to the changes in the previous games we know that the game extracts $\bar{\mathsf{r}}_{i^*,\mathbf{J}_{i^*}} = (\alpha, \varphi, \gamma)$ unless the experiment will abort anyways. Then, in game $\mathbf{G}_{10}$, the game will insert $\mathsf{Translate}(\mathsf{ck}, \mu_0, \varphi)$ into $\mathcal{L}$.

Fix the first pair (m, σ) in $\mathcal{A}$'s final output such that for $\sigma = (\bar{\sigma}, \varphi_1, \ldots, \varphi_K)$ and $\mu^* := \mathsf{Com}(\mathsf{ck}, \mathsf{m}; \varphi_{i^*})$ we have $\mu^* \notin \mathcal{L}$. Such a pair must exists if $\mathcal{A}$ is successful, see game $\mathbf{G}_1$. Then game $\mathbf{G}_{10}$ aborts if $\mathsf{H}(\mathsf{pk}_{i^*}, \mu^*)$ is not defined yet. Note that $\mathcal{A}$'s success probability in such a case can be at most $1/p$ and hence

$$|\Pr[\mathbf{G}_9 \Rightarrow 1] - \Pr[\mathbf{G}_{10} \Rightarrow 1]| \leq \frac{1}{p}.$$

Game $\mathbf{G}_{11}$: In game $\mathbf{G}_{11}$, we change how the random oracle H is simulated and add a new abort. For every query of the form $\mathsf{H}(\mathsf{pk}_{i^*}, \mu)$ the game independently samples a bit $b[\mu] \in \{0,1\}$ such that the probability that $b[\mu] = 1$ is $1/(q+1)$. Whenever the game adds a value μ to the set $\mathcal{L}$, it aborts if $b[\mu] = 1$. Then, after $\mathcal{A}$ returns its final output, the game determines μ^* as in $\mathbf{G}_{10}$, adds arbitrary values to $\mathcal{L}$ such that all values in $\mathcal{L} \cup \{\mu^*\}$ are distinct and $|\mathcal{L}| = q$ and aborts if $b[\mu^*] = 0$ or there is a $\mu \in \mathcal{L}$ such that $b[\mu] = 1$. Otherwise it continues as before. Note that unless the game aborts, $\mathcal{A}$'s view does not change. As all bits $b[\mu]$ are independent, we derive

$$\begin{aligned}
\Pr[\mathbf{G}_{11} \Rightarrow 1] &= \Pr[\mathbf{G}_{10} \Rightarrow 1] \cdot \Pr[b[\mu^*] = 1 \wedge \forall \mu \in \mathcal{L} : b[\mu] = 0] \\
&= \Pr[\mathbf{G}_{10} \Rightarrow 1] \cdot \frac{1}{q+1} \left(1 - \frac{1}{q+1}\right)^q \\
&= \Pr[\mathbf{G}_{10} \Rightarrow 1] \cdot \frac{1}{q} \left(1 - \frac{1}{q+1}\right)^{q+1} \\
&\geq \Pr[\mathbf{G}_{10} \Rightarrow 1] \cdot \frac{1}{4q},
\end{aligned}$$

where the last inequality follows from $(1 - 1/x)^x \geq 1/4$ for all $x \geq 2$.

Finally, we construct a reduction $\mathcal{B}$ that solves CDH with running time t and advantage ϵ_{CDH} such that

$$\Pr[\mathbf{G}_{11} \Rightarrow 1] \leq \epsilon_{\mathsf{CDH}}.$$

Then, the statement follows by an easy calculation. Reduction $\mathcal{B}$ works as follows:

