An extended abstract of this work appears in *EUROCRYPT'20*. This is the full version.

Blind Schnorr Signatures and Signed ElGamal Encryption in the Algebraic Group Model

Georg Fuchsbauer[1], Antoine Plouviez[2], and Yannick Seurin[3]

[1] TU Wien, Austria
[2] Inria, ENS, CNRS, PSL, Paris, France
[3] ANSSI, Paris, France
first.last@{tuwien.ac.at,ens.fr,m4x.org}

January 16, 2021

Abstract. The Schnorr blind signing protocol allows blind issuing of Schnorr signatures, one of the most widely used signatures. Despite its practical relevance, its security analysis is unsatisfactory. The only known security proof is rather informal and in the combination of the generic group model (GGM) and the random oracle model (ROM) assuming that the "ROS problem" is hard. The situation is similar for (Schnorr-)signed ElGamal encryption, a simple CCA2-secure variant of ElGamal.

We analyze the security of these schemes in the algebraic group model (AGM), an idealized model closer to the standard model than the GGM. We first prove tight security of Schnorr signatures from the discrete logarithm assumption (DL) in the AGM+ROM. We then give a rigorous proof for blind Schnorr signatures in the AGM+ROM assuming hardness of the one-more discrete logarithm problem and ROS.

As ROS can be solved in sub-exponential time using Wagner's algorithm, we propose a simple modification of the signing protocol, which leaves the signatures unchanged. It is therefore compatible with systems that already use Schnorr signatures, such as blockchain protocols. We show that the security of our modified scheme relies on the hardness of a problem related to ROS that appears much harder.

Finally, we give tight reductions, again in the AGM+ROM, of the CCA2 security of signed ElGamal encryption to DDH and signed hashed ElGamal key encapsulation to DL.

Keywords: Schnorr signatures, blind signatures, algebraic group model, ElGamal encryption, blockchain protocols

1 Introduction

<u>Schnorr Signatures.</u> The Schnorr signature scheme [Sch90, Sch91] is one of the oldest and simplest signature schemes based on prime-order groups. Its adoption was hindered for years by a patent which expired in February 2008, but it is by now widely deployed: EdDSA [BDL+12], a specific instantiation based on twisted Edward curves, is used for example in OpenSSL, OpenSSH, GnuPG and more. Schnorr signatures are also expected to be implemented in Bitcoin [Wui18], enabling multi-signatures supporting public key aggregation, which will result in considerable scalability and privacy enhancements [BDN18, MPSW19].

The security of the Schnorr signature scheme has been analyzed in the random oracle model (ROM) [BR93], an idealized model which replaces cryptographic hash functions by truly random functions. Pointcheval and Stern [PS96b, PS00] proved Schnorr signatures secure in the ROM under the discrete logarithm assumption (DL). The proof, based on the so-called Forking Lemma, proceeds by rewinding the adversary, which results in a loose reduction (the success probability of the DL solver is a factor q_{h} smaller than that of the adversary, where q_{h} is the number of the adversary's random oracle queries). Using the "meta reduction" technique, a series of works showed that this security loss is unavoidable when the used reductions are either algebraic [PV05, GBL08, Seu12] or generic [FJS19]. Although the security of Schnorr signatures is well understood (in the ROM), the same cannot be said for two related schemes, namely blind Schnorr signatures and Schnorr-signed ElGamal encryption.

Blind Schnorr Signatures. A blind signature scheme allows a user to obtain a signature from a signer on a message m in such a way that (i) the signer is unable to recognize the signature later (*blindness*, which in particular implies that m remains hidden from the signer) and (ii) the user can compute one single signature per interaction with the signer (*one-more unforgeability*). Blind signature schemes were introduced by Chaum [Cha82] and are a fundamental building block for applications that guarantee user anonymity, e.g. e-cash [Cha82, CFN90, OO92, CHL05, FPV09], e-voting [FOO93], direct anonymous attestation [BCC04], and anonymous credentials [Bra94, CL01, BCC+09, BL13a, Fuc11].

Constructions of blind signature schemes range from very practical schemes based on specific assumptions and usually provably secure in the random oracle model [PS96a, PS00, Abe01, Bol03, FHS15, HKL19] to theoretical schemes provably secure in the standard model from generic assumptions [GRS+11, BFPV13, GG14].

The blind Schnorr signature scheme derives quite naturally from the Schnorr signature scheme [CP93]. It is one of the most efficient blind signature schemes and increasingly used in practice. Anticipating the implementation of Schnorr signatures in Bitcoin, developers are already actively exploring the use of blind Schnorr signatures for *blind* coin swaps, trustless tumbler services, and more [Nic19].

While the hardness of computing discrete logarithms in the underlying group $\mathbb{G}$ is obviously necessary for the scheme to be unforgeable, Schnorr [Sch01] showed that another problem that he named ROS, which only depends on the order p of the group $\mathbb{G}$, must also be hard for the scheme to be secure. Informally, the ROS_ℓ problem, parameterized by an integer ℓ, asks to find $\ell+1$ vectors $\vec{\rho}_i = (\rho_{i,j})_{j\in[\ell]}$ such that the system of $\ell+1$ linear equations in unknowns $c_1, \ldots, c_\ell$ over $\mathbb{Z}_p$

$$\textstyle\sum_{j=1}^{\ell} \rho_{i,j} c_j = \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i) \ , \quad i \in [\ell+1]$$

has a solution, where $\mathsf{H}_{\mathrm{ros}} \colon (\mathbb{Z}_p)^\ell \to \mathbb{Z}_p$ is a random oracle. Schnorr showed that an attacker able to solve the ROS_ℓ problem can produce $\ell+1$ valid signatures while interacting (concurrently) only ℓ times with the signer. Slightly later, Wagner [Wag02] showed that the ROS_ℓ problem can be reduced to the $(\ell+1)$-sum problem, which can solved with time and space complexity $O\big((\ell+1)2^{\lambda/(1+\lfloor \lg(\ell+1)\rfloor)}\big)$, where λ is the bit size of p. For example, for $\lambda = 256$, this attack yields 16 valid signatures after $\ell = 15$ interactions with the signer in time and space close to 2^{55}. For $\ell+1 = 2^{\sqrt{\lambda}}$, the attack has sub-exponential time and space complexity $O(2^{2\sqrt{\lambda}})$, although the number of signing sessions becomes arguably impractical. Asymptotically, this attack can be thwarted by increasing the group order, but this would make the scheme quite inefficient.

From a provable-security point of view, a number of results [FS10, Pas11, BL13b] indicate that blind Schnorr signatures cannot be proven one-more unforgeable under standard assumptions, not even in the ROM. The only positive result by Schnorr and Jakobsson [SJ99] and Schnorr [Sch01] states that blind Schnorr signatures are secure in the combination of the generic group model and the ROM assuming hardness of the ROS problem.

The recent analysis by Hauck, Kiltz, and Loss [HKL19] of blind signatures derived from linear identification schemes does not apply to Schnorr. The reason is that the underlying linear function family $F\colon \mathbb{Z}_p \to \mathbb{G}, x \mapsto xG$ lacks the property of having a pseudo torsion-free element from the kernel (see [HKL19, Definition 3.1]). In particular, F is one-to-one, whereas Hauck et al. reduce blind signature unforgeability to collision resistance of the underlying function family.

THE ALGEBRAIC GROUP MODEL. The *generic group model* (GGM) [Nec94, Sho97] is an idealized model for the security analysis of cryptosystems that are defined over cyclic groups. Instead of receiving concrete group elements, the adversary only gets "handles" for them and has access to an oracle that performs the group operation (denoted additively) on handles. This implies that if the adversary is given a list of (handles of) group elements $(X_1, \ldots, X_n)$ and later returns (a handle of) a group element Z, then by inspecting its oracle calls one can derive a "representation" $\vec{z} = (z_1, \ldots, z_n)$ such that $Z = \sum_{i=1}^{n} z_i X_i$.

Fuchsbauer, Kiltz, and Loss [FKL18] introduced the *algebraic group model* (AGM), a model that lies between the standard model and the GGM. On the one hand, the adversary has direct access to group elements; on the other hand, it is assumed to only produce new group elements by applying the group operation to received group elements. In particular, with every group element Z that it outputs, the adversary also gives a representation $\vec{z}$ of Z in terms of the group elements it has received so far. While the GGM allows for proving information-theoretic guarantees, security results in the AGM are proved via reductions to computationally hard problems, like in the standard model.

Our starting point is the observation that in the combination[4] AGM+ROM Schnorr signatures have a *tight* security proof under the DL assumption. This is because we can give a reduction which works *straight-line*, i.e., unlike the forking-lemma-based reduction [PS96b, PS00], which must rewind the adversary, it runs the adversary only once.[5] Motivated by this, we then turn to blind Schnorr signatures, whose security in the ROM remains elusive, and study their security in the AGM+ROM.

OUR RESULTS ON BLIND SCHNORR SIGNATURES. Our first contribution is a rigorous analysis of the security of blind Schnorr signatures in the AGM+ROM. Concretely, we show that any algebraic adversary successfully producing $\ell + 1$ forgeries after at most ℓ interactions with the signer must either solve the one-more discrete logarithm (OMDL) problem or the ROS_ℓ problem. Although this is not overly surprising in view of the previous results in the GGM [SJ99, Sch01], this gives a more satisfying characterization of the security of this protocol. Moreover, all previous proofs [SJ99, Sch01] were rather informal; in particular, the reduction solving ROS was not explicitly described. In contrast, we provide precise definitions (in particular for the ROS problem, whose exact specification is central for a security proof) and work out the details of the reductions to both OMDL and ROS, which yields the first rigorous proof.

Nevertheless, the serious threat by Wagner's attack for standard-size group orders remains. In order to remedy this situation, we propose a simple modification of the scheme which

[4] This combination of idealized models was already considered when the AGM was first defined [FKL18].

[5] A similar result [ABM15] shows that Schnorr signatures, when viewed as non-interactive proofs of knowledge of the discrete logarithm of the public key, are simulation-sound extractable, via a straight-line extractor. Our proof is much simpler and gives a concrete security statement.

only alters the signing protocol (key generation and signature verification remain the same) and thwarts (in a well-defined way) any attempt at breaking the scheme by solving the ROS problem. The idea is that the signer and the user engage in two parallel signing sessions, of which the signer only finishes one (chosen at random) in the last round. Running this tweak takes thus around twice the time of the original protocol. We show that an algebraic adversary successfully mounting an $(\ell + 1)$-forgery attack against this scheme must either solve the OMDL problem or a *modified* ROS problem, which appears much harder than the standard ROS problem for large values of ℓ, which is precisely when the standard ROS problem becomes tractable.

Our results are especially relevant to applications that impose the signature scheme and for which one then has to design a blind signing protocol. This is the case for blockchain-based systems where modifying the signature scheme used for authorizing transactions is a heavy process that can take years (if possible at all). We see a major motivation for studying blind Schnorr signatures in its real-world relevance for protocols that use Schnorr signatures or will in the near future, such as Bitcoin. For these applications, Wagner's attack represents a significant risk, which can be thwarted by using our modified signing protocol.

Update on the Hardness of the ROS Problem. Shortly after the publication of this work, Benhamouda et al. [BLOR20] discovered a polynomial-time attack against the ROS problem. More precisely, the attack efficiently solves ROS_ℓ when $\ell > \log_2(p)$. This directly implies a polynomial-time attack against the blind Schnorr signature scheme where the attacker produces $\ell + 1$ signatures after $\ell \geq \lceil \log_2 p \rceil$ concurrent interactions with the signer. The attack does not apply to the modified ROS problem, hence our new blind signature scheme is unaffected.

Chosen-Ciphertext-Secure ElGamal Encryption. Recall the ElGamal public-key encryption (PKE) scheme [ElG85]: given a cyclic group $(\mathbb{G}, +)$ of prime order p and a generator G, a secret/public key pair is of the form $(y, yG) \in \mathbb{Z}_p \times \mathbb{G}$. To encrypt a message $M \in \mathbb{G}$, one draws $x \leftarrow_\$ \mathbb{Z}_p$, computes $X := xG$, and outputs ciphertext $(X, M + xY)$. This scheme is IND-CPA-secure under the decisional Diffie-Hellman (DDH) assumption [TY98], that is, no adversary can distinguish encryptions of two messages. Since the scheme is homomorphic, it cannot achieve IND-CCA2 security, where the adversary can query decryptions of any ciphertext (except of the one it must distinguish). However, ElGamal has been shown to be IND-CCA1-secure (where no decryption queries can be made after receiving the challenge ciphertext) in the AGM under a "q-type" variant of DDH [FKL18].[6]

A natural way to make ElGamal encryption IND-CCA2-secure is to add a proof of knowledge of the randomness x used to encrypt. Intuitively, this makes the scheme *plaintext-aware* [BR95], which informally means that for any adversary producing a valid ciphertext, there exists an extractor that can retrieve the corresponding plaintext. The reduction of IND-CCA2 security can then use the extractor to answer the adversary's decryption queries. (For ElGamal, the extractor would extract the randomness x used to produce $(X = xG, C = M + xY)$ from the proof of knowledge and return the plaintext $M = C - xY$.) Since the randomness x together with the first part X of the ciphertext form a Schnorr key pair, a natural idea is to use a

[6] [FKL18] showed IND-CCA1 security for the corresponding key-encapsulation mechanism, which returns a key $K = xY$ and an encapsulation of the key $X = xG$. The ElGamal PKE scheme is obtained by combining it with the one-time-secure data-encapsulation mechanism $M \mapsto M + K$. Generic results on hybrid schemes [HHK10] imply IND-CCA1 security of the PKE.

Schnorr signature [Jak98, TY98], resulting in what is usually called (Schnorr-)signed ElGamal encryption. This scheme has a number of attractive properties: ciphertext validity can be checked without knowledge of the decryption key, and one can work homomorphically with the "core" ElGamal ciphertext (a property sometimes called "submission-security" [Wik08]), which is very useful in e-voting.

Since Schnorr signatures are extractable in the ROM, one would expect that signed ElGamal can be proved IND-CCA2 under, say, the DDH assumption (in the ROM). However, turning this intuition into a formal proof has remained elusive. The main obstacle is that Schnorr signatures are not *straight-line* extractable in the ROM [BNW17]. As explained by Shoup and Gennaro [SG02], the adversary could order its random-oracle and decryption queries in a way that makes the reduction take exponential time to simulate the decryption oracle.

Schnorr and Jakobsson [SJ00] showed IND-CCA2 security in the GGM+ROM, while Tsiounis and Yung [TY98] gave a proof under a non-standard "knowledge assumption", which amounts to assuming that Schnorr signatures are straight-line extractable. On the other hand, impossibility results tend to indicate that IND-CCA2 security cannot be proved in the ROM [ST13, BFW16].

Our Results on Signed ElGamal Encryption. Our second line of contributions is two-fold. First, we prove (via a tight reduction) that in the AGM+ROM, Schnorr-signed ElGamal encryption is IND-CCA2-secure under the DDH assumption. While intuitively this should follow naturally from the straight-line extractability of Schnorr proofs of knowledge for algebraic adversaries, the formal proof is technically quite delicate: since messages are group elements, the "basis" of group-element inputs in terms of which the adversary provides representations contains not only the three group elements of the challenge ciphertext but also grows as the adversary queries the decryption oracle.[7]

We finally consider the "hashed" variant of ElGamal (also known as DHIES) [ABR01], in which a key is derived as $k = \mathsf{H}(xY)$. In the ROM, the corresponding key-encapsulation mechanism (KEM) is IND-CCA2-secure under the strong Diffie-Hellman assumption (which states that CDH is hard even when given a DDH oracle) [CS03]. We propose to combine the two approaches: concretely, we consider the hashed ElGamal KEM together with a Schnorr signature proving knowledge of the randomness used for encapsulating the key and give a *tight* reduction of the IND-CCA2 security of this scheme to the DL problem in the AGM+ROM.

2 Preliminaries

General Notation. We denote the (closed) integer interval from a to b by $[a, b]$. We use $[b]$ as shorthand for $[1, b]$. A function $\mu \colon \mathbb{N} \to [0, 1]$ is *negligible* (denoted $\mu = \mathsf{negl}$) if for all $c \in \mathbb{N}$ there exists $\lambda_c \in \mathbb{N}$ such that $\mu(\lambda) \leq \lambda^{-c}$ for all $\lambda \geq \lambda_c$. A function ν is *overwhelming* if $1 - \nu = \mathsf{negl}$. We let lg denote the logarithm in base 2 and write $x \equiv_p y$ for $x \equiv y \pmod{p}$.

Given a non-empty finite set S, we let $x \leftarrow_{\$} S$ denote the operation of sampling an element x from S uniformly at random. All algorithms are probabilistic unless stated otherwise. By $y \leftarrow \mathcal{A}(x_1, \ldots, x_n)$ we denote the operation of running algorithm $\mathcal{A}$ on inputs $(x_1, \ldots, x_n)$ and uniformly random coins and letting y denote the output. If $\mathcal{A}$ has oracle access to some

[7] Bernhard et al. [BFW16] hastily concluded that, in the AGM+ROM, IND-CCA2-security of signed ElGamal followed from straight-line extractability of Schnorr signatures showed in [ABM15]. Our detailed proof shows that this was a bit optimistic.

algorithm $\textsc{Oracle}$, we write $y \leftarrow \mathcal{A}^{\textsc{Oracle}}(x_1, \ldots, x_n)$. A list $\vec{z} = (z_1, \ldots, z_n)$, also denoted $(z_i)_{i\in[n]}$, is a finite sequence. The length of a list $\vec{z}$ is denoted $|\vec{z}|$. The empty list is denoted $()$.

A *security game* $\mathrm{GAME}_{par}(\lambda)$ indexed by a set of parameters *par* consists of a main procedure and a collection of oracle procedures. The main procedure, on input the security parameter λ, initializes variables and generates input on which an adversary $\mathcal{A}$ is run. The adversary interacts with the game by calling oracles provided by the game and returns some output, based on which the game computes its own output b (usually a single bit), which we write $b \leftarrow \mathrm{GAME}^{\mathcal{A}}_{par}(\lambda)$. When the game outputs the truth value of a predicate, we identify **false** with 0 and **true** with 1. Games are either computational or decisional. The *advantage* of $\mathcal{A}$ in $\mathrm{GAME}_{par}(\lambda)$ is defined as $\mathsf{Adv}^{\mathrm{game}}_{par,\mathcal{A}}(\lambda) := \Pr[1 \leftarrow \mathrm{GAME}^{\mathcal{A}}_{par}(\lambda)]$ if the game is computational and as $\mathsf{Adv}^{\mathrm{game}}_{par,\mathcal{A}}(\lambda) := 2 \cdot \Pr[1 \leftarrow \mathrm{GAME}^{\mathcal{A}}_{par}(\lambda)] - 1$ if it is decisional, where the probability is taken over the random coins of the game and the adversary. We say that GAME_{par} is *hard* if for any probabilistic polynomial-time (p.p.t.) adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathrm{game}}_{par,\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$. All games considered in this paper are computational unless stated otherwise (we only consider decisional games in Section 6 and Appendix B and C.)

<u>Algebraic Algorithms.</u> A *group description* is a tuple $\mathit{\Gamma} = (p, \mathbb{G}, G)$ where p is an odd prime, $\mathbb{G}$ is an abelian group of order p, and G is a generator of $\mathbb{G}$. We will use additive notation for the group law throughout this paper, and denote group elements (including the generator G) with italic uppercase letters. We assume the existence of a p.p.t. algorithm GrGen which, on input the security parameter 1^λ in unary, outputs a group description $\mathit{\Gamma} = (p, \mathbb{G}, G)$ where p is of bit-length λ. Given an element $X \in \mathbb{G}$, we let $\log_G(X)$ denote the discrete logarithm of X in base G, i.e., the unique $x \in \mathbb{Z}_p$ such that $X = xG$. We write $\log X$ when G is clear from context.

An *algebraic security game* (w.r.t. GrGen) is a game $\mathrm{GAME}_{\mathsf{GrGen}}$ that (among other things) runs $\mathit{\Gamma} \leftarrow \mathsf{GrGen}(1^\lambda)$ and runs the adversary on input $\mathit{\Gamma} = (p, \mathbb{G}, G)$. An algorithm $\mathcal{A}_{\mathrm{alg}}$ executed in an algebraic game $\mathrm{GAME}_{\mathsf{GrGen}}$ is *algebraic* if for all group elements Z that it outputs, it also provides a representation of Z relative to all previously received group elements: if $\mathcal{A}_{\mathrm{alg}}$ has so far received $\vec{X} = (X_0, \ldots, X_n) \in \mathbb{G}^{n+1}$ (where by convention we let $X_0 = G$), then $\mathcal{A}_{\mathrm{alg}}$ must output Z together with $\vec{z} = (z_0, \ldots, z_n) \in (\mathbb{Z}_p)^{n+1}$ such that $Z = \sum_{i=0}^{n} z_i X_i$. We let $Z_{[\vec{z}]}$ denote such an augmented output. When writing $\vec{z}$ explicitly, we simply write $Z_{[z_0,\ldots,z_n]}$ (rather than $Z_{[(z_0,\ldots,z_n)]}$) to lighten the notation.

<u>Algebraic Algorithms in the Random Oracle Model.</u> The original paper [FKL18] considered the algebraic group model augmented by a random oracle and proved tight security of BLS signatures [BLS04] in the AGM+ROM model. The random oracle in that work is of type $\mathsf{H}\colon \{0,1\}^* \to \mathbb{G}$, and as the outputs are group elements, the adversary's group element representations could depend on them.

In this work we analyze Schnorr-type cryptosystems, for which the RO is typically of type $\mathsf{H}\colon \mathbb{G} \times \{0,1\}^* \to \mathbb{Z}_p$. Thus, an algebraic adversary querying H on some input (Z, m) must also provide a representation $\vec{z}$ for the group-element input Z. In a game that implements the random oracle by lazy sampling, to ease readability, we will define an auxiliary oracle $\widetilde{\mathsf{H}}$, which is used by the game itself (and thus does not take representations of group elements as input) and implements the same function as H.

<u>The One-More Discrete Logarithm Problem.</u> We recall the discrete logarithm (DL) problem in Figure 1. The one-more discrete logarithm (OMDL) problem, also defined in

Game $\mathrm{DL}^{\mathcal{A}}_{\mathsf{GrGen}}(\lambda)$	Game $\mathrm{OMDL}^{\mathcal{A}}_{\mathsf{GrGen}}(\lambda)$	Oracle $\textsc{Chal}()$	Oracle $\textsc{DLog}(X)$
$(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda)$	$(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda)$	$x \leftarrow\!\$\, \mathbb{Z}_p\,;\ X := xG$	$q := q + 1$
$x \leftarrow\!\$\, \mathbb{Z}_p\,;\ X := xG$	$\vec{x} := (\,)\,;\ q := 0$	$\vec{x} := \vec{x} \parallel (x)$	$x := \log_G(X)$
$y \leftarrow \mathcal{A}(p, \mathbb{G}, G, X)$	$\vec{y} \leftarrow \mathcal{A}^{\textsc{Chal},\textsc{DLog}}(p, \mathbb{G}, G)$	**return** X	**return** x
return $(y = x)$	**return** $\big(\vec{y} = \vec{x}\ \wedge\ q < \lvert\vec{x}\rvert\big)$		

Fig. 1. The DL and OMDL problems.

Figure 1, is an extension of the DL problem and consists in finding the discrete logarithm of q group elements by making strictly less than q calls to an oracle solving the discrete logarithm problem. It was introduced in [BNPS03] and used for example to prove the security of the Schnorr identification protocol against active and concurrent attacks [BP02].

3 Schnorr Signatures

3.1 Definitions

A signature scheme SIG consists of the following algorithms:

- $par \leftarrow \mathsf{SIG.Setup}(1^\lambda)$: the setup algorithm takes as input the security parameter λ in unary and outputs public parameters par;
- $(sk, pk) \leftarrow \mathsf{SIG.KeyGen}(par)$: the key generation algorithm takes parameters par and outputs a secret key sk and a public key pk;
- $\sigma \leftarrow \mathsf{SIG.Sign}(sk, m)$: the signing algorithm takes as input a secret key sk and a message $m \in \{0,1\}^*$ and outputs a signature σ;
- $b \leftarrow \mathsf{SIG.Ver}(pk, m, \sigma)$: the (deterministic) verification algorithm takes a public key pk, a message m, and a signature σ; it returns 1 if σ is valid and 0 otherwise.

Correctness requires that for any λ and any message m, when running $par \leftarrow \mathsf{SIG.Setup}(1^\lambda)$, $(sk, pk) \leftarrow \mathsf{SIG.KeyGen}(par)$, $\sigma \leftarrow \mathsf{SIG.Sign}(sk, m)$, and $b \leftarrow \mathsf{SIG.Ver}(pk, m, \sigma)$, one has $b = 1$ with probability 1. The standard security notion for a signature scheme is *existential unforgeability under chosen-message attack* (EUF-CMA), formalized via game EUF-CMA, which we recall in Figure 2. The Schnorr signature scheme [Sch91] is specified in Figure 3.

Game $\text{EUF-CMA}^{\mathcal{A}}_{\mathsf{SIG}}(\lambda)$	Oracle $\textsc{Sign}(m)$
$par \leftarrow \mathsf{SIG.Setup}(1^\lambda)$	$\sigma \leftarrow \mathsf{SIG.Sign}(sk, m)$
$(sk, pk) \leftarrow \mathsf{SIG.KeyGen}(par)\,;\ \mathsf{Q} := (\,)$	$\mathsf{Q} := \mathsf{Q} \parallel (m)$
$(m^*, \sigma^*) \leftarrow \mathcal{A}^{\textsc{Sign}}(pk)$	**return** σ
return $\big(m^* \notin \mathsf{Q} \wedge \mathsf{SIG.Ver}(pk, m^*, \sigma^*)\big)$	

Fig. 2. The EUF-CMA security game for a signature scheme SIG.

