One-more Unforgeability of Blind ECDSA

Xianrui Qin, Cailing Cai, and Tsz Hon Yuen

The University of Hong Kong, Hong Kong, China
{xrqin,clingcai,thyuen}@cs.hku.hk

Abstract. In this paper, we give the *first* formal security analysis on the one-more unforgeability of blind ECDSA. We start with giving a general attack on blind ECDSA, which is similar to the ROS attack on the blind Schnorr signature. We formulate the ECDSA-ROS problem to capture this attack.
Next, we give a generic construction of blind ECDSA based on an additive homomorphic encryption and a corresponding zero-knowledge proof. Our concrete instantiation is about 40 times more bandwidth efficient than the blind ECDSA in AsiaCCS 2019.
After that, we give the *first* formal proof of one-more unforgeability for blind ECDSA, under a new model called *algebraic bijective random oracle*. The security of our generic blind ECDSA relies on the hardness of a discrete logarithm-based interactive assumption and an assumption of the underlying elliptic curve.
Finally, we analyze the hardness of the ECDSA-ROS problem in the algebraic bijective random oracle model.

Keywords: Blind signature · ECDSA · One-more unforgeability.

1 Introduction

A blind signature scheme [8] consists of an interactive protocol between a user and a signer. The signer holds a secret key sk and the user holds a message m of its choice. After the interaction, the user learns a valid signature σ on the message m. The signer can neither learn the message that it signs, nor link the transcripts of protocol that it creates.

The blind signature can be applied to various privacy sensitive scenarios, such as anonymous credentials, eCash, and e-voting. In particular, for blind ECDSA, Bitcoin developers are exploring its usage to blind coin swaps, trustless tumbler services, and more [17,22].

The security model of blind signature is called the *one-more unforgeability* against chosen message attack. It means that the adversary cannot generate $\ell + 1$ blind signatures from ℓ interactions with the signer. Blind Schnorr signature was shown to be one-more unforgeable if the *one-more discrete logarithm* (OMDL) assumption and the *ROS* assumption hold [20]. Unfortunately, the ROS assumption is recently broken in [5]. The clause blind Schnorr signature [12] is secure based on the OMDL assumption and the modified ROS assumption in the algebraic group model.

Blind ECDSA. For decades of study, many constructions of blind signatures are proposed [8,6,2,1,10,15,13,14,12,19], including those based on ECDSA [16,22]. ECDSA is one of the most widely deployed signature schemes in practice. For a signing key x, the signature on a message m is a pair $\sigma = (r, s)$ satisfying

$$s = k^{-1}(H(m) + xr) \mod p, \quad r = f(kG),$$

where G is a generator of an ECC group of prime order p, k is randomly chosen from $\mathbb{Z}_p$, H is a hash function, and f is a *conversion function* which returns the x-part of the input ECC point modulus p.

Although blind ECDSA has already been explored in 2004 [16], to the best of our knowledge, no formal security proof of one-more unforgeability has been provided for it yet. In [22], the authors only give a heuristic argument for the security of scheme. On the contrary, the security of blind Schnorr signature has been studied for decades [20,12]. It is mainly because an elegant linearity exists in the Schnorr signature, while it disappears in the ECDSA with the usage of division k^{-1} and the conversion function f during the signature generation.

Our Contributions. In this paper, we solve the open problem of constructing a blind ECDSA signature with a *formal* security proof of one-more unforgeability. We have the following contributions.

1. General Attack on One-more Unforgeability. We first demonstrate an attack on the one-more unforgeability of *any* blind ECDSA signature. We propose an *ECDSA-ROS problem* to capture this attack. The hardness of the ECDSA-ROS problem will be further analyzed in §5.
2. Generic Construction with Efficient Instantiation. We propose a generic construction of blind ECDSA from an additive homomorphic encryption and a corresponding non-interactive zero-knowledge (NIZK) proof.
   Our blind ECDSA can be instantiated with the additive homomorphic Castagnos-Laguillaumie (CL) encryption [7] with the corresponding NIZK proof in [23]. Our scheme is about 40 times more bandwidth efficient than [22].
3. Formal Security Proof. We give the *first* formal security proof of one-more unforgeability of blind ECDSA. The proof uses a new model called Algebraic Bijective Random Oracle (ABRO), which is a non-trivial combination of the bijective random oracle (BRO) [9] and the algebraic group model (AGM) [11]. The new ABRO model is of independent interest.
   We show that the one-more unforgeability of our generic blind ECDSA relies on some assumptions related to the discrete logarithm and the underlying elliptic curve.

A high level summary will be given in the rest of this section.

1.1 ECDSA-ROS Attack on Blind ECDSA

Now we consider the following attack on the one-more unforgeability of blind ECDSA. No matter how the blind ECDSA protocol is implemented, the user

eventually obtains ℓ signatures (r_j, s_j) on messages m_j for $j \in [\ell]$ such that:

$$R_j = s_j^{-1}(h_j G + r_j X), \quad r_j = f(R_j), \quad h_j = H(m_j).$$

The adversary can break the one-more unforgeability if he can find (m^*, R^*, s^*) and a vector $\vec{\rho} = (\rho_1, \ldots, \rho_\ell)$ that satisfies:

$$\frac{H(m^*)}{s^*} = \sum_{j=1}^{\ell} \frac{\rho_j h_j}{s_j}, \tag{1}$$

$$\frac{f(R^*)}{s^*} = \frac{r^*}{s^*} = \sum_{j=1}^{\ell} \frac{\rho_j r_j}{s_j}, \tag{2}$$

$$R^* = \sum_{j=1}^{\ell} \rho_j R_j. \tag{3}$$

The one-more forgery includes the extra signature $(r^* = f(R^*), s^*)$ on the message m^*. We call this attack as the *ECDSA-ROS attack*, because Eq. 1 is similar to the ROS attack on the blind Schnorr signature. The *ECDSA-ROS attack* does not rely on solving the discrete logarithm of the public key X.

Hardness of the ECDSA-ROS Problem. We conjecture that the ECDSA-ROS problem is hard to solve even under the recent attack on the ROS problem [5], since the attacker needs to solve three equations simultaneously, with two non-linear functions f and H involved. The hardness of the ECDSA-ROS problem will be further analyzed in §5.

1.2 Generic Construction

Our generic blind ECDSA can be constructed with any additive homomorphic encryption HE and a corresponding NIZK proof. Suppose that the signer knows a secret key x and the user knows the message m. They jointly compute $R = k_a k_b G$, where k_a (resp. k_b) is chosen by the signer (resp. the user). The user encrypts $H(m)$ and $r = f(R)$ with HE and sends the ciphertext to the signer, with a NIZK proof of the well-formedness of the ciphertext. The signer returns the ciphertext of $k_a^{-1}(H(m)+rx)$ using the additive homomorphic property. The user decrypts it and then divides the plaintext with k_b to obtain s. The blind signature is (r, s).

Efficiency Analysis. The blind ECDSA in [22] used a modified Paillier encryption with a modulus $N = pqt$, where q and t are two random large prime numbers. Hence, it is even more inefficient than the standard Paillier encryption. Furthermore, [22] proposed a NIZK proof for the modified Paillier ciphertext with a binary challenge. In order to achieve a soundness error of $2^{-\ell_s}$, the NIZK proof has to be repeated for ℓ_s times. The resulting blind ECDSA in [22] is inefficient.