- $\mathcal{B}$ gets as input bilinear group parameters $\mathbb{G}, g, p, e$ and group elements $X = g^x, Y = g^y$. The goal of $\mathcal{B}$ is to compute g^{xy}. First, $\mathcal{B}$ samples $i^* \leftarrow\!\!\$\, [K]$. Then, it defines $\mathsf{pk}_{i^*} := X$ (which implicitly defines $\mathsf{sk}_{i^*} := x$) and $\mathsf{sk}_i \leftarrow\!\!\$\, \mathbb{Z}_p, \mathsf{pk}_i := g^{\mathsf{sk}_i}$ for $i \in [K] \setminus \{i^*\}$.
- $\mathcal{B}$ runs adversary $\mathcal{A}$ on input $\mathbb{G}, g, p, e, \mathsf{pk} := (\mathsf{pk}_1, \ldots, \mathsf{pk}_K, \mathsf{ck})$ with oracle access to a signer oracle and random oracles $\mathsf{H}, \mathsf{H}_\mathsf{r}, \mathsf{H}_c, \mathsf{H}'$. To do so, it simulates oracles $\mathsf{H}_\mathsf{r}, \mathsf{H}_c, \mathsf{H}'$ exactly as in $\mathbf{G}_{11}$. The other oracles are provided as follows:
  - For a query of the form $\mathsf{H}(\mathsf{pk}_{i^*}, \mu)$ for which the hash value is not yet defined, it samples a bit $b[\mu] \in \{0,1\}$ such that the probability that $b[\mu] = 1$ is $1/(q+1)$. Then, it defines the hash value as $Y^{b[\mu]} \cdot g^{t[i^*,\mu]}$ for a randomly sampled $t[i^*, \mu] \leftarrow\!\!\$\, \mathbb{Z}_p$. For a query of the form $\mathsf{H}(\mathsf{pk}_i, \mu), i \neq i^*$ for which the hash value is not yet defined it defines the hash value as $g^{t[i,\mu]}$ for a randomly sampled $t[i, \mu] \leftarrow\!\!\$\, \mathbb{Z}_p$. For all other queries it simulates H honestly.
  - When $\mathcal{A}$ starts an interaction with the signer oracle, $\mathcal{B}$ sends N to $\mathcal{B}$ as in the protocol. When $\mathcal{B}$ sends its first message $\mu_0, \mathsf{com}_\mathsf{r}, \mathsf{com}_c$ as its first message, $\mathcal{B}$ behaves as $\mathbf{G}_{11}$. In particular, it tries to extract $\bar{\mathsf{r}}_{i,j}, \bar{c}_{i,j}$ for $(i,j) \in [K] \times [N]$. It then sends $\mathsf{seed}_\mathbf{J}$ to $\mathcal{A}$.

- When $\mathcal{A}$ sends its second message $k_{\mathbf{J}}, \{c_{i,\mathbf{J}_i}, \eta_i\}_{i\in[K]}$, $\mathcal{B}$ aborts under the same conditions as $\mathbf{G}_{11}$ does. In particular, if $\mathcal{B}$ does not abort and the signer oracle does not abort then $\bar{\mathsf{r}}_{i^*,\mathbf{J}_{i^*}} = (\alpha, \varphi, \gamma)$ is defined and $\mathcal{B}$ for $\mu := \mathsf{Translate}(\mathsf{ck}, \mu_0, \varphi)$, $\mathcal{B}$ sets $s_{i^*,\mathbf{J}_{i^*}} := X^{t[i^*,\mu]+\alpha}$. As defined in $\mathbf{G}_{11}$, $\mathcal{B}$ also inserts μ into the set $\mathcal{L}$. It computes $s_{i,\mathbf{J}_i}$ for $i \neq i^*$ as game $\mathbf{G}_{11}$ does, which is possible as $\mathcal{B}$ holds the corresponding sk_i. Then, $\mathcal{B}$ sends $\bar{s} := \prod_{i=1}^{K} s_{i,\mathbf{J}_i}$ to $\mathcal{A}$.

– When $\mathcal{A}$ returns its final output, $\mathcal{B}$ performs all verification steps in $\mathbf{G}_{11}$. In particular, it searches for the first pair (m, σ) in $\mathcal{A}$'s final output such that for $\sigma = (\bar{\sigma}, \varphi_1, \ldots, \varphi_K)$ and $\mu^* := \mathsf{Com}(\mathsf{ck}, \mathsf{m}; \varphi_{i^*})$ we have $\mu^* \notin \mathcal{L}$. As defined in $\mathbf{G}_{11}$, $\mathcal{B}$ aborts if $b[\mu^*] = 0$. Finally, $\mathcal{B}$ defines $\mu_i := \mathsf{Com}(\mathsf{ck}, \mathsf{m}; \varphi_i)$ and returns

$$Z := \bar{\sigma} \cdot X^{-t[i^*,\mu^*]} \cdot g^{-\sum_{i\in[K]\setminus\{i^*\}} t[i,\mu_i]\mathsf{sk}_i}$$

to its challenger.

We first argue that $\mathcal{B}$ perfectly simulates $\mathbf{G}_{11}$ for $\mathcal{A}$. To see that, note that as the $t[i,\mu]$ are sampled uniformly at random, the random oracle is simulated perfectly. To see that $s_{i^*,\mathbf{J}_{i^*}}$ is distributed correctly, note that if the signing oracle and $\mathbf{G}_{11}$ do not abort, then we have

$$c_{i^*,\mathbf{J}_{i^*}}^{\mathsf{sk}_{i^*}} = (\mathsf{H}(\mathsf{pk}_{i^*}, \mu) \cdot g^{\alpha})^{\mathsf{sk}_{i^*}} = \left(Y^{b[\mu]} \cdot g^{t[i^*,\mu]} \cdot g^{\alpha}\right)^{x} = X^{t[i^*,\mu]+\alpha},$$

where the last equality follows from $b[\mu] = 0$, as otherwise $\mathbf{G}_{11}$ would have aborted.

