Efficient Round Optimal Blind Signatures

Sanjam Garg* IBM T. J. Watson

Divya Gupta UCLA

Abstract

Known constructions of blind signature schemes suffer from at least one of the following limitations: (1) rely on parties having access to a common reference string or a random oracle, (2) are not round-optimal, or (3) are prohibitively expensive.

In this work, we construct the *first* blind-signature scheme that does not suffer from any of these limitations. In other words, besides being round optimal and having a standard model proof of security, our scheme is very efficient. Specifically, in our scheme, one signature is of size 6.5 KB and the communication complexity of the signing protocol is roughly 100 KB. An amortized variant of our scheme has communication complexity less that 1 KB.

Keywords: Blind Signatures, Round Complexity, Efficiency, Complexity Leveraging.

*Research conducted while at the IBM Research, T.J.Watson funded by NSF Grant No.1017660.

1 Introduction

Blind signatures, introduced by Chaum [Cha82], allow users to obtain signatures on messages of their choice without revealing the messages itself to the signer. Additionally, the blind signature scheme should satisfy unforgeability, i.e. no user can produce additional signatures on messages without interacting with the signer. Blind signatures have widespread applications such as e-cash, e-voting, and anonymous credentials.

Even after 30 years of research, and with 50+ candidate schemes in the literature, the state of the art is not completely satisfactory. Essentially, all schemes in the literature can be partitioned into two categories – (1) the schemes that rely on a random oracle or a setup, or (2) the schemes which are round inefficient. Examples of constructions argued to be secure using the random oracle methodology [BR94] include [PS96a, PS96b, Poi98, Abe01, BNPS01, Bol03] and using a setup such as a shared random string include [AO09, AHO10, Fis06, Fuc09, KZ06, Oka06, MSF10]. On the other hand, essentially all schemes that avoid the use of the random oracle methodology or a setup [JLO97, CKW04, Oka06, HKKL07] are not round optimal.

The only scheme that does not fall in the above two categories is the recent construction of Garg et al. [GRS+11]. Unfortunately, this scheme is prohibitively expensive. For example, the communication complexity of this protocol is a large polynomial in the security parameter[1]. In this work, we ask the following question:

Can we construct a very efficient round optimal *blind signature scheme without relying on a random oracle or a setup?*

1.1 Our Results

We construct the *first* blind signature scheme that avoids all of the above limitations, namely it is very efficient, round optimal and does not rely on a random oracle or a setup. We obtain parameters for our scheme by using the concept of work factors from [Gal04, BR09]. A summary of the results is highlighted in Table 1.

Table 1: Comparing the Efficiency of Different Round Optimal Blind Signature Schemes. κ is the security parameter of the scheme. $\epsilon > 1$ is an appropriate constant. The concrete parameters above correspond to the setting for 80 bits of security.

Scheme	Communication Complexity		Signature Size	
	Asymptotic	Concrete	Asymptotic	Concrete
[GRS+11]	$\mathsf{poly}(\kappa)$		small[2]	
DLIN (This work)	$O(\kappa^{1+\epsilon})$	100.6KB	$O(\kappa^{\epsilon})$	6.5KB
Amortized (This work)	$O(\kappa^{\epsilon})$	836 Bytes	$O(\kappa^{\epsilon})$	6.5KB
q-SFP (This work)	$O(\kappa^{1+\epsilon})$	100.2KB	$O(\kappa^{\epsilon})$	3.2KB
Amortized (This work)	$O(\kappa^{\epsilon})$	472 bytes	$O(\kappa^{\epsilon})$	3.2KB

[1] To give an estimate on how big this polynomial is, we instantiate the proofs being given in their construction with Dwork-Naor Zaps using Kilian-Petrank NIZKs and get communication complexity of at least $O(\kappa^9)$ bits. One can also use asymptotically more efficient ZAPs instantiated with PCP based Groth NIZKs with ultimate proof size being $O(\kappa^5 \text{poly}\log(\kappa))$. Note that polylog($\kappa$) factor is quite large and for reasonable security parameters proof size would be comparable to $O(\kappa^7)$.

[2] This scheme uses general MPC techniques and can be instantiated using arbitrary signature scheme and thus has small signatures.

- **Standard Setting:** We assume the sub-exponential hardness of Decisional Linear (DLIN) Assumption and a variant of the discrete-log assumption. Then our signature scheme has one signature of size 6.5 KB and the communication complexity of signing protocol roughly 100 KB.

- **Amortized Setting:** A number of applications require a user to obtain multiple signatures from the same signer. In such a setting, for our scheme almost all of the communication costs can be avoided. More specifically an amortized variant of our scheme has communication cost roughly 100 KB when obtaining the first signature. However, for every subsequent signature obtained the communication cost is less that 1 KB.

- **Stronger assumption:** Assuming a stronger assumption, namely sub-exponentially hard q-Simultaneous Flexible Pairing Assumption (SFP) from [AHO10] we can improve the size of a signature and the amortized communication complexity of our signing protocol by roughly a factor of 2.

Qualitative Improvements. [GRS$^+$11] uses *complexity leveraging* to obtain *standard model* round optimal blind signature scheme, and it is the use of these techniques which makes this scheme so inefficient. However, unfortunately, impossibility results of Fischlin et al. [FS10] and Pass [Pas11] roughly indicate that the use of these techniques is essential for getting round optimal scheme in the standard model. Nonetheless, in this work, we introduce new techniques to *reduce* and *optimize* the use of complex leveraging, and thereby obtain a significantly more efficient scheme.

- **Reducing the use of complexity leveraging.** The technique of complexity leveraging works by creating a gap between the power of an adversarial entity and the reduction proving security. However, many a times this gap needs to be created multiple times in a layered fashion leading to larger parameters. The previous scheme of Garg et al. [GRS$^+$11] needed to create this gap twice. However, in our scheme, we only need to create this gap once and this allows us to get smaller parameters.

- **Optimizing the use of complexity leveraging.** Complexity leveraging techniques (particularly for our application) inherently make non-black-box use of the underlying primitives. [GRS$^+$11] in their construction end up rolling out the cryptographic primitive and viewing it as circuit. This leads to prohibitively inefficient schemes. We also make non-black-box use of the underlying primitive but avoid viewing it as a circuit. Instead, we cast it directly as a set of very structured equations which fit the framework of Groth-Sahai proofs, drastically improving the communication complexity of our protocol.

The techniques developed here are very general and we believe that they should be applicable to other settings. We leave this exploration for future work.

1.2 Technical Difficulties and New Ideas

Now we will describe the key ideas behind our scheme. We assume some familiarity with Groth-Sahai proofs. Lets us start by reminding the reader that GS proofs come in two modes – the hiding mode and the binding mode. In hiding mode, proofs reveal nothing about the witness used in the generation of a proof, and in binding mode, no fake proof exists.

Starting Point. The starting point for our construction is to use a blind signature scheme in the common reference string (CRS) model and remove the need for the CRS by letting the signer generate it. Of course this is problematic because a malicious signer can generate the CRS dishonestly (e.g. in a way such that it knows the trapdoors associated with the CRS) and use that to break the blindness property of the scheme. We solve this problem by using a special blind signature scheme for which blindness is statistical as long as the CRS is sampled from a certain "honest" distribution. In this setting, it is enough for the signer to prove that the CRS is sampled from the "honest" distribution. Looking ahead, this "honest" distribution is actually the CRS distribution for GS proofs in the hiding mode. However, we are faced with the following three issues.

Issue 1) First, in order to ensure blindness, the signer needs to prove to the user that the CRS was indeed sampled from the "honest" distribution.

Issue 2) Secondly, for proving unforgeability we will need that the reduction playing as the signer can "simulate" this proof. In other words, we need that the proof does not leak anything to the user.

Issue 3) The third issue is more subtle and arises as an interleaving of the first two issues. Specifically, the reduction for arguing unforgeability should be able to "extract" the messages on which the signatures are being issued and simulate the view of the attacking user. In other words, this extraction and simulation process should go unnoticed in the view of the attacking user. However, if a cheating signer could replicate the same behavior then this would go unnoticed as well. Hence, we certainly need to rule this out.

Before we describe our attempts to solve these issues, we note that for [GRS$^+$11], this proof is the main reason for inefficiency.

Attempt at using range proofs. As mentioned before, complexity leveraging makes non-black-box use of primitives essential making schemes prohibitively inefficient. In order to solve this issue we need to identify a problem such that: (1) the problem can be algebraically stated in groups of prime order p and has an efficient Groth-Sahai proof, (2) but solving the problem should be much easier than solving discrete log in the group of order p. The first property of the problem ensures efficiency of the proof. The second property as we will see later will be essential in making the complexity leveraging argument. We start by using a simple problem of solving discrete-log when the domain is restricted to some subspace. In particular, the problem we consider is: Given $C = g^c$ such that $c < q$ (where $q << p$), one needs to find c. We then show that it satisfies both the above properties. In particular, we will show that this problem can be cast in the language of efficient Groth-Sahai proofs thus satisfying the first requirement. Secondly, improvement in the brute-force attack when the sample space is restricted to $c < q$ is easy to see.

For the protocol, our idea is that user sends the value g^c for $c < q$ to the signer. Further instead of having the signer prove that the CRS was sampled honestly we have him prove that either the CRS was honestly generated or that it is aware of c. This immediately solves our problems 1 and 2 from above. We know that a cheating signer will not be able to recover c and hence will not be able to cheat. At the same time we can have the reduction for unforgeability extract c and thereby generate simulated proofs.

However our solution to issues 1 and 2 has created a 4^{th} issue. A cheating user may cheat by generating g^c such that $c \geq q$. Next, we will show how issues 3 and 4 can be solved.

Solving issue 3. Very interestingly we can resolve issue 3 by requiring that the signer generates the proof above under the CRS he had sampled for the underling blind signature scheme. This is very counter-intuitive as we are requiring the signer to generate a proof under a CRS that it generates on its own. The key idea is based on the observation that all we need is that the signer generates the CRS from the hiding distribution for Groth-Sahai proofs. If this CRS is indeed hiding then the whole exercise of having a proof is redundant. On the other hand, if this CRS is actually generated dishonestly from the binding distribution then the signer is only hurting itself as it will not be able to generate his proof. [3]

Solving issue 4. Recall that the 4^{th} issue was that the user might generate g^c in a way such that $c > q$. We solve this problem by having the user provide a Groth-Sahai proof that the value c used is less that q. A question is under what CRS should this proof be give such that this proof does not leak c to the signer? Of course, we can not use the CRS that the signer generated for the underlying scheme. Our key observation here again is that we need to worry about this proof only if the original CRS has been generated maliciously, or in other words, if this CRS is binding. Recall that a binding CRS for Groth-Sahai proofs is a DLIN tuple. Our key idea here is that if $(g, g_1, g_2, h_1, h_2, h)$ is a DLIN tuple then its shift $(g, g_1, g_2, h_1, h_2, h \cdot g)$ can not be a DLIN tuple and hence the user can give his proof under this shifted CRS.[4]

2 Preliminaries

In this section, we recall and define basic notation and primitives used. Let λ denote the security parameter. We call a function *negligible* in λ if it is asymptotically smaller than any inverse polynomial. For a detailed description of blind signatures and its security properties refer to Appendix B. Now, we define the primitives used in our construction.