$\underline{\mathsf{Sch.Setup}(1^\lambda)}$ $(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda)$ Select $\mathsf{H}\colon \{0,1\}^* \to \mathbb{Z}_p$ **return** $par := (p, \mathbb{G}, G, \mathsf{H})$	$\underline{\mathsf{Sch.KeyGen}(par)}$ $(p, \mathbb{G}, G, \mathsf{H}) := par;\ x \leftarrow\!\!\$\, \mathbb{Z}_p;\ X := xG$ $sk := (par, x);\ pk := (par, X)$ **return** (sk, pk)
$\underline{\mathsf{Sch.Sign}(sk, m)}$ $(p, \mathbb{G}, G, \mathsf{H}, x) := sk;\ r \leftarrow\!\!\$\, \mathbb{Z}_p;\ R := rG$ $c := \mathsf{H}(R, m);\ s := r + cx \bmod p$ **return** $\sigma := (R, s)$	$\underline{\mathsf{Sch.Ver}(pk, m, \sigma)}$ $(p, \mathbb{G}, G, \mathsf{H}, X) := pk;\ (R, s) := \sigma$ $c := \mathsf{H}(R, m)$ **return** $(sG = R + cX)$

Fig. 3. The Schnorr signature scheme $\mathsf{Sch}[\mathsf{GrGen}]$ based on a group generator GrGen.

3.2 Security of Schnorr Signatures in the AGM

As a warm-up and to introduce some of the techniques used later, we reduce security of Schnorr signatures to hardness of DL in the AGM+ROM.

Theorem 1. *Let* GrGen *be a group generator. Let* $\mathcal{A}_{\mathrm{alg}}$ *be an algebraic adversary against the EUF-CMA security of the Schnorr signature scheme* $\mathsf{Sch}[\mathsf{GrGen}]$ *running in time at most* τ *and making at most* q_{s} *signature queries and* q_{h} *queries to the random oracle. Then there exists an algorithm* $\mathcal{B}$ *solving the DL problem w.r.t.* GrGen*, running in time at most* $\tau + O(q_{\mathrm{s}} + q_{\mathrm{h}})$*, such that*

$$\mathsf{Adv}^{\text{euf-cma}}_{\mathsf{Sch}[\mathsf{GrGen}],\mathcal{A}_{\mathrm{alg}}}(\lambda) \le \mathsf{Adv}^{\mathrm{dl}}_{\mathsf{GrGen},\mathcal{B}}(\lambda) + \frac{q_{\mathrm{s}}(q_{\mathrm{s}} + q_{\mathrm{h}}) + q_{\mathrm{h}} + 1}{2^{\lambda-1}} \ .$$

We start with some intuition for the proof. In the random oracle model, Schnorr signatures can be simulated without knowledge of the secret key by choosing random c and s, setting $R := sG - cX$ and then programming the random oracle so that $\mathsf{H}(R, m) = c$. On the other hand, an adversary that returns a signature forgery $(m^*, (R^*, s^*))$ can be used to compute the discrete logarithm of the public key X. In the ROM this can be proved by rewinding the adversary and using the Forking Lemma [PS96b, PS00], which entails a security loss.

In the AGM+ROM, extraction is straight-line and the security proof thus tight. After querying the signing oracle on messages $m_1, \ldots, m_{q_{\mathrm{s}}}$, the adversary obtains $(R_i, s_i)_{1 \le i \le q_{\mathrm{s}}}$ that verify $R_i = s_i G - c_i X$ with $c_i = \mathsf{H}(R_i, m_i)$. A valid forgery satisfies

$$R^* = s^* G - c^* X \ , \tag{1}$$

with $c^* := \mathsf{H}(R^*, m^*)$. On the other hand, since the adversary is algebraic, when it made its first query $\mathsf{H}(R^*, m^*)$, it provided a representation of R^* in basis $(G, X, R_1, \ldots, R_{q_{\mathrm{s}}})$, that is, $(\gamma^*, \xi^*, \vec{\rho}^*)$ with $R^* = \gamma^* G + \xi^* X + \sum_{i=1}^{q_{\mathrm{s}}} \rho_i^* R_i = \gamma^* G + \xi^* X + \sum_{i=1}^{q_{\mathrm{s}}} \rho_i^* s_i G - \sum_{i=1}^{q_{\mathrm{s}}} \rho_i^* c_i X$. Together with (1), this yields

$$\big(c^* + \xi^* - \textstyle\sum_{i=1}^{q_{\mathrm{s}}} \rho_i^* c_i\big) X = \big(s^* - \gamma^* - \textstyle\sum_{i=1}^{q_{\mathrm{s}}} \rho_i^* s_i\big) G \ .$$

Since c^* was chosen at random *after* the adversary chose $\xi^*, \rho_1^*, \ldots$ and $\rho_{q_{\mathrm{s}}}^*$, the probability that $c^* + \xi^* - \sum_{i=1}^{q_{\mathrm{s}}} \rho_i^* c_i \not\equiv_p 0$ is overwhelming, in which case we can compute the discrete logarithm of X from the above equation.

Game_0 $\left(\text{EUF-CMA}^{\mathcal{A}_{\text{alg}}}_{\mathsf{Sch[GrGen]}}\right)$, $\boxed{\mathsf{Game}_1}$, $\boxed{\mathsf{Game}_2}$

$(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda)$
$j := 0\,;\ x \leftarrow\!\!\$\ \mathbb{Z}_p\,;\ X := xG$
$\mathsf{Q} := (\,)\,;\ \mathsf{T} := (\,)\,;\ /\!\!/\ \mathsf{U} := (\,)$
$\left(m^*, (R^*_{[\gamma,\xi,\vec{\rho}]}, s^*)\right) \leftarrow \mathcal{A}^{\mathsf{H},\textsc{Sign}}_{\text{alg}}(p, \mathbb{G}, G, X)$
$\quad /\!\!/\ R^* = \gamma G + \xi X + \sum_{i=1}^{|\mathsf{Q}|} \rho_i R_i$
if $m^* \in \mathsf{Q}$ **then return** 0
$c^* := \mathsf{H}(R^*_{[\gamma,\xi,\vec{\rho}]}, m^*)$
$/\!\!/\ \ (\gamma^*, \xi^*, \vec{\rho}^*) := \mathsf{U}(R^*, m^*)$
$/\!\!/\ \ x = (s^* - \gamma^* - \sum_{i=1}^{|\mathsf{Q}|} \rho^*_i s_i)(\xi^* - \sum_{i=1}^{|\mathsf{Q}|} \rho^*_i c_i + c^*)^{-1} \bmod p$
return $(s^* G = R^* + c^* X)$

Oracle $\mathsf{H}(R_{[\gamma,\xi,\vec{\rho}]}, m)$

$\quad /\!\!/\ R = \gamma G + \xi X + \sum_{i=1}^{|\mathsf{Q}|} \rho_i R_i$
if $\mathsf{T}(R, m) = \bot$ **then**
$\quad \mathsf{T}(R, m) \leftarrow\!\!\$\ \mathbb{Z}_p\,;\ /\!\!/\ \mathsf{U}(R, m) := (\gamma, \xi, \vec{\rho})$
$\quad$ **if** $\xi - \sum_{i=1}^{|\mathsf{Q}|} \rho_i c_i \equiv_p -\mathsf{T}(R, m)$ **then**
$\quad\quad$ **abort game and return** 0 (I)
return $\mathsf{T}(R, m)$

Oracle $\widetilde{\mathsf{H}}(R, m)$

if $\mathsf{T}(R, m) = \bot$ **then**
$\quad \mathsf{T}(R, m) \leftarrow\!\!\$\ \mathbb{Z}_p$
return $\mathsf{T}(R, m)$

Oracle $\textsc{Sign}(m)$ $/\!\!/$ in Game_0 and Game_1

$j := j + 1\,;\ r_j \leftarrow\!\!\$\ \mathbb{Z}_p\,;\ R_j := r_j G$
$c_j := \widetilde{\mathsf{H}}(R_j, m)\,;\ s_j := r_j + c_j x \bmod p$
$\mathsf{Q} := \mathsf{Q} \| (m)$
return (R_j, s_j)

Oracle $\textsc{Sign}(m)$ $/\!\!/$ in Game_2

$j := j + 1\,;\ c_j, s_j \leftarrow\!\!\$\ \mathbb{Z}_p\,;\ R_j := s_j G - c_j X$
if $\mathsf{T}(R_j, m) = \bot$ **then** $\mathsf{T}(R_j, m) := c_j$
else abort game and return 0 (II)
$\mathsf{Q} := \mathsf{Q} \| (m)$
return (R_j, s_j)

Fig. 4. Games in the proof of Theorem 1. Game_0 is defined by ignoring all boxes; $\boxed{\text{boxes}}$ are included in Game_1 and Game_2; $\boxed{\text{gray boxes}}$ are only included in Game_2. The dark-gray comments show how $\mathcal{B}$ solves DL.

Proof of Theorem 1. Let $\mathcal{A}_{\text{alg}}$ be an algebraic adversary in $\text{EUF-CMA}_{\mathsf{Sch[GrGen]}}$ that makes at most q_{s} signature queries and q_{h} RO queries. We proceed by a sequence of games specified in Figure 4.

$\underline{\mathsf{Game}_0}$. The first game is EUF-CMA (Figure 2) for the Schnorr signature scheme (Figure 3) with a random oracle H. The game maintains a list Q of queried messages and T of values sampled for H. Since the adversary is algebraic, it must provide a representation $(\gamma, \xi, \vec{\rho})$ for its forgery $(m^*, (R^*_{[\gamma,\xi,\vec{\rho}]}, s^*))$ such that $R^* = \gamma G + \xi X + \sum_{i=1}^{|\mathsf{Q}|} \rho_i R_i$, and similarly for each RO query $\mathsf{H}(R_{[\gamma,\xi,\vec{\rho}]}, m)$. To prepare the change to Game_1, we have written the finalization of the game in an equivalent way: it first checks that $m^* \notin \mathsf{Q}$ and then runs $\mathsf{Sch.Ver}(X, m^*, (R^*, s^*))$, which we have written explicitly. Also, the game calls $c^* := \mathsf{H}(R^*_{[\gamma,\xi,\vec{\rho}]}, m^*)$ rather than $c^* := \widetilde{\mathsf{H}}(R^*, m^*)$ but in Game_0 these two oracles are equivalent. By definition,

$$\mathsf{Adv}^{\mathsf{game}_0}_{\mathcal{A}_{\text{alg}}}(\lambda) = \mathsf{Adv}^{\text{euf-cma}}_{\mathsf{Sch[GrGen]},\mathcal{A}_{\text{alg}}}(\lambda)\ . \tag{2}$$

$\underline{\mathsf{Game}_1}$. In Game_1, when $\mathsf{H}(R_{[\gamma,\xi,\vec{\rho}]}, m)$ is called, the game returns 0 in case the freshly drawn random value $\mathsf{T}(R, m)$ satisfies $\xi - \sum_{i=1}^{|\mathsf{Q}|} \rho_i c_i \equiv_p -\mathsf{T}(R, m)$ (line (I)). Clearly, for each call to H this happens with probability $1/p \leq 1/2^{\lambda-1}$. Since H is called at most q_{h} times by the adversary and once by the game when checking the signature, by the union bound, one has

$$\mathsf{Adv}^{\mathsf{game}_1}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) \geq \mathsf{Adv}^{\mathsf{game}_0}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) - \frac{q_{\mathrm{h}}+1}{2^{\lambda-1}} \ . \tag{3}$$

<u>Game_2</u>. In the final game we use the standard method for Schnorr signatures of simulating the Sign oracle without the secret key x by programming the random oracle. Game_1 and Game_2 are identical unless Game_2 returns 0 in line (II). For each signature query, R_j is uniformly random, and the size of table T is at most $q_{\mathrm{s}}+q_{\mathrm{h}}$, hence the game aborts in line (II) with probability at most $(q_{\mathrm{s}}+q_{\mathrm{h}})/p \leq (q_{\mathrm{s}}+q_{\mathrm{h}})/2^{\lambda-1}$. By summing over the at most q_{s} signature queries, we have

$$\mathsf{Adv}^{\mathsf{game}_2}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) \geq \mathsf{Adv}^{\mathsf{game}_1}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) - \frac{q_{\mathrm{s}}(q_{\mathrm{s}}+q_{\mathrm{h}})}{2^{\lambda-1}} \ . \tag{4}$$

<u>Reduction to DL</u>. We now construct an adversary $\mathcal{B}$ solving DL with the same probability as $\mathcal{A}_{\mathrm{alg}}$ wins Game_2. On input group description $(p, \mathbb{G}, G)$ and X, the adversary runs $\mathcal{A}_{\mathrm{alg}}$ on input $(p, \mathbb{G}, G, X)$ and simulates Game_2, which can be done without knowledge of $\log_G(X)$. It also uses an auxiliary table U that for each query $\mathsf{H}(R_{[\gamma,\xi,\vec{\rho}]}, m)$ stores the representation $(\gamma, \xi, \vec{\rho})$ of R. Assume that the adversary wins Game_2 by returning $(m^*, (R^*, s^*))$ and let $c^* := \mathsf{T}(R^*, m^*)$ (which is necessarily defined after the call $c^* := \mathsf{H}(R^*_{[\gamma,\xi,\vec{\rho}]}, m^*)$). We argue that necessarily $(\gamma^*, \xi^*, \vec{\rho^*}) := \mathsf{U}(R^*, m^*) \neq \perp$ and

$$\xi^* - \textstyle\sum_{i=1}^{|\mathsf{Q}|} \rho_i^* c_i \neq -c^* \bmod p. \tag{5}$$

First note that $m^* \notin \mathsf{Q}$, as otherwise the game would already have returned 0. Hence $\mathsf{T}(R^*, m^*)$ can only have been defined either (1) during a call to H by the adversary or (2), if it is still undefined when $\mathcal{A}_{\mathrm{alg}}$ stops, by the game when calling $c^* := \mathsf{H}(R^*_{[\gamma,\xi,\vec{\rho}]}, m^*)$. In both cases, this call sets both $\mathsf{T}(R^*, m^*)$ and $\mathsf{U}(R^*, m^*)$ and (5) is necessarily satisfied as otherwise the game would have returned 0 at line (I).

Since $R^* = \gamma^* G + \xi^* X + \sum_{i=1}^{|\mathsf{Q}|} \rho_i^* R_i$ with $R_i = s_i G - c_i X$ and validity of the forgery implies that $s^* G = R^* + c^* X$, it follows that $\left(s^* - \gamma^* - \sum_{i=1}^{|\mathsf{Q}|} \rho_i^* s_i\right) G = \left(\xi^* - \sum_{i=1}^{|\mathsf{Q}|} \rho_i^* c_i + c^*\right) X$ and $\mathcal{B}$ can compute

$$\log X = (s^* - \gamma^* - \textstyle\sum_{i=1}^{|\mathsf{Q}|} \rho_i^* s_i)(\xi^* - \sum_{i=1}^{|\mathsf{Q}|} \rho_i^* c_i + c^*)^{-1} \bmod p \ .$$

We thus have

$$\mathsf{Adv}^{\mathrm{dl}}_{\mathsf{GrGen},\mathcal{B}}(\lambda) = \mathsf{Adv}^{\mathsf{game}_2}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) \geq \mathsf{Adv}^{\mathrm{euf\text{-}cma}}_{\mathsf{Sch}[\mathsf{GrGen}],\mathcal{A}_{\mathrm{alg}}}(\lambda) - \frac{q_{\mathrm{s}}(q_{\mathrm{s}}+q_{\mathrm{h}})+q_{\mathrm{h}}+1}{2^{\lambda-1}} \ ,$$

where the inequality follows from Eqs. (2)–(4). Assuming that scalar multiplications in $\mathbb{G}$ and assignments in tables T and U take unit time, the running time of $\mathcal{B}$ is $\tau + O(q_{\mathrm{s}}+q_{\mathrm{h}})$. □

4 Blind Schnorr Signatures

4.1 Definitions

We start with defining the syntax and security of blind signature schemes and focus on schemes with a 2-round (i.e., 4 messages) signing protocol for concreteness.

Game $\mathrm{UNF}^{\mathcal{A}}_{\mathsf{BS}}(\lambda)$	Oracle $\textsc{Sign}_1(\mathit{msg})$
$\mathit{par} \leftarrow \mathsf{BS.Setup}(1^\lambda)$ $(\mathit{sk}, \mathit{pk}) \leftarrow \mathsf{BS.KeyGen}(\mathit{par})$ $k_1 := 0\,;\ k_2 := 0\,;\ \mathcal{S} := \emptyset$ $(m_i^*, \sigma_i^*)_{i \in [n]} \leftarrow \mathcal{A}^{\textsc{Sign}_1, \textsc{Sign}_2}(\mathit{pk})$ **return** $\big(k_2 < n$ $\quad \wedge\ \forall\, i \neq j \in [n] : (m_i^*, \sigma_i^*) \neq (m_j^*, \sigma_j^*)$ $\quad \wedge\ \forall\, i \in [n] : \mathsf{BS.Ver}(\mathit{pk}, m_i^*, \sigma_i^*) = 1\big)$	$k_1 := k_1 + 1$ // session id $(\mathit{msg}', \mathit{state}_{k_1}) \leftarrow \mathsf{BS.Sign}_1(\mathit{sk}, \mathit{msg})$ $\mathcal{S} := \mathcal{S} \cup \{k_1\}$ // open sessions **return** (k_1, msg') Oracle $\textsc{Sign}_2(j, \mathit{msg})$ **if** $j \notin \mathcal{S}$ **then return** $\perp$ $(\mathit{msg}', b) \leftarrow \mathsf{BS.Sign}_2(\mathit{state}_j, \mathit{msg})$ **if** $b = 1$ **then** $\mathcal{S} := \mathcal{S} \setminus \{j\}\,;\ k_2 := k_2 + 1$ **return** msg'

Fig. 5. The (strong) unforgeability game for a blind signature scheme BS with a 2-round signing protocol.

<u>Syntax.</u> A blind signature scheme BS consists of the following algorithms:

- $\mathit{par} \leftarrow \mathsf{BS.Setup}(1^\lambda)$: the setup algorithm takes the security parameter λ in unary and returns public parameters par;
- $(\mathit{sk}, \mathit{pk}) \leftarrow \mathsf{BS.KeyGen}(\mathit{par})$: the key generation algorithm takes the public parameters par and returns a secret/public key pair $(\mathit{sk}, \mathit{pk})$;
- $(b, \sigma) \leftarrow \langle \mathsf{BS.Sign}(\mathit{sk}), \mathsf{BS.User}(\mathit{pk}, m)\rangle$: an interactive protocol is run between the signer with private input a secret key sk and the user with private input a public key pk and a message m; the signer outputs $b = 1$ if the interaction completes successfully and $b = 0$ otherwise, while the user outputs a signature σ if it terminates correctly, and $\perp$ otherwise. For a 2-round protocol the interaction can be realized by the following algorithms:

$$\begin{aligned}
(\mathit{msg}_{U,0}, \mathit{state}_{U,0}) &\leftarrow \mathsf{BS.User}_0(\mathit{pk}, m)\\
(\mathit{msg}_{S,1}, \mathit{state}_{S}) &\leftarrow \mathsf{BS.Sign}_1(\mathit{sk}, \mathit{msg}_{U,0})\\
(\mathit{msg}_{U,1}, \mathit{state}_{U,1}) &\leftarrow \mathsf{BS.User}_1(\mathit{state}_{U,0}, \mathit{msg}_{S,1})\\
(\mathit{msg}_{S,2}, b) &\leftarrow \mathsf{BS.Sign}_2(\mathit{state}_{S}, \mathit{msg}_{U,1})\\
\sigma &\leftarrow \mathsf{BS.User}_2(\mathit{state}_{U,1}, \mathit{msg}_{S,2})
\end{aligned}$$

  (Typically, $\mathsf{BS.User}_0$ just initiates the session, and thus $\mathit{msg}_{U,0} = (\,)$ and $\mathit{state}_{U,0} = (\mathit{pk}, m)$.)
- $b \leftarrow \mathsf{BS.Ver}(\mathit{pk}, m, \sigma)$: the (deterministic) verification algorithm takes a public key pk, a message m, and a signature σ, and returns 1 if σ is valid on m under pk and 0 otherwise.

Correctness requires that for any λ and any message m, when running $\mathit{par} \leftarrow \mathsf{BS.Setup}(1^\lambda)$, $(\mathit{sk}, \mathit{pk}) \leftarrow \mathsf{BS.KeyGen}(\mathit{par})$, $(b, \sigma) \leftarrow \langle \mathsf{BS.Sign}(\mathit{sk}), \mathsf{BS.User}(\mathit{pk}, m)\rangle$, and $b' \leftarrow \mathsf{BS.Ver}(\mathit{pk}, m, \sigma)$, we have $b = 1 = b'$ with probability 1.

<u>Unforgeability.</u> The standard security notion for blind signatures demands that no user, after interacting arbitrary many times with a signer and k of these interactions were considered successful by the signer, can produce more than k signatures. Moreover, the adversary can schedule and interleave its sessions with the signer in any arbitrary way.

In game $\mathrm{UNF}^{\mathcal{A}}_{\mathsf{BS}}$ defined in Figure 5 the adversary has access to two oracles $\textsc{Sign}_1$ and $\textsc{Sign}_2$ corresponding to the two phases of the interactive protocol. The game maintains two

$$
\begin{array}{lcl}
\underline{\mathsf{BlSch.Sign}((p,\mathbb{G},G,\mathsf{H}),x)} & & \underline{\mathsf{BlSch.User}(((p,\mathbb{G},G,\mathsf{H}),X),m)} \\
r \leftarrow\!\!\$\, \mathbb{Z}_p\,;\ R \coloneqq rG & \xrightarrow{\quad R \quad} & \alpha,\beta \leftarrow\!\!\$\, \mathbb{Z}_p \\
 & & R' \coloneqq R + \alpha G + \beta X \\
 & & c' \coloneqq \mathsf{H}(R',m) \\
 & \xleftarrow{\quad c \quad} & c \coloneqq c' + \beta \bmod p \\
s \coloneqq r + cx \bmod p & \xrightarrow{\quad s \quad} & \\
 & & \textbf{if } sG \neq R + cX \textbf{ then return } \bot \\
 & & s' \coloneqq s + \alpha \bmod p \\
\textbf{return } 1 & & \textbf{return } \sigma \coloneqq (R',s')
\end{array}
$$

Fig. 6. The signing protocol of the blind Schnorr signature scheme.

counters k_1 and k_2 (initially set to 0), where k_1 is used as session identifier, and a set $\mathcal{S}$ of "open" sessions. Oracle $\textsc{Sign}_1$ takes the user's first message (which for blind Schnorr signatures is empty), increments k_1, adds k_1 to $\mathcal{S}$ and runs the first round on the signer's side, storing its state as $state_{k_1}$. Oracle $\textsc{Sign}_2$ takes as input a session identifier j and a user message; if $j \in \mathcal{S}$, it runs the second round on the signer's side; if successful, it removes j from $\mathcal{S}$ and increments k_2, which thus represents the number of successful interactions.

We say that BS satisfies unforgeability if $\mathsf{Adv}^{\mathrm{unf}}_{\mathsf{BS},\mathcal{A}}(\lambda)$ is negligible for all p.p.t. adversaries $\mathcal{A}$. Note that we consider "strong" unforgeability, which only requires that all pairs (m_i^*, σ_i^*) returned by the adversary (rather than all messages m_i^*) are distinct.

<u>Blindness.</u> Blindness requires that a signer cannot link a message/signature pair to a particular execution of the signing protocol. Formally, the adversary chooses two messages m_0 and m_1 and the experiment runs the signing protocol acting as the user with the adversary, first obtaining a signature σ_b on m_b and then σ_{1-b} on m_{1-b} for a random bit b. If both signatures are valid, the adversary is given (σ_0, σ_1) and must determine the value of b. A formal definition can be found in Appendix C.

<u>Blind Schnorr signatures.</u> A blind signature scheme BlSch is obtained from the scheme Sch in Figure 3 by replacing $\mathsf{Sch.Sign}$ with the interactive protocol specified in Figure 6 (the first message $msg_{U,0}$ from the user to the signer is empty and is not depicted). Correctness follows since a signature (R', s') obtained by the user after interacting with the signer satisfies $\mathsf{Sch.Ver}$:

$$
\begin{aligned}
s'G = sG + \alpha G = (r + cx)G + \alpha G &= R + \alpha G + \beta X + (-\beta + c)X \\
&= R' + c'X = R' + \mathsf{H}(R', m)\, X\ .
\end{aligned}
$$

Moreover, Schnorr signatures achieve perfect blindness [CP93].

4.2 The ROS Problem

The security of blind Schnorr signatures is related to the ROS (<u>R</u>andom inhomogeneities in an <u>O</u>verdetermined, <u>S</u>olvable system of linear equations) problem, which was introduced by Schnorr [Sch01]. Consider the game $\mathrm{ROS}_{\mathsf{GrGen},\ell,\Omega}$ in Figure 7, parameterized by a group generator GrGen,[8] an integer $\ell \geq 1$, and a set Ω (we omit GrGen and Ω from the notation

[8] The group generator GrGen is only used to generate a prime p of length λ; the group $\mathbb{G}$ is not used in the game.