Our blind ECDSA can be instantiated with the additive homomorphic CL encryption [7] with the corresponding NIZK proof in [23]. Consider 128 bit security level, the ciphertext size of the CL encryption is 3654 bits, about 55% of the

ciphertext size of the modified Paillier encryption (6656 bits). The NIZK proof for CL encryption in [23] is 1488 bytes, but the NIZK proof for modified Paillier encryption in [22] is 69120 bytes for a soundness error of 2^{-80}. Our scheme is about 40 times more bandwidth efficient than [22].

1.3 Algebraic Bijective Random Oracle Model

We propose a new Algebraic Bijective Random Oracle (ABRO) model for the security analysis of blind ECDSA. The idea comes from the bijective random oracle (BRO) [9] and the algebraic group model (AGM) [11]. The BRO was used to prove the unforgeability of ECDSA in [9]. The AGM was used to prove the one-more unforgeability of the clause blind Schnorr signature [12]. Hence, the ABRO model is a reasonable security model to analyze the security of blind ECDSA.

The BRO models the algebraically disruptive behaviour of f similar to the random oracles, which model the disordered behavior of cryptographic hash functions. The BRO decomposes the conversion function f into three independent functions as:

$$f = \varphi \circ \Pi \circ \psi,$$

where φ is a function which maps a group element G into $\{0,1\}^L$ (which is the x-part of G for ECDSA). Π is a bijective mapping from $\{0,1\}^L$ to $[0..2^L-1]$, and ψ is a mapping from $[0..2^L-1]$ to $\mathbb{Z}_p$. The BRO requires that the adversary must query the oracles for the computation of Π and Π^{-1}.

The AGM lies between the standard model and the generic group model. With every group element Z that the adversary outputs, he also gives a representation $\vec{z}$ of Z in terms of the group elements it has received so far. The AGM can be instantiated in an *algebraic wrapper*, as introduced in [3].

New ABRO Model. Our new ABRO model is not a trivial combination of BRO and AGM. Our goal is to minimize the use of AGM in the security model. The ABRO model only requires the adversary to output a representation for the group element R asked in the query $\Pi(\varphi(R))$. The representations are in terms of group elements that has received so far, including the output of the signing oracle and the Π^{-1} oracle.

In contrast with the AGM, the ABRO model does not require the adversary to output representations for group elements used in other oracle queries. Since the mapping Π does not appear in the real scheme, our model does not need to be instantiated with the algebraic wrapper [3].

1.4 Security Proof of Blind ECDSA

ECDSA. As a stepping stone for understanding the security proof for blind ECDSA, we briefly describe the security proof of ECDSA in the ABRO model. The public key X comes from the discrete logarithm (DL) problem instance (G, X). For a valid forgery signature (r^*, s^*) on a message m^*, we have $s^*R^* =$

$H(m^*)G + r^*X$ and $r^* = f(R^*)$. In the BRO, either Π or Π^{-1} must be queried for R^* or r^*. By the setting of the ABRO, either the representation of R^* is known when $\Pi(\varphi(R^*))$ is asked, or the representation of R^* is set by the simulator when $\Pi^{-1}(\psi^{-1}(r^*))$ is asked. In either case, the representation of R^* can be expressed as a pair (a, b), where $R^* = aG + bX$. Hence, the answer to the DL problem $\log_G X$ can be computed from $s^*(aG + bX) = H(m^*)G + r^*X$.

Blind ECDSA. The main contribution of the paper is the reduction of the one-more unforgeability of blind ECDSA to the Multi-Base Discrete Logarithm (MBDL) assumption in the ABRO model. The (n, q)-MBDL is the generalization of the $(n, 1)$-MBDL problem in [4]: Given group elements $(G, X, R_1, \ldots, R_n)$ and q DL oracle queries (which takes (i, P) as input and outputs $\log_{R_i} P$), output $\log_G X$, with the restriction that i must be distinct in all DL oracle queries. This restriction is essential because $\log_G X$ can be easily computed from $\log_{R_i} G$ and $\log_{R_i} X$.

In the security proof of one-more unforgeability, the blind signing oracle is simulated by the DL oracle of the (n, q)-MBDL problem. Roughly speaking, the simulator sets $kG = R_i$ first. If the adversary asks the blind signature s with respect to kG for some r and a message m, the simulator asks the DL oracle with input $(i, H(m)G + rX)$. The DL oracle returns s such that $sR_i = H(m)G + rX$. Hence s can be used to build a valid answer for the blind signing oracle query. Similar to the security proof of ECDSA, we eventually get $aG + bX = 0$ from the forgery signature for some $a, b \in \mathbb{Z}_p$. The value $\log_G X$ can be extracted to answer the MBDL problem. For the case of $a = 0$, we prove that it happens in negligible probability. During the security proof, we also need an assumption on the underlying elliptic curve that for all points on the elliptic curve, there is only one subgroup whose order is p. This assumption is needed to ensure the correctness of the simulation of the BRO oracle. This assumption holds for most common elliptic curves defined in various standards.

1.5 Related Work

In the security analysis of the blind ECDSA in [22], the authors claimed that 'the proposed blind signature scheme has unforgeability if the ECDSA is unforgeable.' However, this claim is not formally proven. In particular, they did not discuss one-more type assumption (like other blind signature schemes) or ROS type assumption (like the blind Schnorr signature).

2 Preliminaries

Notations. We denote the (closed) integer interval from a to b by $[a, b]$. We use $[b]$ as shorthand for $[1, b]$.

2.1 ECDSA

We define a group generation algorithm for elliptic curve.

– GpGen. On input a security parameter λ, it picks a prime q, an elliptic curve E defined over $\mathbb{F}_p$, and a cyclic group $\mathbb{G}$ in $E(\mathbb{F}_p)$ with a generator G of prime order p. Finally, it outputs $(p, \mathbb{G}, G)$.

ECDSA requires two independent functions (denoted as H and f) to map the messages and group elements into a field $\mathbb{Z}_p$ respectively. The function H is a cryptographic hash function. The function f is known as the *conversion function*, mapping a point A to $A.x \bmod p$, which is an encoding of the x-coordinate of A as an integer.

If x is a signing key and $X = xG$ is the corresponding verification key, a signature on a message m is a pair $\sigma = (r, s)$ satisfying $r = f(kG)$ and $s = k^{-1}(H(m) + xr) \bmod p$. Signatures are verified by recovering $R = \frac{H(m)}{s}G + \frac{r}{s}X$ and checking that $f(R) = r$.

2.2 Blind Signature

Syntax. We follow the definition of [12]. A blind signature scheme BS consists of the following algorithms:

– $\mathsf{par} \leftarrow \mathsf{BS.Setup}(1^\lambda)$: the setup algorithm takes the security parameter λ in unary and returns public parameters par;
– $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{BS.KeyGen}(\mathsf{par})$: the key generation algorithm takes the public parameters par and returns a secret/public key pair $(\mathsf{sk}, \mathsf{pk})$;
– $(b, \sigma) \leftarrow \langle \mathsf{BS.Sign}(\mathsf{sk}), \mathsf{BS.User}(\mathsf{pk}, m)\rangle$: 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:

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

(Typically, $\mathsf{BS.User}_0$ just initiates the session, thus $msg_{U,0} = ()$ and $\mathrm{state}_{U,0} = (\mathsf{pk}, m)$.)
– $b' \leftarrow \mathsf{BS.Ver}(\mathsf{pk}, m, \sigma)$: the (deterministic) verification algorithm takes a public key pk, a message m and a signature σ as input, 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 $\mathsf{par} \leftarrow \mathsf{BS.Setup}(1^\lambda)$, $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{BS.KeyGen}(\mathsf{par})$, $(b, \sigma) \leftarrow \langle \mathsf{BS.Sign}(\mathsf{sk}), \mathsf{BS.User}(\mathsf{pk}, m)\rangle$, and $b' \leftarrow \mathsf{BS.Ver}(\mathsf{pk}, m, \sigma)$, we have $b = 1 = b'$ with probability 1.