Commitment scheme on groups. We describe a perfectly binding commitment scheme based on the decisional linear (DLIN) assumption (see definition 1) with the special property that both the message space and the commitment comprise only of group elements. Let $(p, \mathbb{G}, \mathbb{G}_T, g, e)$ be a prime order bilinear pairing group. Then the function $\mathsf{Com}_{\mathbb{G}}(\cdot)$ generates a commitment to an element $m \in \mathbb{G}$ by first sampling $g_1, g_2 \xleftarrow{\$} \mathbb{G}$, $x, y \xleftarrow{\$} \mathbb{Z}_p$ and then outputting $(g, g_1, g_2, g_1^x, g_2^y, m \cdot g^{x+y})$. For description of hiding and binding properties of commitment schemes see Appendix A.2.

Structure-Preserving Signatures. A signature scheme $(\mathsf{SPGen}, \mathsf{SPSign}, \mathsf{SPVerify})$ is said to be a *structure preserving signature scheme* over a prime order bilinear group $(p, \mathbb{G}, \mathbb{G}_T, g, e)$ (see Appendix A.1), if public keys, signatures and messages to be signed are vectors of group elements and verification only evaluates pairing product equations. For definition of security of signature schemes refer to Appendix A.3. Structure preserving signature schemes that sign a vector of group elements are known under different assumptions [Gro06, AHO10, ACD$^+$12]. The first feasibility result was given by Groth [Gro06]. This scheme is inefficient as the signature size grows linearly with the number of group elements in the message to be signed and the constants are quite big. In our scheme, we will use constant size structure preserving signatures [AHO10, ACD$^+$12]. Both of

[3] In the final construction (Figure 1), the signer will prove under the CRS he had sampled that it is aware of c. An honest signer who generates a hiding CRS will be able to simulate this proof successfully.

[4] A similar idea was also used by [Gro06] to get perfectly sound NIWI in the standard model using statistically sound NIZKs.

these results have been summarized in the table given below. The size of different parameters are in terms of number of group elements.

Table 2: Efficiency of Structure Preserving Schemes

Scheme	$\|msg\|$	$\|gk\| + \|vk\|$	$\|\sigma\|$	#(PPE)	Assumption
AHO10	k	$2k + 12$	7	2	q-SFP
ACDKNO12	k	$2k + 25$	17	9	DLIN

When k is a constant, a public key as well as a signature generated consist of a constant number of group elements only. Hence, these schemes are highly efficient for constant size messages. From the security of these schemes, it follows that under assumptions which are hard to break in time $\mathrm{T}\cdot\mathsf{poly}(\lambda)$, these schemes are secure against existential forgery under chosen message attack for adversaries running in time $\mathrm{T}\cdot\mathsf{poly}(\lambda)$. More precisely, these schemes are T-$\mathsf{eu\text{-}cma\text{-}secure}$ (see Appendix A.3) under hardness of T-q-SFP and T-DLIN, respectively.

2.1 Two-CRS Non-interactive Zero-Knowledge Proofs.

In this section, we will define a special notion of NIZK proofs that work in the setting with two common reference strings.

Let R be an efficiently computable binary relation. For pairs $(x, w) \in R$ we call x the statement and w the witness. Let L be the language consisting of statements in R. A *Two-CRS* non-interactive proof system for a relation R consists of three common reference string (CRS) generation algorithms K_B, Shift and Shift^{-1}, a prover algorithm $\mathcal{P}$ and a verification algorithm $\mathcal{V}$. We require that all these algorithms be efficient, i.e. polynomial time. The CRS generation algorithm K_B takes the security parameter 1^λ as input and produces a common reference string crs along with an extraction key τ. Both Shift and Shift^{-1} are deterministic algorithms. They take as input a string crs and output another string crs'. The prover algorithm $\mathcal{P}$ takes as input (crs, x, w) and produces a proof π. The verification algorithm $\mathcal{V}$ takes as input (crs, x, π) and outputs 1 or 0. We require that:

CRS Indistinguishability. For all PPT adversaries $\mathcal{A}$, we define $\mathsf{Adv}^{\mathsf{CRS-distinguish}}_{\mathcal{A}}$ as

$$\mathsf{Adv}^{\mathsf{CRS-distinguish}}_{\mathcal{A}}(1^\lambda) = 2\cdot\Pr\left[b = b' \;\middle|\; \begin{array}{l} (\mathsf{crs}, \tau) \leftarrow K_B(1^\lambda); \mathsf{crs}' \leftarrow \mathsf{Shift}(\mathsf{crs}); \mathsf{crs}'' \leftarrow \mathsf{Shift}^{-1}(\mathsf{crs}) \\ b \xleftarrow{\$} \{0,1\}; \quad \text{if } b = 0, (\mathsf{crs}_1, \mathsf{crs}_2) := (\mathsf{crs}, \mathsf{crs}') \\ \quad \text{else } (\mathsf{crs}_1, \mathsf{crs}_2) := (\mathsf{crs}'', \mathsf{crs}) \\ b' \leftarrow \mathcal{A}(\mathsf{crs}_1, \mathsf{crs}_2) \end{array}\right] - 1 \,.$$

We say that a Two-CRS NIZK system has CRS indistinguishability if for all PPT adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{CRS-distinguish}}_{\mathcal{A}}$ is negligible in λ.

Perfect Completeness. Completeness requires that an honest prover with a valid witness can always make an honest verifier output 1. For $K \in \{K_B, \mathsf{Shift} \circ K_B, \mathsf{Shift}^{-1} \circ K_B\}$, where $\circ$ is the the composition of functions, we require that for all x, w such that $(x, w) \in R$:

$$\Pr\left[\mathcal{V}(\mathsf{crs}, x, \pi) = 1 \;\middle|\; \mathsf{crs} \leftarrow K(1^\lambda); \pi \leftarrow \mathcal{P}(\mathsf{crs}, x, w)\right] = 1.$$

Perfect Knowledge Extraction. We require that there exists a probabilistic polynomial time knowledge extractor $\mathcal{E}$ such that for every $(\mathsf{crs}, \tau) \leftarrow K_B(1^\lambda)$, x and purported proof π such that $\mathcal{V}(\mathsf{crs}, x, \pi) = 1$ then we have

$$\Pr\left[(x, w) \in R \;\middle|\; w := \mathcal{E}(\mathsf{crs}, \tau, x, \pi)\right] = 1.$$

Note that since perfect knowledge extraction implies the existence of a witness for the statement being proven, it implies *perfect soundness*.

Perfect Zero-Knowledge. A proof system is zero-knowledge if the proofs do not reveal any information about the witnesses. We require that there exists a polynomial time simulator $\mathcal{S}$ such that for all $(\mathsf{crs}, \tau) \leftarrow K_B(1^\lambda), \mathsf{crs}' := \mathsf{Shift}(\mathsf{crs})$ (or, $\mathsf{crs}' := \mathsf{Shift}^{-1}(\mathsf{crs})$) we have that for all $x \in L$ the distributions $\mathcal{P}(\mathsf{crs}', x, w)$ and $\mathcal{S}(\mathsf{crs}', \tau, x)$ are identical.

Efficient realization of Two-CRS NIZKs based on Groth-Sahai Proofs. Groth-Sahai proofs [GS08] can be used to give efficient Two-CRS NIZKs (under the DLIN assumption) for special languages, namely pairing product equations, multi-scalar multiplication equations, and quadratic equations (described below) in the setting of symmetric bilinear groups. For details, refer to Appendix C. We also show that the range equations also fit this framework (Appendix C.1). Next, we describe these equations formally.

- **Pairing Product Equation.** A pairing product equation (PPE) over the variables $X_1, \ldots X_n \in \mathbb{G}$ is an equation of the form[5]

$$\prod_{i=1}^{n} e(\mathcal{A}_i, X_i) \cdot \prod_{i=1, j \geq i}^{n,n} e(X_i, X_j)^{\gamma_{i,j}} = 1,$$

  determined by constants $\mathcal{A}_i \in \mathbb{G}$ and $\gamma_{i,j} \in \mathbb{Z}_p$.

- **Multiscalar Multiplication Equation.** A multiscalar multiplication equation over the variables $X_1, \ldots X_n \in \mathbb{G}$ and $y_1, y_2, \ldots, y_m \in \{0,1\}$ is of the form[6]

$$\prod_{j=1}^{m} \mathcal{A}_j^{y_j} \cdot \prod_{i=1}^{n} X_i^{b_i} \cdot \prod_{i=1}^{n} \prod_{j=1}^{m} X_i^{\gamma_{i,j} y_j} = \mathcal{T},$$

  determined by constants $\mathcal{A}_j \in \mathbb{G}$, $b_i, \gamma_{i,j} \in \mathbb{Z}_p$, and $\mathcal{T} \in \mathbb{G}$.

- **Quadratic Equation.** A quadratic equation in $\mathbb{Z}_p$ over variables $y_1, y_2, \ldots, y_n \in \{0,1\}$ is of the form[7]

$$\sum_{i=1}^{n} a_i y_i \; + \sum_{i=1, j \geq i}^{n,n} \gamma_{i,j} y_i y_j = t,$$

  determined by constants $a_i \in \mathbb{Z}_p$, $\gamma_{i,j} \in \mathbb{Z}_p$, and $t \in \mathbb{Z}_p$.

- **Range Equation.** The range equation over the variable $c \in \mathbb{Z}_p$ is of the form.

$$\exists c \; : \; g^c = C \bigwedge c < q,$$

  determined by constants $C \in \mathbb{G}$ and $q < p$. We note that the range equation is not explicitly a part of the Groth-Sahai framework but is implied by it. We establish this in Appendix C.1.

[5] General form of PPE can have any $\mathcal{T} \in \mathbb{G}_T$ on the R.H.S. Since GS NIZKs are only known for PPE having 1 on the R.H.S., we use only such equations in our construction.

[6] Multiscalar Multiplication Equations can be defined for more general setting when $y_1, \ldots, y_m$ come from $\mathbb{Z}_p$. In our case, we define it for a more restricted setting because of technical reason discussed in Appendix C.

