Designated-Verifier Linkable Ring Signatures with unconditional anonymity

Danai Balla[0000-0002-5117-679X], Pourandokht Behrouz[0000-0002-0653-1430], Panagiotis Grontas[0000-0001-7584-0643], Aris Pagourtzis[0000-0002-6220-3722], Marianna Spyrakou[0000-0001-5694-8405], and Giannis Vrettos[0000-0002-3453-7872]

School of Electrical and Computer Engineering
National Technical University of Athens,
9, Iroon Polytechniou St, Postal Code 157 80, Athens, Greece
balla.danai@gmail.com, pbehrouz@mail.ntua.gr, pgrontas@corelab.ntua.gr, pagour@cs.ntua.gr, mspyrakou@mail.ntua.gr, gianniszvrettos@gmail.com

Abstract. We propose Designated-Verifier Linkable Ring Signatures with unconditional anonymity, a cryptographic primitive that protects the privacy of signers in two ways: Firstly, it allows them to hide inside a ring (i.e. an anonymity set) they can create by collecting a set of public keys all of which must be used for verification. Secondly, it allows a designated entity to simulate signatures thus making it difficult for an adversary to deduce their identity from the content of the exchanged messages. Our scheme differs from similar proposals since the anonymity guarantees are unconditional.

Keywords: ring signatures, unconditional anonymity, designated verifier, non-transferability, linkability

1 Introduction

Ring signatures (RS) [14] provide anonymity to the signer of a message by allowing them to hide within a crowd of peers. While an RS is created by a single private key, it is verified using a *set* of public keys. This set is called a ring and can be created ad-hoc. In RS constructions, anonymity can be unconditional, which means that the signer can be revealed with probability no better than that of a random guess. However, anonymity can be a double-edged sword; while it is necessary for some applications, it may be a thwarting factor in others. For instance, in e-voting, it allows voters to freely express their preferences. If left uncontrolled, though, it can be an enabler for an attacker e.g. by facilitating double-voting. Linkable Ring Signatures (LRS) [10] limit anonymity while preserving its essence; signatures can be grouped by signer (i.e. linked) without giving away its identity. To achieve this, LRS embed a *linking tag* or *pseudoidentity* in the signature which can be used for linking. This tag, however, downgrades LRS anonymity from unconditional to computational. This allows a strong (e.g. quantum) adversary to learn the identity of the signer.

The protection offered by RS is also vulnerable to a side-channel attack against anonymity; the messages exchanged. Indeed, they may contain information that 'leaks' the signer identity. To thwart such an attack, an additional entity able to simulate signatures can be used. Designated-Verifier signatures (DVS) [5] facilitate this scenario, by including the verifier's public key during signing. As a result, a signature effectively states that it originates either from the original signer or the designated verifier (DV). When the DV receives a signature it did not create, only the DV can be sure of its authenticity. This does not apply to the public, as a signature verifies successfully, both when created by the DV (a simulation) or by the original signer.

Designated-Verifier Linkable Ring Signatures (DVLRS) [2], a recent proposal, combines LRS and DVS. Its aim is to increase the privacy of the ring members by also allowing a DV to add noise through messages with simulated signatures. Therefore, it is more difficult to identify a ring member by the content of their messages, because one cannot tell if they are original or simulations. However, DVLRS provide only computational anonymity, a feature inherited from LRS.

Our contribution. This work solves an open problem of [2] by enhancing DVLRS with unconditional anonymity. Our proposed primitive is called *UDVLRS* (Designated-Verifier Linkable Ring Signatures with Unconditional anonymity). Our construction yields a smaller signature than [2]. We formally define the security model of UDVLRS, propose an instantiation and prove its security.

Related work. Liu et al. [10] provide the first LRS construction and define its security properties: *unforgeability*, *linkability* and *anonymity*. They prove that the former two properties hold if the Discrete Logarithm Problem (DLOG) is hard, while the latter rests on the Decisional Diffie Hellman (DDH) assumption. These security properties were refined in [12, 11, 4]. Liu et al. [9] proposed the first unconditionally anonymous LRS scheme. Based upon their construction we introduce a more realistic model for anonymity, by allowing the adversary to access previously known signatures and corrupted public keys. Their work also defined a new security property, *non - slanderability*, which aimed to prevent an adversary from arbitrarily linking signatures to signers that did not create them. DVLRS combined this property with linkability, thus making the security model of the primitive simpler. We find however that this stronger definition of linkability is not compatible with anonymity.

The pseudoidentity in [10] is a function of the ring and the secret signing key. As a result, linking can be performed only to signatures originating from the same ring. In [9] the pseudoidentity is generalized, since it is a function of the secret signing key and a commonly shared string called event ($\mathtt{ev}$). As a result, signatures from different rings can also be linked.

The essential security property for DVS, besides unforgeability, is *non - transferability*. It was formalized in [15] and states that no party except the DV can be convinced that a signature is not a simulation. DVS can be either publicly or privately verifiable. In the latter case they are called *strong* designated-verifier signatures and require the secret key of the DV for verification.

RS and DVS share a similarity [14] facilitated by anonymity. In both cases the public cannot tell who created a signature. Therefore, a ring of two members provides in effect designated-verifier signatures. However, as stated in [2], linkability breaks this connection, because the pseudoidentity can be used to prove that a particular signature is not a simulation. As a result, there have been some attempts in the literature to combine LRS and DVS, but none of them employs the semantics of DVLRS and UDVLRS. In [16, 6] only private verifiability is provided, while in [6, 7] linkability is not considered. DVLRS was the first scheme to combine linkability with a designated verifier and public verifiability, thus achieving noise via simulated signatures. UDVLRS extend DVLRS with unconditional anonymity, while keeping public verifiability and perfect non-transferability. Unforgeability and linkability remain conditional to the hardness of DLOG. Lastly, UDVLRS generalize linking by allowing signatures to be linked from different rings and different designated verifiers.

2 Preliminaries

2.1 Notation

We denote by λ the security parameter. L is a ring consisting of n public keys. L is a subset of the set of all possible public keys $\mathcal{U}$, the size of which is $\mu(\lambda)$. As usual, $[n]$ is the set $\{1, \ldots, n\}$. Equality is denoted with $=$, assignment with $\leftarrow$, definition with $\triangleq$, while $\leftarrow\!\!\$$ denotes a selection of an item from a set uniformly at random. Our security definitions are in the form of games between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. Their input is always the security parameter and their output is the truth value of the condition that will make $\mathcal{A}$ win the game; for brevity, our games return the condition itself. The absence of an output is denoted by $\perp$. Values of no interest in a particular context are denoted by '$\cdot$'. $\mathcal{A}$ has state which is maintained throughout successive operations but is always omitted. We refer to the cryptographic parameters of our constructions as params. They are required in all our algorithms, but we do not include them for conciseness. We denote a public key as pk and a secret key as sk. The pseudoidentity is denoted by pid and the set of all possible pid's by $\mathcal{PID}$. Linking is based on a common string $\mathtt{ev}$ which originates from a set $\mathcal{EID}$. The index of the designated verifier is D, while the index of the signer in the ring is π. Most of the algorithms that constitute our primitive take as common input the values $\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D$ to denote respectively an event, a ring, a message, and the public key of the designated verifier. We collectively refer to these values as the signature parameters.