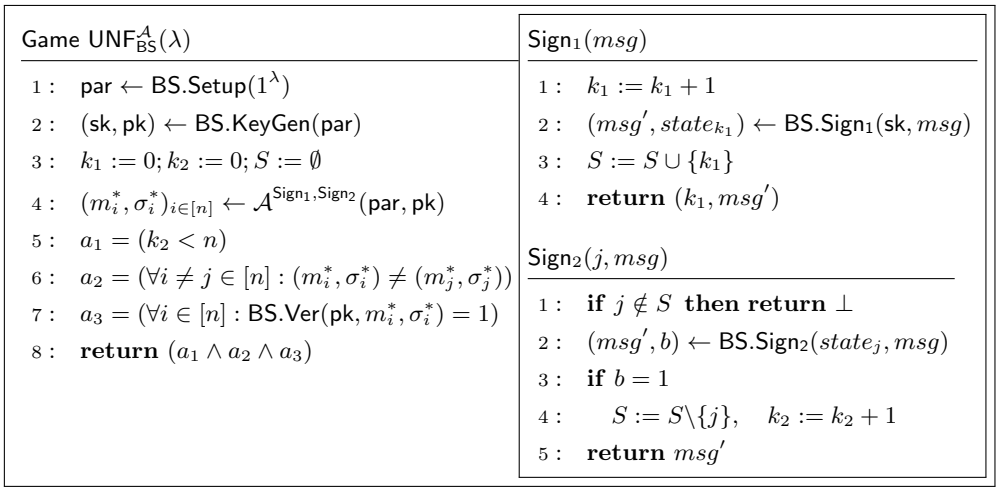

Fig. 1. The one-more unforgeability game for a blind signature scheme BS.

One-more Unforgeability. The standard security notion of blind signatures demands that no user, after arbitrary interactions with a signer and ℓ of these interactions were considered successful by the signer, can produce more than ℓ signatures. Moreover, the adversary can schedule and interleave its sessions with the signer in any arbitrary way.

In Game $\mathsf{UNF}^{\mathcal{A}}_{\mathsf{BS}}(\lambda)$ defined in Fig.1 the adversary has access to two oracles Sign_1 and Sign_2 corresponding to the two phases of the interactive protocol. The game maintains two 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 Sign_1 takes the user's first message, 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 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, representing the number of successful interactions.

Blindness. Blindness requires that a signer cannot link a message/signature pair to a particular execution of the signing protocol. The formal security model of blindness is given in Appendix B.

3 Algebraic Bijective Random Oracle Model

In this paper, we propose a new model called *algebraic bijective random oracle* model (ABRO) for proving the security of blind ECDSA. It is developed from the Bijective Random Oracle (BRO) model and the Algebraic Group Model (AGM).

3.1 AGM and BRO

Algebraic Group Model. The *algebraic group model* (AGM) [11] lies between the standard model and the generic group model. 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 receive 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. Security results in the AGM are proved via reductions to computationally hard problems, like in the standard model.

Bijective Random Oracle Model. As introduced in §1, the BRO does not allow the adversary to compute the conversion function f completely by itself (which is similar to the restriction on computing the hash function in the random oracle model). The BRO decomposes the function f as $f = \varphi \circ \Pi \circ \psi$ for some bijective mapping Π. The adversary has to query an oracle BRO to compute Π, and an oracle BRO^{-1} to compute Π^{-1}.

3.2 Algebraic Bijective Random Oracle Model

We define a new model by demanding the algebraic representation in the BRO. All queries to the BRO oracle with a group element input must come with the corresponding algebraic representation. We call this new model as Algebraic Bijective Random Oracle (ABRO) Model.

Two changes are made from the definition of the BRO model with oracles BRO and BRO^{-1}.

1. The BRO^{-1} oracle takes as input an integer $x \in [0..2^L - 1]$ (L is the bit length of x) and returns $y = \Pi^{-1}(x)$. In addition, the outputs of $\varphi^{-1}(y)$ are added to the list of received group elements. [1]
2. The BRO oracle takes as input a group element R and its algebraic representation $\vec{r}$. The representation is in terms of group elements that it has received so far, including the public key, signatures obtained from the signing oracle, and the group elements defined above by BRO^{-1}. The oracle outputs $x = \Pi(\psi(R))$.

We can see that the algebraic representation requirement in the ABRO is only needed for the query of the BRO oracle. This is the advantage of the ABRO as compared to the trivial combination of the BRO and the AGM.

4 Blind ECDSA

We first give the blind ECDSA protocol. It uses an additive homomorphic encryption and a corresponding non-interactive zero-knowledge (NIZK) proof.

[1] Since φ is a semi-injective function for ECDSA, two group elements are added for each BRO^{-1} query for the case of ECDSA.

4.1 Building Blocks

Additive Homomorphic Encryption. Denote $\mathsf{HE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ as an additive homomorphic encryption scheme, such that $\mathsf{Enc}_{\mathsf{pk}}(m_1) \cdot \mathsf{Enc}_{\mathsf{pk}}(m_2) = \mathsf{Enc}_{\mathsf{pk}}(m_1 + m_2)$. Paillier encryption [18], modified Paillier encryption [22] and CL encryption [7] are some examples of additive homomorphic encryption.

Denote the message space of HE as $\mathcal{M}$. For HE that has a message space $\mathcal{M}$ different from $\mathbb{Z}_p$ (e.g., Paillier encryption), we further require that $\mathcal{M}$ is larger than $p^3 + p^2$.

NIZK Proof. Denote $\mathsf{NIZK} = (\mathsf{Setup}, \mathsf{Pf}, \mathsf{Vf})$ as a non-interactive zero-knowledge proof for the relation $\mathcal{R}$ for the ciphertext of HE:

$$\mathcal{R} = \{(m_1, m_2, r_1, r_2) : c_1 = \mathsf{Enc}_{\mathsf{pk}}(m_1; r_1) \wedge c_2 = \mathsf{Enc}_{\mathsf{pk}}(m_2; r_2) \wedge m_1, m_2 \in \mathbb{Z}_p\},$$

where r_1 (resp. r_2) is the randomness used to encrypt the message m_1 (resp. m_2). If the HE has a message space of $\mathbb{Z}_p$ (e.g., modified Paillier encryption [22], or CL encryption [7]), the range proof of $m_1, m_2 \in \mathbb{Z}_p$ is not needed.

4.2 Construction

The blind ECDSA protocol BS is as follows.

- Setup. On input a security parameter λ, it runs $(p, \mathbb{G}, G) \leftarrow \mathsf{GpGen}(1^\lambda)$ and picks a cryptographic hash function $H : \{0,1\}^* \to \mathbb{Z}_p$. It runs $\mathsf{par}' \leftarrow \mathsf{HE.Setup}(1^\lambda)$ and $\mathsf{crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$. It returns $\mathsf{par} = (p, \mathbb{G}, G, H, \mathsf{par}', \mathsf{crs})$.
- KeyGen. On input par, it picks $\mathsf{sk} := x \leftarrow\!\!\$\, \mathbb{Z}_p$ and computes $\mathsf{pk} := X = xG$.
- Sign, User. The user runs $(\mathsf{upk}, \mathsf{usk}) \leftarrow \mathsf{HE.KeyGen}(\mathsf{par})$ and sends upk to the signer. Then they run the interactive blind signing protocols in Fig. 2.
- Verify. To verify a signature $\sigma = (r, s)$ for a message m and a public key X, it computes $R = \frac{H(m)}{s}G + \frac{r}{s}X$. It returns 1 if $r = f(R)$, or returns 0 otherwise.