[7] Quadratic Equations can be defined for more general setting when $y_1, \ldots, y_m$ come from $\mathbb{Z}_p$. In our case, we define it for a more restricted setting because of technical reason discussed in Appendix C.

Remark 1. *We note that for the first three kinds of equations, under the above mentioned realization of Two-CRS NIZKs, the proof size grows only linearly with the number of variables and the number of equations. This follows directly from the GS proofs as explained in Appendix C.*

Remark 2. *As shown in Appendix C.1, a range equation can be expressed as one multiscalar multiplication equation and* $\log_2 q$ *quadratic equations over* $\log_2 q$ *variables in* $\mathbb{Z}_p$.

3 Blind Signature Scheme: Construction

We begin by giving an informal description of the scheme. In our scheme, we will use a bilinear group $\mathbb{G}$ of prime order p, a structure preserving signature scheme for signing vectors of elements in this group, Two-CRS NIZKs, and commitment scheme $\mathsf{Com}_{\mathbb{G}}$.

During the key generation phase, the signer generates the verification key vk and the secret key sk for the blind signature scheme as follows. vk consists of a verification key $\mathrm{vk}_{\mathsf{SP}}$ for the structure preserving signature scheme, two CRSes, crs_1 and crs_2 under Two-CRS NIZK proof system, and a parameter $q = p^{\epsilon}$ for some constant $\epsilon \in (0, 1)$. crs_1 is sampled from K_B and crs_2 is set to be the shifted crs_1, i.e. $\mathrm{crs}_2 \leftarrow \mathsf{Shift}(\mathrm{crs}_1)$. sk consists of the signing key $\mathrm{sk}_{\mathsf{SP}}$ corresponding to $\mathrm{vk}_{\mathsf{SP}}$ and the extraction key τ for crs_1.

Next, the two round blind signature scheme proceeds as follows: In the first round, the user generates its message as follows: It begins by checking whether crs_2 equals $\mathsf{Shift}(\mathrm{crs}_1)$. It aborts, if this is not the case. Next, it blinds its message m by generating a commitment m_{blind} using $\mathsf{Com}_{\mathbb{G}}$ under randomness r. Then, it samples a random $c < q$ and sets $C = g^c$. Finally, it generates a proof π under crs_1 for the NP-statement Φ: $\exists\ c \ \mid\ g^c = C \ \bigwedge\ c < q$. It sends $(m_{\mathsf{blind}}, C, \pi)$ as the first round message to the signer.

In the second round, the signer generates its message as follows: It begins by checking if the proof π is valid under crs_1. It aborts, if this is not the case. Next, it extracts the witness c from the proof π using extraction key τ. Then it generates a fresh proof π' for the statement Φ under crs_2. Finally, it generates a signature σ_{SP} on m_{blind} using signing key $\mathrm{sk}_{\mathsf{SP}}$. It sends $(\pi', \sigma_{\mathsf{SP}})$ as the second round message to the user.

On receiving the above message from the user, it computes the signature on m as follows: User aborts if π' is not a valid proof under crs_2. It then checks if σ_{SP} is a valid signature on m_{blind} under $\mathrm{vk}_{\mathsf{SP}}$. It aborts if this is not the case. Otherwise, it outputs σ as the proof under crs_2 of the NP-statement Ψ: $\exists\ (m_{\mathsf{blind}}, r, \sigma_{\mathsf{SP}}) \ \mid\ m_{\mathsf{blind}} = \mathsf{Com}_{\mathbb{G}}(m; r) \bigwedge \mathsf{SPVerify}(\mathrm{vk}_{\mathsf{SP}}, m_{\mathsf{blind}}, \sigma_{\mathsf{SP}}) = 1$. In other words, the user proves that there exists $(m_{\mathsf{blind}}, r, \sigma_{\mathsf{SP}})$ such that m_{blind} is the commitment of m using randomness r under commitment scheme $\mathsf{Com}_{\mathbb{G}}$ and σ_{SP} is a valid signature on m_{blind}.

To verify a signature σ on message m, check whether σ is a valid proof for the statement Ψ under crs_2.

Formal Description. Let $\mathsf{SPSig} = (\mathsf{SPGen}, \mathsf{SPSign}, \mathsf{SPVerify})$ be any structure preserving signature scheme which is existentially unforgeable, $(K_B, \mathsf{Shift}, \mathsf{Shift}^{-1}, \mathcal{P}, \mathcal{V})$ be a Two-CRS NIZK proof system, $\mathsf{Com}_{\mathbb{G}}$ be the DLIN based commitment scheme for elements in $\mathbb{G}$ (Section 2). Formal description of the blind signature scheme $(\mathsf{Gen}, \mathcal{S}, \mathcal{U}, \mathsf{Vrfy})$ is given in Figure 1.

4 Proof of Unforgeability

Let $\mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q}$ be the time it takes to break the discrete log problem in $\mathbb{G}$ when exponents are chosen from $\mathbb{Z}_q$.

Recalling from Section 2, let $(\mathsf{SPGen}, \mathsf{SPSign}, \mathsf{SPVerify})$ be an existentially unforgeable structure preserving signature scheme, $(K_B, \mathsf{Shift}, \mathsf{Shift}^{-1}, \mathcal{P}, \mathcal{V})$ be a Two-CRS NIZK proof system and $\mathsf{Com}_{\mathbb{G}}$ be a group based commitment scheme. And let $0 < \epsilon < 1$ be an appropriate (specified later) constant parameter.

Key Generation Gen**:** On input 1^λ, choose an appropriate bilinear group $(p, \mathbb{G}, \mathbb{G}_T, g, e)$[a] and proceed as follows:

- Sample a key pair for the structure preserving signature scheme $(\mathrm{sk}_{\mathsf{SP}}, \mathrm{vk}_{\mathsf{SP}}) \leftarrow \mathsf{SPGen}(1^\lambda)$.
- Sample a CRS $(\mathrm{crs}_1, \tau) \leftarrow K_B(1^\lambda)$ and generate its shift $\mathrm{crs}_2 \leftarrow \mathsf{Shift}(\mathrm{crs}_1)$.
- Output the verification-key for the blind signature scheme as $vk = (\mathrm{vk}_{\mathsf{SP}}, \mathrm{crs}_1, \mathrm{crs}_2, q = p^\epsilon)$ and the secret-key as $sk = (\mathrm{sk}_{\mathsf{SP}}, \tau)$.

Signing Protocol: The user $\mathcal{U}$ with input $m \in \mathbb{G}, \mathrm{vk}_{\mathsf{SP}}$ and the signer $\mathcal{S}$ with input $\mathrm{sk}_{\mathsf{SP}}$ proceed as follows.

- **Round 1:** The user $\mathcal{U}$ generates its first message as follows:
  - Abort if $\mathrm{crs}_2 \neq \mathsf{Shift}(\mathrm{crs}_1)$.
  - Sample $m_{\mathsf{blind}} \leftarrow \mathsf{Com}_{\mathbb{G}}(m; r)$.
  - Samples a uniformly random c such that $c < q$ and sets $C := g^c$. Next sample a proof $\pi \leftarrow \mathcal{P}(\mathrm{crs}_1, \Phi, c)$ where Φ is the NP-statement:

$$\exists\ c \ \mid\ g^c = C \bigwedge c < q. \tag{1}$$

  - Send $(m_{\mathsf{blind}}, C, \pi)$ to the signer.
- **Round 2:** $\mathcal{S}$ generates the second round message as:
  - If $\mathcal{V}(\mathsf{crs}_1, \Phi, \pi) \neq 1$ then abort, otherwise obtain $c := \mathcal{E}(\mathrm{crs}_1, \tau, \Phi, \pi)$ and sample a proof $\pi' \leftarrow \mathcal{P}(\mathrm{crs}_2, \Phi, c)$.
  - Sample a signature $\sigma_{\mathsf{SP}} := \mathsf{SPSign}(\mathrm{sk}_{\mathsf{SP}}, m_{\mathsf{blind}})$.
  - Send $(\pi', \sigma_{\mathsf{SP}})$ to the user $\mathcal{U}$.
- **Signature Generation:** $\mathcal{U}$ aborts if $\mathcal{V}(\mathrm{crs}_2, \Phi, \pi') \neq 1$. $\mathcal{U}$ also aborts if $\mathsf{SPVerify}(\mathrm{vk}_{\mathsf{SP}}, m_{\mathsf{blind}}, \sigma_{\mathsf{SP}}) \neq 1$ and otherwise outputs $\sigma \leftarrow \mathcal{P}(\mathrm{crs}_2, \Psi, (m_{\mathsf{blind}}, r, \sigma_{\mathsf{SP}}))$ where Ψ is the NP-statement:

$$\exists\ (m_{\mathsf{blind}}, r, \sigma_{\mathsf{SP}}) \ \mid\ m_{\mathsf{blind}} = \mathsf{Com}_{\mathbb{G}}(m; r) \bigwedge \mathsf{SPVerify}(\mathrm{vk}_{\mathsf{SP}}, m_{\mathsf{blind}}, \sigma_{\mathsf{SP}}) = 1 \tag{2}$$

Signature Verification Vrfy**:** For input a claimed signature σ on message m, output $\mathcal{V}(\mathrm{crs}_2, \Psi, \sigma)$.

[a] All algorithms take this bilinear group as an implicit input.