2.2 Security Assumptions

The security of some facets of our construction rests on a variation of the Discrete Logarithm Problem (DLOG), which is more easy to use. It was first defined in [9] and it was used as a computationally equivalent problem to DLOG. A similar version was introduced in [1] as the Discrete Logarithm Relation. We will be using [9]'s version, which we reintroduce as the Modified Discrete Logarithm Relation (MDLR):

Definition 1. *Modified Discrete Logarithm Relation (MDLR). Let $\mathbb{G}$ be a cyclic group of prime order q generated by g and $Y_1, Y_2, \ldots, Y_n \leftarrow\!\$\ \mathbb{G}, Y_1 \neq 1_{\mathbb{G}}$. A solution to the MDLR problem is a tuple $(\phi_1, \phi_2, \ldots, \phi_n) \in \mathbb{Z}_q^n$ such that $Y_1 \cdot Y_2^{\phi_2} \cdots Y_n^{\phi_n} = g^{\phi_1}$ and $\sum_{i=1}^n \phi_i \neq 0 \pmod q$.*

Note that for $n = 1$, MDLR is the standard DLOG problem.

Proposition 1. DLOG *is computationally equivalent to MDLR.*

Proof. Assuming a DLOG oracle and a MDLR instance $Y_1, Y_2, \ldots, Y_n$ compute $x_1, x_2, \ldots, x_n$ such that $Y_i = g^{x_i}, i \in [n]$. Select $\{\phi_i \in \mathbb{Z}_q\}_{i=2}^n$ and set $\phi_1 \leftarrow x_1 + \sum_{i=2}^n \phi_i x_i \mod q$. In (the negligible) case that $\sum_{i=1}^n \phi_i = 0 \pmod q$ repeat the process. Clearly, $g^{\phi_1} = g^{x_1 + \sum_{i=2}^n \phi_i x_i} = Y_1 \cdot Y_2^{\phi_2} \cdots Y_n^{\phi_n}$.

Assuming a MDLR oracle and a DLOG instance $Y = g^x$, select $x_2, \ldots, x_n$ and compute $\{Y_i \leftarrow g^{x_i}\}_{i=2}^n$. Query the MDLR oracle with $Y, Y_2, \ldots, Y_n$ and receive $\phi_1, \phi_2, \ldots, \phi_n \in \mathbb{Z}_q$ such that $Y \cdot Y_2^{\phi_2} \cdots Y_n^{\phi_n} = g^{\phi_1}$ and $\sum_{i=1}^n \phi_i \neq 0 \pmod q$. Thus $Y = g^{\phi_1 - \sum_{i=2}^n \phi_i x_i}$ and the DLOG of Y is $x = \phi_1 - \sum_{i=2}^n \phi_i x_i \pmod q$. □

3 UDVLRS definition and security model

3.1 UDVLRS definition

Our definition of UDVLRS is a combination of the respective ones in [2, 9]:

Definition 2. *A UDVLRS scheme is a tuple of PPT algorithms $\Pi =$ (Setup, KGen, Sign, Extract, Sim, Vrfy, Link) where:*

- params $\leftarrow$ Setup(λ). *The* Setup *algorithm generates the cryptographic groups for UDVLRS operations as well as the message, signature, pseudoidentity and event spaces ($\mathcal{MSG}, \mathcal{SG}, \mathcal{PID}, \mathcal{EID}$ respectively).*
- (sk, pk) $\leftarrow$ KGen(). *The key generation algorithm.*
- $\sigma \leftarrow$ Sign(ev, L, m, pk$_D$, sk$_\pi$). Sign *is the algorithm that is used to sign a message* m *by some $\pi \in [n]$ for event* ev*, ring L, and designated verifier D.*
- pid $\leftarrow$ Extract(σ). Extract *is an algorithm that can obtain the pseudoidentity* pid *from a signature. It is publicly executable.*
- $\sigma \leftarrow$ Sim(ev, L, m, pk$_D$, sk$_D$, pid). Sim *is the signature simulation algorithm that allows the designated verifier D to produce indistinguishable signatures for pseudoidentity* pid*.*
- $\{0, 1\} \leftarrow$ Vrfy(ev, L, m, pk$_D$, σ). Vrfy *is the public verification algorithm which outputs* 1 *if the signature is valid or* 0 *if it is not.*
- $\{0, 1\} \leftarrow$ Link(σ_1, ev$_1$, σ_2, ev$_2$). Link *is the public linking algorithm which outputs* 1 *if σ_1 and σ_2 originate from the same signer or if they are simulated to look like they originate from the same signer.*

Note that the Sim algorithm requires a pid for a linkable simulation. Thus, it must have seen a signature before. This does not restrict the applications of our primitive, since an initialization phase could be designed where each signer would publish a signed registration message, without sensitive data. Alternatively, signatures with a random pid could be used before or during the registration phase. These would not be linked to any signer.

3.2 Correctness

The completeness of UDVLRS rests on the following two properties:

Verification Correctness. A signature or simulation, for a specific event, ring, message and designated verifier is valid if and only if it was honestly generated (i.e. $\mathsf{Vrfy}(\mathtt{ev},\ L, \mathtt{m}, \mathsf{pk}_D, \sigma) = 1 \Leftrightarrow \sigma = \mathsf{Sign}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{sk}_\pi),\ \mathsf{pk}_\pi \in L$ or $\sigma = \mathsf{Sim}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{sk}_D, \mathsf{pid}),\ (\mathsf{sk}_D, \mathsf{pk}_D) = \mathsf{KGen}(), \mathsf{pid} \in \mathcal{PID}$).

Linking Correctness. Two valid signatures $\sigma_1,\ \sigma_2$ are linked, $(\mathsf{Link}(\sigma_1, \mathtt{ev}_1, \sigma_2, \mathtt{ev}_2) = 1)$, if and only if $\mathtt{ev}_1 = \mathtt{ev}_2$ and one of the following conditions hold:

i $\sigma_1 = \mathsf{Sign}(\mathtt{ev}_1, L_1, \mathtt{m}_1, \mathsf{pk}_{D_1}, \mathsf{sk}_\pi), \mathsf{pk}_\pi \in L_1$ and $\sigma_2 = \mathsf{Sign}(\mathtt{ev}_2, L_2, \mathtt{m}_2, \mathsf{pk}_{D_2}, \mathsf{sk}_\pi)$, $\mathsf{pk}_\pi \in L_2$. Both signatures are honestly generated by the same signer π.

ii $\sigma_1 = \mathsf{Sign}(\mathtt{ev}_1, L_1, \mathtt{m}_1, \mathsf{pk}_{D_1}, \mathsf{sk}_\pi), \sigma_2 = \mathsf{Sim}(\mathtt{ev}_2, L_2, \mathtt{m}_2, \mathsf{pk}_{D_2}, \mathsf{sk}_{D_2}, \mathsf{Extract}(\sigma_1))$, $\mathsf{pk}_\pi \in L_1,\ (\mathsf{sk}_{D_2}, \mathsf{pk}_{D_2}) \leftarrow \mathsf{KGen}()$. σ_1 is honestly generated by signer π and σ_2 by designated verifier D_2 using the pseudoidentity extracted by σ_1.

iii $\sigma_1 = \mathsf{Sim}(\mathtt{ev}_1, L_1, \mathtt{m}_1, \mathsf{pk}_{D_1}, \mathsf{sk}_{D_1}, \mathsf{pid}),\ (\mathsf{sk}_{D_1}, \mathsf{pk}_{D_1}) = \mathsf{KGen}(),\ \mathsf{pid} \in \mathcal{PID}$ and $\sigma_2 = \mathsf{Sim}(\mathtt{ev}_2, L_2, \mathtt{m}_2, \mathsf{pk}_{D_2}, \mathsf{sk}_{D_2}, \mathsf{pid}), (\mathsf{sk}_{D_2}, \mathsf{pk}_{D_2}) = \mathsf{KGen}()$. Both σ_1, σ_2 are simulated by designated verifiers $D_1,\ D_2$ using the same pid.