4.3 Assumptions

Before proving the security of ECDSA in the ABRO model, we first introduce the assumptions needed for the security proof.

Firstly, we propose an assumption on the underlying elliptic curves chosen by GpGen in the Setup phase above.

Assumption 1. Let $(p, \mathbb{G}, G) \leftarrow \mathsf{GpGen}(\lambda)$ and $E(\mathbb{F}_q)$ is a group of points on the elliptic curve E. There is only one subgroup in $E(\mathbb{F}_q)$ whose order is p.

Assumption 1 is not always true for all elliptic curves. In general, $E(\mathbb{F}_q)$ can contain more than one subgroup whose order is a prime $p > 2$. For example, let E be the elliptic curve $y^2 = x^3 + 2$ over $\mathbb{F}_7$. Then $E(\mathbb{F}_7) = \{\infty, (0,3), (0,4), (3,1), (3,6), (5,1), (5,6), (6,1), (6,6)\}$, and there are four different subgroups whose order is 3:

$$\{(0,3), (0,4), \infty\}, \{(3,1), (3,6), \infty\}, \{(5,1), (5,6), \infty\}, \{(6,1), (6,6), \infty\}.$$

$\mathsf{BS.Sign}((\mathsf{par}, \mathsf{upk}, X), x)$		$\mathsf{BS.User}((\mathsf{par}, \mathsf{upk}, X), m, \mathsf{usk})$
$k_a \xleftarrow{\$} \mathbb{Z}_p$		
$R = k_a G$. *Abort* **if** $R = \mathcal{O}$	$\xrightarrow{R}$	$k_b \xleftarrow{\$} \mathbb{Z}_p, \quad h = H(m)$
		$(r, \cdot) = \tilde{R} = k_b R = k_a k_b G$
		Abort **if** $\tilde{R} = \mathcal{O} \vee r = 0$
		$c_1 = \mathsf{HE.Enc}_{\mathsf{upk}}(r; r_1)$
		$c_2 = \mathsf{HE.Enc}_{\mathsf{upk}}(h; r_2)$
if $\mathsf{NIZK.Vf}((c_1, c_2, \mathsf{upk}), \pi) = 1,$	$\xleftarrow{c_1, c_2, \pi}$	$\pi = \mathsf{NIZK.Pf}((r, h, r_1, r_2), (c_1, c_2, \mathsf{upk}))$
$t \xleftarrow{\$} [0, \lvert\mathcal{M}\rvert/p - p^2 - p]$		
$\bar{k} := k_a^{-1} \mod p$		
$c = c_1^{x\bar{k}} \cdot c_2^{\bar{k}} \cdot \mathsf{HE.Enc}_{\mathsf{upk}}(tp)$	$\xrightarrow{c}$	$s = \mathsf{HE.Dec}_{\mathsf{usk}}(c)/k_b \mod p$
		Abort **if** $s = 0 \wedge \mathsf{Verify}((r, s), m, X) = 0$
		return (r, s)

Fig. 2. The signing protocol of the blind ECDSA signature scheme. If the message space of HE is equal to $\mathbb{Z}_p$, then t can be fixed to 0. Here, r_1, r_2 are Encrandomness. For the range of t, firstly, here t is used for rerandomizing the message($xr\bar{k} + h\bar{k}$) encrypted in the ciphertext c. Secondly, the range in which t is chosen is set to prevent $|xr\bar{k}+h\bar{k}+tp| \geq |\mathcal{M}|$, Otherwise it will result in decryption error of c, i.e., $|xr\bar{k}+h\bar{k}| \neq \mathsf{HE.Dec}_{\mathsf{usk}}(c) \mod p$.

However in practice, the common elliptic curves defined in various standards have the order of the subgroup $\mathbb{G}$ that is closed to the order of the curves themselves. In other words, Assumption 1 holds in practice. For example, for NIST P-256,

$q =$ 0xffffffff 00000001 00000000 00000000 00000000 ffffffff ffffffff ffffffff,
$p =$ 0xffffffff 00000000 ffffffff ffffffff bce6faad a7179e84 f3b9cac2 fc632551.

For secp256k1 used in Bitcoin,

$q =$ 0xffffffff ffffffff ffffffff ffffffff ffffffff fffffffe fffffc2f,
$p =$ 0xffffffff ffffffff ffffffff fffffffe baaedce6 af48a03b bfd25e8c d0364141.

Remark 1. Assumption 1 is crucial for our proof, since the argument that we use in the proof "$pU = \mathcal{O}$ implies $U \in E(\mathbb{F}_q)$ falls in the subgroup $\mathbb{G}$ whose order is p" is not always correct without using assumption 1.

(n, q)-MBDL Problem. We give the (n, q)-MBDL problem in Fig. 3, which is the generalization of the Multi-base Discrete Logarithm (MBDL) problem in [4]. The $(n, 1)$-MBDL problem is introduced by Bellare and Dai [4], and they give a tight security reduction of the Schnorr signature to the $(1, 1)$-MBDL problem.

Game $\mathrm{MBDL}^{\mathcal{A}}_{\mathsf{GpGen},n,q}(\lambda)$	Oracle $\mathrm{DLO}(i, W)$
$(p, \mathbb{G}, G) \leftarrow \mathsf{GpGen}(1^\lambda); S := \emptyset; y \leftarrow\$ \mathbb{Z}_p; Y := yG$	**if** $i \in S$ or $i \notin [n]$, **return** $\perp$
for $i = 1, \ldots, n$,	$S := S \cup \{i\}$
$\quad x_i \leftarrow\$ \mathbb{Z}_p; X_i := x_i G$	**return** $\log_{X_i}(W)$
$y' \leftarrow \mathcal{A}^{\mathrm{DLO}}(p, \mathbb{G}, G, Y, X_1, \ldots, X_n)$	
return $(y' = y \wedge \lvert S \rvert \leq q)$	

Fig. 3. The (n, q)-MBDL problem.

4.4 Security Proof

Theorem 1. *Assume that assumption 1 holds for* GpGen*. Let* $\mathcal{A}_{\mathsf{alg}}$ *be an algebraic adversary against the one-more unforgeability security of the blind ECDSA signature running in time at most* τ *and making at most* ℓ *queries to* Sign_1*,* q_r *queries to the random oracle* H *and* q_b *queries to the bijective random oracles. If* NIZK *has soundness, then there exists an algorithm* $\mathcal{B}_1$ *solving the* (ℓ, ℓ)*-MBDL problem, and an algorithm* $\mathcal{B}_2$ *breaking the collision resistant of* H*, both running in time at most* $\tau + O(\ell + q_r + q_b)$*, such that:*

$$\mathsf{Adv}^{\mathsf{om\text{-}unf}}_{\mathcal{A}_{\mathsf{alg}}}(\lambda) \leq \mathsf{Adv}^{(\ell,\ell)-\mathsf{MBDL}}_{\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{cr}}_{\mathcal{B}_2}(\lambda) + \frac{q_b^2}{(p-1)/2 - q_b} + \left(\frac{q_r \cdot q_b}{p}\right)^{\ell+1}.$$

Proof. The one-more unforgeability of our blind ECDSA is proved by a sequence of games.