Figure 1: Blind Signature Scheme

Theorem 1. *For any PPT malicious user $\mathcal{U}^*$ for the unforgeability game against the blind signature scheme given in Section 3 the following holds:*

$$\mathsf{Adv}^{\mathsf{Unforge}}_{\mathcal{U}^*, \mathsf{BS}}(\lambda) \leq \mathsf{Adv}^{\mathsf{CRS-distinguish}}_{\mathcal{B}}(\lambda) + \mathsf{Adv}^{\mathsf{Unforge}}_{\widehat{\mathcal{U}^*}, \mathsf{SPSig}}(\lambda),$$

where $\mathcal{B}$ is an adversary against the CRS indistinguishability property of the two-CRS NIZK proof system such that $\mathbf{T}(\mathcal{B}) = k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)$ and $\widehat{\mathcal{U}^*}$ is the adversary against the unforgeability

of the underlying structure preserving signature scheme SPSig *such that* $\mathbf{T}(\widehat{\mathcal{U}^*}) = k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)$. *Also,* $\mathcal{U}^*$ *and* $\widehat{\mathcal{U}^*}$ *make at most* k *signing queries.*

If we use GS proof system based Two-CRS NIZKs (see Appendix C) in our construction, the above theorem immediately implies the following corollary:

Corollary 1. *For any PPT malicious user* $\mathcal{U}^*$ *for the unforgeability game against the blind signature scheme given in Section 3 the following holds:*

$$\mathsf{Adv}^{\mathsf{Unforge}}_{\mathcal{U}^*,\mathsf{BS}}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{dlin}}_{\mathcal{B},\mathbb{G}}(\lambda) + \mathsf{Adv}^{\mathsf{Unforge}}_{\widehat{\mathcal{U}^*},\mathsf{SPSig}}(\lambda),$$

where $\mathcal{B}$ *is an adversary against the DLIN assumption in* $\mathbb{G}$ *such that* $\mathbf{T}(\mathcal{B}) = k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)$ *and* $\widehat{\mathcal{U}^*}$ *is the adversary against the unforgeability of the underlying structure preserving signature scheme* SPSig *such that* $\mathbf{T}(\widehat{\mathcal{U}^*}) = k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)$. *Also,* $\mathcal{U}^*$ *and* $\widehat{\mathcal{U}^*}$ *make at most* k *signing queries.*

Following is a corollary of the above theorem:

Theorem 2. *Assume that* $\mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q}$*-DLIN holds in* $\mathbb{G}$ *and* SPSig *is* $\mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q}$-$\mathsf{eu\text{-}cma}$*-unforgeable. Then the blind signature scheme in Section 3 is unforgeable.*

Proof. (of Theorem 1) Let $\mathcal{U}^*$ be any PPT malicious user then we will prove our theorem by considering a sequence of games starting with the unforgeabilty game from Definition 7 (see Appendix B).

- Game_0**:** This is the challenger-adversary game between the challenger following the honest signer $\mathcal{S}$ specification and the malicious user $\mathcal{U}^*$. More specifically, the game starts with the challenger generating a key pair (sk, vk). The challenger then sends vk to $\mathcal{U}^*$. At this point the challenger (playing as the honest signer) and $\mathcal{U}^*$ proceed by interacting in k executions of the signing protocol. Note that the challenger knows the secret key sk and uses it to participate as the signer in the executions of the signing protocol. Finally $\mathcal{U}^*$ outputs $k+1$ message/signature pairs (m_i, σ_i). $\mathcal{U}^*$ is said to win if all the messages are distinct and all signatures verify under vk.

- Game_1**:** Recall that in the second round of the signing protocol the challenger (acting as the signer) obtains the secret value c using the extraction algorithm $\mathcal{E}$. Game_1 is same as the Game_0 except that in each of the k instances of the signing protocol, instead of extracting the secret c using the extraction algorithm, the challenger obtains c by evaluating the discrete log of C assuming that it is less than q. (The challenger aborts if no values less than q is a valid dlog of C.)

  Note that since crs_1 is sampled from K_B, proofs under crs_1 are perfectly sound. This implies that the value c that challenger extracts by solving discrete log is exactly the same as the one that challenger would have extracted using the extraction algorithm in Game_0.

  Note that the views of the malicious user $\mathcal{U}^*$ in games Game_0 and Game_1 are identical.

  It also follows from the perfect soundness of the two-CRS NIZK proof system that the challenger in Game_1 runs in time $k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + poly(\lambda)$, where $\mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q}$ is the time it takes to break discrete log problem in $\mathbb{G}$ when the exponent is chosen from $\mathbb{Z}_q$.

- Game_2**:** Game_2 is same as Game_1 except that the challenger generates the CRSes differently. Instead of generating the CRSes by first sampling $(\mathrm{crs}_1, \tau) \leftarrow K_B(1^\lambda)$ and then generating its

shift $\text{crs}_2 \leftarrow \mathsf{Shift}(\text{crs}_1)$, it reverses the order in which the CRSes are generated. This reverses the security properties of proofs under the two CRSes. More specifically the challenger first samples $(\text{crs}_2, \tau) \leftarrow K_B(1^\lambda)$ and then sets $\text{crs}_1 := \mathsf{Shift}^{-1}(\text{crs}_2)$. Note that now we get perfect zero-knowledge for crs_1 and perfect soundness for crs_2.

Indistinguishability of Game_1 and Game_2 follows from the CRS-Indistinguishability property of the two-CRS NIZK proof system. More precisely, the success probability of $\mathcal{U}^*$ can change by at most $\mathsf{Adv}^{\mathsf{CRS-distinguish}}_{\mathcal{B}}$, where $\mathcal{B}$ is an adversary against the CRS indistinguishability property of the two-CRS NIZK proof system such that $\mathbf{T}(\mathcal{B}) = k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)$.

Now we will show how $\mathcal{U}^*$ who wins in Game_2 can be used to construct a malicious user $\widehat{\mathcal{U}^*}$ that winning the existential unforgeability game of the underlying structure preserving signature scheme.

$\widehat{\mathcal{U}^*}$ starts by obtaining the verification key vk_{SP} from the challenger of the structure preserving signature scheme $(\mathsf{SPGen}, \mathsf{SPSign}, \mathsf{SPVerify})$. Furthermore, it samples $(\text{crs}_2, \tau) \leftarrow K_B(1^\lambda)$, sets $\text{crs}_1 := \mathsf{Shift}^{-1}(\text{crs}_2)$ and invokes $\mathcal{U}^*$ with $(\text{vk}_{\mathsf{SP}}, \text{crs}_1, \text{crs}_2, q)$ as input. At this point, the user $\mathcal{U}^*$ expects to interact in k instances of the signing protocol. In each of these executions, it provides its challenger (the adversary $\widehat{\mathcal{U}^*}$ in our case) with its first round message $(m_{\mathsf{blind}}, C, \pi)$. Our adversary $\widehat{\mathcal{U}^*}$ obtains c by solving the discrete log problem (aborting if $c \geq q$) and uses the extracted value to generate the response proof π'. Additionally, it obtains the signature σ_{SP} on m_{blind} from the signing oracle and passes $(\pi', \sigma_{\mathsf{SP}})$ to $\mathcal{U}^*$. After k such executions, $\mathcal{U}^*$ returns $k+1$ pairs (m_j, σ_j). Note that each σ_j given by $\mathcal{U}^*$ is a proof of knowledge of $(m_{\mathsf{blind},j}, r_j, \sigma_{\mathsf{SP},j})$ under crs_2. Furthermore, since $\widehat{\mathcal{U}^*}$ generates crs_2 in the binding setting, therefore τ can be used to extract $(m_{\mathsf{blind},j}, r_j, \sigma_{\mathsf{SP},j})$ for each j by invoking $\mathcal{E}(\text{crs}_2, \tau, \Psi, \sigma_j)$. Since all messages m_j are distinct and $\mathsf{Com}_{\mathbb{G}}$ is perfectly binding, all $m_{\mathsf{blind},j}$ will also be distinct. Since all $m_{\mathsf{blind},j}$ are distinct there exists at least one m_{blind,j^*} among these that $\widehat{\mathcal{U}^*}$ never queried its challenger. $\widehat{\mathcal{U}^*}$ outputs $(m_{\mathsf{blind},j^*}, \sigma_{\mathsf{SP},j^*})$ as its output.

Hence, the advantage of $\mathcal{U}^*$ in producing a valid forgery in Game_3 is at most the advantage of $\widehat{\mathcal{U}^*}$ in producing a valid forgery against the underlying structure preserving signature scheme, i.e. $\mathsf{Adv}^{\mathsf{Unforge}}_{\mathcal{U}^*,\mathsf{BS},\mathsf{Game}_3} \leq \mathsf{Adv}^{\mathsf{Unforge}}_{\widehat{\mathcal{U}^*},\mathsf{SPSig}}(\lambda)$, where $\widehat{\mathcal{U}^*}$ runs in time $k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)$. □

5 Proof of Blindness

Theorem 3. *For any PPT malicious signer $\mathcal{S}^*$ for the blindness game against the blind signature scheme given in Section 3, which successfully completes the blindness game, the following holds*

$$\mathsf{Adv}^{\mathsf{Unblind}}_{\mathcal{S}^*,\mathsf{BS}}(\lambda) < 2 \cdot \mathsf{Adv}^{\mathsf{hid}}_{\mathcal{A},\mathsf{Com}_{\mathbb{G}}} + \mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{B},\mathbb{G},q}$$

where $\mathcal{A}$ is an adversary against the non-uniform hiding property of $\mathsf{Com}_{\mathbb{G}}$ such that $\mathbf{T}(\mathcal{A}) = \mathbf{T}(\mathcal{S}^*) + poly(\lambda)$ and $\mathcal{B}$ is an adversary against the non-uniform discrete log problem in $\mathbb{G}$ when exponents are chosen uniformly randomly in $\mathbb{Z}_q$ such that $\mathbf{T}(\mathcal{B}) = \mathbf{T}(\mathcal{S}^*) + poly(\lambda)$.

Since the hiding property of the commitment scheme $\mathsf{Com}_{\mathbb{G}}$ holds under the DLIN assumption (definition 1) in $\mathbb{G}$, the above theorem immediately implies the following corollary.

Corollary 2. *For any PPT malicious signer $\mathcal{S}^*$ for the blindness game against the blind signature scheme given in Section 3, which successfully completes the blindness game, the following holds*

$$\mathsf{Adv}^{\mathsf{Unblind}}_{\mathcal{S}^*,\mathsf{BS}}(\lambda) < 4 \cdot \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{C},\mathbb{G}} + \mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{B},\mathbb{G},q}$$

where $\mathcal{C}$ is an adversary against the non-uniform DLIN *assumption in $\mathbb{G}$ such that $\mathbf{T}(\mathcal{C}) = \mathbf{T}(\mathcal{S}^*) + poly(\lambda)$ and $\mathcal{B}$ is an adversary against the non-uniform discrete log problem in $\mathbb{G}$ when exponents are chosen uniformly randomly in $\mathbb{Z}_q$ such that $\mathbf{T}(\mathcal{B}) = \mathbf{T}(\mathcal{S}^*) + poly(\lambda)$.*

Following is a corollary of the above.

Theorem 4. *Assume that non-uniform DLIN assumption holds in $\mathbb{G}$ and the non-uniform discrete log assumption holds in $\mathbb{G}$ even when the exponents are chosen uniformly randomly from $\mathbb{Z}_q$. Then the blind signature scheme from Section 3 is blind.*

Proof. (of Theorem 3) Let $\mathcal{S}^*$ be any PPT malicious signer then we will prove our theorem by considering a sequence of games starting with the blindness game from Definition 8 (see Appendix B).