It remains to show that if $\mathbf{G}_{11}$ outputs 1, then we have $Z = g^{xy}$. This follows directly from the verification equation and $b[\mu^*] = 1$. To see this, note that

$$\begin{aligned}\prod_{i=1}^{K} e\left(\mathsf{H}(\mathsf{pk}_i, \mu_i), \mathsf{pk}_i\right) &= e\left(Y^{b[\mu^*]} \cdot g^{t[i^*,\mu^*]}, X\right) \cdot \prod_{i\in[K]\setminus\{i^*\}} e\left(g^{t[i,\mu_i]}, g^{\mathsf{sk}_i}\right) \\ &= e\left(g, g\right)^{xy+t[i^*,\mu^*]x} \cdot e\left(g, g\right)^{\sum_{i\in[K]\setminus\{i^*\}} t[i,\mu_i]\mathsf{sk}_i}.\end{aligned}$$

Using the verification equation, this implies that

$$g^{xy} = \bar{\sigma} \cdot g^{-\left(t[i^*,\mu^*]x + \sum_{i\in[K]\setminus\{i^*\}} t[i,\mu_i]\mathsf{sk}_i\right)}$$

Concluded. □

We note that instead of giving games $\mathbf{G}_{10}, \mathbf{G}_{11}$ and the reduction from CDH explicitly, one can also directly reduce from the security of the BLS signature scheme to $\mathbf{G}_9$, leading to the very same bound in total. This tells us that one can use (up to losing $\log(K)$ bits [7] of security) the same curves as for BLS.

Corollary 1 (Informal). *Under the same conditions as in Theorem 4, the scheme* $\mathsf{PIKA}_{\mathsf{CDH}}$ *satisfies* $q_{\max}$*-one-more unforgeability, if the BLS signature scheme [5] is unforgeable under chosen message attacks relative to* PGGen*, where the concrete security loss is (up to statistically negligible terms) given by* K*.*

[7] In our concrete instantiation, $\log(K) \approx 6.5$.

5 A Concrete Scheme based on RSA

In addition to our concrete scheme from CDH, we also construct a concrete scheme $\mathsf{BS}_{\mathsf{RSA}}$ based on the RSA assumption. We postpone the details to the full version and only give a short overview here.

Our scheme is based on the Okamoto-Guillou-Quisquater (OGQ) [26] linear function. That is, we start with this function in our generic transformation from Section 3. Informally, the function has domain $\mathcal{D} := \mathbb{Z}_\lambda \times \mathbb{Z}_N^*$, scalar space $\mathcal{S} := \mathbb{Z}_\lambda$ and range $\mathbb{Z}_N^*$, where N is an RSA modulus and λ is a prime with $\gcd(N, \lambda) = \gcd(\varphi(N), \lambda) = 1$. As we can not aggregate signatures efficiently, we can not mimic the K-repetition technique from our CDH-based scheme. Thus, we still rely on the loose bound of the underlying linear blind signature scheme. To solve this issue and obtain practical parameter sizes, we note that the bound becomes acceptable, once we increase the parameter λ. Our insight is that this can be done independently from the modulus N.

Although this improves the bound and thus concrete parameters, we still have a rather large communication complexity, due to the logarithmic number of $R_i \in \mathbb{Z}_N^*$ that are sent in our generic transformation. Here, our solution is to send a short random seed (e.g. 128 bit) and derive the values R_i using a random oracle. Now, the signer has to recover the preimages of the R_i to continue the protocol. We show that the OGQ linear function admits a trapdoor that allows to sample preimages, solving this problem as well.