$\underline{\mathsf{Game}_0}$. As shown in Fig. 4, the first game is the one-more unforgeability game for scheme BS played with $\mathcal{A}_{\mathsf{alg}}$ in the ABRO model. Hence we have $\mathsf{Adv}^{\mathsf{om\text{-}unf}}_{\mathcal{A}_{\mathsf{alg}}}(\lambda) = \mathsf{Adv}^{\mathsf{Game}_0}_{\mathcal{A}_{\mathsf{alg}}}(\lambda)$.

$\underline{\mathsf{Game}_1}$. By Fig. 4, Game_1 is the same as Game_0 except that in the Fin procedure, it rejects the collision of the challenge message m^* with any message queried in the signing oracle. Hence $\mathsf{Adv}^{\mathsf{Game}_0}_{\mathcal{A}_{\mathsf{alg}}}(\lambda) \leq \mathsf{Adv}^{\mathsf{Game}_1}_{\mathcal{A}_{\mathsf{alg}}}(\lambda) + \mathsf{Adv}^{\mathsf{cr}}_{\mathcal{B}_2}(\lambda)$, where $\mathsf{Adv}^{\mathsf{cr}}_{\mathcal{B}_2}(\lambda)$ is the probability of breaking the collision resistance of H by some algorithm $\mathcal{B}_2$.

$\underline{\mathsf{Game}_2}$. By Fig. 5, Game_2 is the same as Game_1, except that the bijection Π is now implemented by:

- BRO: lazy sampling in $\mathbb{B}$.
- BRO^{-1}: lazy sampling α' in $\mathbb{A}$ and then try to convert it to an ECC point U. If $pU = \mathcal{O}$, it means that $U \in \mathbb{G}$ by Assumption 1. We picks a random $v \in \mathbb{Z}_p$ and sets $\alpha = \varphi(vG)$. Otherwise, we just set $\alpha = \alpha'$.

The input-output pairs (α, β) to these two oracles are stored in Π. By assumption of the BRO model, $\mathcal{A}$ does not output a forgery without having posed the corresponding bijective random oracle query first. Procedure Fin is not affected by the switch to sampling and remains unmodified.

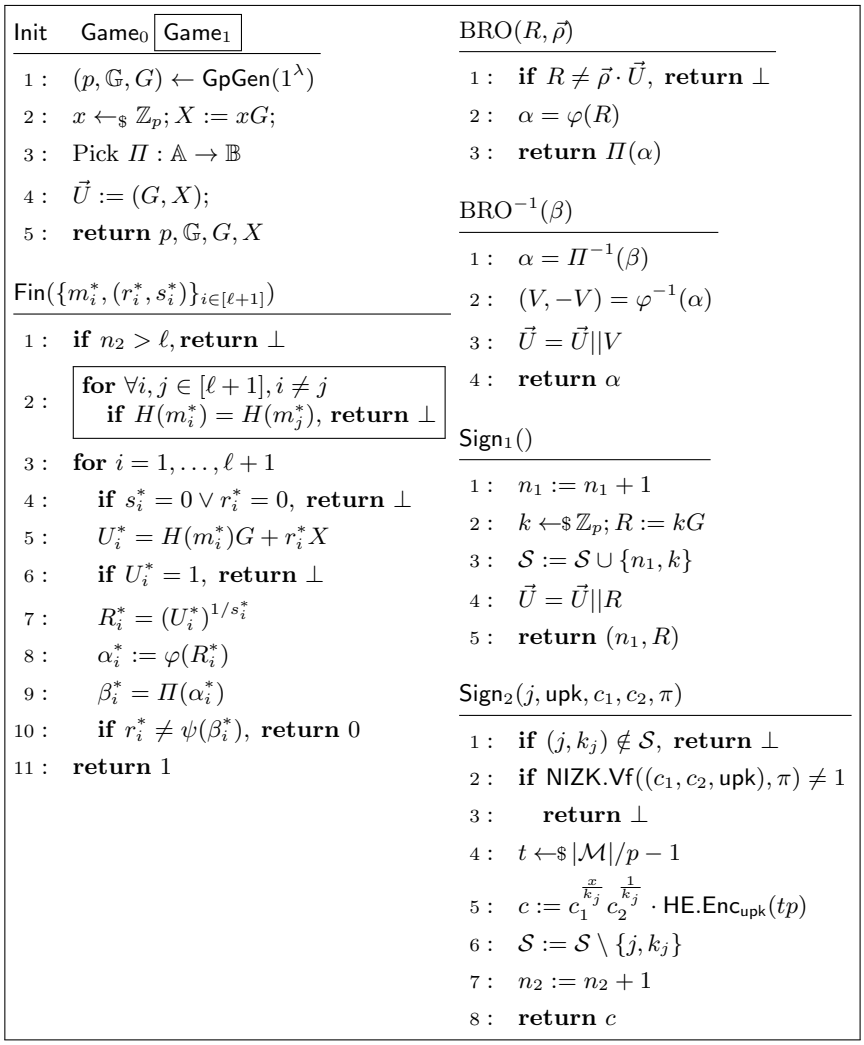

Fig. 4. Game_0 and Game_1 used in the proof of Blind ECDSA

We assess the probability that Game_2 aborts in BRO^{-1} as follows: a uniformly distributed value of a set of cardinality at least $2^L - q_b$ is sampled and checked for containedness in a set of at most q_b elements. That is, the probability of Game_2 aborts in BRO^{-1} is at most $q_b/(2^L - q_b)$. As these lines are executed at most q_b times in total, the overall probability of abort is bounded by $q_b^2/(2^L - q_b)$. Since φ is semi-injective, we have $(p-1)/2 \leq 2^L$. Since Game_1 and Game_2 are identical, if no abort happens, we obtain $\mathsf{Adv}^{\mathsf{Game}_1}_{\mathcal{A}_{\mathsf{alg}}}(\lambda) \leq \mathsf{Adv}^{\mathsf{Game}_2}_{\mathcal{A}_{\mathsf{alg}}}(\lambda) + q_b^2/((p-1)/2 - q_b)$.

Final Reduction. In our last step, we construct an algorithm $\mathcal{B}_1$ solving the (ℓ, ℓ)-MBDL problem whenever $\mathcal{A}_{\mathsf{alg}}$ wins Game_2. Algorithm $\mathcal{B}_1$, which has

$\mathrm{BRO}(R, \vec{\rho})$ $\quad \mathsf{Game}_2$	$\mathrm{BRO}^{-1}(\beta)$
1 : **if** $R \neq \vec{\rho} \cdot \vec{U}$, **return** $\perp$	1 : **if** $(\cdot, \beta) \in \Pi$, **return** $\Pi^{-1}(\beta)$
2 : $\alpha = \varphi(R)$	2 : $\alpha' \leftarrow\$ \mathbb{A} \setminus Dom(\Pi)$
3 : **if** $(\alpha, \cdot) \in \Pi$, **return** $\Pi(\alpha)$	3 : $(U, -U) := \varphi^{-1}(\alpha')$
4 : $\beta \leftarrow\$ \mathbb{B} \setminus Rng(\Pi)$	4 : **if** $pU = \mathcal{O}$
5 : $\Pi \leftarrow \Pi \cup \{(\alpha, \beta)\}$	5 : $\quad v \leftarrow\$ \mathbb{Z}_p; V := vG$
6 : **return** β	6 : **else**
	7 : $\quad V := U$
	8 : $\alpha = \varphi(V)$
	9 : **if** $(\alpha, \cdot) \in \Pi : Abort$
	10 : $\Pi \leftarrow \Pi \cup \{(\alpha, \beta)\}$
	11 : $\vec{U} = \vec{U} \| V$
	12 : **return** α