Game $\mathrm{ROS}^{\mathcal{A}}_{\mathsf{GrGen},\ell,\Omega}(\lambda)$	Oracle $\mathsf{H}_{\mathrm{ros}}(\vec{\rho}, aux)$
$(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda)$; $\mathsf{T}_{\mathrm{ros}} := (\,)$ $\big((\vec{\rho}_i, \mathrm{aux}_i)_{i\in[\ell+1]}, (c_j)_{j\in[\ell]}\big) \leftarrow \mathcal{A}^{\mathsf{H}_{\mathrm{ros}}}(p)$ // $\vec{\rho}_i = (\rho_{i,1}, \dots, \rho_{i,\ell})$ **return** $\big(\forall\, i \neq i' \in [\ell+1] : (\vec{\rho}_i, \mathrm{aux}_i) \neq (\vec{\rho}_{i'}, \mathrm{aux}_{i'})$ $\quad\wedge\ \forall\, i \in [\ell+1] : \sum_{j=1}^{\ell} \rho_{i,j} c_j \equiv_p \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, \mathrm{aux}_i)\big)$	**if** $\mathsf{T}_{\mathrm{ros}}(\vec{\rho}, aux) = \bot$ **then** $\quad\mathsf{T}_{\mathrm{ros}}(\vec{\rho}, aux) \leftarrow\!\!\$\, \mathbb{Z}_p$ **return** $\mathsf{T}_{\mathrm{ros}}(\vec{\rho}, aux)$

Fig. 7. The ROS game, where $\mathsf{H}_{\mathrm{ros}} \colon (\mathbb{Z}_p)^\ell \times \Omega \to \mathbb{Z}_p$ is a random oracle.

when they are clear from context). The adversary $\mathcal{A}$ receives a prime p and has access to a random oracle $\mathsf{H}_{\mathrm{ros}}$ taking as input $(\vec{\rho}, aux)$ where $\vec{\rho} \in (\mathbb{Z}_p)^\ell$ and $aux \in \Omega$. Its goal is to find $\ell+1$ distinct pairs $(\vec{\rho}_i, aux_i)_{i\in[\ell+1]}$ together with a solution $(c_j)_{j\in[\ell]}$ to the linear system $\sum_{j=1}^{\ell} \rho_{i,j} c_j \equiv_p \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, aux_i)$, $i \in [\ell+1]$.[9]

The lemma below, which refines Schnorr's observation [Sch01], shows how an algorithm $\mathcal{A}$ solving the ROS_ℓ problem can be used to break the one-more unforgeability of blind Schnorr signatures.

Lemma 1. *Let* GrGen *be a group generator. Let* $\mathcal{A}$ *be an algorithm for game* $\mathrm{ROS}_{\mathsf{GrGen},\ell,\Omega}$*, where* $\Omega = (\mathbb{Z}_p)^2 \times \{0,1\}^*$*, running in time at most* τ *and making at most* q_{h} *random oracle queries. Then there exists an (algebraic) adversary* $\mathcal{B}$ *running in time at most* $\tau + O(\ell + q_{\mathrm{h}})$*, making at most* ℓ *queries to* Sign_1 *and* Sign_2 *and* q_{h} *random oracle queries, such that*

$$\mathsf{Adv}^{\mathrm{unf}}_{\mathsf{BlSch}[\mathsf{GrGen}],\mathcal{B}}(\lambda) \geq \mathsf{Adv}^{\mathrm{ros}}_{\mathsf{GrGen},\ell,\Omega,\mathcal{A}}(\lambda) - \frac{q_{\mathrm{h}}^2 + (\ell+1)^2}{2^{\lambda-1}}\ .$$

Proof. We first consider a slightly modified game $\mathrm{ROS}'_{\mathsf{GrGen},\ell,\Omega}$, which differs from ROS in that it first draws $x, r_1, \dots, r_\ell \leftarrow\!\!\$\, \mathbb{Z}_p$ and returns 0 if one of the following two events occurs:

- E_1: when $\mathcal{A}$ queries $\mathsf{H}_{\mathrm{ros}}(\vec{\rho}, (\gamma, \xi, m))$, there has been a previous query $\mathsf{H}_{\mathrm{ros}}(\vec{\rho}\,', (\gamma', \xi', m'))$ such that $m = m'$ and
$$\gamma + \xi x + \textstyle\sum_{j=1}^{\ell} \rho_j r_j \equiv_p \gamma' + \xi' x + \sum_{j=1}^{\ell} \rho'_j r_j\ ;$$
- E_2: when $\mathcal{A}$ returns $\big((\vec{\rho}_i, (\gamma_i, \xi_i, m_i))_{i\in[\ell+1]}, (c_j)_{j\in[\ell]}\big)$, there exists $i \neq i' \in [\ell+1]$ such that $m_i = m_{i'}$ and
$$\gamma_i + \xi_i x + \textstyle\sum_{j=1}^{\ell} \rho_{i,j} r_j \equiv_p \gamma_{i'} + \xi_{i'} x + \sum_{j=1}^{\ell} \rho_{i',j} r_j\ .$$

Games ROS and ROS$'$ are identical unless E_1 or E_2 occurs in ROS$'$. Note that we could defer the random selection of $x, r_1, \dots, r_\ell$ and the check whether E_1 or E_2 occurred to the very end of the game. Consider two *distinct* random oracle queries $(\vec{\rho}, (\gamma, \xi, m))$ and $(\vec{\rho}\,', (\gamma', \xi', m'))$; if $m \neq m'$ then E_1 cannot occur; if $m = m'$, then $(\gamma, \xi, \vec{\rho}) \neq (\gamma', \xi', \vec{\rho}\,')$ and by the Schwartz-Zippel Lemma,

$$(\gamma - \gamma') + (\xi - \xi')x + \textstyle\sum_{j=1}^{\ell} (\rho_j - \rho'_j) r_j \equiv_p 0$$

[9] The original definition of the problem by Schnorr [Sch01] sets $\Omega := \emptyset$. Our more general definition does not seem to significantly modify the hardness of the problem while allowing to prove Theorem 2.

with probability $1/p \le 1/2^{\lambda-1}$ over the draw of $x, r_1, \ldots, r_\ell$. Hence, event E_1 occurs with probability at most $q_{\mathrm{h}}^2/2^{\lambda-1}$. Similarly, event E_2 occurs with probability at most $(\ell+1)^2/2^{\lambda-1}$. Hence,

$$\mathsf{Adv}^{\mathrm{ros}'}_{\mathsf{GrGen},\ell,\Omega,\mathcal{A}}(\lambda) \ge \mathsf{Adv}^{\mathrm{ros}}_{\mathsf{GrGen},\ell,\Omega,\mathcal{A}}(\lambda) - \frac{q_{\mathrm{h}}^2 + (\ell+1)^2}{2^{\lambda-1}} \ . \tag{6}$$

We now construct an adversary $\mathcal{B}$ for the game $\mathrm{UNF}_{\mathsf{BlSch[GrGen]}}$ as follows. Adversary $\mathcal{B}$, which takes as input $(p, \mathbb{G}, G, X)$ and has access to random oracle H and signing oracles $\textsc{Sign}_1$ and $\textsc{Sign}_2$, simulates game ROS′ as follows. First, $\mathcal{B}$ initiates ℓ parallel instances of the protocol by querying $(j, R_j) \leftarrow \textsc{Sign}_1()$ for $j \in [\ell]$. Then, it runs $\mathcal{A}(p)$. When $\mathcal{A}$ queries $\mathsf{H}_{\mathrm{ros}}(\vec{\rho}, (\gamma, \xi, m))$ where $\vec{\rho} = (\rho_j)_{j\in[\ell]} \in (\mathbb{Z}_p)^\ell$ and $(\gamma, \xi, m) = aux \in (\mathbb{Z}_p)^2 \times \{0,1\}^*$, $\mathcal{B}$ computes $R \coloneqq \gamma G + \xi X + \sum_{j=1}^{\ell} \rho_j R_j$ and returns $\mathsf{H}(R, m) + \xi$, unless there has been a previous query $\mathsf{H}_{\mathrm{ros}}(\vec{\rho}', (\gamma', \xi', m'))$ with $m = m'$ and $R = \gamma' G + \xi' X + \sum_{j=1}^{\ell} \rho'_j R_j$, in which case $\mathcal{B}$ aborts. It is easy to see that $\mathcal{B}$ perfectly simulate game ROS′. Eventually, $\mathcal{A}$ returns $((\vec{\rho}_i, (\gamma_i, \xi_i, m_i^*))_{i\in[\ell+1]}, (c_j)_{j\in[\ell]})$. Then $\mathcal{B}$ closes all signing sessions by calling $s_j \leftarrow \textsc{Sign}_2(j, c_j)$ for $j \in [\ell]$. Finally, for $i \in [\ell+1]$, it computes

$$\begin{aligned} R_i^* &\coloneqq \gamma_i G + \xi_i X + \textstyle\sum_{j=1}^{\ell} \rho_{i,j} R_j \\ s_i^* &\coloneqq \gamma_i + \textstyle\sum_{j=1}^{\ell} \rho_{i,j} s_j \bmod p \end{aligned}$$

and returns $\ell+1$ forgeries $(m_i^*, (R_i^*, s_i^*))_{i\in[\ell+1]}$.

Assume that $\mathcal{A}$ wins game ROS′. Then, in particular, (i) all pairs (m_i^*, R_i^*) are distinct and (ii) for all $i \in [\ell+1]$,

$$\textstyle\sum_{j=1}^{\ell} \rho_{i,j} c_j \equiv_p \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, (\gamma_i, \xi_i, m_i^*)) \equiv_p \mathsf{H}(R_i^*, m_i^*) + \xi_i \ ,$$

where the second equality follows from the way $\mathcal{B}$ answers $\mathcal{A}$'s queries to $\mathsf{H}_{\mathrm{ros}}$. While (i) implies that all forgeries $(m_i^*, (R_i^*, s_i^*))$ are distinct, (ii) implies that all forgeries are valid since for all $i \in [\ell+1]$,

$$s_i^* G = \gamma_i G + \sum_{j=1}^{\ell} \rho_{i,j}(r_j + c_j x) G = \underbrace{\gamma_i G + \sum_{j=1}^{\ell} \rho_{i,j} R_j}_{R_i^* - \xi_i X} + \underbrace{\left(\sum_{j=1}^{\ell} \rho_{i,j} c_j\right)}_{\mathsf{H}(R_i^*, m_i^*) + \xi_i} X = R_i^* + \mathsf{H}(R_i^*, m_i^*) X \ .$$

Thus, $\mathcal{B}$ successfully breaks unforgeability of $\mathsf{BlSch[GrGen]}$, and thus

$$\mathsf{Adv}^{\mathrm{unf}}_{\mathsf{BlSch[GrGen]},\mathcal{B}}(\lambda) = \mathsf{Adv}^{\mathrm{ros}'}_{\mathsf{GrGen},\ell,\Omega,\mathcal{A}}(\lambda) \ . \tag{7}$$

Clearly, $\mathcal{B}$ runs in time at most $\tau + O(\ell + q_{\mathrm{h}})$ and makes at most ℓ queries to $\textsc{Sign}_1$ and $\textsc{Sign}_2$ and q_{h} random oracle queries. Combining Eqs. (6) and (7) concludes the proof. □

The hardness of the ROS problem critically depends on ℓ. In particular, for small values of ℓ, the ROS problem is statistically hard, as captured by the following lemma.

Lemma 2. *Let* GrGen *be a group generator,* $\ell \ge 1$*, and* Ω *be some arbitrary set. Then for any adversary* $\mathcal{A}$ *making at most* q_{h} *queries to* $\mathsf{H}_{\mathrm{ros}}$*,*

$$\mathsf{Adv}^{\mathrm{ros}}_{\mathsf{GrGen},\ell,\Omega,\mathcal{A}}(\lambda) \le \frac{\binom{q_{\mathrm{h}}}{\ell+1} + 1}{2^{\lambda-1}} \ .$$

Proof. Consider a modified game $\mathrm{ROS}^*_{\mathsf{GrGen},\ell,\Omega}$ that is identical to ROS, except that it returns 0 when the adversary outputs $((\vec{\rho}_i, aux_i)_{i\in[\ell+1]}, (c_j)_{j\in[\ell]})$ such that for some $i \in [\ell+1]$ it has not made the query $\mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, aux_i)$. Games ROS and ROS* are identical unless in game ROS the adversary wins and has not made the query $\mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, aux_i)$ for some i, which happens with probability at most $1/p \le 1/2^{\lambda-1}$. Hence,

$$\mathsf{Adv}^{\mathrm{ros}}_{\mathsf{GrGen},\ell,\Omega,\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathrm{ros}^*}_{\mathsf{GrGen},\ell,\Omega,\mathcal{A}}(\lambda) + \frac{1}{2^{\lambda-1}} \ .$$

In order to win the modified game ROS*, $\mathcal{A}$ must in particular make $\ell+1$ distinct random oracle queries $(\vec{\rho}_i, aux_i)_{i\in[\ell+1]}$ such that the system

$$\textstyle\sum_{j=1}^{\ell} \rho_{i,j} c_j \equiv_p \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, aux_i), \quad i \in [\ell+1] \tag{8}$$

with unknowns $c_1, \dots, c_\ell$ has a solution. Consider any subset of $\ell+1$ queries $(\vec{\rho}_i, aux_i)_{i\in[\ell+1]}$ made by the adversary to the random oracle and let M denote the $(\ell+1)\times\ell$ matrix whose i-th row is $\vec{\rho}_i$ and let $t \le \ell$ denote its rank. Then, Eq. (8) has a solution if and only if the row vector $\vec{h} := (\mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, aux_i))^{\mathrm{T}}_{i\in[\ell+1]}$ is in the span of the columns of M. Since $\vec{h}$ is uniformly random, this happens with probability $p^t/p^{\ell+1} \le 1/p \le 1/2^{\lambda-1}$. By the union bound,

$$\mathsf{Adv}^{\mathrm{ros}^*}_{\mathsf{GrGen},\ell,\Omega,\mathcal{A}}(\lambda) \le \frac{\binom{q_{\mathrm{h}}}{\ell+1}}{2^{\lambda-1}} \ ,$$

which concludes the proof. □

On the other hand, the ROS$_\ell$ problem can be reduced the $(\ell+1)$-sum problem, for which Wagner's generalized birthday algorithm [Wag02, MS12, NS15] can be used. More specifically, consider the $(\ell+1)\times\ell$ matrix

$$(\rho_{i,j}) = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ & & \ddots & \\ 0 & \cdots & 0 & 1 \\ 1 & \cdots & \cdots & 1 \end{pmatrix}$$

and let $\vec{\rho}_i$ denote its i-th line, $i \in [\ell+1]$. Let $q := 2^{\lambda/(1+\lfloor \lg(\ell+1)\rfloor)}$. The solving algorithm builds lists $L_i = (\mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, aux_{i,k}))_{k\in[q]}$ for $i \in [\ell]$ and $L_{\ell+1} = (-\mathsf{H}_{\mathrm{ros}}(\vec{\rho}_{\ell+1}, aux_{\ell+1,k}))_{k\in[q]}$ for arbitrary values $aux_{i,k}$ and uses Wagner's algorithm to find an element e_i in each list L_i such that $\sum_{i=1}^{\ell+1} e_i \equiv_p 0$. Then, it is easily seen that $((\vec{\rho}_i, aux_i)_{i\in[\ell+1]}, (e_j)_{j\in[\ell]})$, where aux_i is such that $e_i = \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, aux_i)$, is a solution to the ROS problem. This algorithm makes $q_{\mathrm{h}} = (\ell+1)2^{\lambda/(1+\lfloor \lg(\ell+1)\rfloor)}$ random oracle queries, runs in time an space $O((\ell+1)2^{\lambda/(1+\lfloor \lg(\ell+1)\rfloor)})$, and succeeds with constant probability.

4.3 Security of Blind Schnorr Signatures

We now formally prove that blind Schnorr signatures are unforgeable assuming the hardness of the one-more discrete logarithm problem and the ROS problem.

Theorem 2. *Let* GrGen *be a group generator. Let* $\mathcal{A}_{\mathrm{alg}}$ *be an algebraic adversary against the UNF security of the blind Schnorr signature scheme* $\mathsf{BlSch}[\mathsf{GrGen}]$ *running in time at most* τ *and making at most* q_{s} *queries to* $\textsc{Sign}_1$ *and* q_{h} *queries to the random oracle. Then there exist* *an algorithm $\mathcal{B}_{\mathrm{ros}}$ for the ROS$_{q_{\mathrm{s}}}$ problem making at most $q_{\mathrm{h}} + q_{\mathrm{s}} + 1$ random oracle queries and an algorithm $\mathcal{B}_{\mathrm{omdl}}$ for the OMDL problem w.r.t.* GrGen *making at most q_{s} queries to its oracle* DLog*, both running in time at most $\tau + O(q_{\mathrm{s}} + q_{\mathrm{h}})$, such that*

$$\mathsf{Adv}^{\mathrm{unf}}_{\mathsf{BlSch}[\mathsf{GrGen}],\mathcal{A}_{\mathrm{alg}}}(\lambda) \leq \mathsf{Adv}^{\mathrm{omdl}}_{\mathsf{GrGen},\mathcal{B}_{\mathrm{omdl}}}(\lambda) + \mathsf{Adv}^{\mathrm{ros}}_{\ell,\mathcal{B}_{\mathrm{ros}}}(\lambda) \ .$$

We start with explaining the proof idea. Consider an adversary in the unforgeability game, let X be the public key and $R_1, \dots, R_\ell$ be the elements returned by the oracle Sign_1 and let (R_i^*, s_i^*) be the adversary's forgeries on messages m_i^*. As $\mathcal{A}_{\mathrm{alg}}$ is algebraic, it must also output a representation $(\gamma_i, \xi_i, \vec{\rho}_i)$ for R_i^* w.r.t. the group elements received from the game: $R_i^* = \gamma_i G + \xi_i X + \sum_{j=1}^{\ell} \rho_{i,j} R_j$. Validity of the forgeries implies another representation, namely $R_i^* = s_i^* G - c_i^* X$ with $c_i^* = \mathsf{H}(R_i^*, m_i^*)$. Together, these yield

$$(c_i^* + \xi_i^*)X + \textstyle\sum_{j=1}^{\ell} \rho_{i,j}^* R_j = (s_i^* - \gamma_i^*)G \ , \tag{9}$$

which intuitively can be used to compute $\log X$.

However, the reduction also needs to simulate Sign_2 queries, for which, contrary to the proof for standard Schnorr signatures (Theorem 1), it cannot rely on programming the random oracle. In fact, the reduction can only win OMDL, which is an *easier* game than DL. In particular, the reduction obtains $X, R_1, \dots, R_q$ from its challenger and must compute their logarithms. It can make q logarithm queries, which it uses to simulate the Sign_2 oracle: on input (j, c_j), it simply returns $s_j \leftarrow \mathrm{DLog}(R_j + c_j X)$.

But this means that in Eq. (9) the reduction does not know the logarithms of the R_j's; all it knows is $R_j = s_j G - c_j X$, which, when plugged into Eq. (9) yields

$$(\underbrace{c_i^* + \xi_i^* - \textstyle\sum_{j=1}^{\ell} \rho_{i,j}^* c_j}_{=:\chi_i})X = (s_i^* - \gamma_i^* - \textstyle\sum_{j=1}^{\ell} \rho_{i,j}^* s_j)G \ .$$

Thus, if for some i, $\chi_i \neq 0$, the reduction can compute $x = \log X$, from which it can derive $r_j = \log R_j = s_j - c_j\, x$. Together, $x, r_1, \dots, r_q$ constitute an OMDL solution.

On the other hand, we can show that if $\chi_i = 0$ for *all* i, then the adversary has actually found a solution to the ROS problem (Figure 7): A reduction to ROS would answer the adversary's queries $\mathsf{H}(R_{[\gamma,\xi,\vec{\rho}]}, m)$ by $\mathsf{H}_{\mathrm{ros}}(\vec{\rho}, (\gamma, \xi, m)) - \xi$; then $\chi_i = 0$ implies (recall that $c_i^* = \mathsf{H}(R_i^*, m^*)$)

$$0 = \chi_i = \mathsf{H}(R_i^*, m_i^*) + \xi_i^* - \textstyle\sum_{j=1}^{\ell} \rho_{i,j}^* c_j = \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i^*, (\gamma_i^*, \xi_i^*, m_i^*)) - \textstyle\sum_{j=1}^{\ell} \rho_{i,j}^* c_j \ ,$$

meaning $((\vec{\rho}_i^*, (\gamma_i^*, \xi_i^*, m_i^*))_i, (c_j)_j)$ is a solution to ROS.

To simplify the proof we first show the following lemma.

Lemma 3. *Let* GrGen *be a group generator and let $\mathcal{A}$ be an adversary against the UNF security of the blind Schnorr signature scheme* $\mathsf{BlSch}[\mathsf{GrGen}]$ *running in time at most τ and making at most q_{s} queries to Sign_1 and q_{h} queries to the random oracle. Then there exists an adversary $\mathcal{B}$ that makes* exactly q_{s} *queries to Sign_1 and q_{s} queries to Sign_2 that do not return $\perp$, and returns $q_{\mathrm{s}} + 1$ forgeries, running in time at most $\tau + O(q_{\mathrm{s}})$, such that*

$$\mathsf{Adv}^{\mathrm{unf}}_{\mathsf{BlSch}[\mathsf{GrGen}],\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathrm{unf}}_{\mathsf{BlSch}[\mathsf{GrGen}],\mathcal{B}}(\lambda) \ .$$

Proof. We construct the following adversary that plays game UNF (Figure 5). On input pk, adversary $\mathcal{B}$ runs $\mathcal{A}(pk)$ and relays all oracle queries and responses between its challenger and $\mathcal{A}$. Let q be the number of $\mathcal{A}$'s $\textsc{Sign}_1$ queries, let $R_1, \ldots, R_q$ be the answers, and let $\mathcal{C}$ be the completed sessions, that is, the set of values j such that $\mathcal{A}$ queried $\textsc{Sign}_2$ on some input $(j, *)$ and $\textsc{Sign}_2$ did not reply $\perp$. Let $(m_i^*, (R_i^*, s_i^*))_{i\in[n]}$ be $\mathcal{A}$'s output, for which we must have $k = |\mathcal{C}| < n$ when $\mathcal{A}$ wins.

$\mathcal{B}$ then makes $q_\mathrm{s} - q$ queries to $\textsc{Sign}_1$ to receive $R_{q+1}, \ldots, R_{q_\mathrm{s}}$. Next, $\mathcal{B}$ completes all $q_\mathrm{s} - k$ open signing sessions for distinct messages by following the protocol in Figure 6: for every $j \in \mathcal{S} := [1, \ldots, q_\mathrm{s}] \setminus \mathcal{C}$, adversary $\mathcal{B}$ picks a fresh message $m_j \notin \{m_i^*\}_{i\in[n]} \cup \{m_i\}_{i\in\mathcal{S}\setminus[j]}$ and $\alpha_j, \beta_j \leftarrow\!\!\$\, \mathbb{Z}_p$, computes $R'_j := R_j + \alpha_j G + \beta_j X$, queries $\mathsf{H}(R', m_j)$ to get c'_j, computes $c_j := c'_j + \beta_j \bmod p$ and queries (j, c_j) to $\textsc{Sign}_2$. Upon receiving s_j, $\mathcal{B}$ computes $s'_j := s_j + \alpha_j \bmod p$, which yields a signature (R'_j, s'_j) on message m_j.

Finally, $\mathcal{B}$ concatenates $\mathcal{A}$'s output with $q_\mathrm{s}+1-n \le q_\mathrm{s}-k$ signatures: let $\mathcal{S} = \{j_1, \ldots, j_{q_\mathrm{s}-k}\}$; then $\mathcal{B}$ returns $(m_i^*, (R_i^*, s_i^*))_{i\in[n]} \parallel (m_{j_i}, (R'_{j_i}, s'_{j_i}))_{i\in[q_\mathrm{s}+1-n]}$. When $\mathcal{A}$ wins the game, all tuples $(m_i^*, (R_i^*, s_i^*))$ are different; as all remaining messages also differ, all tuples output by $\mathcal{B}$ are distinct. By correctness of the scheme, $\mathcal{B}$'s signatures are valid. Thus whenever $\mathcal{A}$ wins, then so does $\mathcal{B}$. □

Proof of Theorem 2. Let $\mathcal{A}_\mathrm{alg}$ be an algebraic adversary making at most q_s queries to $\textsc{Sign}_1$ and q_h random oracle queries. By the above lemma, we can assume that $\mathcal{A}_\mathrm{alg}$ makes exactly $\ell := q_\mathrm{s}$ queries to $\textsc{Sign}_1$, closes all sessions, and returns $\ell + 1$ valid signatures. We proceed with a sequence of games defined in Figure 8.

<u>Game_0</u>. The first game is the UNF game (Figure 5) for scheme $\mathsf{BlSch}[\mathsf{GrGen}]$ played with $\mathcal{A}_\mathrm{alg}$ in the random oracle model. We have written the finalization of the game in a different but equivalent way. In particular, instead of checking that $(m_i^*, (R_i^*, s_i^*)) \neq (m_{i'}^*, (R_{i'}^*, s_{i'}^*))$ for all $i \neq i' \in [\ell + 1]$, we simply check that $(m_i^*, R_i^*) \neq (m_{i'}^*, R_{i'}^*)$. This is equivalent since for any pair (m, R), there is a single $s \in \mathbb{Z}_p$ such that (R, s) is a valid signature for m. Hence, if the adversary returns $(m_i^*, (R_i^*, s_i^*))$ and $(m_{i'}^*, (R_{i'}^*, s_{i'}^*))$ with $(m_i^*, R_i^*) = (m_{i'}^*, R_{i'}^*)$ and $s_i^* \neq s_{i'}^*$, at least one of the two forgeries is invalid. Thus,

$$\mathsf{Adv}^{\mathsf{game}_0}_{\mathcal{A}_\mathrm{alg}}(\lambda) = \mathsf{Adv}^{\mathrm{unf}}_{\mathsf{BlSch}[\mathsf{GrGen}],\mathcal{A}_\mathrm{alg}}(\lambda)\ . \tag{10}$$

<u>Game_1</u>. In Game_1, we make the following changes (which are analogous to those in the proof of Theorem 1). First, we introduce an auxiliary table U that for each query $\mathsf{H}(R_{[\gamma,\xi,\vec{\rho}]}, m)$ stores the representation $(\gamma, \xi, \vec{\rho})$ of R. Second, when the adversary returns its forgeries $(m_i^*, (R^*_{i\,[\gamma_i,\xi_i,\vec{\rho}_i]}, s_i^*))_{i\in[\ell+1]}$, then for each $i \in [\ell + 1]$ for which $\mathsf{T}(R_i^*, m_i^*)$ is undefined, we emulate a call to $\mathsf{H}(R^*_{i\,[\gamma_i,\xi_i,\vec{\rho}_i]}, m_i^*)$. Again, this does not change the output of the game, since in Game_0, the value $\mathsf{T}(R_i^*, m_i^*)$ would be randomly assigned when the game calls $\widetilde{\mathsf{H}}$ to check the signature. Finally, for each $i \in [\ell + 1]$, we retrieve $(\gamma_i^*, \xi_i^*, \vec{\rho}_i^*) := \mathsf{U}(R_i^*, m_i^*)$ (which is necessarily defined at this point) and return 0 if $\sum_{i=1}^{\ell} \rho^*_{i,j} c_j \equiv_p c_i^* + \xi_i^*$ for all $i \in [\ell+1]$, where c_j is the (unique) value submitted to $\textsc{Sign}_2$ together with j and not answered by $\perp$.