3.3 Adversarial Capabilities

To model the security UDVLRS, we consider a strong adaptive adversary that has the ability to add new users, corrupt any set of users, and request signatures and simulations from any user and designated verifier. To formally model the capabilities of $\mathcal{A}$ we use the following oracles similar to [9, 8, 2]:

- $\mathsf{pk} \leftarrow \mathcal{JO}()$. The *Joining Oracle*, adds a public key to the list of public keys $\mathcal{U}$, and returns it.
- $\mathsf{sk} \leftarrow \mathcal{CO}(\mathsf{pk})$. The *Corruption Oracle*, receives a public key pk that is an output of $\mathcal{JO}$ and returns the corresponding secret key sk. It models the ability of $\mathcal{A}$ to control some members of $\mathcal{U}$.
- $\sigma \leftarrow \mathcal{SO}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{pk}_\pi)$. The *Signing Oracle* receives the signature parameters and public key $\mathsf{pk}_\pi \in L$ and outputs a signature σ such that $\sigma \leftarrow \mathsf{Sign}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{sk}_\pi)$ and $(\mathsf{pk}_\pi, \mathsf{sk}_\pi) \leftarrow \mathsf{KGen}()$.
- $\sigma \leftarrow \mathcal{MO}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{pid})$. The *Simulation Oracle* receives the signature parameters and a pseudoidentity pid and outputs a signature σ such that $\sigma \leftarrow \mathsf{Sim}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{sk}_D, \mathsf{pid})$ and $(\mathsf{pk}_D, \mathsf{sk}_D) \leftarrow \mathsf{KGen}()$.

We model hash functions as random oracles [3] denoted as $\mathcal{RO}$. For simplicity, we abuse notation, and also denote the set of an oracle's answers by its name.

3.4 Unforgeability

Unforgeability requires that only a ring member or the designated verifier can produce signatures or simulations that verify successfully. To formally define it for a UDVLRS scheme Π, we consider the experiment $\mathsf{Exp}^{\text{unf}}_{\mathcal{A},\Pi}$ in Game 1.1. The adversary queries all the oracles according to any adaptive strategy. Then $\mathcal{A}$ chooses the signature parameters and creates a forged signature σ^*. The adversary succeeds if the signature verifies, and none of the keys of L, nor pk_D, have been queried to $\mathcal{CO}$ and if the signature is not a query output of $\mathcal{SO}$ or $\mathcal{MO}$.

Definition 3. *Unforgeability. A UDVLRS scheme Π is unforgeable if for any* PPT *adversary $\mathcal{A}$:* $\mathsf{Adv}^{\text{unf}}_{\mathcal{A}}(\lambda) \triangleq \Pr\Big[\mathsf{Exp}^{\text{unf}}_{\mathcal{A},\Pi}(\lambda) = 1\Big] \leq \mathsf{negl}(\lambda)$.

3.5 Anonymity

Unconditional anonymity means that for any (unbounded) adversary $\mathcal{A}$ it should be impossible to find the public key of the signer of a specific signature for a ring L with probability greater than that of random sampling.

More formally, consider the interaction between an unbounded adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ in Game 1.2. The adversary queries the $\mathcal{JO}, \mathcal{CO}, \mathcal{SO}, \mathcal{MO}$ according to any adaptive strategy, sets pk_D, and forms a ring L with any subset of n private keys. $\mathcal{MO}$ does not give any advantage to $\mathcal{A}$ that cannot be obtained with the calls to $\mathcal{SO}$ since the signatures are not constructed with any key from the ring, but are simulations. Assume that from the calls of the $\mathcal{CO}$, $\mathcal{A}$ has obtained m_1 private keys. Using its unlimited computational power and the pseudoidentity, $\mathcal{A}$ might also obtain m_2 private keys using signatures from $\mathcal{SO}$, where the signer is known. The public keys obtained in this way may be contained in L, but $n > m_1 + m_2 + 1$ should hold. $\mathcal{A}$ gives $\mathcal{C}$ an event $\mathtt{ev}$, a message $\mathtt{m}$, the set of public keys L, and the designated verifier public key pk_D. $\mathcal{C}$ picks $\pi \leftarrow\!\!\$\ [n]$ and constructs a challenge signature $\sigma_c = \mathsf{Sign}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{sk}_\pi)$ and gives it to $\mathcal{A}$, who must guess π. The adversary wins the game if it correctly guesses the signer, and its private key has not been obtained through the $\mathcal{CO}$ or $\mathcal{SO}$.

Definition 4. *Anonymity. A UDVLRS scheme Π is unconditionally anonymous if for any unbounded adversary $\mathcal{A}$:* $\mathsf{Adv}^{\text{anon}}_{\mathcal{A}}(\lambda) \triangleq |\Pr[\mathsf{Exp}^{\text{anon}}_{\mathcal{A},\Pi}(\lambda) = 1] - \frac{1}{n-m_1-m_2}| = 0$.

Note that in schemes like [10, 2], where there is *one-to-one* correspondence between private and public key, unconditional anonymity is unattainable, since an unbounded adversary might recover the key from the linking tag by 'reversing' the function that connects them. Our construction avoids this pitfall.

3.6 Non-Transferability

Non-Transferability means that the public cannot tell if a valid signature originates from the Sign or the Sim algorithm. In essence, this property ensures that a simulation is only distinguishable by the designated verifier and the owner of the pseudoidentity used to create it. The formal definition is given in Game 1.3.

The adversary is considered to be computationally unbounded, and it is given access to $\mathcal{CO}, \mathcal{SO}, \mathcal{MO}$ for the same reasons as in subsection 3.5. It may query them with any adaptive strategy at any point. The adversary $\mathcal{A}$ chooses the signature parameters. The challenger then produces the signature σ and the simulation σ' with the same pseudoidentity pid and one of the two is given randomly to $\mathcal{A}$, who must now guess whether it received a signature or a simulation.

Definition 5. *Non-transferability. A UDVLRS scheme Π is perfectly non-transferable if for any unbounded adversary $\mathcal{A}$:* $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{trans}}(\lambda) \triangleq \Pr\left[\mathsf{Exp}_{\mathcal{A},\Pi}^{\mathrm{trans}}(\lambda) = 1\right] - \frac{1}{2} = 0$.

3.7 Linkability

Linkability means that all signatures of the same signer should be linked, while all other security properties are preserved. Simulated signatures generated by the designated verifier for a specific pid are linked to this pid's signatures. Game 1.4 captures the definition of linkability. In this game, the adversary $\mathcal{A}$ tries to generate two unlinked signatures with *a single* secret key. $\mathcal{A}$ queries all the oracles according to any adaptive strategy to generate two signatures σ_1 and σ_2. The adversary can pick the public keys from two separate rings and select two different designated verifiers. $\mathcal{A}$ wins if both signatures verify and are not linked.

Definition 6. *Linkability. A UDVLRS scheme Π is linkable if for any* PPT *adversary $\mathcal{A}$:* $\mathsf{Adv}_{\mathcal{A}}^{\mathrm{link}}(\lambda) \triangleq \Pr\left[\mathsf{Exp}_{\mathcal{A},\Pi}^{\mathrm{link}}(\lambda) = 1\right] \leq \mathsf{negl}(\lambda)$.

Note that our model in Game 1.4 differs from the corresponding one in [2] which stated that $\mathcal{A}$ could not produce $k+1$ pairwise unlinked signatures by having access to k signing keys. In our case, $\mathcal{A}$ can only have access to a single private key instead of more. This restriction of linkability allows our scheme to have unconditional anonymity, but has the downside that the scheme is prone to linkability attacks, if there is collusion of signers or if the adversary has access to more than one private keys. As a result, this weaker linkability together with the unforgeability property do not imply non-slanderability as in [2]. In the next section we define the notion of non-slanderability.