Fig. 5. Game_2 used in the proof of Blind ECDSA.

access to the oracle DLO takes as input a group description $(p, \mathbb{G}, G)$ and $(Y, X_1, \ldots, X_\ell)$. It sets the public key $X := Y$, and runs $\mathcal{A}_{\mathsf{alg}}$ on input $(p, \mathbb{G}, G, X)$. Each time $\mathcal{A}_{\mathsf{alg}}$ makes a $\mathsf{Sign}_1()$ query, $\mathcal{B}_1$ sets $R_j = X_j$. It simulates $\mathsf{Sign}_2(j, c_1, c_2, \pi)$ in the following way: firstly it uses the extractor of the NIZK to extract $r_j, h_j \in \mathbb{Z}_p$ from the proof π_j. Then it queries the oracle upon $(j, h_j G + r_j X)$ and get $s_j = \log_{R_j}(h_j G + r_j X)$. Finally, $\mathcal{B}_1$ returns c as $\mathsf{HE.Enc}_{\mathsf{upk}}(s_j + tp)$, where $t \leftarrow\$ |\mathcal{M}|/p - 1$[2].

Finally, $\mathcal{A}_{\mathsf{alg}}$ returns (m_i^*, r_i^*, s_i^*) for all $i \in \{\ell + 1\}$. Since the i-th forgery is valid, we have $r_i^* = f(R_i^*)$ and:

$$s_i^* R_i^* = H(m_i^*)G + r_i^* X. \tag{4}$$

There are two possible cases:

1. For all $i \in [\ell + 1]$, $r_i^* = f(R_i^*)$ is queried via BRO.
2. For some $i \in [\ell + 1]$, $\pm R_i^* = f^{-1}(r_i^*)$ is queried in BRO^{-1}.

Case 1. $(\gamma_i^*, \xi_i^*, \vec{\zeta}_i^*, , \vec{\mu}_i^*)$ is a representation of R_i^* asked in BRO, i.e.,

$$R_i^* = \gamma_i^* G + \xi_i^* X + \sum_{j=1}^{\ell} \zeta_{i,j}^* R_j + \sum_{j=1}^{q_b} \mu_{i,j}^* V_j. \tag{5}$$

Denote $\bar{V}_j$ (resp. $V_j' := v_j G$) as the ECC points generated from line 6 (resp. line 4) of BRO^{-1} and $\bar{\mu}_{i,j}$ (resp. $\mu_{i,j}'$) as the corresponding coefficients in equation

[2] This range is set to ensure that the distribution of simulated message $s_j + tp$ is the same as the distribution of the real message $xr\bar{k} + h\bar{k} + tp$.

(5). If $\sum_{\forall j} \bar{\mu}_{i,j}\bar{V}_j \neq 0$, $\mathcal{B}_1$ aborts since R_i^* is not in $\mathbb{G}$. Otherwise, combining Equations (4) and (5), we get

$$s_i^*((\gamma_i^* + \sum_{\forall j} \mu'_{i,j} v_j)G + \xi_i^* X + \sum_{j=1}^{\ell} \zeta_{i,j}^* R_j) = H(m_i^*)G + r_i^* X. \tag{6}$$

By the Sign_2 oracle query, we have $s_j R_j = h_j G + r_j X$. Then we have:

$$\underbrace{[s_i^*(\xi_i^* + \sum_{j=1}^{\ell} \frac{\zeta_{i,j}^* r_j}{s_j}) - r_i^*]}_{=:\chi_i} X + \underbrace{[s_i^*(\gamma_i^* + \sum_{\forall j} \mu'_{i,j} v_j + \sum_{j=1}^{\ell} \frac{\zeta_{i,j}^* h_j}{s_j}) - H(m_i^*)]}_{=:\theta_i} G = 0.$$

If $\chi_i = 0$ for all $i \in [\ell + 1]$, it means

$$s_i^* = (\xi_i^* + \sum_{j=1}^{\ell} \frac{\zeta_{i,j}^* r_j}{s_j})^{-1} r_i^*.^{3} \tag{7}$$

for all $i \in [\ell + 1]$. Plugging (7) in (4), we have

$$r_i^* C_i^* = H(m_i^*)G \tag{8}$$

where $C_i^* = (\xi_i^* + \sum_{j=1}^{\ell} \frac{\zeta_{i,j}^* r_j}{s_j})^{-1} R_i^* - X$. Observe that C_i^* is fixed when $f(R_i^*)$ is queried via BRO[4]. Since r_i^* is randomly chosen after fixing C_i^*, and $h_i^* = H(m_i^*)$ is also randomly chosen from $\mathbb{Z}_p$ in the random oracle $H(\cdot)$, the probability that equation (8) holds is $\frac{q_r \cdot q_b}{p}$ at maximum for each i.

Case 2. By the simulation of BRO^{-1}, if it is computed by line 6 of BRO^{-1}, $\mathcal{B}_1$ aborts since $R_i^* \notin \mathbb{G}$[5]. Otherwise, $R_i^* = \pm v_i G$. Combining it with Equation (4), we have $\pm v_i s_i^* G = H(m_i^*)G + r_i^* X$. Then $\mathcal{B}_1$ can return $(\pm v_i s_i^* - H(m_i^*))/r_i^*$ as the solution to the MBDL problem since $r_i^* \neq 0$.

Hence, we have $\mathsf{Adv}^{\mathsf{Game}_3}_{\mathcal{A}_{\mathsf{alg}}}(\lambda) \leq \mathsf{Adv}^{(\ell,\ell)-\mathsf{MBDL}}_{\mathcal{B}_1}(\lambda) + (\frac{q_r \cdot q_b}{p})^{\ell+1}$ □

Theorem 2. *The blind ECDSA has blindness if* HE *is IND-CPA secure and* NIZK *has zero-knowledge property.*

The security proof of blindness is given in Appendix B.

4.5 EUF-CMA Security of ECDSA in the ABRO Model

Similar to the proof of one-more unforgeability for blind ECDSA, we can prove the EUF-CMA security of ECDSA directly in the ABRO model. As compared to the ECDSA security proof in [9], the advantage of our proof is that our reduction does not involve rewinding, and hence the reduction is tight.

The high level idea of the EUF-CMA security of ECDSA is described in §1.4. We omit the details due to the space limit for the paper submission.

[3] Note that $r_i^* \neq 0$ and $s_i^* \neq 0$ for a valid ECDSA signature. If $\chi_i = 0$, it means that $\xi_i^* + \sum_{j=1}^{\ell} \frac{\zeta_{i,j}^* r_j}{s_j}$ would not be 0.

[4] Also, $f(-R_i^*)$ refers to the same C_i^*.

[5] If R_i^* satisfies $s_i^* R_i^* = H(m_i^*)G + r_i^* X$, there must be $R_i^* \in \mathbb{G}$.

5 Hardness of the ECDSA-ROS Problem

In the previous section, we prove the security of our blind ECDSA without directly using any assumption related to the ECDSA-ROS attack mentioned in §1. In this section, we want to show that the ECDSA-ROS problem is hard to solve if the DL assumption holds in the ABRO and the random oracle model. Hence, we do not need to have an extra ECDSA-ROS assumption in the security proof.