Game_0 and Game_1 are identical unless Game_1 returns 0 in line (I). We reduce indistinguishability of the games to ROS by constructing an algorithm $\mathcal{B}_\mathrm{ros}$ solving the ROS_ℓ problem whenever Game_1 stops in line (I). Algorithm $\mathcal{B}_\mathrm{ros}$, which has access to oracle H_ros, runs $\mathcal{A}_\mathrm{alg}$ and simulates Game_1 in a straightforward way, except for using its H_ros oracle to define the entries of T.

$\mathsf{Game}_0\ \left(\mathrm{UNF}^{\mathcal{A}_{\mathrm{alg}}}_{\mathsf{BlSch}[\mathsf{GrGen}]}(\lambda)\right),\ \boxed{\mathsf{Game}_1}$

$(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda)$
$x \leftarrow\!\$\ \mathbb{Z}_p\,;\ X := xG\,;\ k_1 := 0\,;\ k_2 := 0\,;$
$\mathcal{S} := \emptyset\,;\ \mathsf{T} := (\,)\,;\ \boxed{\mathsf{U} := (\,)}$
$\left(m_i^*, (R^*_{i\,[\gamma_i,\xi_i,\vec{\rho}_i]}, s_i^*)\right)_{i\in[\ell+1]}$
$\qquad\leftarrow \mathcal{A}_{\mathrm{alg}}^{\mathsf{H},\textsc{Sign}_1,\textsc{Sign}_2}(p, \mathbb{G}, G, X)$
$/\!\!/\ R_i^* = \gamma_i G + \xi_i X + \sum_{j=1}^{\ell} \rho_{i,j} R_j$
if $k_2 > \ell$ **then return** 0
if $\exists\, i \neq i' \in [\ell+1] : (m_i^*, R_i^*) = (m_{i'}^*, R_{i'}^*)$ **then**
  return 0

$\boxed{\begin{array}{l} \textbf{for } i = 1 \ldots \ell+1 \textbf{ do} \\ \quad \textbf{if } \mathsf{T}(R_i^*, m_i^*) = \bot \textbf{ then} \\ \quad\quad \mathsf{T}(R_i^*, m_i^*) \leftarrow\!\$\ \mathbb{Z}_p \\ \quad\quad /\!\!/\ \mathsf{T}(R_i^*, m_i^*) := \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, (\gamma_i, \xi_i, m_i^*)) - \xi_i \\ \quad\quad \mathsf{U}(R_i^*, m_i^*) := (\gamma_i, \xi_i, \vec{\rho}_i) \end{array}}$

for $i = 1 \ldots \ell+1$ **do**
  $c_i^* := \widetilde{\mathsf{H}}(R_i^*, m_i^*)$ $/\!\!/$ doesn't modify T in Game_1

$\boxed{\begin{array}{l} \quad (\gamma_i^*, \xi_i^*, \vec{\rho}_i^*) := \mathsf{U}(R_i^*, m_i^*) \\ \textbf{if } \forall\, i \in [\ell+1] : \sum_{j=1}^{\ell} \rho_{i,j}^* c_j \equiv_p c_i^* + \xi_i^* \\ \quad \textbf{then return } 0 \qquad\qquad\qquad \text{(I)} \\ \quad /\!\!/\ ((\vec{\rho}_i^*, (\gamma_i^*, \xi_i^*, m_i^*))_{i\in[\ell+1]}, \vec{c}) \text{ solves ROS} \end{array}}$

return $(\forall\, i \in [\ell+1] : s_i^* G = R_i^* + c_i^* X)$

Oracle $\widetilde{\mathsf{H}}(R, m)$

if $\mathsf{T}(R, m) = \bot$ **then** $\mathsf{T}(R, m) \leftarrow\!\$\ \mathbb{Z}_p$
return $\mathsf{T}(R, m)$

Oracle $\mathsf{H}(R_{[\gamma,\xi,\vec{\rho}]}, m)$

$/\!\!/\ R = \gamma G + \xi X + \sum_{j=1}^{|\vec{\rho}|} \rho_j R_j$
if $\mathsf{T}(R, m) = \bot$ **then**
  $\mathsf{T}(R, m) \leftarrow\!\$\ \mathbb{Z}_p$
  $/\!\!/\ \mathsf{T}(R, m) := \mathsf{H}_{\mathrm{ros}}(\vec{\rho}, (\gamma, \xi, m)) - \xi$
  $\boxed{\mathsf{U}(R, m) := (\gamma, \xi, \vec{\rho})}$
return $\mathsf{T}(R, m)$

Oracle $\textsc{Sign}_1()$

$k_1 := k_1 + 1\,;\ r_{k_1} \leftarrow\!\$\ \mathbb{Z}_p$
$R_{k_1} := r_{k_1} G$ $/\!\!/\ R_{k_1} \leftarrow \textsc{Chal}()$
$\mathcal{S} := \mathcal{S} \cup \{k_1\}$
return (k_1, R_{k_1})

Oracle $\textsc{Sign}_2(j, c_j)$

if $j \notin \mathcal{S}$ **then return** $\bot$
$s_j := r_j + c_j x$ $/\!\!/\ s_j \leftarrow \textsc{DLog}(R_j + c_j X)$
$\mathcal{S} := \mathcal{S} \setminus \{j\}\,;\ k_2 := k_2 + 1$
return s_j

Fig. 8. Games used in the proof of Theorem 2. Game_0 ignores all boxes. The light-gray comments in Game_1 and oracle H show how reduction $\mathcal{B}_{\mathrm{ros}}$ solves ROS; the dark-gray comments in the $\textsc{Sign}$ oracles show how $\mathcal{B}_{\mathrm{omdl}}$ embeds its challenges and simulates Game_1.

In particular, consider a query $\mathsf{H}(R_{[\gamma,\xi,\vec{\rho}]}, m)$ by $\mathcal{A}_{\mathrm{alg}}$ such that $\mathsf{T}(R, m) = \bot$. Then $\mathcal{B}_{\mathrm{ros}}$ pads the vector $\vec{\rho}$ with 0's to make it of length ℓ (at this point, not all $R_1, \ldots, R_\ell$ are necessarily defined, so $\vec{\rho}$ might not be of length ℓ), and assigns $\mathsf{T}(R, m) := \mathsf{H}_{\mathrm{ros}}(\vec{\rho}, (\gamma, \xi, m)) - \xi$ (cf. comments in Figure 8). Similarly, when $\mathcal{A}_{\mathrm{alg}}$ returns its forgeries $(m_i^*, (R^*_{i\,[\gamma_i,\xi_i,\vec{\rho}_i]}, s_i^*))_{i\in[\ell+1]}$, then for each $i \in [\ell+1]$ with $\mathsf{T}(R_i^*, m_i^*) = \bot$, reduction $\mathcal{B}_{\mathrm{ros}}$ assigns $\mathsf{T}(R_i^*, m_i^*) := \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i, (\gamma_i, \xi_i, m_i^*)) - \xi_i$. Since $\mathsf{H}_{\mathrm{ros}}$ returns uniformly random elements in $\mathbb{Z}_p$, the simulation is perfect.

If Game_1 aborts in line (I), then $\mathcal{B}_{\mathrm{ros}}$ returns $((\vec{\rho}_i^*, (\gamma_i^*, \xi_i^*, m_i^*))_{i\in[\ell+1]}, (c_j)_{j\in[\ell]})$, where $(\gamma_i^*, \xi_i^*, \vec{\rho}_i^*) := \mathsf{U}(R_i^*, m_i^*)$. We show that this is a valid ROS solution.

First, for all $i \neq i' \in [\ell+1]$: $(\vec{\rho}_i^*, (\gamma_i^*, \xi_i^*, m_i^*)) \neq (\vec{\rho}_{i'}^*, (\gamma_{i'}^*, \xi_{i'}^*, m_{i'}^*))$. Indeed, otherwise we would have $(m_i^*, R_i^*) = (m_{i'}^*, R_{i'}^*)$ and the game would have returned 0 earlier. Second, since the game returns 0 in line (I), we have $\sum_{j=1}^{\ell} \rho_{i,j}^* c_j \equiv_p c_i^* + \xi_i^*$ for all $i \in [\ell+1]$. Hence, to show that the ROS solution is valid, it is sufficient to show that for all $i \in [\ell+1]$, $c_i^* = \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_i^*, (\gamma_i^*, \xi_i^*, m_i^*)) - \xi_i^*$. This is clearly the case if $\mathsf{T}(R_i^*, m_i^*) = \bot$ when the adversary returns its forgeries. Indeed, in that case $(\gamma_i^*, \xi_i^*, \vec{\rho}_i^*) = (\gamma_i, \xi_i, \vec{\rho}_i)$ and

$$c_i^* = \mathsf{T}(R_i^*, m_i^*) = \mathsf{H}_{\text{ros}}(\vec{\rho}_i, (\gamma_i, \xi_i, m_i^*)) - \xi_i = \mathsf{H}_{\text{ros}}(\vec{\rho}_i^*, (\gamma_i^*, \xi_i^*, m_i^*)) - \xi_i^* \ .$$

Otherwise, $\mathsf{T}(R_i^*, m_i^*)$ was necessarily assigned during a call to H, and this call was of the form $\mathsf{H}(R^*_{i\,[\gamma_i^*, \xi_i^*, \vec{\rho}_i^*]}, m_i^*)$, which implies that $c_i^* = \mathsf{T}(R_i^*, m^*) = \mathsf{H}_{\text{ros}}(\vec{\rho}_i^*, (\gamma_i^*, \xi_i^*, m_i^*)) - \xi_i^*$. Hence,

$$\mathsf{Adv}^{\mathsf{game}_1}_{\mathcal{A}_{\text{alg}}}(\lambda) \geq \mathsf{Adv}^{\mathsf{game}_0}_{\mathcal{A}_{\text{alg}}}(\lambda) - \mathsf{Adv}^{\text{ros}}_{\ell, \mathcal{B}_{\text{ros}}}(\lambda) \ . \tag{11}$$

Moreover, it is easy to see that $\mathcal{B}_{\text{ros}}$ makes at most $q_{\text{h}} + \ell + 1$ queries to H_{ros} and runs in time at most $\tau + O(\ell + q_{\text{h}})$, assuming scalar multiplications in $\mathbb{G}$ and table assignments take unit time.

<u>REDUCTION TO OMDL</u>. In our last step, we construct an algorithm $\mathcal{B}_{\text{omdl}}$ solving the OMDL problem whenever $\mathcal{A}_{\text{alg}}$ wins Game_1. Algorithm $\mathcal{B}_{\text{omdl}}$, which has access to two oracles CHAL and DLOG (see Figure 1) takes as input a group description $(p, \mathbb{G}, G)$, makes a first query $X \leftarrow$ CHAL(), and runs $\mathcal{A}_{\text{alg}}$ on input $(p, \mathbb{G}, G, X)$, simulating Game_1 as follows (cf. comments in Figure 8). Each time $\mathcal{A}_{\text{alg}}$ makes a $\text{SIGN}_1()$ query, $\mathcal{B}_{\text{omdl}}$ queries its CHAL oracle to obtain R_j. It simulates $\text{SIGN}_2(j, c)$ without knowledge of x and r_j by querying $s_j \leftarrow \text{DLOG}(R_j + cX)$.

Assume that Game_1 returns 1, which implies that all forgeries (R_i^*, s_i^*) returned by $\mathcal{A}_{\text{alg}}$ are valid. We show how $\mathcal{B}_{\text{omdl}}$ solves OMDL. First, note that $\mathcal{B}_{\text{omdl}}$ made exactly ℓ calls to its oracle DLOG in total (since it makes exactly one call for each (valid) SIGN_2 query made by $\mathcal{A}_{\text{alg}}$).

Since Game_1 did not return 0 in line (I), there exists $i \in [\ell + 1]$ such that

$$\textstyle\sum_{j=1}^{\ell} \rho_{i,j}^* c_j \not\equiv_p c_i^* + \xi_i^* \ . \tag{12}$$

For all i, the adversary returned a representation $(\gamma_i^*, \xi_i^*, \vec{\rho}_i^*)$ of R_i^*, thus

$$R_i^* = \gamma_i^* G + \xi_i^* X + \textstyle\sum_{j=1}^{\ell} \rho_{i,j}^* R_j \ . \tag{13}$$

On the other hand, validity of the i-th forgery yields another representation: $R_i^* = s_i^* G + c_i^* X$. Combining these two, we get

$$(c_i^* + \xi_i^*) X + \textstyle\sum_{j=1}^{\ell} \rho_{i,j}^* R_j = (s_i^* - \gamma_i^*) G \ . \tag{14}$$

Finally, for each $j \in [\ell]$, s_j was computed with a call $s_j \leftarrow \text{DLOG}(R_j + c_j X)$, hence

$$R_j = s_j G - c_j X \ . \tag{15}$$

Injecting Eq. (15) in Eq. (14), we obtain

$$\Big(c_i^* + \xi_i^* - \textstyle\sum_{j=1}^{\ell} \rho_{i,j}^* c_j\Big) X = \Big(s_i^* - \gamma_i^* - \textstyle\sum_{j=1}^{\ell} \rho_{i,j}^* s_j\Big) G \ . \tag{16}$$

Since by Eq. (12) the coefficient in front of X is non-zero, this allows $\mathcal{B}_{\text{omdl}}$ to compute $x := \log X$. Furthermore, from Eq. (15) we have $r_j := \log R_j = s_j - c_j x$ for all $j \in [\ell]$. By returning $(x, r_1, \ldots, r_\ell)$, $\mathcal{B}_{\text{omdl}}$ solves the OMDL problem whenever $\mathcal{A}_{\text{alg}}$ wins Game_1, which implies

$$\mathsf{Adv}^{\text{omdl}}_{\mathsf{GrGen}, \mathcal{B}_{\text{omdl}}}(\lambda) = \mathsf{Adv}^{\mathsf{game}_1}_{\mathcal{A}_{\text{alg}}}(\lambda) \ . \tag{17}$$

The theorem now follows from Equations (10), (11) and (17). $\square$

$\underline{\mathsf{CBlSch.Sign}((p, \mathbb{G}, G, \mathsf{H}), x)}$		$\underline{\mathsf{CBlSch.User}(((p, \mathbb{G}, G, \mathsf{H}), X), m)}$
$r_0, r_1 \leftarrow\!\!\$\, \mathbb{Z}_p$		
$R_0 := r_0 G\,; R_1 := r_1 G$	$\xrightarrow{R_0, R_1}$	
		$\alpha_0, \beta_0, \alpha_1, \beta_1 \leftarrow\!\!\$\, \mathbb{Z}_p$
		$R'_0 := R_0 + \alpha_0 G + \beta_0 X$
		$R'_1 := R_1 + \alpha_1 G + \beta_1 X$
		$c'_0 := \mathsf{H}(R'_0, m)$
		$c'_1 := \mathsf{H}(R'_1, m)$
		$c_0 := c'_0 + \beta_0 \bmod p$
	$\xleftarrow{c_0, c_1}$	$c_1 := c'_1 + \beta_1 \bmod p$
$b \leftarrow\!\!\$\, \{0, 1\}$		
$s := r_b + c_b x \bmod p$	$\xrightarrow{b, s}$	
		if $sG \neq R_b + c_b X$ **then return** $\perp$
		$s' := s + \alpha_b \bmod p$
return 1		**return** $\sigma := (R'_b, s')$

Fig. 9. The clause blind Schnorr signing protocol.

5 The Clause Blind Schnorr Signature Scheme

We present a variation of the blind Schnorr signature scheme that only modifies the signing protocol. The scheme thus does not change the signatures themselves, meaning that it can be very smoothly integrated in existing applications.

The signature issuing protocol is changed so that it prevents the adversary from attacking the scheme by solving the ROS problem using Wagner's algorithm [Wag02, MS12]. The reason is that, as we show in Theorem 3, the attacker must now solve a *modified* ROS problem, which we define in Figure 10.

We start with explaining the modified signing protocol, formally defined in Figure 9. In the first round the signer and the user execute two parallel runs of the blind signing protocol from Figure 6, of which the signer only finishes one at random in the last round, that is, it finishes ($\mathrm{Run}_1 \vee \mathrm{Run}_2$): the clause from which the scheme takes its name.

This minor modification has major consequences. Recall that in the attack against the standard blind signature scheme from Section 4.2, the adversary opens ℓ signing sessions, receiving $R_1, \ldots, R_\ell$, then searches a solution $\vec{c}$ to the ROS problem and closes the signing sessions by sending $c_1, \ldots, c_\ell$. Our modified signing protocol prevents this attack, as now for every opened session the adversary must *guess* which of the two challenges the signer will reply to. Only if all its guesses are correct is the attack successful. As the attack only works for large values of ℓ, this probability vanishes exponentially.

In Theorem 3 we make this intuition formal; that is, we define a modified ROS game, which we show any successful attacker (which does not solve OMDL) must solve.

We have used two parallel executions of the basic protocol for the sake of simplicity, but the idea can be straightforwardly generalized to $t > 2$ parallel runs, of which the signer closes only one at random in the last round, that is, it closes ($\mathrm{Run}_1 \vee \ldots \vee \mathrm{Run}_t$). This decreases the probability that the user correctly guesses which challenges will be answered by the signer in ℓ concurrent sessions.

<u>The Modified ROS Problem.</u> Consider Figure 10. The difference to the original ROS problem (Figure 7) is that the queries to the $\mathsf{H}_{\mathrm{ros}}$ oracle consist of *two* vectors $\vec{\rho}_0, \vec{\rho}_1$ and additional *aux* information. Analogously, the adversary's task is to return $\ell + 1$ tuples $(\vec{\rho}_{i,0}, \vec{\rho}_{i,1}, aux_i)$,

Game $\mathrm{MROS}^{\mathcal{A}}_{\mathsf{GrGen},\ell,\Omega}(\lambda)$

$(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda)$
$\mathsf{T}_{\mathrm{ros}} := ()$
$(\vec{\rho}_{i,0}, \vec{\rho}_{i,1}, aux_i)_{i \in [\ell+1]} \leftarrow \mathcal{A}^{\mathsf{H}_{\mathrm{ros}}, \textsc{Select}}(p)$
$\quad /\!\!/ \ \vec{\rho}_{i,b} = (\rho_{i,b,1}, \dots, \rho_{i,b,\ell})$
return $\big(\forall\, i \neq i' : (\vec{\rho}_{i,0}, \vec{\rho}_{i,1}, aux_i) \neq (\vec{\rho}_{i',0}, \vec{\rho}_{i',1}, aux_{i'})$
$\quad \wedge\ \forall\, i \in [\ell+1] : \sum_{j=1}^{\ell} \rho_{i,b_j,j} c_j \equiv_p \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_{i,0}, \vec{\rho}_{i,1}, aux_i)$
$\quad \wedge\ \forall\, i \in [\ell+1] \forall\, j \in [\ell] : \rho_{i,1-b_j,j} = 0\big)$

Oracle $\mathsf{H}_{\mathrm{ros}}(\vec{\rho}_0, \vec{\rho}_1, aux)$

if $\mathsf{T}_{\mathrm{ros}}(\vec{\rho}_0, \vec{\rho}_1, aux) = \perp$ **then**
$\quad \mathsf{T}_{\mathrm{ros}}(\vec{\rho}_0, \vec{\rho}_1, aux) \leftarrow\!\!\$\ \mathbb{Z}_p$
return $\mathsf{T}_{\mathrm{ros}}(\vec{\rho}_0, \vec{\rho}_1, aux)$

Oracle $\textsc{Select}(j, c'_0, c'_1)$

$/\!\!/$ must be queried $\forall\, j \in [\ell]$
$b_j \leftarrow\!\!\$\ \{0,1\}\,;\ c_j := c'_{b_j}$
return b_j

Fig. 10. The modified ROS problem.

except that the ROS solution $c_1^*, \dots, c_\ell^*$ is selected as follows: for every index $j \in [\ell]$ the adversary must query an additional oracle $\textsc{Select}(j, c_{j,0}, c_{j,1})$, which flips a random bit b_j and sets the j-th coordinate of the solution to $c_j^* := c_{j,b_j}$.

Up to now, nothing really changed, as an adversary could always choose $\vec{\rho}_{i,0} = \vec{\rho}_{i,1}$ and $c_{j,0} = c_{j,1}$ for all indices, and solve the standard ROS problem. What complicates the task for the adversary considerably is the additional winning condition, which demands that in *all* tuples returned by the adversary, the ρ values that correspond to the complement of the selected bit must be zero, that is, for all $i \in [\ell+1]$ and all $j \in [\ell]$: $\rho_{i,1-b_j,j} = 0$. The adversary thus must commit to the solution coordinate c_j^* before it learns b_j, which then restricts the format of its ρ values.

We conjecture that the best attack against this modified ROS problem is to guess the ℓ bits b_j and to solve the standard ROS problem based on this guess using Wagner's algorithm. Hence, the complexity of the attack is increased by a factor 2^ℓ and requires time

$$O\big(2^\ell \cdot (\ell+1) 2^{\lambda/(1+\lfloor \lg(\ell+1) \rfloor)}\big)\ .$$

This estimated complexity is plotted for $\lambda \in \{256, 512\}$ in Figure 11. This should be compared to the standard Wagner attack with $\ell + 1 = 2^{\sqrt{\lambda}}$ running in time 2^{32} and 2^{45}, respectively, for the same values of the security parameter.

<u>Unforgeability of the Clause Blind Schnorr Signature Scheme.</u> We now prove that the Schnorr signature scheme from Figure 3, with the signing algorithm replaced by the protocol in Figure 9 is secure under the OMDL assumption for the underlying group and hardness of the modified ROS problem.

Theorem 3. *Let* GrGen *be a group generator. Let* $\mathcal{A}_{\mathrm{alg}}$ *be an algebraic adversary against the UNF security of the clause blind Schnorr signature scheme* $\mathsf{CBlSch}[\mathsf{GrGen}]$ *running in time at most* τ *and making at most* q_{s} *queries to* $\textsc{Sign}_1$ *and* q_{h} *queries to the random oracle. Then there exist an algorithm* $\mathcal{B}_{\mathrm{mros}}$ *for the* $\mathrm{MROS}_{q_{\mathrm{s}}}$ *problem making at most* $q_{\mathrm{h}} + q_{\mathrm{s}} + 1$ *random oracle queries and an algorithm* $\mathcal{B}_{\mathrm{omdl}}$ *for the OMDL problem w.r.t.* GrGen *making at most* q_{s} *queries to its oracle* $\textsc{DLog}$*, both running in time at most* $\tau + O(q_{\mathrm{s}} + q_{\mathrm{h}})$*, such that*

$$\mathsf{Adv}^{\mathrm{unf}}_{\mathsf{BlSch}[\mathsf{GrGen}],\mathcal{A}_{\mathrm{alg}}}(\lambda) \leq \mathsf{Adv}^{\mathrm{omdl}}_{\mathsf{GrGen},\mathcal{B}_{\mathrm{omdl}}}(\lambda) + \mathsf{Adv}^{\mathrm{mros}}_{\ell,\mathcal{B}_{\mathrm{mros}}}(\lambda)\ .$$

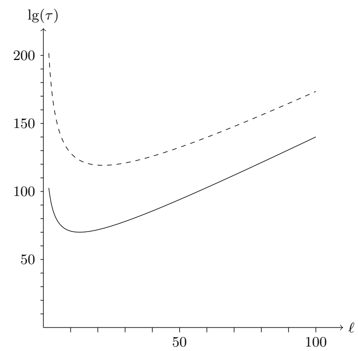

Fig. 11. Estimated complexity τ of conjectured best attack against the modified ROS problem as a function of parameter ℓ for $\lambda = 256$ (solid line) and $\lambda = 512$ (dashed line).

The theorem follows by adapting the proof of Theorem 2; we therefore discuss the changes and refer to Figure 12, which compactly presents all the details.

The proof again proceeds by one game hop, where an adversary behaving differently in the two games is used to break the modified ROS problem; the only change to the proof of Theorem 2 is that when simulating Sign_2, the reduction $\mathcal{B}_{\text{mros}}$ calls $\text{Select}(j, c_{j,0}, c_{j,1})$ to obtain bit b instead of choosing it itself. By definition, Game_1 aborts in line (I) if and only if $\mathcal{B}_{\text{mros}}$ has found a solution for MROS.

The difference in the reduction to OMDL of the modified game is that the adversary can fail to solve MROS in two ways: (1) its values $((\rho_{i,b_j,j})_{i,j}, (c_j)_j)$ are not a ROS solution; in this case the reduction can solve OMDL as in the proof of Theorem 2; (2) these values *are* a ROS solution, but for some i, j, we have $\rho_{i,1-b_j,j} \neq 0$. We show that in this case the OMDL reduction can compute the discrete logarithm of one of the values $R_{j,1-b_j}$.