References

1. Agrawal, S., Kirshanova, E., Stehlé, D., Yadav, A.: Can round-optimal lattice-based blind signatures be practical? Cryptology ePrint Archive, Report 2021/1565 (2021), `https://eprint.iacr.org/2021/1565`
2. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme. Journal of Cryptology 16(3), 185–215 (Jun 2003)
3. Benhamouda, F., Lepoint, T., Loss, J., Orrù, M., Raykova, M.: On the (in)security of ROS. In: Canteaut, A., Standaert, F.X. (eds.) Advances in Cryptology – EUROCRYPT 2021, Part I. Lecture Notes in Computer Science, vol. 12696, pp. 33–53. Springer, Heidelberg, Germany, Zagreb, Croatia (Oct 17–21, 2021)
4. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In: Desmedt, Y. (ed.) PKC 2003: 6th International Workshop on Theory and Practice in Public Key Cryptography. Lecture Notes in Computer Science, vol. 2567, pp. 31–46. Springer, Heidelberg, Germany, Miami, FL, USA (Jan 6–8, 2003)
5. Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. In: Boyd, C. (ed.) Advances in Cryptology – ASIACRYPT 2001. Lecture Notes in Computer Science, vol. 2248, pp. 514–532. Springer, Heidelberg, Germany, Gold Coast, Australia (Dec 9–13, 2001)
6. Camenisch, J., Groß, T.: Efficient attributes for anonymous credentials. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM CCS 2008: 15th Conference on Computer and Communications Security. pp. 345–356. ACM Press, Alexandria, Virginia, USA (Oct 27–31, 2008)

7. Camenisch, J., Lysyanskaya, A.: An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In: Pfitzmann, B. (ed.) Advances in Cryptology – EUROCRYPT 2001. Lecture Notes in Computer Science, vol. 2045, pp. 93–118. Springer, Heidelberg, Germany, Innsbruck, Austria (May 6–10, 2001)
8. Camenisch, J., Michels, M.: Proving in zero-knowledge that a number is the product of two safe primes. In: Stern, J. (ed.) Advances in Cryptology – EUROCRYPT'99. Lecture Notes in Computer Science, vol. 1592, pp. 107–122. Springer, Heidelberg, Germany, Prague, Czech Republic (May 2–6, 1999)
9. Chaum, D.: Blind signatures for untraceable payments. In: Chaum, D., Rivest, R.L., Sherman, A.T. (eds.) Advances in Cryptology – CRYPTO'82. pp. 199–203. Plenum Press, New York, USA, Santa Barbara, CA, USA (1982)
10. Fischlin, M.: Round-optimal composable blind signatures in the common reference string model. In: Dwork, C. (ed.) Advances in Cryptology – CRYPTO 2006. Lecture Notes in Computer Science, vol. 4117, pp. 60–77. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 20–24, 2006)
11. Fuchsbauer, G., Hanser, C., Slamanig, D.: Practical round-optimal blind signatures in the standard model. In: Gennaro, R., Robshaw, M.J.B. (eds.) Advances in Cryptology – CRYPTO 2015, Part II. Lecture Notes in Computer Science, vol. 9216, pp. 233–253. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 16–20, 2015)
12. Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Shacham, H., Boldyreva, A. (eds.) Advances in Cryptology – CRYPTO 2018, Part II. Lecture Notes in Computer Science, vol. 10992, pp. 33–62. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 19–23, 2018)
13. Fuchsbauer, G., Plouviez, A., Seurin, Y.: Blind schnorr signatures and signed ElGamal encryption in the algebraic group model. In: Canteaut, A., Ishai, Y. (eds.) Advances in Cryptology – EUROCRYPT 2020, Part II. Lecture Notes in Computer Science, vol. 12106, pp. 63–95. Springer, Heidelberg, Germany, Zagreb, Croatia (May 10–14, 2020)
14. Garg, S., Gupta, D.: Efficient round optimal blind signatures. In: Nguyen, P.Q., Oswald, E. (eds.) Advances in Cryptology – EUROCRYPT 2014. Lecture Notes in Computer Science, vol. 8441, pp. 477–495. Springer, Heidelberg, Germany, Copenhagen, Denmark (May 11–15, 2014)
15. Garg, S., Rao, V., Sahai, A., Schröder, D., Unruh, D.: Round optimal blind signatures. In: Rogaway, P. (ed.) Advances in Cryptology – CRYPTO 2011. Lecture Notes in Computer Science, vol. 6841, pp. 630–648. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 14–18, 2011)
16. Ghadafi, E.: Efficient round-optimal blind signatures in the standard model. In: Kiayias, A. (ed.) FC 2017: 21st International Conference on Financial Cryptography and Data Security. Lecture Notes in Computer Science, vol. 10322, pp. 455–473. Springer, Heidelberg, Germany, Sliema, Malta (Apr 3–7, 2017)
17. Goldberg, S., Reyzin, L., Sagga, O., Baldimtsi, F.: Efficient noninteractive certification of RSA moduli and beyond. In: Galbraith, S.D., Moriai, S. (eds.) Advances in Cryptology – ASIACRYPT 2019, Part III. Lecture Notes in Computer Science, vol. 11923, pp. 700–727. Springer, Heidelberg, Germany, Kobe, Japan (Dec 8–12, 2019)
18. Goldreich, O., Goldwasser, S., Micali, S.: How to construct random functions (extended abstract). In: 25th Annual Symposium on Foundations of Computer Science. pp. 464–479. IEEE Computer Society Press, Singer Island, Florida (Oct 24–26, 1984)