3.8 Non-Slanderability

Non-slanderability prevents framing, by not allowing adversarial attempts to link signatures to a specific ring member. Therefore, if a signature is linked to another one, it is either generated by the same signer or by the corresponding designated verifier. This is captured in Game 1.5. The adversary queries all the oracles, according to any adaptive strategy, and chooses the signature parameters and the public key of a selected signer pk_π, and gives them to challenger $\mathcal{C}$. Then, $\mathcal{C}$

using the Sign algorithm for the private key sk_π, produces a signature σ_1. Note that pk_π - chosen by $\mathcal{A}$ - should not have been asked to $\mathcal{CO}$ or included as public key of any query to $\mathcal{SO}$. Then, $\mathcal{A}$ queries the oracles with the same restrictions for pk_π and produces new signature parameters (except for $\mathtt{ev}$) and σ_2, different from σ_1. $\mathcal{A}$ wins if σ_2 verifies and σ_1 and σ_2 are linked.

Definition 7. *Non-Slanderability. A UDVLRS scheme Π is non-slanderable if for any* PPT *adversary $\mathcal{A}$:* $\mathsf{Adv}_{\mathcal{A}}^{\text{sland}}(\lambda) \triangleq \Pr\Big[\mathsf{Exp}_{\mathcal{A},\Pi}^{\text{sland}}(\lambda) = 1\Big] \leq \mathsf{negl}(\lambda)$.

4 Our construction

Our proposed construction that implements the functionalities of section 3 is depicted in Figure 1. Our signatures (excluding the linking tag) consist of $2n+4$ elements in contrast with [2] that consist of $3n+1$. Thus, while they remain linear to the size of the ring they are in practice shorter.

5 Security Analysis

The completeness of our construction is straightforward. We now prove that both an honestly generated plain and simulated signatures verify correctly and that the scheme has linking correctness.

Lemma 1. *An honestly generated* UDVLRS σ *verifies correctly.*

Proof. The completeness of the signature scheme follows from Eq. 1 since:

$$g^{\tilde{x}}h^{\tilde{y}}\prod_{i=1}^{n} Z_i^{c_i+w_i} = g^{r_x-(c_\pi+w_\pi)x_\pi}h^{r_y-(c_\pi+w_\pi)y_\pi}\prod_{i=1}^{n} Z_i^{c_i+w_i} =$$

$$g^{r_x}h^{r_y}(g^{x_\pi}h^{y_\pi})^{-(c_\pi+w_\pi)}\prod_{i=1}^{n} Z_i^{c_i+w_i} = g^{r_x}h^{r_y}Z_\pi^{-(c_\pi+w)}Z_\pi^{(c_\pi+w)}\prod_{\substack{i\in[n]\\ i\neq\pi}} Z_i^{c_i+w_i} = K$$

Similarly:

$$e^{\tilde{x}}t^{\sum_{i=1}^{n} c_i+w_i} = e^{r_x-(c_\pi+w_\pi)x_\pi}t^{c_\pi+w_\pi}t^{\sum_{\substack{i\in[n]\\ i\neq\pi}} c_i+w_i} =$$

$$e^{r_x}t^{-(c_\pi+w_\pi)}t^{c_\pi+w_\pi}t^{\sum_{\substack{i\in[n]\\ i\neq\pi}} c_i+w_i} = e^{r_x}t^{\sum_{\substack{i\in[n]\\ i\neq\pi}} c_i+w_i} = K'$$

It is also clear by construction that for an honestly generated signature indeed:

$$h^s\mathsf{pk}_D^r\prod_{i=1}^{n} g^{w_i} = K''$$

Therefore $\mathsf{Vrfy}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \sigma) = 1$ $\square$

Game 1.1: Unforgeability experiment $\mathsf{Exp}^{\text{unf}}_{\mathcal{A},\Pi}$

```
$\mathsf{params} \leftarrow \Pi.\mathsf{Setup}(1^\lambda)$
$\mathcal{U} \leftarrow \{(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \Pi.\mathsf{KGen}()\}_{i=1}^{\mu(\lambda)}$
$(\sigma^*, \mathtt{ev}, L = \{\mathsf{pk}_i\}_{i=1}^{n}, \mathtt{m}, \mathsf{pk}_D) \leftarrow \mathcal{A}^{\mathcal{RO},\mathcal{IO},\mathcal{CO},\mathcal{SO},\mathcal{MO}}(\mathcal{U})$
return $\mathsf{Vrfy}(\mathtt{ev}, \sigma, L, \mathtt{m}, \mathsf{pk}_D) = 1$ AND $\forall i \in \mathcal{CO},\ \mathsf{pk}_i \notin L$ AND $D \notin \mathcal{CO}$ AND
$\sigma^* \notin \mathcal{SO}$ AND $\sigma^* \notin \mathcal{MO}$
```

Game 1.2: Anonymity experiment $\mathsf{Exp}^{\text{anon}}_{\mathcal{A},\Pi}$

```
$\mathsf{params} \leftarrow \Pi.\mathsf{Setup}(1^\lambda)$
$\mathcal{U} \leftarrow \{(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \Pi.\mathsf{KGen}()\}_{i=1}^{\mu(\lambda)}$
$(\mathtt{ev}, L = \{\mathsf{pk}_i\}_{i=1}^{n}, \mathtt{m}, \mathsf{pk}_D) \leftarrow \mathcal{A}^{\mathcal{IO},\mathcal{CO},\mathcal{SO},\mathcal{MO}}(\mathcal{U})$
$\pi \leftarrow\!\!\$\ [n]$
$\sigma_c \leftarrow \Pi.\mathsf{Sign}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{sk}_\pi)$
$\xi \leftarrow \mathcal{A}^{\mathcal{CO},\mathcal{SO},\mathcal{MO}}(L, \mathtt{m}, \sigma_c)$
return $\xi \neq \perp$ AND $\xi = \pi$ AND $\pi \notin \mathcal{CO}$ AND $\pi$ cannot be obtained from $\sigma \in \mathcal{SO}$
```

Game 1.3: Non-transferability experiment $\mathsf{Exp}^{\text{trans}}_{\mathcal{A},\Pi}$

```
$\mathsf{params} \leftarrow \Pi.\mathsf{Setup}(1^\lambda)$
$\mathcal{U} \leftarrow \{(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \Pi.\mathsf{KGen}()\}_{i=1}^{\mu(\lambda)}$
$(\mathtt{ev}, L = \{\mathsf{pk}_i\}_{i=1}^{n}, \mathtt{m}, \mathsf{pk}_D, \mathsf{pk}_\pi) \leftarrow \mathcal{A}^{\mathcal{IO},\mathcal{CO},\mathcal{SO},\mathcal{MO}}(\mathcal{U})$
$\sigma_0 \leftarrow \Pi.\mathsf{Sign}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{sk}_\pi)$
$\mathsf{pid} \leftarrow \Pi.\mathsf{Extract}(\sigma_0)$
$\sigma_1 \leftarrow \Pi.\mathsf{Sim}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{sk}_D, \mathsf{pid})$
$b \leftarrow\!\!\$\ \{0,1\}$
$b' \leftarrow \mathcal{A}^{\mathcal{CO},\mathcal{SO},\mathcal{MO}}(L, \mathtt{m}, \sigma_b)$
return $b = b'$
```