More in detail, the main difference to Theorem 2 is that the representation of the values R_i^* in the adversary's forgery depend on both the $R_{j,0}$ and the $R_{j,1}$ values; we can thus write them as

$$R_i^* = \gamma_i^* G + \xi_i^* X + \textstyle\sum_{j=1}^{\ell} \rho_{i,b_j,j}^* R_{j,b_j} + \sum_{j=1}^{\ell} \rho_{i,1-b_j,j}^* R_{j,1-b_j}$$

(this corresponds to Eq. (13) in the proof of Theorem 2). Validity of the forgery implies $R_i^* = s_i^* G - c_i^* X$, which together with the above yields

$$(c_i^* + \xi_i^*) X + \textstyle\sum_{j=1}^{\ell} \rho_{i,b_j,j}^* R_{j,b_j} = (s_i^* - \gamma_i^*) G - \sum_{j=1}^{\ell} \rho_{i,1-b_j,j}^* R_{j,1-b_j}$$

(cf. Eq. (14)). By definition of s_j, we have $R_{j,b_j} = s_j G - c_j X$ for all $j \in [\ell]$; the above equation becomes thus

$$\big(c_i^* + \xi_i^* - \textstyle\sum_{j=1}^{\ell} \rho_{i,b_j,j}^* c_j\big) X = \big(s_i^* - \gamma_i^* - \sum_{j=1}^{\ell} \rho_{i,b_j,j}^* s_j\big) G - \sum_{j=1}^{\ell} \rho_{i,1-b_j,j}^* R_{j,1-b_j} \qquad (18)$$

(which corresponds to Eq. (16) in Theorem 2). In Theorem 2, not solving ROS implied that for some i, the coefficient of X in the above equation was non-zero, which allowed computation of $\log X$.

However, if the adversary sets all these coefficients to 0, it could still fail to solve MROS if $\rho_{i^*,1-b_{j^*},j^*}^* \neq 0$ for some i^*, j^* (this is case (2) defined above). In this case Game_1 does not abort

$\mathsf{Game}_0 \ \big(\mathrm{UNF}^{\mathcal{A}_{\mathrm{alg}}}_{\mathsf{CBlSch[GrGen]}}(\lambda)\big)$, $\boxed{\mathsf{Game}_1}$

$(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda)$
$x \leftarrow\!\!\$\, \mathbb{Z}_p\,;\ X \coloneqq xG$
$k_1 \coloneqq 0\,;\ k_2 \coloneqq 0\,;\ \mathcal{S} \coloneqq \emptyset\,;\ \mathsf{T} \coloneqq (\,)\,;\ \boxed{\mathsf{U} \coloneqq (\,)}$
$(m_i^*, (R^*_{i\,[\gamma_i,\xi_i,\vec{\rho}_{i,0},\vec{\rho}_{i,1}]}, s_i^*))_{i\in[\ell+1]}$
$\qquad \leftarrow \mathcal{A}_{\mathrm{alg}}^{\mathsf{H},\textsc{Sign}_1,\textsc{Sign}_2}(p, \mathbb{G}, G, X)$
// $R_i^* = \gamma_i G + \xi_i X + \sum \rho_{i,0,j} R_{j,0} + \sum \rho_{i,1,j} R_{j,1}$
if $k_2 > \ell$ **then return** 0
if $\exists\, i \neq i' \in [\ell+1] : (m_i^*, R_i^*) = (m_{i'}^*, R_{i'}^*)$
  then return 0

$\boxed{\text{Game}_1}$:
for $i = 1 \ldots \ell+1$ **do**
  if $\mathsf{T}(R_i^*, m_i^*) = \bot$ **then**
    $\mathsf{T}(R_i^*, m_i^*) \leftarrow\!\!\$\, \mathbb{Z}_p$
    // $\mathsf{T}(R_i^*, m_i^*) \coloneqq \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_{i,0}, \vec{\rho}_{i,1}, (\gamma_i, \xi_i, m_i^*)) - \xi_i$
    $\mathsf{U}(R_i^*, m_i^*) \coloneqq (\gamma_i, \xi_i, \vec{\rho}_{i,0}, \vec{\rho}_{i,1})$

for $i = 1 \ldots \ell+1$ **do**
  $c_i^* \coloneqq \widetilde{\mathsf{H}}(R_i^*, m_i^*)$ // does not modify T in Game_1

$\boxed{\text{Game}_1}$:
  $(\gamma_i^*, \xi_i^*, \vec{\rho}^{\,*}_{i,0}, \vec{\rho}^{\,*}_{i,1}) \coloneqq \mathsf{U}(R_i^*, m_i^*)$
if $\forall\, i \in [\ell+1] : \sum_{j=1}^{\ell} \rho^*_{i,b_j,j} c_j \equiv_p c_i^* + \xi_i^*$
  $\wedge\, \forall\, i \in [\ell+1], \forall\, j \in [\ell] : \rho^*_{i,1-b_j,j} = 0$
  then return 0 (I)
  // $((\vec{\rho}^{\,*}_{i,0}, \vec{\rho}^{\,*}_{i,1}, (\gamma_i^*, \xi_i^*, m_i^*))_{i\in[\ell+1]})$ solves MROS

return $(\forall\, i \in [\ell+1] : s_i^* G = R_i^* + c_i^* X)$

// $\left\{\begin{array}{l}
\varphi_i \coloneqq s_i^* - \gamma_i^* - \sum_{j=1}^{\ell} \rho^*_{i,b_j,j}\, s_j \\
\textbf{if } \chi_i \coloneqq c_i^* + \xi_i^* - \sum_{j=1}^{\ell} \rho^*_{i,b_j,j}\, c_j \not\equiv_p 0 \\
\quad x \coloneqq \chi_i^{-1} \varphi_i \bmod p \\
\quad \textbf{for } j \in [\ell] : r_{j,1-b_j} \leftarrow \textsc{DLog}(R_{j,1-b_j}) \\
\textbf{else if } \psi \coloneqq \rho^*_{i,1-b_{\hat{\jmath}},\hat{\jmath}} \neq 0 \text{ for some } i, \hat{\jmath} \\
\quad \textbf{for } j \neq \hat{\jmath} : r_{j,1-b_j} \leftarrow \textsc{DLog}(R_{j,1-b_j}) \\
\quad r_{\hat{\jmath},1-b_{\hat{\jmath}}} \coloneqq \psi^{-1}(\varphi_i - \sum_{j\neq\hat{\jmath}} \rho^*_{i,1-b_j,j}\, r_{j,1-b_j}) \\
\quad x \leftarrow \textsc{DLog}(X) \\
\textbf{for } j \in [\ell] : r_{j,b_j} \coloneqq s_j - c_j x \\
(x, r_{1,0}, \ldots, r_{\ell,0}, r_{1,1}, \ldots, r_{\ell,1}) \text{ solves OMDL}
\end{array}\right.$

Oracle $\widetilde{\mathsf{H}}(R, m)$

if $\mathsf{T}(R, m) = \bot$ **then**
  $\mathsf{T}(R, m) \leftarrow\!\!\$\, \mathbb{Z}_p$
return $\mathsf{T}(R, m)$

Oracle $\mathsf{H}(R_{[\gamma,\xi,\vec{\rho}_0,\vec{\rho}_1]}, m)$

// $R = \gamma G + \xi X + \sum \rho_{0,j} R_{j,0} + \sum \rho_{1,j} R_{j,1}$
if $\mathsf{T}(R, m) = \bot$ **then**
  $\mathsf{T}(R, m) \leftarrow\!\!\$\, \mathbb{Z}_p$
  // $\mathsf{T}(R, m) \coloneqq \mathsf{H}_{\mathrm{ros}}(\vec{\rho}_0, \vec{\rho}_1, (\gamma, \xi, m)) - \xi$
  $\boxed{\mathsf{U}(R, m) \coloneqq (\gamma, \xi, \vec{\rho}_0, \vec{\rho}_1)}$
return $\mathsf{T}(R, m)$

Oracle $\textsc{Sign}_1()$

$k_1 \coloneqq k_1 + 1$
$r_{k_1,0}, r_{k_1,1} \leftarrow\!\!\$\, \mathbb{Z}_p$
$R_{k_1,0} \coloneqq r_{k_1,0} G$ // $R_{k_1,0} \leftarrow \textsc{Chal}()$
$R_{k_1,1} \coloneqq r_{k_1,1} G$ // $R_{k_1,1} \leftarrow \textsc{Chal}()$
$\mathcal{S} \coloneqq \mathcal{S} \cup \{k_1\}$
return $(k_1, R_{k_1,0}, R_{k_1,1})$

Oracle $\textsc{Sign}_2(j, c_{j,0}, c_{j,1})$

if $j \notin \mathcal{S}$ **then return** $\bot$
$b_j \leftarrow\!\!\$\, \{0,1\}$
  // $b_j \leftarrow \textsc{Select}(j, c_{j,0}, c_{j,1})$
$c_j \coloneqq c_{j,b_j}$
$s_j \coloneqq r_{j,b_j} + c_j x$
  // $s_j \leftarrow \textsc{DLog}(R_{j,b_j} + c_j X)$
$\mathcal{S} \coloneqq \mathcal{S} \setminus \{j\}$
$k_2 \coloneqq k_2 + 1$
return (b_j, s_j)

Fig. 12. Games used in the proof of Theorem 3. Game_0 is the unforgeability game for the clause blind Schnorr signature scheme in the ROM for an algebraic adversary $\mathcal{A}_{\mathrm{alg}}$. The comments in light gray show how $\mathcal{B}_{\mathrm{mros}}$ solves MROS; the dark comments show how $\mathcal{B}_{\mathrm{omdl}}$ solves OMDL.

and the OMDL reduction $\mathcal{B}_{\text{omdl}}$ must succeed. Since in this case the left-hand side of Eq. (18) is then 0, $\mathcal{B}_{\text{omdl}}$ can, after querying $\text{DLOG}(R_{j,1-b_j})$ for all $j \neq j^*$, compute $\text{DLOG}(R_{j^*,1-b_{j^*}})$, which breaks OMDL.

We finally note that the above case distinction was merely didactic, as the same OMDL reduction can handle both cases simultaneously, which means that our reduction does not introduce any additional security loss. In particular, the reduction obtains X and all values $(R_{j,0}, R_{j,1})$ from its OMDL challenger, then handles case (2) as described, and case (1) by querying $R_{1,1-b_1}, \ldots, R_{\ell,1-b_\ell}$ to its DLOG oracle. In both cases it made 2ℓ queries to DLOG and computed the discrete logarithms of all $2\ell + 1$ challenges.

Figure 12 presents the unforgeability game and Game_1, which aborts if the adversary solved MROS. The gray and dark gray comments also precisely define how a reduction $\mathcal{B}_{\text{mros}}$ solves MROS whenever Game_1 aborts in line (I), and how a reduction $\mathcal{B}_{\text{omdl}}$ solves OMDL whenever $\mathcal{A}_{\text{alg}}$ wins Game_1.

BLINDNESS OF CLAUSE BLIND SCHNORR SIGNATURES. Blindness of the "clause" variant in Figure 9 follows via a hybrid argument from blindness of the standard scheme (Figure 6). In the game defining blindness (see Figure 20 in Appendix C), the adversary impersonates a signer and selects two messages m_0 and m_1. The game flips a bit b, runs the signing protocol with the adversary for m_b and then for m_{1-b}. If both sessions terminate, the adversary is given the resulting signatures and must determine b.

In the blindness game for scheme CBlSch, the challenger runs *two* instances of the issuing protocol from BlSch for m_b of which the signer finishes one, as determined by its message (β_b, s_b) in the third round (β_b corresponds to b in Figure 9), and then *two* instances for m_{1-b}.

If $b = 0$, the challenger thus asks the adversary for signatures on m_0, m_0, m_1 and then m_1. We define a hybrid game where the order of the messages is $\underline{m_1}, m_0, \underline{m_0}, m_1$; this game thus lies between the blindness games for $b = 0$ and $b = 1$, where the messages are m_1, m_1, m_0, m_0. The original games differ from the hybrid game by exactly one message pair; intuitively, they are thus indistinguishable by blindness of BlSch.

A technical detail is that the above argument only works when $\beta_0 = \beta_1$, as otherwise in the reduction to BlSch blindness, both reductions (between each original game and the hybrid game) abort one session and do not get any signatures from its challenger. The reductions thus guess the values β_0 and β_1 (and return a random bit if the guess turns out wrong). The hybrid game then replaces the β_0-th message of the first two and the β_1-th of the last two (as opposed to the ones underlined as above). Following this argument, in Appendix C we prove the following:

Theorem 4. *Let $\mathcal{A}$ be a p.p.t. adversary against blindness of the scheme* CBlSch*. Then there exist two p.p.t. algorithms $\mathcal{B}_1$ and $\mathcal{B}_2$ against blindness of* BlSch *such that*

$$\mathsf{Adv}^{\text{blind}}_{\mathsf{CBlSch},\mathcal{A}}(\lambda) \leq 4 \cdot \left(\mathsf{Adv}^{\text{blind}}_{\mathsf{BlSch},\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\text{blind}}_{\mathsf{BlSch},\mathcal{B}_2}(\lambda)\right) .$$

Since the (standard) blind Schnorr signature scheme is perfectly blind [CP93], by the above, our variant also satisfies perfect blindness.

6 Schnorr-Signed ElGamal Encryption

A public key for the ElGamal public-key encryption (PKE) scheme is a group element $Y \in \mathbb{G}$. Messages are group elements $M \in \mathbb{G}$ and to encrypt M under Y, one samples a random

$$
\begin{array}{l}
\underline{\text{Game } \mathrm{DDH}^{\mathcal{A}}_{\mathsf{GrGen}}(\lambda)} \\
(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda)\,;\ b \leftarrow\!\!\$\ \{0,1\}\,;\ x, y, z \leftarrow\!\!\$\ \mathbb{Z}_p \\
X := xG\,;\ Y := yG\,;\ Z_0 := xyG\,;\ Z_1 := zG \\
b' \leftarrow \mathcal{A}(p, \mathbb{G}, G, X, Y, Z_b) \\
\textbf{return } (b = b')
\end{array}
$$

Fig. 13. The DDH problem.

$x \in \mathbb{Z}_p$ and derives an ephemeral key $K := xY$ to blind the message: $C := xY + M$. Given in addition the value $X := xG$, the receiver that holds $y = \log Y$ can derive $K := yX$ and recover $M := C - K$.

Under the decisional Diffie-Hellman (DDH) assumption (see Figure 13), ciphertexts of different messages are computationally indistinguishable: replacing K by a random value K' makes the ciphertext C perfectly hide the message. In the AGM, ElGamal, viewed as a key-encapsulation mechanism (KEM) was shown to satisfy CCA1-security (where the adversary can only make decryption queries before seeing the challenge key) under a parametrized variant of DDH [FKL18].

The idea of *Schnorr-signed* ElGamal is to accompany the ciphertext by a proof of knowledge of the randomness $x = \log X$ used to encrypt, in particular, a Schnorr signature on the pair (X, C) under the public key X. The scheme is detailed in Figure 14. (Note that we changed the argument order in the hash function call compared to Section 3 so that it is the same as in ciphertexts.)

The strongest security notion for PKE is indistinguishability of ciphertexts under adaptive chosen-ciphertext attack (IND-CCA2), where the adversary can query decryptions of ciphertexts of its choice even after receiving the challenge. The (decisional) game IND-CCA2 is defined in Figure 15.

When ephemeral keys are hashed (that is, defined as $k := \mathsf{H}'(xY)$) and the scheme is viewed as a KEM, then CCA2-security can be reduced to the *strong* Diffie-Hellman (SDH)

$$
\begin{array}{ll}
\underline{\mathsf{SEG.Setup}(\lambda)} & \underline{\mathsf{SEG.KeyGen}(par)} \\
(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda) & (p, \mathbb{G}, G, \mathsf{H}) := par\,;\ y \leftarrow\!\!\$\ \mathbb{Z}_p\,;\ Y := yG \\
\text{Select } \mathsf{H}\colon \{0,1\}^* \to \mathbb{Z}_p & sk := (par, y)\,;\ pk := (par, Y) \\
\textbf{return } par := (p, \mathbb{G}, G, \mathsf{H}) & \textbf{return } (sk, pk) \\
 & \\
\underline{\mathsf{SEG.Enc}(pk, M)} & \underline{\mathsf{SEG.Dec}(sk, (X, C, R, s))} \\
(p, \mathbb{G}, G, \mathsf{H}, Y) := pk\,;\ x, r \leftarrow\!\!\$\ \mathbb{Z}_p & (p, \mathbb{G}, G, \mathsf{H}, y) := sk \\
X := xG\,;\ R := rG\,;\ C := xY + M & \textbf{if } sG \neq R + \mathsf{H}(X, C, R) \cdot X \textbf{ then} \\
s := r + \mathsf{H}(X, C, R) \cdot x \bmod p & \quad \textbf{return } \perp \\
\textbf{return } (X, C, R, s) & \textbf{return } M := C - yX
\end{array}
$$

Fig. 14. The Schnorr-Signed ElGamal PKE scheme $\mathsf{SEG}[\mathsf{GrGen}]$.

Game $\text{IND-CCA2}^{\mathcal{A}}_{\mathsf{PKE}}(\lambda)$	Oracle $\textsc{Enc}(m_0, m_1)$ // one time
$par \leftarrow \mathsf{PKE.Setup}(\lambda)$	$c^* \leftarrow \mathsf{PKE.Enc}(pk, m_b)$
$(pk, sk) \leftarrow \mathsf{PKE.KeyGen}(par)$	**return** c^*
$b \leftarrow_\$ \{0,1\}$	
$b' \leftarrow \mathcal{A}^{\textsc{Enc},\textsc{Dec}}(pk)$	Oracle $\textsc{Dec}(c)$
return $(b = b')$	**if** $c = c^*$ **then return** $\perp$
	return $\mathsf{PKE.Dec}(sk, c)$

Fig. 15. The IND-CCA2 security game for a PKE scheme PKE.

assumption[10] [ABR01, CS03] in the ROM. In Appendix B we show that when key hashing is applied to the Schnorr-signed ElGamal scheme from Figure 14, then in the AGM+ROM we can directly reduce CCA2 security of the corresponding KEM to the DL assumption (Figure 1); in particular, we do so using a *tight* security proof (note that SDH is equivalent to DL in the AGM [FKL18] but the reduction from DL to SDH is non-tight). Here we prove that the Schnorr-signed ElGamal PKE is IND-CCA2-secure in the AGM+ROM under the DDH assumption.

Theorem 5. *Let* GrGen *be a group generator. Let* $\mathcal{A}_{\text{alg}}$ *be an algebraic adversary against the IND-CCA2 security of the Schnorr-signed ElGamal PKE scheme* $\mathsf{SEG}[\mathsf{GrGen}]$ *making at most* q_{d} *decryption queries and* q_{h} *queries to the random oracle. Then there exist two algorithms* $\mathcal{B}_1$ *and* $\mathcal{B}_2$ *solving respectively the DL problem and the DDH problem w.r.t.* GrGen*, such that*

$$\mathsf{Adv}^{\text{ind-cca2}}_{\mathsf{SEG}[\mathsf{GrGen}],\mathcal{A}_{\text{alg}}}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\text{ddh}}_{\mathsf{GrGen},\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{\text{dl}}_{\mathsf{GrGen},\mathcal{B}_1}(\lambda) + \frac{q_{\text{d}} + \frac{1}{2^{\lambda-1}}(q_{\text{d}} + q_{\text{h}})}{2^{\lambda-1}} \ .$$

We start with the proof idea. The full proof can be found in Appendix A. Let Y be the public key, let P_0 and P_1 denote the challenge plaintexts, and let $(X^* = x^*G, C^* = x^*Y + P_b, R^*, s^*)$ be the challenge ciphertext. Under the DDH assumption, given Y and X^*, the value x^*Y looks random. We can thus replace x^*Y by a random group element Z^*, which perfectly hides P_b and leads to a game where the adversary gains no information about the challenge bit b.

It remains to show how the reduction can simulate the game without knowledge of $\log X^*$ (needed to sign the challenge ciphertext) and $\log Y$ (needed to answer decryption queries). The Schnorr signature under X^* contained in the challenge ciphertext can be simulated by programming the random oracle H as for Theorem 1.

Decryption queries leverage the fact that the Schnorr signature contained in a queried ciphertext (X, C, R, s) proves knowledge of x with $X = xG$. Thus, intuitively, the reduction should be able to answer a query by extracting x and returning $M = C - xY$. However, this extraction is a lot trickier than in the proof of Theorem 1: During the game the adversary obtains group elements Y, X^*, C^*, and R^*, as well as the answers $M_1, \ldots, M_{q_{\text{d}}}$ to its queries to $\textsc{Dec}$. The adversary's representations of group elements can thus depend on all these elements. In particular, since $\textsc{Dec}$ on input $(X, C, \ldots)$ computes $M \coloneqq C - yX$, by successive calls to $\textsc{Dec}$, the adversary can obtain arbitrary powers of y.

[10] SDH states that given $X = xG$ and Y it is infeasible to compute xY even when given access to an oracle which on input (Y', Z') returns 1 if $Z' = xY'$ and 0 otherwise.

In our proof we first show that from a representation given by the adversary, we can always (efficiently) derive a representation in basis

$$(G, X^*, Y = yG, \ldots, y^{q_{\mathrm{d}}+1}G, x^*yG, \ldots, x^*y^{q_{\mathrm{d}}+1}G) \ .$$

Now consider a decryption query (X, C, R, s), each group element represented as

$$X = \gamma_x G + \xi_x X^* + \textstyle\sum_{i=1}^{q_d+1} \upsilon_x^{(i)} y^i G + \sum_{i=1}^{q_d+1} \zeta_x^{(i)} x^* y^i G \ , \qquad R = \gamma_r G + \ldots \tag{19}$$

We show that each query falls into one of three categories:
(1) The choice of $c = \mathsf{H}(X, C, R)$ was unlucky, which only happens with negligible probability (this corresponds to an abort in line (I) in Figure 16 in Appendix A).
(2) The representation of X is independent of Y, that is, $X = \gamma_x G + \xi_x X^*$. Then xY (and hence the answer $M = C - xY$ to the query) can be computed as $xY := \gamma_x Y + \xi_x Z^*$ (where $Z^* := x^* Y$ is known by the reduction).
(3) Otherwise we show that the adversary has actually computed $\log Y$ (corresponding to an abort in line (III) in Figure 16): If the Dec query was valid then $sG = R + cX$, which, by plugging in the representations (19) yields

$$0 = (\gamma_r + c\gamma_x - s)G + (\xi_r + c\xi_x)X^* + \sum_{i=1}^{q_d+1} \big(\underbrace{(\upsilon_r^{(i)} + x^*\zeta_r^{(i)}) + c\,\overbrace{(\upsilon_x^{(i)} + x^*\zeta_x^{(i)})}^{=:\beta^{(i)}}}_{=:\alpha^{(i)}}\big) y^i G$$

If $\beta^{(i)} \equiv_p 0$ for all i, we are in case (2). If $\beta^{(j)} \not\equiv_p 0$ for some j and $\alpha^{(i)} \equiv_p 0$ for all i, then $c \equiv_p -(\upsilon_r^{(j)} + x^*\zeta_r^{(j)}) \cdot (\beta^{(j)})^{-1}$ was an unlucky choice (made *after* the adversary chose its representations from (19)) (case (1)). Otherwise $\alpha^{(j)} \equiv_p 0$ for some j and

$$0 = \gamma_r + c\gamma_x - s + (\xi_r + c\xi_x)x^* + \textstyle\sum_{i=1}^{q_d+1} \alpha^{(i)} y^i$$

can be solved for y. (Note that the reduction to DL chooses x^* itself.)

Acknowledgements. The first author is supported by the Vienna Science and Technology Fund (WWTF) through project VRG18-002. Parts of this work were done during a visit to the Simons Institute for the Theory of Computing while at Inria. This work is funded in part by the MSR–Inria Joint Centre.

References

[Abe01] Masayuki Abe. A secure three-move blind signature scheme for polynomially many signatures. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 136–151. Springer, Heidelberg, May 2001.

[ABM15] Michel Abdalla, Fabrice Benhamouda, and Philip MacKenzie. Security of the J-PAKE password-authenticated key exchange protocol. In *2015 IEEE Symposium on Security and Privacy*, pages 571–587. IEEE Computer Society Press, May 2015.

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The oracle Diffie-Hellman assumptions and an analysis of DHIES. In David Naccache, editor, *CT-RSA 2001*, volume 2020 of *LNCS*, pages 143–158. Springer, Heidelberg, April 2001.

[BCC04] Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *ACM CCS 2004*, pages 132–145. ACM Press, October 2004.

[BCC+09] Mira Belenkiy, Jan Camenisch, Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Hovav Shacham. Randomizable proofs and delegatable anonymous credentials. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 108–125. Springer, Heidelberg, August 2009.

[BDL+12] Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang. High-speed high-security signatures. *Journal of Cryptographic Engineering*, 2(2):77–89, September 2012.

[BDN18] Dan Boneh, Manu Drijvers, and Gregory Neven. Compact multi-signatures for smaller blockchains. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part II*, volume 11273 of *LNCS*, pages 435–464. Springer, Heidelberg, December 2018.

[BFPV13] Olivier Blazy, Georg Fuchsbauer, David Pointcheval, and Damien Vergnaud. Short blind signatures. *Journal of Computer Security*, 21(5):627–661, 2013.

[BFW16] David Bernhard, Marc Fischlin, and Bogdan Warinschi. On the hardness of proving CCA-security of signed ElGamal. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part I*, volume 9614 of *LNCS*, pages 47–69. Springer, Heidelberg, March 2016.

[BL13a] Foteini Baldimtsi and Anna Lysyanskaya. Anonymous credentials light. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 1087–1098. ACM Press, November 2013.

[BL13b] Foteini Baldimtsi and Anna Lysyanskaya. On the security of one-witness blind signature schemes. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 82–99. Springer, Heidelberg, December 2013.

[BLOR20] Fabrice Benhamouda, Tancrède Lepoint, Michele Orrù, and Mariana Raykova. On the (in)security of ROS. Cryptology ePrint Archive, Report 2020/945, 2020.

[BLS04] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. *Journal of Cryptology*, 17(4):297–319, September 2004.

[BNPS03] Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme. *Journal of Cryptology*, 16(3):185–215, June 2003.

[BNW17] David Bernhard, Ngoc Khanh Nguyen, and Bogdan Warinschi. Adaptive proofs have straightline extractors (in the random oracle model). In Dieter Gollmann, Atsuko Miyaji, and Hiroaki Kikuchi, editors, *ACNS 17*, volume 10355 of *LNCS*, pages 336–353. Springer, Heidelberg, July 2017.

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 31–46. Springer, Heidelberg, January 2003.

[BP02] Mihir Bellare and Adriana Palacio. GQ and Schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 162–177. Springer, Heidelberg, August 2002.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby, editors, *ACM CCS 93*, pages 62–73. ACM Press, November 1993.