- Game_0: This is a challenger-adversary game between the challenger following the honest user strategy and the malicious signer $\mathcal{S}^*$. The malicious signer $\mathcal{S}^*$ has full control over the scheduling of instances of the user in an arbitrary order. Since our scheme is only two round, we can fix it to be the worst case ordering. Since $\mathcal{S}^*$ does not receive any response to the message it sends to the user, we can assume that $\mathcal{S}^*$ first gathers all the incoming messages from the user and then sends its responses. Thus, without loss of generality, the Game_0 proceeds as follows: $\mathcal{S}^*$ first outputs the public key vk and the challenge messages m_0, m_1. $\mathcal{S}^*$ then expects the two incoming blinded messages $m_{blind,0}$ and $m_{blind,1}$ from the user corresponding to m_b, m_{1-b} for a random bit b. After receiving both the messages, $\mathcal{S}^*$ outputs its responses to the challenger. Our challenger at this point outputs the signature on (m_0, m_1) generated in the two protocol executions. Finally the malicious signer $\mathcal{S}^*$ outputs a bit b' and its advantage $\mathsf{Adv}^{\mathsf{Unblind}}_{\mathcal{S}^*,\mathsf{BS}}$ is equal to $|2 \cdot \Pr[b = b'] - 1|$.
- Game_1: Same as Game_0 except the following: The challenger after receiving the public key vk, figures out whether crs_2 is in the range of K_B or not. The challenger may execute in unbounded time when figuring this out; storing the extraction key τ for later use. Now it proceeds as follows:
  - crs_2 is in the range of K_B: In this case, our challenger proceeds just as in Game_0, except that if the first instance of the signing protocol completes successfully then our challenger outputs `DL-Abort`.
  - crs_2 is not in the range of K_B: Proceed as in Game_0.

  Note that conditioned on the fact that `DL-Abort` does not happen, we have that Game_0 and Game_1 are identical. Next we will show that the probability of `DL-Abort` happening is bounded by $\mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{B},\mathbb{G},q}$.

  Lemma 1. *The probability of `DL-Abort` happening is bounded by $\mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{B},\mathbb{G},q}$, with $\mathbf{T}(\mathcal{B}) = \mathbf{T}(\mathcal{S}^*) + poly(\lambda)$, $\mathcal{B}$ is an adversary against the non-uniform discrete log problem in $\mathbb{G}$ when exponents are chosen uniformly randomly in $\mathbb{Z}_q$.*

  Proof. We will show that an $\mathcal{S}^*$ that can make our challenger output `DL-Abort` can be used to construct an adversary $\mathcal{B}$ that breaks the non-uniform discrete log problem in $\mathbb{G}$ when the exponent is restricted to $< q$.

Constructing the adversary $\mathcal{B}$. Given this cheating signer $\mathcal{S}^*$, there exists random coins for $\mathcal{S}^*$ such that our challenger in Game-1 outputs DL-Abort. We will hard-code the random coins of $\mathcal{S}^*$ such that our challenger outputs DL-Abort with maximum probability. Note that we are in the case when crs_2 is binding and hence crs_1 is hiding. Next, our adversary $\mathcal{B}$ or the challenger of the blindness game on receiving this public key vk will run in unbounded time to compute the extraction key τ for crs_2. Thus, the adversary $\mathcal{B}$ we constructed is a non-uniform adversary with auxiliary input as the random coins of $\mathcal{S}^*$ (specified above) and the extraction key τ corresponding to vk.

Our adversary $\mathcal{B}$ obtains as input D (such that $D = g^d$ with $d < q$) and it wins if it outputs d. On receiving D, $\mathcal{B}$ proceeds as the challenger does in Game_1 except that it sets $C := D$ instead of choosing a fresh value for C. Also, invoking perfect zero-knowledge property of crs_1, $\mathcal{B}$ generates π as $\mathcal{S}(\text{crs}_1, \tau, \Phi)$, where $\mathcal{S}$ is the zero-knowledge simulator. At this point $\mathcal{S}^*$ must output a proof π' such that $\mathcal{V}(\text{crs}_2, \Phi, \pi') = 1$ for the challenger in Game_1 to output DL-Abort. On obtaining the proof π', $\mathcal{B}$ outputs $\mathcal{E}(\text{crs}_2, \tau, \Phi, \pi')$ as the discrete log of D. By perfect extraction under crs_2, the extracted value will be the discrete log of D.

Note that after receiving the challenge D, $\mathcal{B}$ runs in polynomial time. Thus, the probability of DL-Abort when we fix the worst case random coins of $\mathcal{S}^*$ (as described above) is bounded by $\mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{B},\mathbb{G},q}$. Hence, it holds that the probability of DL-Abort in Game_1 is bounded by $\mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{B},\mathbb{G},q}$. □

- Game_2: Game_2 is identical to Game_1 except for the following modifications. Instead of generating the final signatures honestly, the challenger simulates them. More specifically, instead of generating the signatures as $\mathcal{P}(\text{crs}_2, \Psi, (m_{\mathsf{blind}}, r, \sigma_{\mathsf{SP}}))$, in Game_2 the challenger generates signatures as $\mathcal{S}(\text{crs}_2, \tau, \Psi)$.

  Game_2 and Game_1 are perfectly indistinguishable based on the non-uniform perfect zero-knowledge property of the two-CRS NIZK proof system.

- Game_3: Now, we modify Game_2 and remove all dependencies on the input messages m_0 and m_1. That is, we let the user algorithm compute the blinded message $m_{blind,0}$ as $\mathsf{Com}_{\mathbb{G}}(0)$ instead of $\mathsf{Com}_{\mathbb{G}}(m_b)$. We proceed similarly for m_{1-b}.

  The indistinguishability between Game_3 and Game_2 follows from the non-uniform computational hiding property of the commitment scheme $\mathsf{Com}_{\mathbb{G}}$.

Since in Game_3 the entire transcript is independent of the message, $\mathsf{Adv}^{\mathsf{Unblind}}_{\mathcal{S}^*,\mathsf{BS},\mathsf{Game}_3}$ is 0. □

6 Concrete Efficiency

In this section we will compute the communication complexity for the blind signature scheme described in Section 3 and the size of the final blind signature. First we need to compute the group size p and number q which will give us the desired level of security. For this we will calculate the work factors for different adversaries as discussed below. We begin by defining the work factors.

Work Factors. These have been used in [Gal04, BR09] to calculate concrete parameters. This text has been taken verbatim from [BR09]. For any adversary running in time $\mathbf{T}(\mathcal{A})$ and gaining advantage ϵ, we define the *w*ork factor of $\mathcal{A}$ to be $\mathbf{WF}(\mathcal{A}) \leq \mathbf{T}(A)/\epsilon$. The ratio of $\mathcal{A}$'s running time to its advantage provides a measure of efficiency of the adversary. Generally speaking, to resist an adversary with work factor $\mathbf{WF}(\mathcal{A})$, a scheme should have its security parameter (bits of security) be $\kappa \geq \log \mathbf{WF}(\mathcal{A})$. Note that for a particular ϵ, this means a run time of $\mathbf{T}(A) \leq \epsilon 2^{\kappa}$.

Similar to [Gal04, BR09], in the discussion that follows we will assume that Pollard Rho's algorithm for finding discrete logs in $\mathbb{G}$ is the best known attack[8] against DLIN in group $\mathbb{G}$ of prime order p. The work factor of Pollard's algorithm ends up being

$$\mathbf{WF}(\mathcal{P}) = \frac{T(\mathcal{P})}{\epsilon_p} = \frac{0.88}{e}\sqrt{p}\frac{\log^2(p)}{10^3}$$

For security we require that the work factor of any adversary $\mathcal{A}$ against DLIN is at most the work factor of Pollard's algorithm, i.e. $\mathbf{WF}(\mathcal{A}) \leq \mathbf{WF}(\mathcal{P})$.

Parameters. In Appendix D, we calculate the values of p and q using the work factors for adversaries against the blindness game and unforgeability game. We summarize the parameters obtained in Table 3.

Table 3: Suggested parameters, where k is the number of signature queries and the adversary is allowed to run in time $t \cdot T_R$ where T_R is the time taken by the reduction.

k	t	$\log q$	$\log \lvert\mathbb{G}\rvert$
2^{20}	2^{30}	155	291
2^{20}	2^{40}	155	311
2^{30}	2^{30}	155	331
2^{30}	2^{40}	155	351

6.1 Efficiency

Verification key size. In our blind signature scheme, the verification key is $vk = (\mathrm{vk}_{\mathsf{SP}}, \mathrm{crs}_1, \mathrm{crs}_2, q = p^{\epsilon})$, where $\mathrm{vk}_{\mathsf{SP}}$ is the verification key of the structure preserving signature scheme in $\mathbb{G}$ and crs_1 and crs_2 are two CRSes for Two-CRS NIZK. Furthermore, as can be seen in Table 2, to sign k group elements, $\mathrm{vk}_{\mathsf{SP}}$ has $2k + 25$ group elements. Since in our case $k = 6$, there are 37 group elements in $\mathrm{vk}_{\mathsf{SP}}$. In GS proof system, we need 6 group elements in $\mathbb{G}$ to represent crs_1 and crs_2. Hence, the size of the verification key for our scheme is 43 group elements. Taking the number of bits to represent a group element as 291 bits, we get the key size to be 1.6KB.

Signature size. The final signature is a Groth-Sahai [GS08] proof of knowledge in $\mathbb{G}$ using crs_2 as the common reference string. Under the DLIN assumption, the proof size is three group elements for each variable and nine group elements for each pairing product equation (see Figure 2 in [GS08]) that is proved. The variables are $m_{blind}, \sigma_{\mathsf{SP}}, r$. By $\mathsf{Com}_{\mathbb{G}}$, m_{blind} has six group elements and in order to prove $m_{blind} = \mathsf{Com}_{\mathbb{G}}(m; r)$, we will have two additional variables (which capture the randomness r used in commitment) and three pairing product equations in total. Furthermore, as can be seen in Table 2, σ_{SP} has 17 group elements and nine pairing product equations in verification algorithm. Hence, the size of the final blind signature will be 183 group elements in $\mathbb{G}$. Taking the number of bits to represent a group element as 291 bits, we get the signature size to be 6.5KB.

[8] If there is a faster attack against discrete log or DLIN problem for prime order groups, it can be used to obtain the parameters for our blind signature scheme.

Communication complexity. We begin by computing the communication complexity of the user step by step as follows:

- $\mathcal{U}$ computes a commitment m_{blind} in $\mathbb{G}$ which consists of six group elements in $\mathbb{G}$.
- It computes a range proof π for an NP-statement which consists of $\log_2 q$ quadratic equations and one multiscalar multiplication equation over $\log_2 q$ variables in $\mathbb{Z}_p$ (Appendix C). In GS proof system, each quadratic equations adds six group elements, multiscalar multiplication equation adds nine group elements and each variable in $\mathbb{Z}_p$ adds three group elements to the proof ([GS08], Figure 2). Using this, π consists of $9\log_2 q + 9$ group elements of $\mathbb{G}$.