Game 1.4: Linkability experiment $\mathsf{Exp}^{\text{link}}_{\mathcal{A},\Pi}$

```
$\mathsf{params} \leftarrow \Pi.\mathsf{Setup}(1^\lambda)$
$\mathcal{U} \leftarrow \{(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \Pi.\mathsf{KGen}()\}_{i=1}^{\mu(\lambda)}$
$(\sigma_1, \sigma_2, \mathtt{ev}, L_1 = \{\mathsf{pk}_i\}_{i=1}^{n_1}, L_2 = \{\mathsf{pk}_i\}_{i=1}^{n_2}, \mathtt{m}_1, \mathtt{m}_2, \mathsf{pk}_{D_1}, \mathsf{pk}_{D_2}) \leftarrow$
  $\mathcal{A}^{\mathcal{RO},\mathcal{IO},\mathcal{CO},\mathcal{SO},\mathcal{MO}}(\mathcal{U})$
return $|\mathcal{CO}| = 1$ AND $\sigma_1, \sigma_2 \notin \mathcal{SO}$ AND $\mathsf{Vrfy}(\mathtt{ev}, \sigma_1, L_1, \mathtt{m}_1, \mathsf{pk}_{D_1}) =$
$1$ AND $\mathsf{Vrfy}(\mathtt{ev}, \sigma_2, L_2, \mathtt{m}_2, \mathsf{pk}_{D_2}) = 1$ AND $\mathsf{Link}(\sigma_1, \mathtt{ev}, \sigma_2, \mathtt{ev}) = 0$
```

Game 1.5: Non-slanderability experiment $\mathsf{Exp}^{\text{sland}}_{\mathcal{A},\Pi}$

```
$\mathsf{params} \leftarrow \Pi.\mathsf{Setup}(1^\lambda)$
$\mathcal{U} \leftarrow \{(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \Pi.\mathsf{KGen}()\}_{i=1}^{\mu(\lambda)}$
$(\mathtt{ev}, L_1 = \{\mathsf{pk}_i\}_{i=1}^{n_1}, \mathtt{m}_1, \mathsf{pk}_{D_1}, \mathsf{pk}_\pi) \leftarrow \mathcal{A}^{\mathcal{RO},\mathcal{IO},\mathcal{CO},\mathcal{SO},\mathcal{MO}}(\mathcal{U})$
$\sigma_1 \leftarrow \Pi.\mathsf{Sign}(\mathtt{ev}, L_1, \mathtt{m}_1, \mathsf{pk}_{D_1}, \mathsf{sk}_\pi)$
$(\sigma_2, L_2 = \{\mathsf{pk}_i\}_{i=1}^{n_2}, \mathtt{m}_2, \mathsf{pk}_{D_2}) \leftarrow \mathcal{A}^{\mathcal{RO},\mathcal{IO},\mathcal{CO},\mathcal{SO},\mathcal{MO}}(\mathcal{U})$
return $\sigma_2 \neq \sigma_1$ AND $\mathsf{Vrfy}(\mathtt{ev}, \sigma_2, L_2, \mathtt{m}_2, \mathsf{pk}_{D_2}) = 1$ AND $\sigma_2 \notin \mathcal{SO}$ AND $\sigma_2 \notin$
$\mathcal{MO}$ AND $\pi \notin \mathcal{CO}$ AND $D_2 \notin \mathcal{CO}$ AND $\mathsf{Link}(\sigma_1, \mathtt{ev}, \sigma_2, \mathtt{ev}) = 1$
```

- $\mathsf{params} \leftarrow \mathsf{Setup}(\lambda)$:
  Return $\mathbb{G}$ of order q, where the DLOG problem is hard, two random generators $g, h \in \mathbb{G}$ and two hash functions $\mathsf{H}_{\mathbb{G}} : \{0,1\}^* \to \mathbb{G}$ and $\mathsf{H}_q : \{0,1\}^* \to \mathbb{Z}_q$. Also return $\mathcal{MSG} = \{0,1\}^*$, $\mathcal{SG} = \mathbb{G} \times \mathbb{Z}_q^{2n+4}$, $\mathcal{PID} = \mathbb{G}$, $\mathcal{EID} = \{0,1\}^*$. Note that the signature space is related to the size of the ring and that the relative discrete logarithms of g, h should not be known.
- $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}()$:
  Each user i samples $x_i, y_i \leftarrow\$ \mathbb{Z}_q$, computes $Z_i \leftarrow g^{x_i} h^{y_i}$ and sets $\mathsf{sk}_i \leftarrow (x_i, y_i)$, $\mathsf{pk}_i = Z_i$. The designated verifier sets $\mathsf{sk}_D \leftarrow (x_D, y_D), \mathsf{pk}_D \leftarrow Z_D$.
- $\sigma \leftarrow \mathsf{Sign}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{sk}_\pi)$:
  The signer π picks $r_x, r_y, r, s, \{c_i\}_{\substack{i \in [n] \\ i \neq \pi}}, \{w_i\}_{i \in [n]} \leftarrow\$ \mathbb{Z}_q$ and computes:
$$e \leftarrow \mathsf{H}_{\mathbb{G}}(\mathtt{ev}), \qquad t \leftarrow e^{x_\pi}, \qquad K \leftarrow g^{r_x} h^{r_y} \cdot \prod_{\substack{i \in [n] \\ i \neq \pi}} Z_i^{c_i + w_i},$$
$$K' \leftarrow e^{r_x} \cdot t^{\sum_{\substack{i \in [n] \\ i \neq \pi}} c_i + w_i}, \qquad K'' \leftarrow h^s \mathsf{pk}_D^r \cdot \prod_{i=1}^n g^{w_i}$$
  Then it computes c_π such that $\sum_{i=1}^n c_i \mod q = \mathsf{H}_q(\mathtt{m}, L, \mathtt{ev}, t, K, K', K'')$ and computes:
$$\tilde{x} \leftarrow (r_x - (c_\pi + w_\pi) x_\pi) \mod q, \qquad \tilde{y} \leftarrow (r_y - (c_\pi + w_\pi) y_\pi) \mod q$$
  The signature is the tuple $\sigma = (t, \tilde{x}, \tilde{y}, r, s, \{c_i\}_{i=1}^n, \{w_i\}_{i=1}^n)$.
- $\mathsf{pid} \leftarrow \mathsf{Extract}(\sigma)$
  Parse σ as the tuple $(t, \tilde{x}, \tilde{y}, r, s, \{c_i\}_{i=1}^n, \{w_i\}_{i=1}^n)$ and return t.
- $\sigma \leftarrow \mathsf{Sim}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \mathsf{sk}_D, \mathsf{pid})$: The designated verifier picks $\chi, \psi, \alpha, \beta, \gamma$, $\{c_i\}_{i=2}^n$, $\{w_i\}_{i=2}^n \leftarrow\$ \mathbb{Z}_q$, and $t = \mathsf{pid}$ as pseudoidentity and then computes:
$$K_D \leftarrow g^\chi \cdot h^\psi \cdot Z_1^\alpha \cdot \prod_{i=2}^n Z_i^{c_i + w_i},$$
$$K'_D \leftarrow e^\chi t^{\alpha + \sum_{i=2}^n c_i + w_i}, \qquad K''_D \leftarrow g^\beta h^\gamma \cdot \prod_{i=2}^n g^{w_i}$$
  Computes c_1 such that $\sum_{i=1}^n c_i \mod q = \mathsf{H}_q(\mathtt{m}, L, \mathtt{ev}, t, K_D, K'_D, K''_D)$ and sets:
$$\tilde{x} \leftarrow \chi, \qquad \tilde{y} \leftarrow \psi, \qquad w_1 \leftarrow \alpha - c_1 \mod q,$$
$$r \leftarrow (\beta - w_1) x_D^{-1} \mod q, \qquad s \leftarrow \gamma - r y_D \mod q$$
  The simulated signature is the tuple $\sigma = (t, \tilde{x}, \tilde{y}, r, s, \{c_i\}_{i=1}^n, \{w_i\}_{i=1}^n)$.
  Note that Sim can use any c_k, w_k for any $k \in [n]$ instead of c_1, w_1.
- $\{0,1\} \leftarrow \mathsf{Vrfy}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \sigma)$:
  Parse σ as the tuple $(t, \tilde{x}, \tilde{y}, r, s, \{c_i\}_{i=1}^n, \{w_i\}_{i=1}^n)$ and compute the value:
$$c_0 \leftarrow \mathsf{H}_q(\mathtt{m}, L, \mathtt{ev}, t, g^{\tilde{x}} h^{\tilde{y}} \cdot \prod_{i=1}^n Z_i^{c_i + w_i}, e^{\tilde{x}} \cdot t^{\sum_{i=1}^n c_i + w_i}, h^s \mathsf{pk}_D^r \cdot \prod_{i=1}^n g^{w_i}) \qquad (1)$$
  Return 1 (valid) if $c_0 = \sum_{i=1}^n c_i \pmod q$ else return 0 (invalid).
- $\{0,1\} \leftarrow \mathsf{Link}(\sigma_1, \mathtt{ev}_1, \sigma_2, \mathtt{ev}_2)$:
  Return 1 (linked) if $\mathtt{ev}_1 = \mathtt{ev}_2$ AND $\mathsf{Extract}(\sigma_1) = \mathsf{Extract}(\sigma_2)$ and both signatures verify otherwise return 0 (unlinked).