[BR95] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In Alfredo De Santis, editor, *EUROCRYPT'94*, volume 950 of *LNCS*, pages 92–111. Springer, Heidelberg, May 1995.

[Bra94] Stefan Brands. Untraceable off-line cash in wallets with observers (extended abstract). In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 302–318. Springer, Heidelberg, August 1994.

[CFN90] David Chaum, Amos Fiat, and Moni Naor. Untraceable electronic cash. In Shafi Goldwasser, editor, *CRYPTO'88*, volume 403 of *LNCS*, pages 319–327. Springer, Heidelberg, August 1990.

[Cha82] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO'82*, pages 199–203. Plenum Press, New York, USA, 1982.

[CHL05] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 302–321. Springer, Heidelberg, May 2005.

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 93–118. Springer, Heidelberg, May 2001.

[CP93] David Chaum and Torben P. Pedersen. Wallet databases with observers. In Ernest F. Brickell, editor, *CRYPTO'92*, volume 740 of *LNCS*, pages 89–105. Springer, Heidelberg, August 1993.

[CS03] Ronald Cramer and Victor Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[FHS15] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 233–253. Springer, Heidelberg, August 2015.

[FJS19] Nils Fleischhacker, Tibor Jager, and Dominique Schröder. On tight security proofs for Schnorr signatures. *Journal of Cryptology*, 32(2):566–599, April 2019.

[FKL18] Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 33–62. Springer, Heidelberg, August 2018.

[FOO93] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A practical secret voting scheme for large scale elections. In Jennifer Seberry and Yuliang Zheng, editors, *AUSCRYPT'92*, volume 718 of *LNCS*, pages 244–251. Springer, Heidelberg, December 1993.

[FPV09] Georg Fuchsbauer, David Pointcheval, and Damien Vergnaud. Transferable constant-size fair e-cash. In Juan A. Garay, Atsuko Miyaji, and Akira Otsuka, editors, *CANS 09*, volume 5888 of *LNCS*, pages 226–247. Springer, Heidelberg, December 2009.

[FS10] Marc Fischlin and Dominique Schröder. On the impossibility of three-move blind signature schemes. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 197–215. Springer, Heidelberg, May / June 2010.

[Fuc11] Georg Fuchsbauer. Commuting signatures and verifiable encryption. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 224–245. Springer, Heidelberg, May 2011.

[GBL08] Sanjam Garg, Raghav Bhaskar, and Satyanarayana V. Lokam. Improved bounds on security reductions for discrete log based signatures. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 93–107. Springer, Heidelberg, August 2008.

[GG14] Sanjam Garg and Divya Gupta. Efficient round optimal blind signatures. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 477–495. Springer, Heidelberg, May 2014.

[GRS+11] Sanjam Garg, Vanishree Rao, Amit Sahai, Dominique Schröder, and Dominique Unruh. Round optimal blind signatures. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 630–648. Springer, Heidelberg, August 2011.

[HHK10] Javier Herranz, Dennis Hofheinz, and Eike Kiltz. Some (in)sufficient conditions for secure hybrid encryption. *Inf. Comput.*, 208(11):1243–1257, 2010.

[HKL19] Eduard Hauck, Eike Kiltz, and Julian Loss. A modular treatment of blind signatures from identification schemes. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 345–375. Springer, Heidelberg, May 2019.

[Jak98] Markus Jakobsson. A practical mix. In Kaisa Nyberg, editor, *EUROCRYPT'98*, volume 1403 of *LNCS*, pages 448–461. Springer, Heidelberg, May / June 1998.

[MPSW19] Gregory Maxwell, Andrew Poelstra, Yannick Seurin, and Pieter Wuille. Simple Schnorr multi-signatures with applications to Bitcoin. *Des. Codes Cryptogr.*, 87(9):2139–2164, 2019.

[MS12] Lorenz Minder and Alistair Sinclair. The extended k-tree algorithm. *Journal of Cryptology*, 25(2):349–382, April 2012.

[Nec94] V. I. Nechaev. Complexity of a determinate algorithm for the discrete logarithm. *Mathematical Notes*, 55(2):165–172, 1994.

[Nic19] Jonas Nick. Blind signatures in scriptless scripts. Presentation given at *Building on Bitcoin* 2019, 2019. Slides and video available at https://jonasnick.github.io/blog/2018/07/31/blind-signatures-in-scriptless-scripts/.

[NS15] Ivica Nikolic and Yu Sasaki. Refinements of the k-tree algorithm for the generalized birthday problem. In Tetsu Iwata and Jung Hee Cheon, editors, *ASIACRYPT 2015, Part II*, volume 9453 of *LNCS*, pages 683–703. Springer, Heidelberg, November / December 2015.

[OO92] Tatsuaki Okamoto and Kazuo Ohta. Universal electronic cash. In Joan Feigenbaum, editor, *CRYPTO'91*, volume 576 of *LNCS*, pages 324–337. Springer, Heidelberg, August 1992.

[Pas11] Rafael Pass. Limits of provable security from standard assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *43rd ACM STOC*, pages 109–118. ACM Press, June 2011.

[PS96a] David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT'96*, volume 1163 of *LNCS*, pages 252–265. Springer, Heidelberg, November 1996.

[PS96b] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 387–398. Springer, Heidelberg, May 1996.

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

[PV05] Pascal Paillier and Damien Vergnaud. Discrete-log-based signatures may not be equivalent to discrete log. In Bimal K. Roy, editor, *ASIACRYPT 2005*, volume 3788 of *LNCS*, pages 1–20. Springer, Heidelberg, December 2005.

[Sch90] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *CRYPTO'89*, volume 435 of *LNCS*, pages 239–252. Springer, Heidelberg, August 1990.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, January 1991.

[Sch01] Claus-Peter Schnorr. Security of blind discrete log signatures against interactive attacks. In Sihan Qing, Tatsuaki Okamoto, and Jianying Zhou, editors, *ICICS 01*, volume 2229 of *LNCS*, pages 1–12. Springer, Heidelberg, November 2001.

[Seu12] Yannick Seurin. On the exact security of Schnorr-type signatures in the random oracle model. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 554–571. Springer, Heidelberg, April 2012.

[SG02] Victor Shoup and Rosario Gennaro. Securing threshold cryptosystems against chosen ciphertext attack. *Journal of Cryptology*, 15(2):75–96, March 2002.

[Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *EUROCRYPT'97*, volume 1233 of *LNCS*, pages 256–266. Springer, Heidelberg, May 1997.

[SJ99] Claus-Peter Schnorr and Markus Jakobsson. Security of discrete log cryptosystems in the random oracle and the generic model, 1999. Available at https://core.ac.uk/download/pdf/14504220.pdf.

[SJ00] Claus-Peter Schnorr and Markus Jakobsson. Security of signed ElGamal encryption. In Tatsuaki Okamoto, editor, *ASIACRYPT 2000*, volume 1976 of *LNCS*, pages 73–89. Springer, Heidelberg, December 2000.

[ST13] Yannick Seurin and Joana Treger. A robust and plaintext-aware variant of signed ElGamal encryption. In Ed Dawson, editor, *CT-RSA 2013*, volume 7779 of *LNCS*, pages 68–83. Springer, Heidelberg, February / March 2013.

[TY98] Yiannis Tsiounis and Moti Yung. On the security of ElGamal based encryption. In Hideki Imai and Yuliang Zheng, editors, *PKC'98*, volume 1431 of *LNCS*, pages 117–134. Springer, Heidelberg, February 1998.

[Wag02] David Wagner. A generalized birthday problem. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 288–303. Springer, Heidelberg, August 2002.

[Wik08] Douglas Wikström. Simplified submission of inputs to protocols. In Rafail Ostrovsky, Roberto De Prisco, and Ivan Visconti, editors, *SCN 08*, volume 5229 of *LNCS*, pages 293–308. Springer, Heidelberg, September 2008.

[Wui18] Pieter Wuille. Schnorr signatures for secp256k1. Bitcoin Improvement Proposal, 2018. See https://github.com/sipa/bips/blob/bip-schnorr/bip-schnorr.mediawiki.

A Proof of Theorem 5

Consider games Game_0–Game_4 in Figure 16, where Game_0 is $\text{IND-CCA2}^{\mathcal{A}_{\text{alg}}}_{\mathsf{SEG}[\mathsf{GrGen}]}$ and the adversary's advantage in Game_4 is 0. We prove the theorem by bounding the probability that the adversary behaves differently in two consecutive games Game_i and Game_{i+1}.

First, we establish some notation regarding the representation of group elements. Let (P_0, P_1) be the two messages of the adversary's call to Enc. At the beginning of the experiment, the only group-element inputs to $\mathcal{A}_{\text{alg}}$ are G and the challenge public key Y. When the adversary queries its Enc oracle, it receives three more group elements (X^*, C^*, R^*) in the answer.

Furthermore, as the adversary queries its Dec oracle, it receives additional group elements $M_1, \ldots, M_{q_d}$ in response. All in all, representations provided by the adversary are w.r.t. $(G, Y, X^*, C^*, R^*, M_1, \ldots, M_{q_d})$ and for a group element A we write

$$A = \gamma_a G + \upsilon_a Y + \xi_a X^* + \kappa_a C^* + \rho_a R^* + \sum_{j=1}^{q_d} \mu_{a,j} M_i \ ,$$

with the convention that $(\xi_a, \kappa_a, \rho_a) = (0,0,0)$ before the call to Enc and $\mu_{a,j} = 0$ before the j-th call to Dec.

Claim 1. *Consider a query* $\mathsf{H}(X_{[\gamma_x,\upsilon_x,\xi_x,\kappa_x,\rho_x,(\mu_{x,j})_j]}, C_{[\ldots]}, R_{[\ldots]})$ *or* $\mathrm{Dec}(X_{[\ldots]}, C_{[\ldots]}, R_{[\ldots]}, s)$ *in* Game_0*. Then there exist efficiently computable coefficients* $(\bar{\gamma}_r, \bar{\xi}_r, (\bar{\upsilon}_r^{(i)})_i, (\bar{\zeta}_r^{(i)})_i, \bar{\gamma}_x, \bar{\xi}_x, (\bar{\upsilon}_x^{(i)})_i, (\bar{\zeta}_x^{(i)})_i)$ *which only depend on* c^**,* s^**, and on the coefficients of the representations of* X*,* C*,* R *and of the group elements contained in previous* Dec *queries such that*

$$X = \bar{\gamma}_x G + \bar{\xi}_x X^* + \sum_{i=1}^{q_d+1} \bar{\upsilon}_x^{(i)} y^i G + \sum_{i=1}^{q_d+1} \bar{\zeta}_x^{(i)} x^* y^i G \tag{20}$$

$$R = \bar{\gamma}_r G + \bar{\xi}_r X^* + \sum_{i=1}^{q_d+1} \bar{\upsilon}_r^{(i)} y^i G + \sum_{i=1}^{q_d+1} \bar{\zeta}_r^{(i)} x^* y^i G \ . \tag{21}$$

Proof. When the adversary queries H on a tuple (X, C, R) or Dec on a tuple (X, C, R, s), it provides a representation of X, C, and R in terms of the group elements received so far:

$$X = \gamma_x G + \upsilon_x Y + \xi_x X^* + \kappa_x C^* + \rho_x R^* + \textstyle\sum_{j=1}^{q_d} \mu_{x,j} M_j \tag{22}$$

$$C = \gamma_c G + \upsilon_c Y + \xi_c X^* + \kappa_c C^* + \rho_c R^* + \textstyle\sum_{j=1}^{q_d} \mu_{c,j} M_j \tag{23}$$

$$R = \gamma_r G + \upsilon_r Y + \xi_r X^* + \kappa_r C^* + \rho_r R^* + \textstyle\sum_{j=1}^{q_d} \mu_{r,j} M_j \ . \tag{24}$$

Let $\mathcal{B}_j = (G, X^*, yG, \ldots, y^j G, x^* yG, \ldots, x^* y^j G)$. We will show the following:

(i) the j-th message M_j returned by Dec can be represented over $\mathcal{B}_{j+1}$;
(ii) C^* and R^* can be represented over $\mathcal{B}_{q_d+1}$.

Combined with (22) and (24), this will prove the claim.

We first show (i) for the Dec calls before the Enc query (if any). Consider the first Dec call before the Enc query and let (X_1, C_1, R_1, s_1) be its input. Then (22) and (23) simplify to $X_1 = \gamma_{x,1} G + \upsilon_{x,1} Y$ and $C_1 = \gamma_{c,1} G + \upsilon_{c,1} Y$. The output of Dec is thus

$$M_1 \coloneqq C_1 - yX_1 = \underbrace{\gamma_{c,1}}_{=:\gamma_{m,1}} G + \underbrace{(\upsilon_{c,1} - \gamma_{x,1})}_{=:\upsilon_{m,1}^{(1)}} Y + \underbrace{(-\upsilon_{x,1})}_{=:\upsilon_{m,1}^{(2)}} y^2 G \ .$$

In the second Dec call preceding the Enc query, the representation of the arguments (X_2, C_2, R_2, s_2) can also depend on M_1, so analogously, we can define coefficients $\gamma_{m,2}, \upsilon_{m,2}^{(1)}, \upsilon_{m,2}^{(2)}$ and $\upsilon_{m,2}^{(3)}$ such that the second output M_2 of Dec satisfies

$$M_2 = \gamma_{m,2} G + \upsilon_{m,2}^{(1)} Y + \upsilon_{m,2}^{(2)} y^2 G + \upsilon_{m,2}^{(3)} y^3 G \ .$$

More generally, the j-th message returned by the Dec oracle before the Enc query can be written as

$$M_j = \gamma_{m,j} G + \sum_{i=1}^{j+1} \upsilon_{m,j}^{(i)} y^i G \ . \tag{25}$$

In the following, we let k denote the number of queries to Dec before the Enc call.

When the adversary queries its Enc oracle, it provides the representation of the challenge messages P_0 and P_1. We thus have for $b \in \{0,1\}$, $P_b = \gamma_{p,b} G + \upsilon_{p,b} Y + \sum_{j=1}^{k} \mu_{p,b,j} M_j$. Analogously to the above, using (25) we can define coefficients such that

$$P_b = \bar{\gamma}_{p,b} G + \sum_{i=1}^{k+1} \bar{\upsilon}_{p,b}^{(i)} y^i G \ .$$

By inspection of the code of Enc, it follows that

$$C^* = x^* Y + P_b = x^* Y + \bar{\gamma}_{p,b} G + \sum_{i=1}^{k+1} \bar{\upsilon}_{p,b}^{(i)} y^i G \quad \text{and} \tag{26}$$

$$R^* = s^* G - c^* X^* \ , \tag{27}$$

which proves (ii). Consider now the first Dec query after the Enc query. Plugging in (26) and (27) into Equation (22) yields

$$X = (\gamma_x + \kappa_x \bar{\gamma}_{p,b} + \rho_x s^*) G + (\xi_x - \rho_x c^*) X^* + (\upsilon_x + \kappa_x \bar{\upsilon}_{p,b}^{(1)}) Y + \kappa_x x^* Y + \sum_{i=2}^{k+1} \kappa_x \bar{\upsilon}_{p,b}^{(i)} y^i G + \sum_{j=1}^{k} \mu_{x,j} M_j \ .$$

Moreover, the M_i's are still of the form as in (25), hence we obtain

$$X = (\underbrace{\gamma_x + \kappa_x \bar{\gamma}_{m,b} + \rho_x s^* + \textstyle\sum_{j=1}^{k} \mu_{x,j} \gamma_{m,j}}_{=:\bar{\gamma}_x}) G + (\underbrace{\xi_x - \rho_x c^*}_{=:\bar{\xi}_x}) X^* + (\underbrace{\upsilon_x + \kappa_x \bar{\upsilon}_{p,b}^{(1)} + \textstyle\sum_{j=1}^{k} \mu_{x,j} \upsilon_{m,j}^{(1)}}_{=:\bar{\upsilon}_x^{(1)}}) Y + \sum_{i=2}^{k+1} (\underbrace{\kappa_x \bar{\upsilon}_{p,b}^{(i)} + \textstyle\sum_{j=1}^{k} \mu_{x,j} \upsilon_{m,j}^{(i)}}_{=:\bar{\upsilon}_x^{(i)}}) y^i G + \underbrace{\kappa_x}_{=:\bar{\zeta}_x} x^* Y$$

and similarly for C. Hence, the output $M_{k+1} = C - yX$ of Dec is of the form

$$M_{k+1} = \gamma_{m,k+1} G + \xi_{m,k+1} X^* + \sum_{i=1}^{k+2} \upsilon_{m,k+1}^{(i)} y^i G + \sum_{i=1}^{2} \zeta_{m,k+1}^{(i)} x^* y^i G \ .$$

More generally, the j-th message returned by Dec, $j > k$, can be written

$$M_j = \gamma_{m,j} G + \xi_{m,j} X^* + \sum_{i=1}^{j+1} \upsilon_{m,j}^{(i)} y^i G + \sum_{i=1}^{j+1-k} \zeta_{m,j}^{(i)} x^* y^i G \ ,$$

which proves (i) for all j. ■

To prove Theorem 5, we start with the difference between Game_0 and Game_1. First note that at any point $\mathsf{T}(X^*, C^*, R^*)$ is the only value in T that might not have been set during an adversary's call to H or Dec, and that could thus does *not* have a corresponding entry in U. Moreover, if Dec does not reply $\perp$, we must have $(X, C, R) \neq (X^*, C^*, R^*)$, since otherwise by the 3rd line in Dec, we have $s = \log R^* + \mathsf{T}(X^*, C^*, R^*) \log X^* = s^*$ and the oracle would have returned $\perp$ in the 1st line.

The values $(\bar{\gamma}_x, \bar{\xi}_x, (\bar{\upsilon}_x^{(i)})_i, (\bar{\zeta}_x^{(i)})_i, \bar{\gamma}_r, \bar{\xi}_r, (\bar{\upsilon}_r^{(i)})_i, (\bar{\zeta}_r^{(i)})_i)$ as defined in Dec were (implicitly) chosen by the adversary before $\mathsf{T}(X, C, R) = c$ was randomly drawn, which, as we argued

Game_0 $\boxed{\mathsf{Game}_1}$ $\boxed{\mathsf{Game}_2}$ $\boxed{\mathsf{Game}_3}$ $\boxed{\mathsf{Game}_4}$

$(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(\lambda)$
$y \leftarrow\!\!\$\, \mathbb{Z}_p$; $Y := yG$
$b \leftarrow\!\!\$\, \{0,1\}$
$\mathsf{T} := ()$; $\boxed{\mathsf{U} := ()}$
$b' \leftarrow \mathcal{A}_{\text{alg}}^{\mathsf{H},\textsc{Enc},\textsc{Dec}}((p, \mathbb{G}, G), Y)$
return $(b = b')$

Oracle $\widetilde{\mathsf{H}}(X, C, R)$

if $\mathsf{T}(X, C, R) = \perp$ **then**
  $\mathsf{T}(X, C, R) \leftarrow\!\!\$\, \mathbb{Z}_p$
return $\mathsf{T}(X, C, R)$

Oracle $\textsc{Enc}(P_{0,[\gamma_{p,0}, \upsilon_{p,0}, (\mu_{p,0,j})_j]}, P_{1,[\ldots]})$

  // one-time; only in Game_0 and Game_1
  // $P_b = \gamma_{p,b} G + \upsilon_{p,b} Y + \sum_{j=1}^{k} \mu_{p,b,j} M_j$
$x^*, r^* \leftarrow\!\!\$\, \mathbb{Z}_p$; $X^* := x^* G$
$C^* = x^* Y + P_b$
$R^* := r^* G$; $c^* := \widetilde{\mathsf{H}}(X^*, C^*, R^*)$
$s^* := r^* + c^* x^* \bmod p$
return (X^*, C^*, R^*, s^*)

Oracle $\textsc{Enc}(P_{0,[\gamma_{p,0}, \upsilon_{p,0}, (\mu_{p,0,j})_j]}, P_{1,[\ldots]})$

  // one-time; only in $\mathsf{Game}_2 - \mathsf{Game}_4$
$x^*, c^*, s^* \leftarrow\!\!\$\, \mathbb{Z}_p$; $X^* := x^* G$
$Z^* := x^* Y$
$\boxed{Z^* \leftarrow\!\!\$\, \mathbb{G}}$
$C^* = Z^* + P_b$
$R^* := s^* G - c^* X^*$
if $\mathsf{T}(X^*, C^*, R^*) = \perp$ **then**
  $\mathsf{T}(X^*, C^*, R^*) := c^*$
else abort game and
  return $b' \leftarrow\!\!\$\, \{0,1\}$ (II)
return (X^*, C^*, R^*, s^*)

Oracle $\mathsf{H}((X_{[\gamma_x, \upsilon_x, \xi_x, \kappa_x, \rho_x, (\mu_{x,j})_j]}, C_{[\ldots]}, R_{[\ldots]})$

  // $X = \gamma_x G + \upsilon_x Y + \xi_x X^* + \kappa_x C^* + \rho_x R^* + \sum_j \mu_{x,j} M_j$

let $\bar{\gamma}_x, \bar{\xi}_x, (\bar{\upsilon}_x^{(i)})_i, (\bar{\zeta}_x^{(i)})_i, \bar{\gamma}_r, \bar{\xi}_r, (\bar{\upsilon}_r^{(i)})_i, (\bar{\zeta}_r^{(i)})_i$ such that
$X = \bar{\gamma}_x G + \bar{\xi}_x X^* + \sum_i \bar{\upsilon}_x^{(i)} y^i G + \sum_i \bar{\zeta}_x^{(i)} x^* y^i G$
$R = \bar{\gamma}_r G + \bar{\xi}_r X^* + \sum_i \bar{\upsilon}_r^{(i)} y^i G + \sum_i \bar{\zeta}_r^{(i)} x^* y^i G$

if $\mathsf{T}(X, C, R) = \perp$ **then**
  $\mathsf{T}(X, C, R) \leftarrow\!\!\$\, \mathbb{Z}_p$
  $\boxed{\mathsf{U}(X, C, R) := (\bar{\gamma}_x, \bar{\xi}_x, (\bar{\upsilon}_x^{(i)})_i, (\bar{\zeta}_x^{(i)})_i, \bar{\gamma}_r, \bar{\xi}_r, (\bar{\upsilon}_r^{(i)})_i, (\bar{\zeta}_r^{(i)})_i)}$
return $\mathsf{T}(X, C, R)$

Oracle $\textsc{Dec}(X_{[\gamma_x, \upsilon_x, \xi_x, \kappa_x, \rho_x, (\mu_{x,j})_j]}, C_{[\ldots]}, R_{[\ldots]}, s)$

if $(X, C, R, s) = (X^*, C^*, R^*, s^*)$ **then return** $\perp$
$c := \mathsf{H}(X_{[\gamma_x, \upsilon_x, \xi_x, \kappa_x, \rho_x, (\mu_{x,j})_j]}, C_{[\ldots]}, R_{[\ldots]})$
if $sG \neq R + cX$ **then return** $\perp$

$(\bar{\gamma}_x, \bar{\xi}_x, (\bar{\upsilon}_x^{(i)})_i, (\bar{\zeta}_x^{(i)})_i, \bar{\gamma}_r, \bar{\xi}_r, (\bar{\upsilon}_r^{(i)})_i, (\bar{\zeta}_r^{(i)})_i) := \mathsf{U}(X, C, R)$
for all $i : \alpha^{(i)} := (\bar{\upsilon}_r^{(i)} + x^* \bar{\zeta}_r^{(i)}) + c\,(\bar{\upsilon}_x^{(i)} + x^* \bar{\zeta}_x^{(i)}) \bmod p$
if $\exists j : \bar{\upsilon}_x^{(j)} + x^* \bar{\zeta}_x^{(j)} \not\equiv_p 0$ **and** $\forall i : \alpha^{(i)} = 0$ **then**
  abort game and return $b' \leftarrow\!\!\$\, \{0,1\}$ (I)

if $\exists j : \alpha^{(j)} \neq 0$ **then**
  abort game and return $b' \leftarrow\!\!\$\, \{0,1\}$ (III)
// solve for y: $\sum_i \alpha^{(i)} y^i + (\bar{\xi}_r + c\bar{\xi}_x) x^*$
// $+(\bar{\gamma}_r + c\bar{\gamma}_x - s) \equiv_p 0$

// $\forall i : \bar{\upsilon}_x^{(i)} = \bar{\zeta}_x^{(i)} = 0 \Rightarrow x = \bar{\gamma}_x + \bar{\xi}_x x^*$ and $yX = \bar{\gamma}_x Y + \bar{\xi}_x Z^*$
$M := C - yX$
$\boxed{M := C - (\bar{\gamma}_x Y + \bar{\xi}_x Z^*)}$
return M

Fig. 16. The IND-CCA2 game for the Schnorr-Signed ElGamal scheme (Game_0) and games Game_1–Game_4 used in the proof. The comments in dashed boxes show how the DL reduction simulates Game_3 without knowledge of y.

above, must have been by a call from the adversary. The value c is thus independent of $(\bar{\gamma}_x, \ldots, (\bar{\zeta}_r^{(i)})_i)$.

Game_0 and Game_1 behave identically unless Game_1 aborts during a DEC call in line (I), thus in particular for some j: $\beta^{(j)} \coloneqq \bar{\upsilon}_x^{(j)} + x^* \bar{\zeta}_x^{(j)} \not\equiv_p 0$ and $\alpha^{(j)} \coloneqq (\bar{\upsilon}_r^{(j)} + x^* \bar{\zeta}_r^{(j)}) + c\,(\bar{\upsilon}_x^{(j)} + x^* \bar{\zeta}_x^{(j)}) \equiv_p 0$. By the above argument, the probability that c was chosen such as $c = -(\bar{\upsilon}_r^{(j)} + x^* \bar{\zeta}_r^{(j)}) \cdot (\beta^{(j)})^{-1} \bmod p$ is upper-bounded by $\frac{1}{2^{\lambda-1}}$.