Now we compute the communication complexity of signer step by step as follows:

- It computes σ_{SP} consisting of 17 elements in $\mathbb{G}$ as explained above.
- It also computes a range proof π' for the same NP-statement as the user. As above, π' consists of $9\log_2 q + 9$ group elements of $\mathbb{G}$.

Hence, the overall communication complexity of our blind signature protocol is $18\log_2 q + 41$ elements in $\mathbb{G}$. Taking $\log_2 q$ as 155 and $\log_2 p$ as 291, the communication complexity is 100.56KB.

7 Acknowledgements

We thank Jens Groth for useful discussions relating to this work. We also thank the anonymous reviewers of EUROCRYPT 2014 for their insightful comments and an observation that helped improve the communication complexity of our signing protocol from 200 KB to 100 KB.

References

[Abe01] Masayuki Abe. A secure three-move blind signature scheme for polynomially many signatures. In *EUROCRYPT*, pages 136–151, 2001.

[ACD+12] Masayuki Abe, Melissa Chase, Bernardo David, Markulf Kohlweiss, Ryo Nishimaki, and Miyako Ohkubo. Constant-size structure-preserving signatures: Generic constructions and simple assumptions. In *ASIACRYPT*, pages 4–24, 2012.

[AHO10] Masayuki Abe, Kristiyan Haralambiev, and Miyako Ohkubo. Signing on elements in bilinear groups for modular protocol design. IACR ePrint 2010/133, 2010.

[ANN06] Michel Abdalla, Chanathip Namprempre, and Gregory Neven. On the (im)possibility of blind message authentication codes. In David Pointcheval, editor, *CT-RSA 2006*, volume 3860 of *LNCS*, pages 262–279, San Jose, CA, USA, February 13–17, 2006. Springer, Berlin, Germany.

[AO09] Masayuki Abe and Miyako Ohkubo. A framework for universally composable non-committing blind signatures. In *ASIACRYPT*, pages 435–450, 2009.

[BNPS01] Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The power of rsa inversion oracles and the security of chaum's rsa-based blind signature scheme. In *Financial Cryptography*, pages 309–328, 2001.

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-diffie-hellman-group signature scheme. In *Public Key Cryptography*, pages 31–46, 2003.

[BR94] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In *EUROCRYPT*, pages 92–111, 1994.

[BR09] Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for waters' ibe scheme. In *EUROCRYPT*, pages 407–424, 2009.

[Cha82] David Chaum. Blind signatures for untraceable payments. In *CRYPTO*, pages 199–203, 1982.

[CKW04] Jan Camenisch, Maciej Koprowski, and Bogdan Warinschi. Efficient blind signatures without random oracles. In *SCN*, pages 134–148, 2004.

[Fis06] Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In *CRYPTO*, pages 60–77, 2006.

[FS10] Marc Fischlin and Dominique Schröder. On the impossibility of three-move blind signature schemes. In *EUROCRYPT*, pages 197–215, 2010.

[Fuc09] Georg Fuchsbauer. Automorphic signatures in bilinear groups and an application to round-optimal blind signatures. IACR ePrint 2009/320, 2009.

[Gal04] David Galindo. The exact security of pairing based encryption and signature schemes. In *Based on a talk at Workshop on Provable Security, INRIA, Paris, 2004.*, 2004. `http://www.dgalindo.es/galindoEcrypt.pdf`.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[Gro06] Jens Groth. Simulation-sound nizk proofs for a practical language and constant size group signatures. In *ASIACRYPT*, pages 444–459, 2006.

[GRS+11] Sanjam Garg, Vanishree Rao, Amit Sahai, Dominique Schröder, and Dominique Unruh. Round optimal blind signatures. In *CRYPTO*, pages 630–648, 2011.

[GS08] Jens Groth and Amit Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, pages 415–432, 2008.

[HKKL07] Carmit Hazay, Jonathan Katz, Chiu-Yuen Koo, and Yehuda Lindell. Concurrently-secure blind signatures without random oracles or setup assumptions. In *TCC*, pages 323–341, 2007.

[JLO97] Ari Juels, Michael Luby, and Rafail Ostrovsky. Security of blind digital signatures (extended abstract). In *CRYPTO*, pages 150–164, 1997.

[KZ06] Aggelos Kiayias and Hong-Sheng Zhou. Concurrent blind signatures without random oracles. In *SCN*, pages 49–62, 2006.

[MSF10] Sarah Meiklejohn, Hovav Shacham, and David Mandell Freeman. Limitations on transformations from composite-order to prime-order groups: The case of round-optimal blind signatures. In *ASIACRYPT*, pages 519–538, 2010.

[Oka06] Tatsuaki Okamoto. Efficient blind and partially blind signatures without random oracles. In *TCC*, pages 80–99, 2006.

[Pas11] Rafael Pass. Limits of provable security from standard assumptions. In *STOC*, pages 109–118, 2011.

[Poi98] David Pointcheval. Strengthened security for blind signatures. In *EUROCRYPT*, pages 391–405, 1998.

[PS96a] David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In *ASIACRYPT*, pages 252–265, 1996.

[PS96b] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In *EUROCRYPT*, pages 387–398, 1996.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, 2000.

A Preliminaries

Now we will describe various tools and assumptions we use in our round optimal blind signature scheme.

A.1 Bilinear Groups.

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two groups of prime order p. Let g be a generator of $\mathbb{G}$. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map with the following properties:

- $e(g, g)$ does not evaluate to the identity element of $\mathbb{G}_T$. We say that the map e is non-degenerate.
- $\forall u \in \mathbb{G},\ \forall v \in \mathbb{G},\ \forall a, b \in \mathbb{Z}_p : e(u^a, v^b) = e(u, v)^{ab}$. We say that the map e is bilinear.

Bilinear groups $\mathbb{G}$, with efficient group operations, and for which a target group $\mathbb{G}_T$ exists with an efficient corresponding bilinear map e, are well-known. Note that this bilinear map is also symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

Definition 1. [Decisional Linear Assumption (DLIN)] Let $\mathbb{G} = \langle g \rangle$ be a bilinear group of prime order p with security parameter λ. For a non-uniform PPT $\mathcal{A}$, its advantage $\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{A},\mathbb{G}}$ is defined as:

$$\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{A},\mathbb{G}}(1^\lambda) = 2 \cdot \Pr\left[b = b' \;\middle|\; \begin{array}{l} g_1, g_2 \xleftarrow{\$} \mathbb{G};\ x, y, z \xleftarrow{\$} \mathbb{Z}_p \\ b \xleftarrow{\$} \{0,1\};\ \text{if } b = 0,\ \text{set } W = g^{x+y} \text{ else set } W = g^z \\ b' \leftarrow \mathcal{A}(g_1, g_2, g_1^x, g_2^y, W) \end{array} \right] - 1 \,.$$

We say that the DLIN assumption holds for a bilinear group $\mathbb{G}$ if for all PPT algorithms $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{A},\mathbb{G}}$ is negligible in λ. More generally, we say that T-DLIN assumption holds in group $\mathbb{G}$ if for every $T \cdot poly(\lambda)$ time algorithm $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{A},\mathbb{G}}$ is negligible in λ.

Definition 2. [Discrete Log Assumption (DLog)] Let $\mathbb{G} = \langle g \rangle$ be a bilinear group of prime order p with security parameter λ. For a non-uniform PPT $\mathcal{A}$, its advantage $\mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{A},\mathbb{G}}$ is defined as:

$$\mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{A},\mathbb{G}}(1^\lambda) = \Pr\left[c = c' \;\middle|\; c \xleftarrow{\$} \mathbb{Z}_p;\ C = g^c;\ c' \leftarrow \mathcal{A}(C)\right] .$$

We say that the discrete log assumption holds for a group $\mathbb{G}$ if for all PPT algorithms $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{A},\mathbb{G}}$ is negligible in λ.

A.2 Non-interactive Commitment Scheme.

A commitment scheme Com takes as input a message $m \in \{0,1\}^\lambda$ and randomness r and outputs a commitment cmt. To decommit, the committer sends m, r.

Any commitment scheme Com must satisfy two properties: hiding and binding. In our construction, we will use a scheme which is computational hiding and perfectly binding. Below we describe these properties.

Computationally hiding property says that that no PPT adversary can distinguish between the commitments of any two different messages. More precisely, let $\mathcal{A}$ be an adversary against the hiding property of the scheme Com. We define its hiding-advantage as

$$\mathsf{Adv}^{\mathsf{hid}}_{\mathcal{A},\mathsf{Com}}(1^\lambda) = 2 \cdot \Pr\left[b = b' \;\middle|\; \begin{array}{l}(m_0, m_1, \mathsf{st}) \leftarrow \mathcal{A}(1^\lambda); b \xleftarrow{\$} \{0,1\};\\ \mathsf{cmt} = \mathsf{Com}(m_b, r); b' \leftarrow \mathcal{A}(\mathsf{cmt}, \mathsf{st})\end{array}\right] - 1 .$$

Definition 3. Com is $\mathsf{computationally\ hiding}$ if the advantage function $\mathsf{Adv}^{\mathsf{hid}}_{\mathcal{A},\mathsf{Com}}$ is negligible in λ for all PPT adversaries $\mathcal{A}$.

A commitment scheme is perfectly binding if the underlying message is information theoretically fixed by the commitment cmt itself. More precisely,

Definition 4. Com is $\mathsf{perfectly\ binding}$ if there does not exist values $(\mathsf{cmt}, m_0, m_1, r_0, r_1)$ such that $m_0 \neq m_1$, $\mathsf{cmt} = \mathsf{Com}(m_0, r_0)$ and $\mathsf{cmt} = \mathsf{Com}(m_1, r_1)$.

A.3 Digital Signatures.

A digital signature scheme $\mathsf{SP} = (\mathsf{SPGen}, \mathsf{SPSign}, \mathsf{SPVerify})$ with security parameter λ consists of the following algorithms: SPGen outputs a pair $(\mathrm{sk}_{\mathsf{SP}}, \mathrm{vk}_{\mathsf{SP}})$ of signing and verification keys; and $\mathsf{SPSign}(\mathrm{sk}_{\mathsf{SP}}, M)$ outputs a signature σ, which is verified by $\mathsf{SPVerify}(\mathrm{vk}_{\mathsf{SP}}, M, \sigma)$.