Fig. 1. The UDVLRS construction

Lemma 2. *A simulated* UDVLRS σ *verifies correctly.*

Proof. A simulated signature verifies since by Eq. 1:

$$g^{\tilde{x}} h^{\tilde{y}} \prod_{i=1}^{n} Z_i^{c_i+w_i} = g^{\chi} h^{\psi} Z_1^{c_1+w_1} \prod_{i=2}^{n} Z_i^{c_i+w_i} = g^{\chi} h^{\psi} Z_1^{\alpha} \prod_{i=2}^{n} Z_i^{c_i+w_i} = K_D$$

Similarly:

$$e^{\tilde{x}} t^{\sum_{i=1}^{n} c_i+w_i} = e^{\chi} t^{c_1+w_1} t^{\sum_{i=2}^{n} c_i+w_i} = e^{\chi} t^{\alpha+\sum_{i=2}^{n} c_i+w_i} = K'_D$$

For the final part of the signature:

$$h^s \mathsf{pk}_D^r \prod_{i=1}^{n} g^{w_i} = h^s \mathsf{pk}_D^r g^{w_1} \prod_{i=2}^{n} g^{w_i} =$$

$$g^{\beta-rx_D} h^{\gamma-ry_D} \mathsf{pk}_D^r \prod_{i=2}^{n} g^{w_i} = g^{\beta} h^{\gamma} (g^{x_D} h^{y_D})^{-r} \mathsf{pk}_D^r \prod_{i=2}^{n} g^{w_i} = K''_D$$

Therefore $\mathsf{Vrfy}(\mathtt{ev}, L, \mathtt{m}, \mathsf{pk}_D, \sigma) = 1$ □

Lemma 3. UDVLRS *have the verification correctness property.*

Proof. The proof is a direct consequence of Lemma 1 and Lemma 2. □

Lemma 4. UDVLRS *have the linking correctness property.*

Proof. Assume two signatures σ_1, σ_2 generated on the same event $\mathtt{ev}$.

- If both were honestly created by the same signer π with $\mathsf{sk}_\pi = (x_\pi, y_\pi)$ then $\mathsf{Extract}(\sigma_1) = \mathsf{Extract}(\sigma_2) = (H_{\mathbb{G}}(\mathtt{ev}))^{x_\pi}$. Thus $\mathsf{Link}(\sigma_1, \mathtt{ev}, \sigma_2, \mathtt{ev}) = 1$.
- If σ_1 is honestly generated by signer π and σ_2 is a simulation with linking tag $t = \mathsf{Extract}(\sigma_1)$ then $\mathsf{Link}(\sigma_1, \mathtt{ev}, \sigma_2, \mathtt{ev}) = 1$.
- If σ_1, σ_2 are simulations with the same linking tag, $\mathsf{Link}(\sigma_1, \mathtt{ev}, \sigma_2, \mathtt{ev}) = 1$. □

The Security properties are proved by the following theorems.

Theorem 1 (Unforgeability). *Our UDVLRS scheme is unforgeable in the random oracle model if* DLOG *is hard in* $\mathbb{G}$.

Proof. Assume a PPT adversary $\mathcal{A}$ that manages to output a forgery, i.e. a valid signature $\sigma = (t, \tilde{x}, \tilde{y}, r, s, \{c_i\}_{i=1}^n, \{w_i\}_{i=1}^n)$ for some event $\mathtt{ev}$, ring of n members $L = \{Z_i\}_{i=1}^n$ with non-negligible probability. We will construct a PPT adversary $\mathcal{B}$ that using $\mathcal{A}$ will either break the DLOG instance $\{X_D \in \mathbb{G} : X_D = g^{x_D},\, x_D \in \mathbb{Z}_q\}$ or solve the MDLR problem for a non-empty subset of $\{X_i \in \mathbb{G} : X_i = g^{x_i}, x_i \in \mathbb{Z}_q\}_{i=1}^n$ with non-negligible probability.

The input of $\mathcal{B}$ consists of $\mathbb{G}, g, q, \{X_i\}_{i=1}^n \cup \{X_D\}$. $\mathcal{B}$ simulates the environment for $\mathcal{A}$:

- $\mathcal{B}$ returns $\mathbb{G}, g, q$ when $\mathcal{A}$ executes Setup.
- Upon the execution of KGen, $\mathcal{B}$ selects $x' \leftarrow\!\$\ \mathbb{Z}_q$, sets $h = g^{x'}$ and provides g, h to $\mathcal{A}$.
- Simulation of $\mathcal{RO}$ oracle. $\mathcal{B}$ simulates the $\mathsf{H}_{\mathbb{G}}$ random oracle by returning g^a for some $a \leftarrow\!\$\ \mathbb{Z}_q$ and H_q by returning b for some $b \leftarrow\!\$\ \mathbb{Z}_q$.
- Simulation of $\mathcal{JO}$ oracle. Since $\mathcal{A}$ is PPT, the maximum number of queries to $\mathcal{JO}$ will be $n' = \mathsf{poly}(\lambda)$ where $n' \geq n+1$. $\mathcal{B}$ selects uniformly at random a subset of indices $\mathcal{I}_{n+1} \subseteq [n']$. W.l.o.g. $\mathcal{I}_{n+1} = [n+1]$. The i-th $\mathcal{JO}$ query is answered as follows [1]:
  - If $i \in \mathcal{I}_{n+1}$, $\mathcal{B}$ selects $y_i \in \mathbb{Z}_q$ and returns $Z_i = X_i h^{y_i}$.
  - If $i \notin \mathcal{I}_{n+1}$, $\mathcal{B}$ selects $x_i, y_i \in \mathbb{Z}_q$ and returns $g^{x_i} h^{y_i}$.
- Simulation of $\mathcal{CO}$ oracle.
  - If $i \in \mathcal{I}_{n+1}$, $\mathcal{B}$ halts, as it does not know the discrete logarithm of X_i.
  - If $i \notin \mathcal{I}_{n+1}$, $\mathcal{B}$ returns $x_i, y_i \in \mathbb{Z}_q$ as they were returned when answering the respective $\mathcal{JO}$ query.
- Simulation of $\mathcal{SO}$ oracle. The input to this oracle is a message $\mathtt{m}$, event $\mathtt{ev}$, some ring $L = \{Z_i\}_{i=1}^n$, the public key of the designated verifier pk_D, and an index π which indicates that the signer must use the secret key corresponding to Z_π. If $\pi \notin \mathcal{I}_n$ then $\mathcal{B}$ knows the full private key (x_i, y_i) and as a result it signs by executing the Sign algorithm. If $\pi \in \mathcal{I}_n$, then $\mathcal{B}$ must simulate the signature as it only knows y_π:
  - $\mathcal{B}$ sets $t \leftarrow X_\pi^a$ where $e = g^a$ was the answer to $\mathsf{H}_{\mathbb{G}}(\mathtt{ev})$. This means that $t = e^{x_\pi}$.
  - $\mathcal{B}$ picks $\tilde{x}, \tilde{y}, r, s, \{c_i\}_{i=1}^n, \{w_i\}_{i=1}^n \leftarrow\!\$\ \mathbb{Z}_q$ and programs the random oracle to answer $\mathsf{H}_q(\mathtt{m}, L, \mathtt{ev}, t, g^{\tilde{x}} h^{\tilde{y}} \cdot \prod_{i=1}^n Z_i^{c_i + w_i}, e^{\tilde{x}} \cdot t^{\sum_{i=1}^n c_i + w_i}, h^s \mathsf{pk}_D^r \cdot \prod_{i=1}^n g^{w_i})$ with the value $\sum_{i=1}^n c_i$.
  - By construction, the simulated signature $\sigma = (t, \tilde{x}, \tilde{y}, r, s, \{c_i\}_{i=1}^n, \{w_i\}_{i=1}^n)$ is valid and indistinguishable from the output of the Sign algorithm.
- Simulation of $\mathcal{MO}$ oracle. Assume that i_D was the query to $\mathcal{JO}$ that asked the pk_D. If $i_D \notin \mathcal{I}_{n+1}$ then $\mathcal{B}$ knows the full designated-verifier private key (x_D, y_D) and as a result it can simulate using the Sim algorithm. If $i_D \in \mathcal{I}_{n+1}$, then $\mathcal{B}$ must program the random oracle to produce indistinguishable simulated signatures. This is easier to accomplish than the $\mathcal{SO}$ oracle, since in Sim that tag is a random group element:
  - $\mathcal{B}$ selects $t \leftarrow\!\$\ \mathbb{G}$.
  - $\mathcal{B}$ picks $\tilde{x}, \tilde{y}, r, s, \{c_i\}_{i=1}^n, \{w_i\}_{i=1}^n \leftarrow\!\$\ \mathbb{Z}_q$ and programs the random oracle to answer with the value $\sum_{i=1}^n c_i$ to the query $\mathsf{H}_q(\mathtt{m}, L, \mathtt{ev}, t, g^{\tilde{x}} h^{\tilde{y}} \cdot \prod_{i=1}^n Z_i^{c_i + w_i}, e^{\tilde{x}} \cdot t^{\sum_{i=1}^n c_i + w_i}, h^s \mathsf{pk}_D^r \cdot \prod_{i=1}^n g^{w_i})$.
  - By construction, the simulated signature $\sigma = (t, \tilde{x}, \tilde{y}, r, s, \{c_i\}_{i=1}^n, \{w_i\}_{i=1}^n)$ is valid and indistinguishable from the output of the Sim algorithm.

The unforgeability adversary $\mathcal{A}$ manages after interactions with the oracles $\mathcal{RO}, \mathcal{JO}, \mathcal{CO}, \mathcal{SO}, \mathcal{MO}$ controlled by $\mathcal{B}$ to output a signature forgery $\sigma_1^* =$

[1] We assume w.l.o.g that $X_D = X_{n+1}$

$(t_1, \tilde{x}_1, \tilde{y}_1, r_1, s_1, \{c_{i1}\}_{i=1}^n, \{w_{i1}\}_{i=1}^n)$ for some event $\mathtt{ev}$, ring L. In order to produce the signature $\mathcal{A}$ must have queried H_q on $\mathtt{ev}$, L, some message $\mathtt{m}$, some $t \in \mathbb{G}$, i.e. $\mathsf{H}_q(\mathtt{m}, L, \mathtt{ev}, t, K_1, K_1', K_1'')$. Since σ_1^* is a valid forgery:

$$K_1 = g^{\tilde{x}_1} h^{\tilde{y}_1} \cdot \prod_{i=1}^n Z_i^{c_{i1}+w_{i1}}$$

Assume that the value K_1 was asked during the l-th query and $\mathcal{B}$ replied with c_{01}. Again, since σ_1^* is valid:

$$c_{01} = \sum_{i=1}^n c_{i1} \pmod q$$

We will prove that $\mathcal{B}$ can either recover the discrete logarithm of $X_D = g^{x_D}$ or solve the MDLR problem for a subset of $\{X_i \in \mathbb{G} : X_i = g^{x_i}, x_i \in \mathbb{Z}_q\}_{i=1}^n$ using the rewinding technique [13]. $\mathcal{B}$ rewinds $\mathcal{A}$ and answers all queries up to l consistently, but answers the l-th query with $c_{02} \neq c_{01}$. By the forking lemma [13], $\mathcal{A}$ will produce another forgery $\sigma_2^* = (t_2, \tilde{x}_2, \tilde{y}_2, r_2, s_2, \{c_{i2}\}_{i=1}^n, \{w_{i2}\}_{i=1}^n)$ with non-negligible probability in polynomial time. However $K_1 = K_2$, $K_1' = K_2'$, $K_1'' = K_2''$ for both l-th queries. This means that:

$$g^{\tilde{x}_1} h^{\tilde{y}_1} \cdot \prod_{i=1}^n Z_i^{c_{i1}+w_{i1}} = g^{\tilde{x}_2} h^{\tilde{y}_2} \cdot \prod_{i=1}^n Z_i^{c_{i2}+w_{i2}} \text{ and} \tag{2}$$

$$h^{s_1} \cdot \mathsf{pk}_D^{r_1} \cdot g^{\sum_{i=1}^n w_{i1}} = h^{s_2} \cdot \mathsf{pk}_D^{r_2} \cdot g^{\sum_{i=1}^n w_{i2}} \tag{3}$$

By Eq. 3, we have that:

$$g^{r_1 \cdot x_D + \sum_{i=1}^n w_{i1}} \cdot h^{s_1 + r_1 \cdot y_D} = g^{r_2 \cdot x_D + \sum_{i=1}^n w_{i2}} \cdot h^{s_2 + r_2 \cdot y_D} \Rightarrow$$

$$r_1 \cdot x_D + \sum_{i=1}^n w_{i1} = r_2 \cdot x_D + \sum_{i=1}^n w_{i2}$$

If $r_1 \neq r_2$:

$$x_D = \frac{\sum_{i=1}^n w_{i1} - \sum_{i=1}^n w_{i2}}{r_2 - r_1}$$

Therefore, $\mathcal{B}$ has successfully computed the discrete logarithm of X_D.