Denoting by A the event that Game_1 aborts in line (I) during some DEC call, we have $\Pr[A] \leq \frac{q_\mathrm{d}}{2^{\lambda-1}}$. Since $\Pr[1 \leftarrow \mathsf{Game}_0 \,|\, \neg A] = \Pr[1 \leftarrow \mathsf{Game}_1 \,|\, \neg A]$, we have

$$\begin{aligned}\mathsf{Adv}^{\mathsf{game}_0}_{\mathcal{A}_\mathrm{alg}}(\lambda) - \mathsf{Adv}^{\mathsf{game}_1}_{\mathcal{A}_\mathrm{alg}}(\lambda) &= 2\Pr[A]\big(\Pr[1 \leftarrow \mathsf{Game}_0 \,|\, A] - \Pr[1 \leftarrow \mathsf{Game}_1 \,|\, A]\big) \\ &\leq \Pr[A]\ ,\end{aligned}$$

as $\Pr[1 \leftarrow \mathsf{Game}_1 \,|\, A] = \frac{1}{2}$. We thus have

$$\mathsf{Adv}^{\mathsf{game}_1}_{\mathcal{A}_\mathrm{alg}}(\lambda) \geq \mathsf{Adv}^{\mathsf{game}_0}_{\mathcal{A}_\mathrm{alg}}(\lambda) - \frac{q_\mathrm{d}}{2^{\lambda-1}}\ . \tag{28}$$

Game_1 and Game_2 behave identically unless oracle ENC generates values (X^*, C^*, R^*) that have already been assigned a value in the table T. The values X^* and R^* are uniformly random in $\mathbb{G}$. Thus, after the adversary has made q_h queries to H and q_d to DEC, at most $q_\mathrm{h} + q_\mathrm{d}$ values in T are assigned. Thus the probability that (X^*, C^*, R^*) collides with one of the entries is bounded by $\frac{q_\mathrm{h}+q_\mathrm{d}}{(2^{\lambda-1})^2}$, and we thus have

$$\mathsf{Adv}^{\mathsf{game}_2}_{\mathcal{A}_\mathrm{alg}}(\lambda) \geq \mathsf{Adv}^{\mathsf{game}_1}_{\mathcal{A}_\mathrm{alg}}(\lambda) - \frac{q_\mathrm{d} + q_\mathrm{h}}{(2^{\lambda-1})^2}\ . \tag{29}$$

Game_2 and Game_3 behave identically unless for some j: $\alpha^{(j)} \coloneqq (\bar{\upsilon}_r^{(j)} + x^* \bar{\zeta}_r^{(j)}) + c\,(\bar{\upsilon}_x^{(j)} + x^* \bar{\zeta}_x^{(j)}) \not\equiv_p 0$. We show that when this happens, we can build a reduction $\mathcal{B}_1$ that computes the discrete logarithm of Y. The reason is that if DEC does not return $\perp$ then $sG = R + cX$, which, by plugging in (20) and (21), yields

$$\begin{aligned}0 &= (\bar{\gamma}_r + c\bar{\gamma}_x - s)G + (\bar{\xi}_r + c\bar{\xi}_x)X^* + \sum_{i=1}^{q_d+1} (\bar{\upsilon}_r^{(i)} + c\bar{\upsilon}_x^{(i)}) y^i G + \sum_{i=1}^{q_d+1} (\bar{\zeta}_r^{(i)} + c\bar{\zeta}_x^{(i)}) x^* y^i G \\ &= (\bar{\gamma}_r + c\bar{\gamma}_x - s)G + (\bar{\xi}_r + c\bar{\xi}_x)X^* + \sum_{i=1}^{q_d+1} \alpha^{(i)} y^i G\ .\end{aligned} \tag{30}$$

If $\alpha^{(j)} \neq 0$ for some j, then we can solve the above for y.

In more detail, reduction $\mathcal{B}_1$ is given a challenge Y and sets it as the public key. It simulates Game_2 by choosing random values x^*, c^* and s^* during the ENC call. Moreover, it can simulate any DEC query before a potential abort in line (III) without knowledge of y as follows.

Consider a call $\mathrm{DEC}(X_{[\gamma_x, \upsilon_x, \xi_x, \kappa_x, \rho_x, (\mu_{x,i})_i]}, C_{[\ldots]}, R_{[\ldots]}, s)$. The oracle returns $\perp$ if $sG \neq R + \mathsf{H}(X, C, R)X$ or $(X, C, R, s) = (X^*, C^*, R^*, s^*)$. As s is determined by (X, C, R), the latter implies $(X, C, R) \neq (X^*, C^*, R^*)$ if DEC did not return $\perp$.

If furthermore DEC does not abort in line (I) or (III), then $\bar{\upsilon}_x^{(i)} + x^* \bar{\zeta}_x^{(i)} \equiv_p 0$ for all i; thus, by (20): $X = \bar{\gamma}_x G + \bar{\xi}_x X^*$. The reduction can thus compute $yX = \bar{\gamma}_x Y + \bar{\xi}_x x^* Y$ and return

$$M \coloneqq C - (\bar{\gamma}_x + \bar{\xi}_x x^*)Y = C - xY = C - yX\ .$$

(Note that Game_3 also introduced a syntactical change by directly defining the response of $\textsc{Dec}$ as $M := C - \bar{\gamma}_x Y + \bar{\xi}_x Z^* = (\bar{\gamma}_x + \bar{\xi}_x x^*)Y = C - yX$.)

This shows that $\mathcal{B}_1$ can simulate Game_3 until an abort in line (III). In this case, $\mathcal{B}_1$ returns y, the solution of the following equation (cf. (30)):

$$\textstyle\sum_{i=1}^{q_d+1} \alpha^{(i)} y^i + \bar{\gamma}_r + c\bar{\gamma}_x - s + (\bar{\xi}_r + c\bar{\xi}_x)x^* \equiv_p 0 \ .$$

and thus solves the DL challenge Y. This yields

$$\mathsf{Adv}^{\mathsf{game}_3}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) \geq \mathsf{Adv}^{\mathsf{game}_2}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) - \mathsf{Adv}^{\mathrm{dl}}_{\mathsf{GrGen},\mathcal{B}_1}(\lambda) \ . \tag{31}$$

Finally, Game_3 and Game_4 only differ in the definition of Z^*, which in Game_3 is the CDH of X^* and Y, whereas in Game_4 it is random. We build a reduction $\mathcal{B}_2$ to DDH, which is given a DDH challenge (X^*, Y, Z^*) and uses these values to simulate Game_3 (when Z^* is x^*Y) or Game_4 (when it is random). Note that the games can be simulated without knowledge of x^* and y; in particular, the abort condition for (III) can be checked as

$$0 \stackrel{?}{=} \alpha^{(j)} G = (\bar{\upsilon}_r^{(j)} + c\bar{\upsilon}_x^{(j)})G + (\bar{\zeta}_r^{(j)} + c\bar{\zeta}_x^{(j)})X^*$$

and likewise for the condition of abort (I). We thus have

$$\mathsf{Adv}^{\mathsf{game}_4}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) \geq \mathsf{Adv}^{\mathsf{game}_3}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) - 2 \cdot \mathsf{Adv}^{\mathrm{ddh}}_{\mathsf{GrGen},\mathcal{B}_2}(\lambda) \ . \tag{32}$$

Inspecting Game_4, we note that $\mathcal{A}$'s output is independent of b, because the random group element Z^* completely hides the message P_b. And whenever the game aborts, it outputs a random bit; we thus have:

$$\mathsf{Adv}^{\mathsf{game}_4}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) = 2 \cdot \Pr[1 \leftarrow \mathsf{Game}_4] - 1 = 0 \ . \tag{33}$$

The theorem now follows from Equations (28), (29), (31), (32) and (33).

B Schnorr-Signed Hashed ElGamal KEM

A public key for the ElGamal key-encapsulation mechanism (KEM) is a group element $Y \in \mathbb{G}$. To encrypt a message under Y, one samples a random $x \in \mathbb{Z}_p$ and derives an ephemeral key $K := xY$ to encrypt the message. Given the *encapsulation* $X := xG$, the receiver that holds $y = \log Y$ can derive the same key as $K := yX$. Under the decisional Diffie-Hellman assumption (DDH), this scheme is IND-CPA-secure. In the AGM, it was shown to satisfy CCA1 security (where the adversary can only make decryption queries before it has seen the challenge key) under a parameterized variant of DDH [FKL18].

By hashing the key, that is, defining $k := \mathsf{H}(xY)$, the assumption for proving CPA, resp. CCA2 security, can be relaxed to CDH, resp. strong Diffie-Hellman (SDH), in the random-oracle model.

Exactly as in Section 6, the idea of *Schnorr-signed* hashed ElGamal is that, in addition to X, the encapsulation contains a proof of knowledge of the used randomness $x = \log X$, in the form of a Schnorr signature on message X under the public key X. The scheme is detailed in Figure 17.

$\underline{\mathsf{SEGK.Setup}(\lambda)}$

$(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(1^\lambda)$
Select $\mathsf{H}\colon \{0,1\}^* \to \mathbb{Z}_p$
Select $\mathsf{H}'\colon \{0,1\}^* \to \mathcal{K}$
return $par := (p, \mathbb{G}, G, \mathsf{H}, \mathsf{H}')$

$\underline{\mathsf{SEGK.Enc}(pk)}$

$(p, \mathbb{G}, G, \mathsf{H}, \mathsf{H}', Y) := pk$
$x, r \leftarrow_\$ \mathbb{Z}_p\,;\; X := xG\,;\; R := rG$
$k := \mathsf{H}'(xY)\,;\; s := r + \mathsf{H}(R, X) \cdot x \bmod p$
return $(k, (X, R, s))$

$\underline{\mathsf{SEGK.KeyGen}(par)}$

$(p, \mathbb{G}, G, \mathsf{H}, \mathsf{H}') := par$
$y \leftarrow_\$ \mathbb{Z}_p\,;\; Y := yG$
$sk := (par, y)\,;\; pk := (par, Y)$
return (sk, pk)

$\underline{\mathsf{SEGK.Dec}(sk, (X, R, s))}$

$(p, \mathbb{G}, G, \mathsf{H}, \mathsf{H}', y) := sk$
if $sG \neq R + \mathsf{H}(R, X) \cdot X$ **then**
  return $\perp$
return $k := \mathsf{H}'(yX)$

Fig. 17. The Schnorr-signed ElGamal KEM scheme $\mathsf{SEGK[GrGen]}$ for key space $\mathcal{K}$.

The strongest security notion for KEM schemes is indistinguishability of ciphertexts under chosen-ciphertext attack (IND-CCA2), where the adversary can query decryptions of encapsulations of its choice even after receiving the challenge. The (decisional) game IND-CCA2 is defined in Figure 18.

We now prove that the Schnorr-signed hashed ElGamal KEM is tightly IND-CCA2-secure in the AGM+ROM under the discrete logarithm assumption.

Theorem 6. *Let* GrGen *be a group generator. Let* $\mathcal{A}_{\mathrm{alg}}$ *be an algebraic adversary against the IND-CCA2 security of the Schnorr-signed ElGamal KEM scheme* $\mathsf{SEGK[GrGen]}$ *making at most* q_{d} *decryption queries and* q_{h} *queries to both random oracles. Then there exists an algorithm* $\mathcal{B}$ *solving the DL problem w.r.t.* GrGen*, such that*

$$\mathsf{Adv}^{\text{ind-cca2}}_{\mathsf{SEGK[GrGen]},\mathcal{A}_{\mathrm{alg}}}(\lambda) \leq \mathsf{Adv}^{\mathrm{dl}}_{\mathsf{GrGen},\mathcal{B}}(\lambda) + \frac{q_{\mathrm{d}} + \frac{1}{2^{\lambda-1}}(q_{\mathrm{d}} + q_{\mathrm{h}})}{2^{\lambda-1}} \;.$$

We start with the proof idea. Let Y be the public key and let $(X^* = x^*G, R^*, s^*)$ be the challenge ciphertext. If the adversary never queries $\mathsf{H}'(x^*Y)$ then it has no information about

$\underline{\text{Game IND-CCA2}^{\mathcal{A}}_{\mathsf{KEM}}(\lambda)}$

$par \leftarrow \mathsf{KEM.Setup}(\lambda)$
$(pk, sk) \leftarrow \mathsf{KEM.KeyGen}(par)$
$b \leftarrow_\$ \{0,1\}$
$b' \leftarrow \mathcal{A}^{\textsc{Enc},\textsc{Dec}}(pk)$
return $(b = b')$

$\underline{\text{Oracle } \textsc{Enc}() \quad /\!\!/ \text{ one time}}$

$(k_0, c^*) \leftarrow \mathsf{KEM.Enc}(pk)\,;\; k_1 \leftarrow_\$ \mathcal{K}$
return (k_b, c^*)

$\underline{\text{Oracle } \textsc{Dec}(c)}$

if $c = c^*$ **then return** $\perp$
return $\mathsf{KEM.Dec}(sk, c)$

Fig. 18. The IND-CCA2 security game for a KEM scheme KEM.

the challenge key k_b; but in order to query $K^* := x^*Y$, the adversary must solve the CDH problem for (Y, X^*). A CDH solution cannot be recognized by the reduction, so it would have to guess one of $\mathcal{A}$'s H' queries, which would make the proof non-tight.

In the AGM we can give a *tight* reduction to a *weaker* assumption, namely DL: Given a DL challenge Y, we set it as the public key, pick a random z and set $X^* := zY$. If the adversary makes the query $\mathsf{H}'(K^*)$ then we have $K^* = zy^2G$. On the other hand, the adversary must provide a representation $(\gamma, \upsilon, \xi, \rho)$ of K^* w.r.t. (G, Y, X^*, R^*), and thus

$$K^* = \gamma G + \upsilon Y + \xi X^* + \rho R^* = (\gamma + \upsilon y + \xi zy + \rho s^* - \rho c^* zy)G \ , \tag{34}$$

using the fact that $R^* = s^*G - c^*X^*$. Setting these two representations of $\log K^*$ equal yields the following quadratic equation in y:

$$zy^2 - (\upsilon + \xi z - \rho c^* z)y \equiv_p \gamma + \rho s^* \ .$$

If one of the solutions is the DL of Y, we are done; otherwise, the adversary's query was not of the form K^* and the challenge bit remains information-theoretically hidden.

The rest of the game is simulated without knowledge of $\log X^*$ and $\log Y$ as follows: The Schnorr signature under X^* contained in the challenge encapsulation can be simulated by programming the random oracle H as in the proof of Theorem 1. Decryption queries leverage the fact that the Schnorr signature contained in an encapsulation (X, R, s) proves knowledge of x with $X = xG$. By extracting x, the reduction can answer the query with $k = \mathsf{H}'(xY)$, but this extraction is trickier than in the proof of Theorem 1, since both X and R can also depend on Y, X^* and R^* (if the query is made *after* seeing the challenge ciphertext, which is the harder case).

In more detail, given the representations $(\gamma, \upsilon, \xi, \rho)$ and $(\gamma', \upsilon', \xi', \rho')$ of R and X provided by the adversary, we can write (analogously to Eq. (34)):

$$\begin{aligned} r = \log R \ &\equiv_p \ \gamma + \upsilon y + \xi zy + \rho s^* - \rho c^* zy \quad &&\equiv_p \alpha y + (\gamma + \rho s^*) \quad \text{and} \\ x = \log X \ &\equiv_p \gamma' + \upsilon' y + \xi' zy + \rho' s^* - \rho' c^* zy &&\equiv_p \alpha' y + (\gamma' + \rho' s^*) \end{aligned} \tag{35}$$

with $\alpha := \upsilon + (\xi - \rho c^*)z \bmod p$ and $\alpha' := \upsilon' + (\xi' - \rho' c^*)z \bmod p$. Since the signature (R, s) contained in the query must be valid, we have $s \equiv_p r + cx$ with $c = \mathsf{H}(R, X)$. Plugging the above two equations into the latter yields

$$(\alpha + \alpha' c)y \ \equiv_p s - (\gamma + \rho s^*) - (\gamma' + \rho' s^*)c \ .$$

If $\alpha + \alpha' c \not\equiv_p 0$ then solving the above for y solves the challenge DL and the reduction can stop. Since $c = \mathsf{H}(R, X)$ was chosen by the experiment after the adversary provided representations of R and X, which define α and α', we have that $\alpha + \alpha' c \equiv_p 0$ happens with probability $\frac{1}{p}$, unless $\alpha' = 0$.

In the latter case however, from Eq. (35) we have $x = \gamma' + \rho' s^* \bmod p$, meaning the reduction can compute x and can therefore answer the decryption query by returning $\mathsf{H}'(xY) = \mathsf{H}'(yX)$.

Proof of Theorem 6. Consider the games Game_0 through Game_3 in Figure 19, where in Game_3 the adversary's advantage is 0. Game_0 has the same behavior as $\text{IND-CCA2}^{\mathcal{A}_{\text{alg}}}_{\mathsf{SEGK[GrGen]}}$; the only syntactical change is that the value X^* used in the $\text{E}\textsc{nc}$ oracle is already set before running $\mathcal{A}$ (which ensures that in later games it is defined in the abort conditions for lines (I),

Game_0 $\boxed{\mathsf{Game}_1}$ $\boxed{\mathsf{Game}_2}$ $\boxed{\mathsf{Game}_3}$

$(p, \mathbb{G}, G) \leftarrow \mathsf{GrGen}(\lambda)$
$y \leftarrow_\$ \mathbb{Z}_p^* ;\ Y := yG$
$b \leftarrow_\$ \{0,1\}$
$\mathsf{T} := (\,) ;\ \mathsf{T}' := (\,) ;\ \boxed{\mathsf{U} := (\,) ;\ c^* := 0}$
$x^* \leftarrow_\$ \mathbb{Z}_p ;\ X^* := x^* G$
// $z \leftarrow_\$ \mathbb{Z}_p^* ;\ X^* := zY$
$b' \leftarrow \mathcal{A}_{\mathrm{alg}}^{\mathsf{H}, \mathsf{H}', \textsc{Enc}, \textsc{Dec}}((p, \mathbb{G}, G), Y)$
return $(b = b')$

Oracle $\widetilde{\mathsf{H}}(R, X)$

if $\mathsf{T}(R, X) = \perp$ **then** $\mathsf{T}(R, X) \leftarrow_\$ \mathbb{Z}_p$
return $\mathsf{T}(R, X)$

Oracle $\widetilde{\mathsf{H}}'(K)$

if $\mathsf{T}'(K) = \perp$ **then** $\mathsf{T}'(K) \leftarrow_\$ \mathcal{K}$
return $\mathsf{T}'(K)$

Oracle $\textsc{Enc}()$ // one-time oracle

// only Game_0 and Game_1
$r^* \leftarrow_\$ \mathbb{Z}_p ;\ R^* := r^* G$
$c^* := \widetilde{\mathsf{H}}(R^*, X^*)$
$s^* := r^* + c^* x^* \bmod p$
$k_0 := \widetilde{\mathsf{H}}'(x^* Y) ;\ k_1 \leftarrow_\$ \mathcal{K}$
return $(k_b, (X^*, R^*, s^*))$

Oracle $\textsc{Enc}()$ // one-time oracle

$c^*, s^* \leftarrow_\$ \mathbb{Z}_p ;\ R^* := s^* G - c^* X^*$
if $\mathsf{T}(R^*, X^*) = \perp$ **then**
$\mathsf{T}(R^*, X^*) := c^*$
else abort game and
return $b' \leftarrow_\$ \{0,1\}$ (II)
$k_0 := \widetilde{\mathsf{H}}'(x^* Y) ;\ k_1 \leftarrow_\$ \mathcal{K}$
return $(k_1, (X^*, R^*, s^*))$
return $(k_b, (X^*, R^*, s^*))$

Oracle $\mathsf{H}(R_{[\gamma, \upsilon, \xi, \rho]}, X_{[\gamma', \upsilon', \xi', \rho']})$

// $R = \gamma G + \upsilon Y + \xi X^* + \rho R^*$
// $X = \gamma' G + \upsilon' Y + \xi' X^* + \rho' R^*$
// (if $\textsc{Enc}$ has not been called yet,
// then $\xi = \xi' = \rho = \rho' = 0$)
if $\mathsf{T}(R, X) = \perp$ **then**
$\mathsf{T}(R, X) \leftarrow_\$ \mathbb{Z}_p$
$\boxed{\mathsf{U}(R, X) := (\gamma, \upsilon, \xi, \rho, \gamma', \upsilon', \xi', \rho')}$
return $\mathsf{T}(R, X)$

Oracle $\mathsf{H}'(K_{[\gamma, \upsilon, \xi, \rho]})$

if $K = yX^*$ **then**
abort game and return $b' \leftarrow_\$ \{0,1\}$ (III)
// solve $zy^2 - (\upsilon + (\xi - \rho c^*) z) y \equiv_p \gamma + \rho s^*$
if $\mathsf{T}'(K) = \perp$ **then** $\mathsf{T}'(K) \leftarrow_\$ \mathcal{K}$
return $\mathsf{T}'(K)$

Oracle $\textsc{Dec}(X_{[\gamma', \upsilon', \xi', \rho']}, R_{[\gamma, \upsilon, \xi, \rho]}, s)$

if $(X, R, s) = (X^*, R^*, s^*)$ **then return** $\perp$
$c := \mathsf{H}(R_{[\gamma, \upsilon, \xi, \rho]}, X_{[\gamma', \upsilon', \xi', \rho']})$
if $sG \neq R + cX$ **then return** $\perp$

$(\gamma, \upsilon, \xi, \rho, \gamma', \upsilon', \xi', \rho') := \mathsf{U}(R, X)$
$z := x^* y^{-1} \bmod p$
$\alpha := \upsilon + (\xi - \rho c^*) z \bmod p$
$\alpha' := \upsilon' + (\xi' - \rho' c^*) z \bmod p$
if $\alpha + \alpha' c \equiv_p 0$ **and** $\alpha' \neq 0$ **then**
abort game and return $b' \leftarrow_\$ \{0,1\}$ (I)
if $\alpha + \alpha' c \not\equiv_p 0$ **then**
abort game and return $b' \leftarrow_\$ \{0,1\}$ (IV)
// $y = (\alpha + \alpha' c)^{-1} (s - (\gamma + \rho s^*) - (\gamma' + \rho' s^*) c) \bmod p$

// $\alpha' = 0 \Rightarrow x = \gamma' + \rho' s^*$, thus $k = \mathsf{H}'(xY)$
$k := \widetilde{\mathsf{H}}'(yX)$
return k

Fig. 19. The IND-CCA2 security game $\text{IND-CCA2}_{\mathsf{SEGK[GrGen]}}^{\mathcal{A}_{\mathrm{alg}}}$ (Game_0) for the Schnorr-Signed hashed ElGamal KEM scheme and games Game_1–Game_3 used in the proof of Theorem 6. The comments in dashed boxes show how the DL reduction simulates Game_3 without knowledge of x^* and y.

(III) and (IV) even when $\textsc{Enc}$ has not been called yet). We prove the theorem by bounding the probability that the adversary behaves differently in two consecutive games Game_i and Game_{i+1}.

We start with the difference between Game_0 and Game_1, which consists in a possible abort in line (I) in oracle $\textsc{Dec}$. This happens when the experiment randomly chooses c as one particular value. (Note that Game_1 sets $c^* := 0$, so the value is defined in $\textsc{Dec}$ when $\textsc{Enc}$ has not been called yet.)

Observe that at any point $\mathsf{T}(R^*, X^*)$ is the only value in T that might not have been set during an adversary's call to H or Dec, and that could *not* have a corresponding entry in U. Moreover, if a call (X, R, s) to $\textsc{Dec}$ is not answered by $\perp$, we must have $(X, R) \neq (X^*, R^*)$, since otherwise by the 3rd line $s = \log R^* + \mathsf{T}(R^*, S^*) \log X^* = s^*$ and the oracle would have returned $\perp$ in the 1st line.

Game_1 sets the values $(\gamma, \upsilon, \xi, \rho, \gamma', \upsilon', \xi', \rho')$ that were given as the representation of X and R when $\mathsf{T}(X, R) = c$ was randomly drawn. As we argued above, this must have been during a call from the adversary. The value c is thus independent of $(\gamma, \ldots, \rho')$, the values that define α and α'.