Now we define the existential unforgeability of the signature scheme under adaptively chosen message attack (EU-CMA) [GMR88] as follows: Let $\mathcal{A}$ be an adversary against SP. We define its $\mathsf{eu\text{-}cma}$-advantage as

$$\mathsf{Adv}^{\mathsf{eu-cma}}_{\mathsf{SP},\mathcal{A}}(\lambda) = \Pr\left[\begin{array}{r}\mathsf{SPVerify}(\mathrm{vk}_{\mathsf{SP}}, m^*, \sigma^*) = 1 : (\mathrm{sk}_{\mathsf{SP}}, \mathrm{vk}_{\mathsf{SP}}) \leftarrow \mathsf{SPGen}(1^\lambda, gk);\\ (m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{SPSign}(\mathrm{sk}_{\mathsf{SP}}, \cdot)}(\mathrm{vk}_{\mathsf{SP}})\end{array}\right],$$

where gk are public parameters. The adversary $\mathcal{A}$ is not allowed to query $\mathsf{SPSign}(\mathrm{sk}_{\mathsf{SP}}, \cdot)$ on m^*.

Definition 5. SP is said to be T-$\mathsf{eu\text{-}cma\text{-}secure}$ if for all adversaries $\mathcal{A}$ running in time $T \cdot poly(\lambda)$ $\mathsf{Adv}^{\mathsf{eu-cma}}_{\mathsf{SP},\mathcal{A}}$ is negligible in λ.

B Blind Signatures and Their Security

In this section we will recall the notion of blind signatures and define their security. Parts of this section have been taken verbatim from [GRS+11]. By $(a, b) \leftarrow \langle \mathcal{X}(x), \mathcal{Y}(y) \rangle$ we denote interactive execution of algorithms $\mathcal{X}$ and $\mathcal{Y}$, where x (resp., y) is the private input of $\mathcal{X}$ (resp., $\mathcal{Y}$), and a (resp., b) is the private output of $\mathcal{X}$ (resp., $\mathcal{Y}$). We write $\mathcal{X}^{\langle \cdot, \mathcal{Y} \rangle^\infty}$ for $\mathcal{X}$ with oracle access to arbitrarily many interactions with $\mathcal{Y}$. And $\mathcal{X}^{\langle \cdot, \mathcal{Y} \rangle^1}$ for $\mathcal{X}$ with oracle access to arbitrarily a single interaction with $\mathcal{Y}$.

Definition 6. A blind signature scheme BS consists of PPT algorithms $\mathsf{Gen}, \mathsf{Vrfy}$ along with interactive PPT algorithms $\mathcal{S}$,$\mathcal{U}$ such that for any $\lambda \in \mathbb{N}$:

- $\mathsf{Gen}(1^\lambda)$ generates a key pair (sk, vk).
- The joint execution of $\mathcal{S}(sk)$ and $\mathcal{U}(vk, m)$, where $m \in \{0,1\}^\lambda$, generates an output σ for the user and no output for the signer. We write this as $(\perp, \sigma) \leftarrow \langle \mathcal{S}(sk), \mathcal{U}(vk, m) \rangle$.
- Algorithm $\mathsf{Vrfy}(vk, m, \sigma)$ outputs a bit b.

We require completeness i.e., for any $m \in \{0,1\}^\lambda$, and for $(sk, vk) \leftarrow \mathsf{Gen}(1^\lambda)$, and σ output by $\mathcal{U}$ in the joint execution of $\mathcal{S}(sk)$ and $\mathcal{U}(vk, m)$, it holds that $\mathsf{Vrfy}(vk, m, \sigma) = 1$ with overwhelming probability in $\lambda \in \mathbb{N}$.

Note that it is always possible to sign messages of arbitrary length by applying a collision-resistant hash function to the message prior to signing.

Blind signatures must satisfy two properties: unforgeability and blindness [JLO97, PS00].

For unforgeability we require that a user who runs k executions of the signature-issuing protocol should be unable to output $k+1$ valid signatures on $k+1$ distinct messages.

Definition 7. A blind signature scheme $\mathsf{BS} = (\mathsf{Gen}, \mathcal{S}, \mathcal{U}, \mathsf{Vrfy})$ is unforgeable if for any PPT algorithm $\mathcal{U}^*$ the probability that experiment $\mathsf{Unforge}^{\mathsf{BS}}_{\mathcal{U}^*}(\lambda)$ defined in Figure 2 evaluates to 1 is negligible in λ.

Experiment $\mathsf{Unforge}^{\mathsf{BS}}_{\mathcal{U}^*}(\lambda)$
$(sk, vk) \leftarrow \mathsf{Gen}(1^\lambda)$
$((m_1^*, \sigma_1^*), \ldots, (m_{k+1}^*, \sigma_{k+1}^*)) \leftarrow \mathcal{U}^{*\langle \mathcal{S}(sk), \cdot \rangle^\infty}(vk)$
Return 1 iff
$m_i^* \neq m_j^*$ for all i, j with $i \neq j$, and
$\mathsf{Vrfy}(vk, m_i^*, \sigma_i^*) = 1$ for all $i \in [k+1]$, and
at most k interactions with $\mathcal{S}(sk)$
were completed.

Experiment $\mathsf{Unblind}^{\mathsf{BS}}_{\mathcal{S}^*}(\lambda)$
$(vk, m_0, m_1, \mathsf{st}_{\mathsf{find}}) \leftarrow \mathcal{S}^*(\mathsf{find}, 1^\lambda)$
$b \leftarrow \{0,1\}$
$\mathsf{st}_{\mathsf{issue}} \leftarrow \mathcal{S}^{*\langle \cdot, \mathcal{U}(vk, m_b) \rangle^1, \langle \cdot, \mathcal{U}(vk, m_{\bar{b}}) \rangle^1}(\mathsf{issue}, \mathsf{st}_{\mathsf{find}})$
and let $\sigma_b, \sigma_{\bar{b}}$ denote the
(possibly undefined) local outputs
of $\mathcal{U}(vk, m_b)$ and $\mathcal{U}(vk, m_{\bar{b}})$ resp..
set $(\sigma_0, \sigma_1) = (\perp, \perp)$ if $\sigma_0 = \perp$ or $\sigma_1 = \perp$
$b^* \leftarrow \mathcal{S}^*(\mathsf{guess}, \sigma_0, \sigma_1, \mathsf{st}_{\mathsf{issue}})$
return 1 iff $b = b^*$.

Figure 2: Security games of blind signatures.

Blindness says that it should be infeasible for any malicious signer $\mathcal{S}^*$ to decide which of two messages m_0 and m_1 has been signed first in two executions with an honest user $\mathcal{U}$. This condition must hold, even if $\mathcal{S}^*$ is allowed to choose the public key maliciously [ANN06]. If one of these executions has returned an invalid signature, denoted by $\perp$, then the signer is not informed about the other signature either. We define the advantage of $\mathcal{S}^*$ in blindness game with respect to the experiment $\mathsf{Unblind}^{\mathsf{BS}}_{\mathcal{S}^*}(\lambda)$ as

$$\mathsf{Adv}^{\mathsf{Unblind}}_{\mathcal{S}^*,\mathsf{BS}}(\lambda) = \left|2 \cdot \Pr[\mathsf{Unblind}^{\mathsf{BS}}_{\mathcal{S}^*}(\lambda) = 1] - 1\right|$$

Definition 8. A blind signature scheme $\mathsf{BS} = (\mathsf{Gen}, \mathcal{S}, \mathcal{U}, \mathsf{Vrfy})$ satisfies blindness if the advantage function $\mathsf{Adv}^{\mathsf{Unblind}}_{\mathcal{S}^*,\mathsf{BS}}$ is negligible for any $\mathcal{S}^*$ (working in modes find, issue, and guess) running in time $poly(\lambda)$.

A blind signature scheme is secure if it is unforgeable and blind.

C Efficient Realization of Two-CRS NIZKs using Groth-Sahai Proofs

In this section, we describe how Groth-Sahai (GS) proofs [GS08] can be used to construct efficient Two-CRS NIZK proof system described in section 2.1. Groth and Sahai [GS08] gave direct constructions of NIZKs that do not require NP reductions and thus are efficient in practice. They described these for languages which can be written as a set of following equations: pairing product equations, multi-scalar multiplication equations in base group, and quadratic equations in $\mathbb{Z}_p$. For a description of these equations, see Section 2.1. The GS proof system can be instantiated under the DLIN assumption in the setting of symmetric bilinear groups. Next, we give an informal overview of how GS proofs work.

Overview of GS NIZK proofs. GS proofs are given in the common reference string (CRS) model where the CRS consists of commitment keys. GS proofs have two different modes, namely $\mathsf{GSSetup}_{\mathsf{B}}$ and $\mathsf{GSSetup}_{\mathsf{H}}$. These two modes have common reference strings (CRS) coming from two different distributions. In $\mathsf{GSSetup}_{\mathsf{B}}$, the CRS has perfectly binding keys and the proofs are perfectly sound. In particular, the CRS consists of a DLIN tuple. In $\mathsf{GSSetup}_{\mathsf{H}}$, the CRS has perfectly hiding keys and the proofs are perfectly zero-knowledge. In particular, the CRS consists of non-DLIN tuple.

When proving a statement, described as a set of equations, the GS proof system works as follows: The prover first commits to the witness components (using CRS as the commitment key) and then for each equation produces proof elements showing that the corresponding committed values satisfy the equation.

The GS proof system satisfies the following properties:

Completeness. GS proof system is perfectly complete under both $\mathsf{GSSetup}_{\mathsf{B}}$ and $\mathsf{GSSetup}_{\mathsf{H}}$.

Soundness and Witness Extraction. As stated above, in $\mathsf{GSSetup}_{\mathsf{B}}$, GS proofs are perfectly sound. Now we will describe how they achieve witness extraction in $\mathsf{GSSetup}_{\mathsf{B}}$ mode. This mode, along with CRS, also has an extraction key *GSek*. *GSek* consists of the discrete log of the elements in the CRS. Furthermore, GS show a PPT algorithm GSExtr, which given a commitment of a group element, successfully extracts the underlying value. But there is a problem.

Depending on the kind of equations used, the witness can consist of elements in $\mathbb{G}$ as well as elements in $\mathbb{Z}_p$. While GS proofs under binding keys are proofs of knowledge of group elements (using GSExtr), they are not proofs of knowledge of elements in $\mathbb{Z}_p$. In GS proof system, the best one can extract from a commitment to $y \in \mathbb{Z}_p$ is the value g^y. Since we assume that the Discrete Log problem is hard in $\mathbb{G}$, there is no way to extract y given this value.

We will solve this problem by using properties of our equations. In the equations we will use in our scheme, y will either be 0 or 1. Hence, the GS proof will also be a proof of knowledge of elements in $\mathbb{Z}_p$ in our case.