19. Grontas, P., Pagourtzis, A., Zacharakis, A., Zhang, B.: Towards everlasting privacy and efficient coercion resistance in remote electronic voting. In: Zohar, A., Eyal, I., Teague, V., Clark, J., Bracciali, A., Pintore, F., Sala, M. (eds.) FC 2018 Workshops. Lecture Notes in Computer Science, vol. 10958, pp. 210–231. Springer, Heidelberg, Germany, Nieuwpoort, Curaçao (Mar 2, 2019)
20. Hauck, E., Kiltz, E., Loss, J.: A modular treatment of blind signatures from identification schemes. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019, Part III. Lecture Notes in Computer Science, vol. 11478, pp. 345–375. Springer, Heidelberg, Germany, Darmstadt, Germany (May 19–23, 2019)
21. Hauck, E., Kiltz, E., Loss, J., Nguyen, N.K.: Lattice-based blind signatures, revisited. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020, Part II. Lecture Notes in Computer Science, vol. 12171, pp. 500–529. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 17–21, 2020)
22. Heilman, E., Baldimtsi, F., Goldberg, S.: Blindly signed contracts: Anonymous on-blockchain and off-blockchain bitcoin transactions. In: Clark, J., Meiklejohn, S., Ryan, P.Y.A., Wallach, D.S., Brenner, M., Rohloff, K. (eds.) FC 2016 Workshops. Lecture Notes in Computer Science, vol. 9604, pp. 43–60. Springer, Heidelberg, Germany, Christ Church, Barbados (Feb 26, 2016)
23. Juels, A., Luby, M., Ostrovsky, R.: Security of blind digital signatures (extended abstract). In: Kaliski Jr., B.S. (ed.) Advances in Cryptology – CRYPTO'97. Lecture Notes in Computer Science, vol. 1294, pp. 150–164. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 17–21, 1997)
24. Kastner, J., Loss, J., Xu, J.: On pairing-free blind signature schemes in the algebraic group model. In: PKC 2022 (to appear). Lecture Notes in Computer Science, Springer, Heidelberg, Germany (2022)
25. Katz, J., Loss, J., Rosenberg, M.: Boosting the security of blind signature schemes. In: Advances in Cryptology – ASIACRYPT 2021. Lecture Notes in Computer Science, vol. 13093, pp. 468–492. Springer International Publishing, Cham (2021)
26. Okamoto, T.: Provably secure and practical identification schemes and corresponding signature schemes. In: Brickell, E.F. (ed.) Advances in Cryptology – CRYPTO'92. Lecture Notes in Computer Science, vol. 740, pp. 31–53. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 16–20, 1993)
27. Okamoto, T.: Efficient blind and partially blind signatures without random oracles. In: Halevi, S., Rabin, T. (eds.) TCC 2006: 3rd Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 3876, pp. 80–99. Springer, Heidelberg, Germany, New York, NY, USA (Mar 4–7, 2006)
28. Okamoto, T., Ohta, K.: Universal electronic cash. In: Feigenbaum, J. (ed.) Advances in Cryptology – CRYPTO'91. Lecture Notes in Computer Science, vol. 576, pp. 324–337. Springer, Heidelberg, Germany, Santa Barbara, CA, USA (Aug 11–15, 1992)
29. Pointcheval, D.: Strengthened security for blind signatures. In: Nyberg, K. (ed.) Advances in Cryptology – EUROCRYPT'98. Lecture Notes in Computer Science, vol. 1403, pp. 391–405. Springer, Heidelberg, Germany, Espoo, Finland (May 31 – Jun 4, 1998)
30. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. Journal of Cryptology 13(3), 361–396 (Jun 2000)
31. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Shmoys, D.B. (ed.) 46th Annual ACM Symposium on Theory of Computing. pp. 475–484. ACM Press, New York, NY, USA (May 31 – Jun 3, 2014)

32. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: Fumy, W. (ed.) Advances in Cryptology – EUROCRYPT'97. Lecture Notes in Computer Science, vol. 1233, pp. 256–266. Springer, Heidelberg, Germany, Konstanz, Germany (May 11–15, 1997)
33. Tessaro, S., Zhu, C.: Short pairing-free blind signatures with exponential security. Cryptology ePrint Archive, Report 2022/047 (2022), https://eprint.iacr.org/2022/047