Recall that the ECDSA-ROS problem is that, given (r_j, s_j) on messages m_j for $j \in [\ell]$, output (m^*, R^*, s^*) and a vector ρ such that:

$$\frac{H(m^*)}{s^*} = \sum_{j=1}^{\ell} \frac{\rho_j h_j}{s_j},$$
$$\frac{f(R^*)}{s^*} = \frac{r^*}{s^*} = \sum_{j=1}^{\ell} \frac{\rho_j r_j}{s_j},$$
$$R^* = \sum_{j=1}^{\ell} \rho_j R_j.$$

Theorem 3. *Assume that $\mathcal{A}_{\mathsf{alg}}$ is the adversary solving the ECDSA-ROS problem, with q_r queries to the random oracle H and q_b queries to the bijective random oracle. Then there exists an algorithm $\mathcal{B}$ solving the DL problem such that:*

$$\mathsf{Adv}_{\mathcal{A}_{\mathsf{alg}}}(\lambda) \leq \mathsf{Adv}^{\mathsf{DL}}_{\mathcal{B}}(\lambda) + \frac{q_r \cdot q_b}{p}.$$

Proof. The algorithm $\mathcal{B}$ is given a DL problem (G, Y) and wants to solve $\log_G Y$. Assume that there is an adversary $\mathcal{A}_{\mathsf{alg}}$ that can break the ECDSA-ROS problem. Then $\mathcal{B}$ picks $x \leftarrow\$ \mathbb{Z}_p$ and computes $X = xG$, which is forwarded to $\mathcal{A}_{\mathsf{alg}}$. $\mathcal{B}$ samples random messages m_j for $j \in [\ell]$ and computes ECDSA signatures (r_j, s_j). These are given to the adversary $\mathcal{A}_{\mathsf{alg}}$.

When $\mathcal{A}_{\mathsf{alg}}$ queries the BRO with a new input $(R, \vec{\rho})$, $\mathcal{B}$ returns a random $\beta \leftarrow\$ \mathbb{B} \setminus Rng(\Pi)$ as reply. When $\mathcal{A}_{\mathsf{alg}}$ queries the BRO^{-1} with input β, $\mathcal{B}$ picks a random $\delta \leftarrow\$ \mathbb{Z}_p$ and returns $\varphi(\delta Y)$ as reply. The function H is simulated as a normal random oracle.

Finally, $\mathcal{A}_{\mathsf{alg}}$ outputs (m^*, R^*, s^*) and a vector ρ. There are two cases:

1. $r^* = f(R^*)$ is queried via BRO.
2. $\pm R^* = f^{-1}(r^*)$ is queried in BRO^{-1}.

Case 1: Observe that

$$s^* = H(m^*) / \sum_{j=1}^{\ell} \frac{\rho_j h_j}{s_j} = r^* / \sum_{j=1}^{\ell} \frac{\rho_j r_j}{s_j}.$$

Let $z^* = \sum_{j=1}^{\ell} \frac{\rho_j r_j}{s_j} / \sum_{j=1}^{\ell} \frac{\rho_j h_j}{s_j}$, the above means the adversary can find $m^*, \vec{\rho}$ such that

$$\underbrace{H(m^*)}_{\text{random}} = z^* \underbrace{r^*}_{\text{random}} . \tag{9}$$

But r^* is calculated from the output of BRO is randomly distributed in $\mathbb{Z}_p$ and independent from z^* (which is fixed by $\vec{\rho}$ when $f(R^*)$ is queried), and $H(m^*)$ output is also randomly chosen from $\mathbb{Z}_p$, which means (9) happens with the probability of $\frac{q_r \cdot q_b}{p}$ in maximum.

Case 2: By the simulation of BRO^{-1}, if it is computed by line 6 of BRO^{-1}, $\mathcal{B}_1$ aborts since $R^* \notin \mathbb{G}$. Otherwise, $\mathcal{B}$ returns $\varphi(\delta^* Y)$ as a reply for some $\delta^* \neq 0$. If the adversary can find a valid pair of (m^*, r^*, s^*) and a vector $\vec{\rho} = (\rho_1, \ldots, \rho_\ell)$, we have that satisfies:

$$R^* = f^{-1}(r^*) = \delta^* Y = \sum_{j=1}^{\ell} \frac{\rho_j h_j}{s_j} G + \sum_{j=1}^{\ell} \frac{\rho_j r_j}{s_j} X. \tag{10}$$

then $\mathcal{B}$ forwards $(\sum_{j=1}^{\ell} \frac{\rho_j h_j}{s_j} + \sum_{j=1}^{\ell} \frac{\rho_j r_j}{s_j} x)/\delta^*$ to the challenger as the solution to the DL problem. □

The best attack against the ECDSA-ROS problem. We only reduce the ECDSA-ROS problem to the DL problem in the ABRO model and random oracle model above. Their relation in the standard model is not clear.

We conjecture that the best attack against the ECDSA-ROS problem is to use Wagner's generalized birthday algorithm [21]. We left the analysis on the complexity of attack against the ECDSA-ROS problem as an interesting open problem.

6 Conclusion

ECDSA is a significant signature scheme in applications, especially in cryptocurrency like Bitcoin and Ethereum. Blind signature is also popular in constructing privacy-preserving applications. In this paper, we give the first formal security proof for blind ECDSA. One of the assumptions that we use is the MBDL assumption, which is relatively new. An interesting question is whether we can prove the security of blind ECDSA under some well-studied assumptions, like one-more discrete logarithm assumption. We leave this as future work.

References

1. Abe, M.: A secure three-move blind signature scheme for polynomially many signatures. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. Lecture Notes in Computer Science, vol. 2045, pp. 136–151. Springer (2001)

2. Abe, M., Okamoto, T.: Provably secure partially blind signatures. In: Bellare, M. (ed.) CRYPTO 2000. Lecture Notes in Computer Science, vol. 1880, pp. 271–286. Springer (2000)
3. Agrikola, T., Hofheinz, D., Kastner, J.: On instantiating the algebraic group model from falsifiable assumptions. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. Lecture Notes in Computer Science, vol. 12106, pp. 96–126. Springer (2020)
4. Bellare, M., Dai, W.: The multi-base discrete logarithm problem: Tight reductions and non-rewinding proofs for schnorr identification and signatures. In: Bhargavan, K., Oswald, E., Prabhakaran, M. (eds.) INDOCRYPT 2020. Lecture Notes in Computer Science, vol. 12578, pp. 529–552. Springer (2020)
5. Benhamouda, F., Lepoint, T., Loss, J., Orrù, M., Raykova, M.: On the (in) security of ros. In: EUROCRYPT 2021. pp. 33–53. Springer (2021)
6. Camenisch, J., Piveteau, J., Stadler, M.: Blind signatures based on the discrete logarithm problem. In: Santis, A.D. (ed.) EUROCRYPT '94. Lecture Notes in Computer Science, vol. 950, pp. 428–432. Springer (1994)
7. Castagnos, G., Laguillaumie, F.: Linearly homomorphic encryption from DDH. In: Nyberg, K. (ed.) CT-RSA 2015. Lecture Notes in Computer Science, vol. 9048, pp. 487–505. Springer (2015)
8. Chaum, D.: Blind signatures for untraceable payments. In: Chaum, D., Rivest, R.L., Sherman, A.T. (eds.) CRYPTO '82. pp. 199–203. Plenum Press, New York (1983)
9. Fersch, M., Kiltz, E., Poettering, B.: On the provable security of (EC)DSA signatures. In: Weippl, E.R., Katzenbeisser, S., Kruegel, C., Myers, A.C., Halevi, S. (eds.) CCS 2016. pp. 1651–1662. ACM (2016)
10. Fischlin, M.: Round-optimal composable blind signatures in the common reference string model. In: Dwork, C. (ed.) CRYPTO 2006. Lecture Notes in Computer Science, vol. 4117, pp. 60–77. Springer (2006)
11. Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018. Lecture Notes in Computer Science, vol. 10992, pp. 33–62. Springer (2018)
12. Fuchsbauer, G., Plouviez, A., Seurin, Y.: Blind schnorr signatures and signed elgamal encryption in the algebraic group model. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT 2020. Lecture Notes in Computer Science, vol. 12106, pp. 63–95. Springer (2020)
13. Garg, S., Rao, V., Sahai, A., Schröder, D., Unruh, D.: Round optimal blind signatures. In: Rogaway, P. (ed.) CRYPTO 2011. Lecture Notes in Computer Science, vol. 6841, pp. 630–648. Springer (2011)
14. Hauck, E., Kiltz, E., Loss, J.: A modular treatment of blind signatures from identification schemes. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. Lecture Notes in Computer Science, vol. 11478, pp. 345–375. Springer (2019)
15. Kiayias, A., Zhou, H.S.: Equivocal blind signatures and adaptive uc-security. In: Canetti, R. (ed.) TCC 2008. Lecture Notes in Computer Science, vol. 4948, pp. 340–355. Springer (2008)
16. Metet, A.: Blind signatures with dsa/ecdsa? https://www.metzdowd.com/pipermail/cryptography/2004-April/006790.html
17. Nick, J.: Blind signatures in scriptless scripts. https://building-on-bitcoin.com/docs/slides/Jonas_Nick_BoB_2018.pdf
18. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT '99. Lecture Notes in Computer Science, vol. 1592, pp. 223–238. Springer (1999)