The two games Game_0 and Game_1 behave identically unless Game_1 aborts in line (I), that is, if $\alpha + c\alpha' \equiv_p 0$ and $\alpha' \neq 0$. By the above argument, the probability that c was chosen such as $c = -\alpha \cdot (\alpha')^{-1} \bmod p$ is upper-bounded by $\frac{1}{2^{\lambda-1}}$. Denoting by A the event that Game_1 aborts in line (I) during some $\textsc{Dec}$ call, we have $\Pr[A] \leq \frac{q_\mathrm{d}}{2^{\lambda-1}}$. Since $\Pr[1 \leftarrow \mathsf{Game}_0 \,|\, \neg A] = \Pr[1 \leftarrow \mathsf{Game}_1 \,|\, \neg A]$, we have

$$\begin{aligned}\mathsf{Adv}^{\mathsf{game}_0}_{\mathcal{A}_\mathrm{alg}}(\lambda) - \mathsf{Adv}^{\mathsf{game}_1}_{\mathcal{A}_\mathrm{alg}}(\lambda) &= 2\Pr[A]\big(\Pr[1 \leftarrow \mathsf{Game}_0 \,|\, A] - \Pr[1 \leftarrow \mathsf{Game}_1 \,|\, A]\big)\\ &\leq \Pr[A]\ ,\end{aligned}$$

as $\Pr[1 \leftarrow \mathsf{Game}_1 \,|\, A] = \frac{1}{2}$. We thus have:

$$\mathsf{Adv}^{\mathsf{game}_1}_{\mathcal{A}_\mathrm{alg}}(\lambda) \geq \mathsf{Adv}^{\mathsf{game}_0}_{\mathcal{A}_\mathrm{alg}}(\lambda) - \frac{q_\mathrm{d}}{2^{\lambda-1}}\ . \tag{36}$$

The two games Game_1 and Game_2 behave identically unless oracle $\textsc{Enc}$ generates values (R^*, X^*) that have already been assigned a value in the table T. The values R^* and X^* are uniformly random in $\mathbb{G}$. Moreover, after the adversary has made q_h queries to H and q_d to $\textsc{Dec}$, at most $q_\mathrm{h} + q_\mathrm{d}$ values in T are assigned. Thus, the probability that (R^*, X^*) collides with one of the entries is bounded by $\frac{q_\mathrm{h}+q_\mathrm{d}}{(2^{\lambda-1})^2}$, and we thus have

$$\mathsf{Adv}^{\mathsf{game}_2}_{\mathcal{A}_\mathrm{alg}}(\lambda) \geq \mathsf{Adv}^{\mathsf{game}_1}_{\mathcal{A}_\mathrm{alg}}(\lambda) - \frac{(q_\mathrm{d} + q_\mathrm{h})}{(2^{\lambda-1})^2}\ . \tag{37}$$

<u>Reduction to DL</u>. We now construct an adversary $\mathcal{B}$ solving DL whenever Game_3 differs from Game_2, that is, when there is an abort in line (III) or (IV). Given a DL challenge Y, the reduction $\mathcal{B}$ sets Y as the public key, chooses a random $z \leftarrow_\$ \mathbb{Z}_p^*$ and sets $X^* := zY$. It simulates $\boxed{\textsc{Enc}()}$ by computing (R^*, s^*) as prescribed by the oracle, but setting $k_b := k_1$, a random key. (We argue below that when this introduces an inconsistency, the game aborts in line (III) anyway). $\mathcal{B}$ simulates the other oracles in Game_3 for $\mathcal{A}$ without knowledge of y and x^* as follows (cf. the comments in dashed boxes in Figure 19):

– Queries to H': whenever $\mathcal{A}$ queries $K_{[\gamma,\upsilon,\xi,\rho]}$ with

$$K = \gamma G + \upsilon Y + \xi X^* + \rho R^* = (\gamma + \upsilon y + \xi zy + \rho s^* - \rho c^* zy)G \ ,$$

$\mathcal{B}$ checks whether $K = yX^*$ (which equals zy^2G) by solving the following equation for y

$$zy^2 - (\upsilon + \xi z - \rho c^* z)y \equiv_p \gamma + \rho s^*$$

and checking whether some solution y satisfies $Y = yG$ (in this case Game_3 would abort in line (III)); if so, $\mathcal{B}$ stops and returns y.
Note that otherwise, $K \neq yX^*$ and thus k_1 still perfectly simulates $\mathsf{H}'(yX^*) = k_b$.

– Queries to Dec: when queried (X, R, s), Dec returns $\perp$ if $sG \neq R + \mathsf{H}(R, X)X$ or $(X, R, s) = (X^*, R^*, s^*)$. Since s is determined by (R, X), the latter implies $(R, X) \neq (R^*, X^*)$ if Dec did not return $\perp$. By the first lines in the $\boxed{\text{box}}$ in Dec, we have that $(\gamma, \upsilon, \xi, \rho)$ and $(\gamma', \upsilon', \xi', \rho')$ are such that

$$\begin{aligned} R = \gamma G + \upsilon Y + \xi X^* + \rho R^* \ &= \ (\gamma + \upsilon y + \xi zy + \rho s^* - \rho c^* zy)G \\ X = \gamma' G + \upsilon' Y + \xi' X^* + \rho' R^* &= (\gamma' + \upsilon' y + \xi' zy + \rho' s^* - \rho' c^* zy)G \ . \end{aligned} \tag{38}$$

As in Dec, we let $\alpha := \upsilon + (\xi - \rho c^*)z \bmod p$ and $\alpha' := \upsilon' + (\xi' - \rho' c^*)z \bmod p$ and thus from Eq. (38) we have

$$\begin{aligned} r := \log R = \gamma + \rho s^* + \alpha y \bmod p \\ x := \log X = \gamma' + \rho' s^* + \alpha' y \bmod p \ . \end{aligned} \tag{39}$$

Since the oracle did not return $\perp$ in the 3rd line, we have $s \equiv_p r + cx$, and thus, by substituting r and x from Eq. (39):

$$(\alpha + \alpha' c)y \equiv_p s - (\gamma + \rho s^*) - (\gamma' + \rho' s^*)c \ .$$

If $\alpha + \alpha' c \not\equiv_p 0$ then Game_3 would abort in line (IV); in this case $\mathcal{B}$ returns the DL solution $y = (\alpha + \alpha' c)^{-1}(s - (\gamma + \rho s^*) - (\gamma' + \rho' s^*)c) \bmod p$.
If $\alpha + \alpha' c \equiv_p 0$ and $\alpha' \neq 0$ then both Game_2 and Game_3 (and thus $\mathcal{B}$) abort in line (I). Otherwise we must have $\alpha' = 0$ and, from Eq. (39): $x = \gamma' + \rho' s^* \bmod p$. The reduction can thus simulate the decryption query by returning $\mathsf{H}'(xY)$ (which might define a new H' value or not).

This shows that whenever Game_3 differs from Game_2 (in lines (III) or (IV)), reduction $\mathcal{B}$ solves the DL problem, which yields:

$$\mathsf{Adv}^{\mathsf{game}_3}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) \geq \mathsf{Adv}^{\mathsf{game}_2}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) - \mathsf{Adv}^{\mathrm{dl}}_{\mathsf{GrGen},\mathcal{B}}(\lambda) \ . \tag{40}$$

Inspecting Game_3, we note that $\mathcal{A}$'s output is independent of b and if the game aborts it outputs a random bit; we thus have:

$$\mathsf{Adv}^{\mathsf{game}_3}_{\mathcal{A}_{\mathrm{alg}}}(\lambda) = 2 \cdot \Pr[1 \leftarrow \mathsf{Game}_3] - 1 = 0 \ . \tag{41}$$

The theorem now follows from Equations (36), (37), (40) and (41). $\square$

C Blindness of the Clause Blind Schnorr Signatures

In this section we formally prove blindness of the clause blind Schnorr signature scheme CBlSch, whose signing protocol is defined in Figure 9, by reducing it to blindness of the (standard) blind Schnorr signature scheme BlSch (Figure 6).

In the game defining blindness for BlSch, the adversary plays the role of the signer and interacts with oracles that simulate a user running two signing sessions. Oracle U_1 reproduces the first interaction $\mathsf{BlSch.User}_1$ of session i, in which the user sends a challenge c. Oracle U_2 is the second interaction $\mathsf{BlSch.User}_2$, which, once both sessions are finished, outputs the resulting signatures.

The formal game $\mathrm{BLIND}_{\mathsf{BlSch}}$ for adversary $\mathcal{B}$ is specified in Figure 20, where we follow the definition from Hauk, Kiltz and Loss [HKL19]. As usual, $\mathcal{B}$'s advantage is defined as $\mathsf{Adv}^{\mathrm{blind}}_{\mathsf{BlSch},\mathcal{B}}(\lambda) := 2 \cdot \Pr\left[1 \leftarrow \mathrm{BLIND}^{\mathcal{B}}_{\mathsf{BlSch}}(\lambda)\right] - 1$.

Proof of Theorem 4. Figure 20 shows the blindness game for clause blind Schnorr signatures, where we have replaced $\mathsf{CBlSch.User}_1$ and $\mathsf{CBlSch.User}_2$ by their instantiations in terms of $\mathsf{BlSch.User}_1$ and $\mathsf{BlSch.User}_2$: the user first runs two instances of $\mathsf{BlSch.User}_1$, and the signer calls U_2 with an additional input β, which specifies which instance the signer completes.

To reduce blindness of CBlSch to blindness of BlSch, we will guess the bits β_0 and β_1 that the adversary will use in its calls to U_2: game $\mathrm{G}^{\mathcal{A}}(\lambda)$, specified in Figure 20, is defined like $\mathrm{BLIND}^{\mathcal{A}}_{\mathsf{CBlSch}}$, except that it picks two random bits $\hat{\beta}_0$ and $\hat{\beta}_1$ and aborts if its guess was wrong. (We also make a syntactical change in that U_2 continues session $\hat{\beta}_i$ instead of β_i; when $\hat{\beta}_i \neq \beta_i$, the simulation is not correct, but the game ignores $\mathcal{A}$'s output anyway.) When $\hat{\beta}_0 \neq \beta_0$ or $\hat{\beta}_1 \neq \beta_1$, the bit b' is random, so we have

$$\Pr\left[1 \leftarrow \mathrm{G}^{\mathcal{A}}(\lambda) \,\middle|\, \hat{\beta}_0 \neq \beta_0 \vee \hat{\beta}_1 \neq \beta_1\right] = \tfrac{1}{2} \ . \tag{42}$$

On the other hand, when $\hat{\beta}_0 = \beta_0$ and $\hat{\beta}_1 = \beta_1$, the game is the same as the original blindness game, whose output is independent of the guess, which yields

$$\Pr\left[1 \leftarrow \mathrm{G}^{\mathcal{A}}(\lambda) \,\middle|\, \hat{\beta}_0 = \beta_0 \wedge \hat{\beta}_1 = \beta_1\right] = \Pr\left[1 \leftarrow \mathrm{BLIND}^{\mathcal{A}}_{\mathsf{CBlSch}}(\lambda)\right] \ . \tag{43}$$

From Eqs. (42) and (43), we have

$$\Pr\left[1 \leftarrow \mathrm{G}^{\mathcal{A}}(\lambda)\right] = \tfrac{1}{2} \cdot \tfrac{3}{4} + \Pr\left[1 \leftarrow \mathrm{BLIND}^{\mathcal{A}}_{\mathsf{CBlSch}}(\lambda)\right] \cdot \tfrac{1}{4} \ ,$$

and thus

$$\mathsf{Adv}^{\mathrm{G}}_{\mathcal{A}}(\lambda) = 2 \cdot \Pr\left[1 \leftarrow \mathrm{G}^{\mathcal{A}}(\lambda)\right] - 1 = \tfrac{1}{4} \cdot \mathsf{Adv}^{\mathrm{blind}}_{\mathsf{CBlSch},\mathcal{A}}(\lambda) \ . \tag{44}$$

In the remainder of the proof, we will show that the adversary's behavior only changes negligibly when the bit b changes from 0 to 1. To do so, we define $\mathrm{G}^{\mathcal{A}}_0$ and $\mathrm{G}^{\mathcal{A}}_1$ by modifying $\mathrm{G}^{\mathcal{A}}$ as follows: the bit b is fixed to 0 and 1, respectively, and the game *directly* outputs bit b'. The games are specified in Figure 21 and we define $\mathrm{BLIND}^{\mathcal{B}}_{0,\mathsf{BlSch}}$ and $\mathrm{BLIND}^{\mathcal{B}}_{1,\mathsf{BlSch}}$ analogously. We have:

$$\begin{aligned}\mathsf{Adv}^{\mathrm{G}}_{\mathcal{A}}(\lambda) &= \Pr\left[1 \leftarrow \mathrm{G}^{\mathcal{A}}(\lambda) \,\middle|\, b = 1\right] + \Pr\left[1 \leftarrow \mathrm{G}^{\mathcal{A}}(\lambda) \,\middle|\, b = 0\right] - 1 \\ &= \Pr\left[1 \leftarrow \mathrm{G}^{\mathcal{A}}_1(\lambda)\right] - \Pr\left[1 \leftarrow \mathrm{G}^{\mathcal{A}}_0(\lambda)\right] \ . \end{aligned} \tag{45}$$

Game $\mathrm{BLIND}^{\mathcal{B}}_{\mathsf{BlSch}}(\lambda)$

$b \leftarrow_\$ \{0,1\}$
$b_0 := b\,;\ b_1 := 1-b$
$b' \leftarrow \mathcal{B}^{\textsc{Init},\mathrm{U}_1,\mathrm{U}_2}(1^\lambda)$
return $(b' = b)$

$\textsc{Init}(pk, m_0, m_1)$

$sess_0 := \texttt{init}$
$sess_1 := \texttt{init}$

Oracle $\mathrm{U}_1(i, R_i)$

if $i \notin \{0,1\} \vee sess_i \neq \texttt{init}$ **then return** $\perp$
$sess_i := \texttt{open}$
$(state_i, c_i) \leftarrow \mathsf{BlSch.User}_1(pk, R_i, m_{b_i})$
return c_i

Oracle $\mathrm{U}_2(i, s_i)$

if $sess_i \neq \texttt{open}$ **then return** $\perp$
$sess_i := \texttt{closed}$
$\sigma_{b_i} \leftarrow \mathsf{BlSch.User}_2(state_i, s_i)$
if $sess_0 = sess_1 = \texttt{closed}$ **then**
  if $\sigma_0 = \perp \vee \sigma_1 = \perp$ **then** $(\sigma_0, \sigma_1) := (\perp, \perp)$
  return (σ_0, σ_1)
else return ϵ

Game $\mathrm{BLIND}^{\mathcal{A}}_{\mathsf{CBlSch}}(\lambda)$, $\boxed{\mathrm{G}^{\mathcal{A}}(\lambda)}$

$b \leftarrow_\$ \{0,1\}$
$b_0 := b\,;\ b_1 := 1-b$
$\boxed{\hat\beta_0, \hat\beta_1 \leftarrow_\$ \{0,1\}}$
$b' \leftarrow \mathcal{A}^{\textsc{Init},\mathrm{U}_1,\mathrm{U}_2}(1^\lambda)$
$\boxed{\textbf{if } \hat\beta_0 \neq \beta_0 \vee \hat\beta_1 \neq \beta_1 \textbf{ then } b' \leftarrow_\$ \{0,1\}}$
return $(b' = b)$

$\textsc{Init}(pk, m_0, m_1)$

$sess_0 := \texttt{init}$
$sess_1 := \texttt{init}$

Oracle $\mathrm{U}_1(i, R_{i,0}, R_{i,1})$

if $i \notin \{0,1\} \vee sess_i \neq \texttt{init}$ **then return** $\perp$
$sess_i := \texttt{open}$
$(state_{i,0}, c_{i,0}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,0}, m_{b_i})$
$(state_{i,1}, c_{i,1}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,1}, m_{b_i})$
return $(c_{i,0}, c_{i,1})$

Oracle $\mathrm{U}_2(i, s_i, \beta_i)$

if $sess_i \neq \texttt{open}$ **then return** $\perp$
$sess_i := \texttt{closed}$
$\sigma_{b_i} \leftarrow \mathsf{BlSch.User}_2(state_{i,\beta_i}, s_i)$
$\boxed{\sigma_{b_i} \leftarrow \mathsf{BlSch.User}_2(state_{i,\hat\beta_i}, s_i)}$
if $sess_0 = sess_1 = \texttt{closed}$ **then**
  if $\sigma_0 = \perp \vee \sigma_1 = \perp$ **then** $(\sigma_0, \sigma_1) := (\perp, \perp)$
  return (σ_0, σ_1)
else return ϵ

Fig. 20. The blindness game for the blind Schnorr signature scheme BlSch (top) and (bottom) for the clause blind Schnorr signature scheme CBlSch (ignoring boxes) and game G (including the boxes) used in the proof of Theorem 4.

$\mathrm{G}_0^{\mathcal{A}}(\lambda)$ $\boxed{\mathrm{G}_*^{\mathcal{A}}(\lambda)}$ $\boxed{\mathrm{G}_1^{\mathcal{A}}(\lambda)}$

$\hat\beta_0, \hat\beta_1 \leftarrow\!\!\$\ \{0,1\}$
$b' \leftarrow \mathcal{A}^{\textsc{Init},\mathrm{U}_1,\mathrm{U}_2}(1^\lambda)$
if $\hat\beta_0 \neq \beta_0 \vee \hat\beta_1 \neq \beta_1$ **then** $b' \leftarrow\!\!\$\ \{0,1\}$
return b'

$\textsc{Init}(pk, m_0, m_1)$

$sess_0 \coloneqq \texttt{init}$
$sess_1 \coloneqq \texttt{init}$

Oracle $\mathrm{U}_1(i, R_{i,0}, R_{i,1})$ // in G_0 and $\boxed{\mathrm{G}_1}$

if $i \notin \{0,1\} \vee sess_i \neq \texttt{init}$ **then return** $\perp$
$sess_i \coloneqq \texttt{open}$
$(state_{i,0}, c_{i,0}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,0}, m_i)$
$(state_{i,1}, c_{i,1}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,1}, m_i)$
$\boxed{(state_{i,0}, c_{i,0}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,0}, m_{1-i})}$
$\boxed{(state_{i,1}, c_{i,1}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,1}, m_{1-i})}$
return $(c_{i,0}, c_{i,1})$

Oracle $\mathrm{U}_2(i, s_i, \beta_i)$

if $sess_i \neq \texttt{open}$ **then return** $\perp$
$sess_i \coloneqq \texttt{closed}$
$\sigma_i \leftarrow \mathsf{BlSch.User}_2(state_{i,\hat\beta_i}, s_i)$ // only in G_0
$\boxed{\sigma_{1-i} \leftarrow \mathsf{BlSch.User}_2(state_{i,\hat\beta_i}, s_i)}$ // in $\boxed{\mathrm{G}_*}$ and $\boxed{\mathrm{G}_1}$
if $sess_0 = sess_1 = \texttt{closed}$ **then**
  if $\sigma_0 = \perp \vee \sigma_1 = \perp$ **then** $(\sigma_0, \sigma_1) \coloneqq (\perp, \perp)$
  return (σ_0, σ_1)
else return ϵ

Oracle $\mathrm{U}_1(i, R_{i,0}, R_{i,1})$ // only in G_*

if $i \notin \{0,1\} \vee sess_i \neq \texttt{init}$ **then return** $\perp$
$sess_i \coloneqq \texttt{open}$
if $\hat\beta_i = 0$ **then**
  $(state_{i,0}, c_{i,0}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,0}, m_{1-i})$
  $(state_{i,1}, c_{i,1}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,1}, m_i)$
else if $\hat\beta_i = 1$ **then**
  $(state_{i,0}, c_{i,0}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,0}, m_i)$
  $(state_{i,1}, c_{i,1}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,1}, m_{1-i})$
return $(c_{i,0}, c_{i,1})$

Fig. 21. Description of the games G_0 and G_1 which fix the bit b in game G from Figure 20. G_* is a hybrid game that makes the transition between G_0 and G_1.

We now define a hybrid game G_* which lies "between" G_0 and G_1 and is also specified in Figure 21. It differs from G_0 in the $\hat\beta_i$-th message used in signing session i and from G_1 in the $(1-\hat\beta_i)$-th message. Since

$$\mathsf{Adv}^{\mathrm{G}}_{\mathcal{A}}(\lambda) = \Pr[1 \leftarrow \mathrm{G}_1^{\mathcal{A}}(\lambda)] - \Pr[1 \leftarrow \mathrm{G}_*^{\mathcal{A}}(\lambda)] + \Pr[1 \leftarrow \mathrm{G}_*^{\mathcal{A}}(\lambda)] - \Pr[1 \leftarrow \mathrm{G}_0^{\mathcal{A}}(\lambda)] \ , \tag{46}$$

it suffices to bound these two differences. For the first, we construct an adversary $\mathcal{B}_1$ playing game $\mathrm{BLIND}_{\mathsf{BlSch}}$ and simulating G to $\mathcal{A}$ so that if $\mathcal{B}_1$ plays $\mathrm{BLIND}_{0,\mathsf{BlSch}}$, it simulates G_0 to $\mathcal{A}$; whereas if it plays $\mathrm{BLIND}_{1,\mathsf{BlSch}}$, it simulates G_* to $\mathcal{A}$. Adversary $\mathcal{B}_1$ thus embeds its interaction with its challenger as the two sessions that $\mathcal{A}$ will conclude (provided that $\hat\beta_0$ and $\hat\beta_1$ are guessed correctly); it is specified in Figure 22. By inspection, we have

$$\begin{aligned} \Pr\left[1 \leftarrow \mathrm{BLIND}^{\mathcal{B}_1}_{0,\mathsf{BlSch}}(\lambda)\right] &= \Pr\left[1 \leftarrow \mathrm{G}_0^{\mathcal{A}}(\lambda)\right] \ \text{ and} \\ \Pr\left[1 \leftarrow \mathrm{BLIND}^{\mathcal{B}_1}_{1,\mathsf{BlSch}}(\lambda)\right] &= \Pr\left[1 \leftarrow \mathrm{G}_*^{\mathcal{A}}(\lambda)\right] \ . \end{aligned} \tag{47}$$

We also construct an adversary $\mathcal{B}_2$ that simulates game $\mathrm{G}_*^{\mathcal{A}}(\lambda)$ or $\mathrm{G}_1^{\mathcal{A}}(\lambda)$. It embeds its interaction as the sessions that $\mathcal{A}$ will abort and executes the concluding sessions (which are the same in G_* and G_1) on its own. Adversary $\mathcal{B}_2$ is also specified in Figure 22 (note that in

$\mathcal{B}_1^{\textsc{Init}^{\mathcal{B}},\mathrm{U}_1^{\mathcal{B}},\mathrm{U}_2^{\mathcal{B}}}(1^\lambda)$ $\boxed{\mathcal{B}_2^{\textsc{Init}^{\mathcal{B}},\mathrm{U}_1^{\mathcal{B}},\mathrm{U}_2^{\mathcal{B}}}(1^\lambda)}$

$\hat{\beta}_0, \hat{\beta}_1 \leftarrow_{\$} \{0,1\}$
$b' \leftarrow \mathcal{A}^{\textsc{Init}^{\mathcal{A}},\mathrm{U}_1^{\mathcal{A}},\mathrm{U}_2^{\mathcal{A}}}(1^\lambda)$
if $\hat{\beta}_0 \neq \beta_0 \vee \hat{\beta}_1 \neq \beta_1$ **then** // β_i could be $\perp$
  $b' \leftarrow_{\$} \{0,1\}$
return b'

$\textsc{Init}^{\mathcal{A}}(pk, m_0, m_1)$

$\boxed{\begin{array}{l} \mathit{sess}_0 \coloneqq \texttt{init} \\ \mathit{sess}_1 \coloneqq \texttt{init} \end{array}}$
$\textsc{Init}^{\mathcal{B}}(pk, m_0, m_1)$

Oracle $\mathrm{U}_1^{\mathcal{A}}(i, R_{i,0}, R_{i,1})$ // simulated by $\mathcal{B}_1$

if $\hat{\beta}_i = 0$ **then**
  $c_{i,0} \leftarrow \mathrm{U}_1^{\mathcal{B}}(i, R_{i,0})$
  $(\mathit{state}_{i,1}, c_{i,1}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,1}, m_i)$
if $\hat{\beta}_i = 1$ **then**
  $(\mathit{state}_{i,0}, c_{i,0}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,0}, m_i)$
  $c_{i,1} \leftarrow \mathrm{U}_1^{\mathcal{B}}(i, R_{i,1})$
return $(c_{i,0}, c_{i,1})$

Oracle $\mathrm{U}_2^{\mathcal{A}}(i, s_i, \beta_i)$ // simulated by $\mathcal{B}_1$

$\texttt{out} \leftarrow \mathrm{U}_2^{\mathcal{B}}(i, s_i)$
  // $\texttt{out}$ can be ϵ, (σ_0, σ_1), or $(\perp, \perp)$
return $\texttt{out}$

Oracle $\mathrm{U}_2^{\mathcal{A}}(i, s_i, \beta_i)$ // simulated by $\mathcal{B}_2$

if $\mathit{sess}_i \neq \texttt{open}$ **then return** $\perp$
$\mathit{sess}_i \coloneqq \texttt{closed}$
$\sigma_{1-i} \leftarrow \mathsf{BlSch.User}_2(\mathit{state}_{i,\hat{\beta}_i}, s_i)$
if $\mathit{sess}_0 = \mathit{sess}_1 = \texttt{closed}$ **then**
  if $\sigma_0 = \perp \vee \sigma_1 = \perp$ **then** $(\sigma_0, \sigma_1) \coloneqq (\perp, \perp)$
  return (σ_0, σ_1)
else return ϵ

Oracle $\mathrm{U}_1^{\mathcal{A}}(i, R_{i,0}, R_{i,1})$ // simulated by $\mathcal{B}_2$

if $i \notin \{0,1\} \vee \mathit{sess}_i \neq \texttt{init}$ **then return** $\perp$
$\mathit{sess}_i \coloneqq \texttt{open}$
if $\hat{\beta}_i = 0$ **then**
  $(\mathit{state}_{i,0}, c_{i,0}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,1}, m_{1-i})$
  $c_{i,1} \leftarrow \mathrm{U}_1^{\mathcal{B}}(i, R_{i,1})$
if $\hat{\beta}_i = 1$ **then**
  $c_{i,0} \leftarrow \mathrm{U}_1^{\mathcal{B}}(i, R_{i,0})$
  $(\mathit{state}_{i,1}, c_{i,1}) \leftarrow \mathsf{BlSch.User}_1(pk, R_{i,1}, m_{1-i})$
return $(c_{i,0}, c_{i,1})$

Fig. 22. Description of adversaries $\mathcal{B}_1$ and $\mathcal{B}_2$ in the proof of Theorem 4.

its simulation of U_2, the variable $\mathit{state}_{i,\hat{\beta}_i}$ is always defined because of our syntactical change in Figure 20). We have

$$\begin{aligned} \Pr[1 \leftarrow \mathrm{BLIND}_{0,\mathsf{BlSch}}^{\mathcal{B}_2}(\lambda)] &= \Pr[1 \leftarrow \mathrm{G}_*^{\mathcal{A}}(\lambda)] \quad \text{and} \\ \Pr[1 \leftarrow \mathrm{BLIND}_{1,\mathsf{BlSch}}^{\mathcal{B}_2}(\lambda)] &= \Pr[1 \leftarrow \mathrm{G}_1^{\mathcal{A}}(\lambda)] \ . \end{aligned} \tag{48}$$

From Eqs. (46)–(48) we get

$$\mathsf{Adv}_{\mathcal{A}}^{\mathrm{G}}(\lambda) = \mathsf{Adv}_{\mathsf{BlSch},\mathcal{B}_1}^{\mathrm{blind}}(\lambda) + \mathsf{Adv}_{\mathsf{BlSch},\mathcal{B}_2}^{\mathrm{blind}}(\lambda) \ ,$$

which, together with Eq. (44), concludes the proof. $\square$