Zero-Knowledge. As stated above, the proofs under $\mathsf{GSSetup_H}$ are perfect zero-knowledge. In particular, given discrete logs of elements in the CRS, the proofs can be simulated perfectly and efficiently.

CRS indistinguishability. It is easy to observe that the CRSes under $\mathsf{GSSetup_B}$ and $\mathsf{GSSetup_H}$ are indistinguishable under the DLIN assumption in $\mathbb{G}$.

Remark 3. *A GS proof consists of a constant number of group elements for each variable and each equation. For more details see Figure 2 in [GS08].*

Now we give a description of efficient realization of Two-CRS NIZKs (defined in section 2.1) using the algorithms $\mathsf{GSSetup_B}$, *GSek* and GSExtr.

Efficient Realization of Two-CRS NIZKs. Using GS proofs, we get an efficient realization of Two-CRS NIZKs as follows: Define $K_B := \mathsf{GSSetup_B}$ which outputs a DLIN tuple, $\mathsf{crs} = (g, g^a, g^b, g^{ax}, g^{by}, g^{x+y})$. Define $\tau := GSek = (a, b, x, y)$. Define $\mathcal{E} := \mathsf{GSExtr}$. Define Shift and Shift^{-1} as deterministic algorithms which change the last element of CRS output by $\mathsf{GSSetup_B}$ as $\mathsf{crs}' = (g, g^a, g^b, g^{ax}, g^{by}, g^{x+y+1}) \leftarrow \mathsf{Shift}(\mathsf{crs})$ and $\mathsf{crs}'' = (g, g^a, g^b, g^{ax}, g^{by}, g^{x+y-1}) \leftarrow \mathsf{Shift}^{-1}(\mathsf{crs})$. The prover algorithm $\mathcal{P}$ and verification algorithm $\mathcal{V}$ are same as the corrresponding algorithms in GS proofs.

Next, we briefly state why the above system satisfies the properties described for Two-CRS NIZK proof systems in section 2.1.

CRS Indistinguishability. It holds under the DLIN assumption in $\mathbb{G}$.

Perfect Completeness. It holds under the perfect completeness of GS proofs.

Perfect Knowledge Extraction. It holds by perfect soundness and perfect witness extractability of GS proofs under $\mathsf{GSSetup_B}$.

Perfect Zero-Knowledge. Since, the CRS output by both Shift and Shift^{-1} are non-DLIN tuples, they will provide the guarantees given by a key output by $\mathsf{GSSetup_H}$. Also, note that the discrete logs output by $\mathsf{GSSetup_B}$ are a sufficient trapdoor information required to simulate the proofs under crs' and crs''.

C.1 The Range Equation in our scheme fits the GS framework for NIZKs.

In this section, we describe how we can write a range equation as a set of equations for which efficient GS proofs can be given. A range equation over variable $c \in \mathbb{Z}_q$ is of the form

$$\exists c \ : \ g^c = C \ \bigwedge \ c < q,$$

where $q < p$. Let number of bits needed to represent q be n. Let $x_0, x_1, \ldots, x_{n-1}$ be the bit representation of c and $\mathcal{A}_i = g^{2^i}$ for all $i \in \{0, 1, \ldots, n-1\}$. Then the above equation can be written as

$$\prod_{i=0}^{n-1} \mathcal{A}_i^{x_i} = C \bigwedge \forall i \ x_i \in \{0, 1\}$$

over variables $x_i \in \mathbb{Z}_p$. This can be expressed as

$$\prod_{i=0}^{n-1} \mathcal{A}_i^{x_i} = C \bigwedge \forall i \; x_i \cdot (1 - x_i) = 0$$

over variables $x_i \in \mathbb{Z}_p$. This is a combination of multiscalar multiplication and quadratic equations and fits well with the GS framework (described above). Note that all possible values for each of x_i are 0 and 1 and hence they can be extracted by GSExtr.

D Calculating Parameters

In the following subsections, we will calculate our parameters using work factors for adversaries against the blindness game and unforgeability game.

D.1 Equations from Blindness

For any PPT malicious signer $\mathcal{S}^*$ for the blindness game against the blind signature scheme given in Section 3, which successfully completes the blindness game, the following holds

$$\mathsf{Adv}^{\mathsf{Unblind}}_{\mathcal{S}^*,\mathsf{BS}}(\lambda) < 4 \cdot \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{C},\mathbb{G}} + \mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{B},\mathbb{G},q} \tag{3}$$

where $\mathcal{C}$ is an adversary against the DLIN assumption in $\mathbb{G}$ such that $\mathbf{T}(\mathcal{C}) = \mathbf{T}(\mathcal{S}^*) + poly(\lambda)$ and $\mathcal{B}$ is an adversary against the discrete log problem in $\mathbb{G}$ when exponents are chosen uniformly randomly in $\mathbb{Z}_q$ such that $\mathbf{T}(\mathcal{B}) = \mathbf{T}(\mathcal{S}^*) + poly(\lambda)$.

As we will see, $q << p$ and hence, under best known attacks, for any $\mathcal{C}$, $\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{C},\mathbb{G}} << \mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{B},\mathbb{G},q}$. Hence, taking $\mathsf{Adv}^{\mathsf{Unblind}}_{\mathcal{S}^*,\mathsf{BS}}(\lambda) = \eta$ and calculating the work factor of $\mathcal{B}$ we get

$$\mathbf{WF}(\mathcal{B}) = \frac{\mathbf{T}(\mathcal{B})}{\mathsf{Adv}^{\mathsf{dlog}}_{\mathcal{B},\mathbb{G},q}} \approx \frac{\mathbf{T}(\mathcal{S}^*) + poly(\lambda)}{\eta}$$

For security we want $\mathbf{WF}(\mathcal{B}) \leq \frac{0.88}{e} \sqrt{q} \frac{\log^2 q}{10^3}$, i.e.

$$\frac{\mathbf{T}(\mathcal{S}^*) + poly(\lambda)}{\eta} \leq \frac{0.88}{e} \sqrt{q} \frac{\log^2 q}{10^3}$$

Note that if allow our adversary to run in at most 2^{80} steps, i.e. $\mathbf{T}(\mathcal{S}^*) + poly(\lambda) \leq 2^{80}$, and let $\eta = 1$, we will get the following:

$$2^{80} \leq \frac{0.88}{e} \sqrt{q} \frac{\log^2 q}{10^3}$$

Solving this equation, we get $\log_2 q = 155$. Using this value of q we will calculate the value of p from the security against unforgeability.

D.2 Equations from Unforgeability proof

For any PPT malicious user $\mathcal{U}^*$ for the unforgeability game against the blind signature scheme given in Section 3 the following holds:

$$\mathsf{Adv}^{\mathsf{Unforge}}_{\mathcal{U}^*,\mathsf{BS}}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{dlin}}_{\mathcal{B},\mathbb{G}}(\lambda) + \mathsf{Adv}^{\mathsf{Unforge}}_{\widehat{\mathcal{U}^*},\mathsf{SPSig}}(\lambda), \tag{4}$$

where $\mathcal{B}$ is an adversary against the DLIN assumption in $\mathbb{G}$ such that $\mathbf{T}(\mathcal{B}) = k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)$ and $\widehat{\mathcal{U}^*}$ is the adversary against the unforgeability of the underlying structure preserving signature scheme SPSig such that $\mathbf{T}(\widehat{\mathcal{U}^*}) = k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)$. Also, $\mathcal{U}^*$ and $\widehat{\mathcal{U}^*}$ make at most k signing queries.

To calculate the parameters for our blind signature scheme, we use the constant size structure preserving signatures in [ACD+12]. Their scheme SIG1 is unforgeable under the DLIN assumption in group $\mathbb{G}$. In particular, they show the following in Theorem 7 [ACD+12]:

$$\mathsf{Adv}^{\mathsf{Unforge}}_{\widehat{\mathcal{U}^*},\mathsf{SPSig}}(\lambda) \leq (k+3) \cdot \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{C},\mathbb{G}} + 1/p$$

Also, $\mathbf{T}(\mathcal{C}) = \mathbf{T}(\widehat{\mathcal{U}^*}) + poly(\lambda)$, where $poly(\lambda)$ is the time taken in answering k signature queries of the user in the unforgeability game.

Substituting this in Equation 5 we get

$$\mathsf{Adv}^{\mathsf{Unforge}}_{\mathcal{U}^*,\mathsf{BS}}(\lambda) \leq (k+5) \cdot \mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{A},\mathbb{G}} + \frac{1}{p}, \tag{5}$$

where $\mathcal{A}$ is an adversary against the DLIN assumption in $\mathbb{G}$ such that $\mathbf{T}(\mathcal{A}) = k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)$.

Taking $\mathsf{Adv}^{\mathsf{Unforge}}_{\mathcal{U}^*,\mathsf{BS}}(\lambda) = \epsilon$, we calculate the work factor of $\mathcal{A}$ as follows:

$$\mathbf{WF}(\mathcal{A}) = \frac{\mathbf{T}(\mathcal{A})}{\mathsf{Adv}^{\mathsf{DLIN}}_{\mathcal{A},\mathbb{G}}} \approx \frac{k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)}{\frac{\epsilon - \frac{1}{p}}{k+5}}$$

For security we want $\mathbf{WF}(\mathcal{A}) \leq \frac{0.88}{e}\ \sqrt{p}\ \frac{\log^2 p}{10^3}$, i.e.

$$\frac{k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q} + \mathbf{T}(\mathcal{U}^*) + poly(\lambda)}{\frac{\epsilon - \frac{1}{p}}{k+5}} \leq \frac{0.88}{e}\ \sqrt{p}\ \frac{\log^2 p}{10^3}.$$

Using Pollard Rho's algorithm for breaking discrete logs $\mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q}$ will be $\frac{0.88}{e}\ \sqrt{q}\ \frac{\log^2 q}{10^3}$. We will assume that if our reduction runs in time T_R, we allow the adversary to run in time $t \cdot T_R$. Hence, by taking $\mathbf{T}(\mathcal{A})$ to be $t \cdot k \cdot \mathbf{T}^{\mathsf{dlog}}_{\mathbb{G},q}$ and $\epsilon = 1$, we get

$$t \cdot \left(1 - \frac{1}{p}\right) \cdot k \cdot (k+5) \frac{0.88}{e}\ \sqrt{q}\ \frac{\log^2 q}{10^3} \leq \frac{0.88}{e}\ \sqrt{p}\ \frac{\log^2 p}{10^3}.$$

When we solve the above equation for $t = 2^{30}$, $k = 2^{20}$ and $\log_2 q = 155$, we get $\log_2 p = 291$.
In table 3 we show the values of $\log_2 p$ with different values of k and t.