19. Rückert, M.: Lattice-based blind signatures. In: Abe, M. (ed.) ASIACRYPT 2010. Lecture Notes in Computer Science, vol. 6477, pp. 413–430. Springer (2010)
20. Schnorr, C.: Security of blind discrete log signatures against interactive attacks. In: Qing, S., Okamoto, T., Zhou, J. (eds.) ICICS 2001. Lecture Notes in Computer Science, vol. 2229, pp. 1–12. Springer (2001)
21. Wagner, D.A.: A generalized birthday problem. In: Yung, M. (ed.) CRYPTO 2002. Lecture Notes in Computer Science, vol. 2442, pp. 288–303. Springer (2002)
22. Yi, X., Lam, K.Y.: A new blind ecdsa scheme for bitcoin transaction anonymity. In: Galbraith, S.D., Russello, G., Susilo, W., Gollmann, D., Kirda, E., Liang, Z. (eds.) AsiaCCS 2019. pp. 613–620. ACM (2019)
23. Yuen, T.H., Cui, H., Xie, X.: Compact zero-knowledge proofs for threshold ecdsa with trustless setup. In: PKC 2021. pp. 481–511 (2021)

A Comparison with Existing Blind ECDSA Protocols

We compare our blind ECDSA with the blind ECDSA proposed in [22].

Efficiency Analysis. We first analyze the blind ECDSA protocol in [22]. For the security level of 3072-bit RSA, the modified Paillier ciphertext uses N, which is a product of two large primes times p (3328 bits). The modified Paillier ciphertext is an integer modulus N^2, which is 6656 bits. The NIZK proof for each modified Paillier ciphertext is two integers modulus N^2 and an integer modulus p, which is 864 bytes for each NIZK proof. To achieve a soundness error of 2^{-80}, the NIZK proof is run 80 times. The total bandwidth is 69120 bytes.

We can instantiate our blind ECDSA with the CL encryption using class groups of imaginary quadratic order. Consider 128-bit security level, the size of a class group element is 1827 bits [7]. A CL ciphertext has two class group elements, which is 3654 bits. The NIZK proof for CL encryption is 1488 bytes ([23], Table 2) for a soundness error of 2^{-80}.

B Blindness

B.1 Security Model of Blindness

A formal definition of blindness can be found in Fig. 6. 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.

B.2 Security Proof of Blindness

Proof. Let $\mathcal{A}$ be an adversary playing in $\mathsf{Game}\ \mathsf{BLIND}^{\mathcal{B}}_{\mathsf{BS}}(\lambda)$. After its execution, $\mathcal{A}$ holds $(m_0, \sigma_0), (m_1, \sigma_1)$ where σ_0 is a signature on m_0 and σ_1 is a signature on m_1. The adversary $\mathcal{A}$ furthermore learns two transcripts $T_1 =$

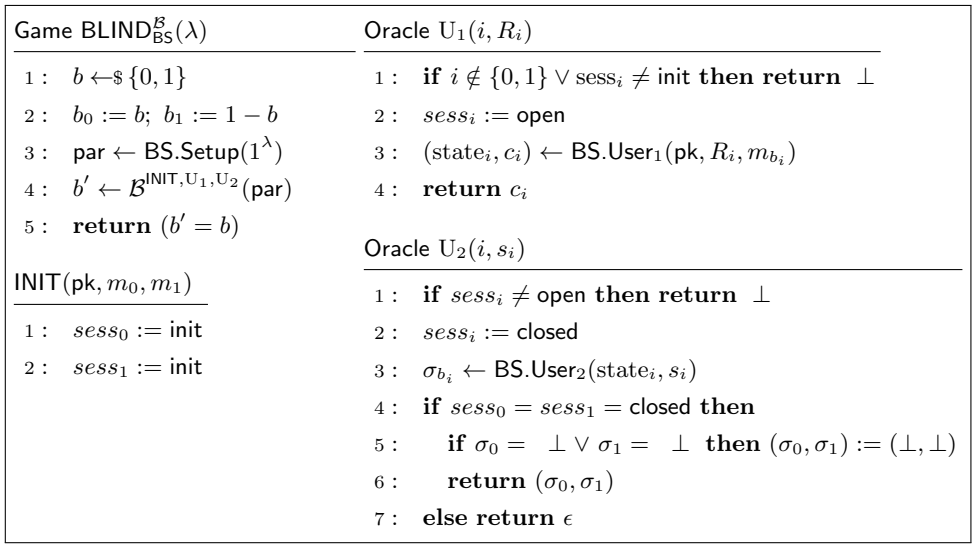

Fig. 6. The blindness game for a blind ECDSA scheme BS.

$(R_1, c'_{1,1}, c'_{1,2}, \pi_1, c_1)$ and $T_2 = (R_2, c'_{2,1}, c'_{2,2}, \pi_2, c_2)$ from its interaction with the first and the second signer session, respectively. The goal of $\mathcal{A}$ is to match the message/signature pairs with the two transcripts.

We show that no adversary is able to distinguish whether the message m_0 was used by the experiment to create the transcript T_1 or T_2. We define Game_1 that is the same as Game BLIND, except that the proof π returned from the Oracle U_1 is replaced by the simulator of the NIZK proof. If NIZK has the zero-knowledge property, then no PPT adversary can distinguish these two games.

We define Game_2 that is the same as Game_1, except that the ciphertext (c_1, c_2) returned from the Oracle U_1 is changed from the encryption of m_{b_i} to m_{1-b_i}. If HE is IND-CPA secure, then no PPT adversary can distinguish these two games.

We define Game_3 that is the same as Game_2, except that the proof π returned from the Oracle U_1 is changed back to the real NIZK proof on message m_{1-b_i}.

Finally, we can see that in Game_3 is the same as the original Game BLIND, except that the bit b is flipped in these two games. □