In the second case, if $r_1 = r_2$ then $\sum_{i=1}^n w_{i1} = \sum_{i=1}^n w_{i2}$ and since $K_1 = K_2$, by equation 2 we get the system of equations:

$$g^{\tilde{x}_1} \cdot h^{\tilde{y}_1} \cdot \prod_{i=1}^n Z_i^{c_{i1}+w_{i1}} = g^{\tilde{x}_2} \cdot h^{\tilde{y}_2} \cdot \prod_{i=1}^n Z_i^{c_{i2}+w_{i2}} \Rightarrow$$

$$g^{\tilde{x}_1 + \sum_{i=1}^n x_i(c_{i1}+w_{i1})} \cdot h^{\tilde{y}_1 + \sum_{i=1}^n y_i(c_{i1}+w_{i1})} = g^{\tilde{x}_2 + \sum_{i=1}^n x_i(c_{i2}+w_{i2})} \cdot h^{\tilde{y}_2 + \sum_{i=1}^n y_i(c_{i2}+w_{i2})} \Rightarrow$$

$$g^{\tilde{x}_1} \cdot \prod_{i=1}^n X_i^{c_{i1}+w_{i1}} = g^{\tilde{x}_2} \cdot \prod_{i=1}^n X_i^{c_{i2}+w_{i2}} \Rightarrow$$

$$\prod_{i=1}^n X_i^{c_{i1}+w_{i1}-(c_{i2}+w_{i2})} = g^{\tilde{x}_2 - \tilde{x}_1} \tag{4}$$

We note that there exists $i \in [n]$ such that $c_{i1} + w_{i1} \neq c_{i2} + w_{i2}$, since if $c_{i1} + w_{i1} = c_{i2} + w_{i2}$, $\forall i \in [n]$ then $\sum_{i=1}^{n} c_{i1} + w_{i1} = \sum_{i=1}^{n} c_{i2} + w_{i2} \xRightarrow{\sum_{i\in[n]} w_{i1} = \sum_{i\in[n]} w_{i2}} \sum_{i=1}^{n} c_{i1} = \sum_{i=1}^{n} c_{i2} \Rightarrow c_{01} = c_{02}$ which yields a contradiction.

Assume that there are exactly k indices, $i_1, i_2, \dots, i_k$ $1 \leq k \leq n$, such that $c_{i_j1} + w_{i_j1} \neq c_{i_j2} + w_{i_j2}$, $j \in [k]$. Then by Eq. 4 we have that:

$$X_{i_1} \cdot X_{i_2}^{\phi_2} \cdots X_{i_k}^{\phi_k} = g^{\phi_1}$$

where $\phi_1 = \frac{\tilde{x}_2 - \tilde{x}_1}{c_{i_11} + w_{i_11} - c_{i_12} - w_{i_12}}$ and $\phi_j = \frac{c_{i_j1} + w_{i_j1} - c_{i_j2} - w_{i_j2}}{c_{i_j1} + w_{i_j1} - c_{i_j2} - w_{i_j2}}$, $j \in [k]$. As a result, we have found a solution $\phi_1, \dots \phi_k$ for the MDLR problem for $\{X_i \in \mathbb{G} : X_i = g^{x_i}, x_i \in \mathbb{Z}_q\}_{i=1}^{k}$.

The forking lemma guarantees that the running time of $\mathcal{B}$ is polynomial and that its success probability is non-negligible, thus arriving at a contradiction. $\square$

Theorem 2 (Anonymity). *Our UDVLRS scheme is unconditionally anonymous.*

Proof. We will prove that our scheme is unconditionally anonymous following the techniques used in [9], properly augmented to accommodate for the introduction of the designated verifier. More specifically, we will show that a signature generated using a non-corrupted public key is equally possible to have been created by any uncompromised signer. In addition the usage of the map $f(x, y) = g^x h^y$, which is a *q-to-one* function whose image is uniformly distributed over $\mathbb{G}$, since g, h are generators, protects against an unbounded adversary. This is due to the fact that even if $\mathcal{A}$ is capable of solving the DLOG problem and thus find the x component of sk, it still cannot find to which public key it corresponds to, without extra information, due to the fact that every public key is possible to be produced using that x value.

For every query that the adversary $\mathcal{A}$ makes to the $\mathcal{JO}$, a value $Z = g^x h^y$ is returned for some random pair (x, y). The challenger, $\mathcal{C}$, generates a challenge signature $\sigma_c = (t, \tilde{x}, \tilde{y}, r, s, \{c_i\}, \{w_i\})$ using the key of a random signer from the given ring. The proof is going to show that for any adversary $\mathcal{A}$, the advantage that it has is zero under no computational assumption. Assume w.l.o.g that the private keys, $\mathcal{A}$ has not obtained through $\mathcal{CO}$ or $\mathcal{SO}$ are the first items in $[n - m_1 - m_2]$.

First we prove that for every signer $\pi \in [n - m_1 - m_2]$ and every possible public key $Z = g^{x_\pi} h^{y_\pi}$ there exists a pair (x_π, y_π) such that $t = e^{x_\pi}$. Furthermore we show that for every such pair there exists randomness (r_{x_π}, r_{y_π}) such that σ_c is constructed using that randomness and lastly we prove that these four values $(x_\pi, y_\pi, r_{x_\pi}, r_{y_\pi})$ are uniformly distributed and as such the challenge signature can come from any signer in the ring.

$\mathcal{A}$ can obtain a value x from $t = e^x$ and a value l from $g = h^l$. Thus, $Z_i = h_i^z, i \in [n - m_1 - m_2]$ and for all $\pi \in [n - m_1 - m_2]$:

$$\begin{aligned} x_\pi &= x \pmod{q} \\ y_\pi &= z_\pi - x_\pi l \pmod{q} \end{aligned}$$

We see that $Z_\pi = h^{z_\pi} = h^{y_\pi + x_\pi l} = g^{x_\pi} h^{y_\pi}$, that is the pair (x_π, y_π) constitutes a private key pair that corresponds to the public key Z_π, and that $t = e^x = e^{x_\pi}$.

For every private key (x_π, y_π), consider the following values:

$$r_{x_\pi} = \tilde{x} - (c_\pi + w_\pi)x_\pi \pmod q$$
$$r_{y_\pi} = \tilde{y} - (c_\pi + w_\pi)y_\pi \pmod q$$

It is easy to see that σ_c can be constructed by the private key (x_π, y_π) using randomness (r_{x_π}, r_{y_π}), by any signer $\pi \in [n - m_1 - m_2]$.

We will now show that the elements constituting every tuple $(x_\pi, y_\pi, r_{x_\pi}, r_{y_\pi})$ follow the same distribution over $\mathbb{Z}_q$. Since $x_\pi = x \pmod q$ and $x \leftarrow\$ \mathbb{Z}_q$ we have that x_π is uniformly distributed over $\mathbb{Z}_q$. By the definition of y_π it follows that it is also uniformly distributed over $\mathbb{Z}_q$. For r_{x_π}, and by the same argument for r_{y_π}, we have that it follows the uniform distribution over $\mathbb{Z}_q$ since it is calculated using the value w_π which is chosen at random from $\mathbb{Z}_q$ when creating the signature σ_c. As a result we have that every tuple $(x_\pi, y_\pi, r_{x_\pi}, r_{y_\pi})$ consists of elements that are distributed uniformly over $\mathbb{Z}_q$ for any signer $\pi \in [n - m_1 - m_2]$. □

Theorem 3 (Non-Transferability). *Our UDVLRS scheme is perfectly non-transferable in the random oracle model.*

Proof. Let σ and σ' be a legitimate signature and simulation respectively, on the same message $\mathtt{m}$, ring L, designated verifier key $\mathsf{pk}_D = g^{x_D} h^{y_D}$, same linking tag t, and same signer π. We will go over every part of the signatures and prove that they follow the same distribution.

- *Linking tag t.* Since σ' is simulating σ they will be the same value and therefore follow the same distribution.
- *$\tilde{x}$ and $\tilde{y}$.* In the case of σ, $\tilde{x}$ and $\tilde{y}$ are calculated using randomness r_x and r_y respectively and, as a result, will be uniformly distributed over $\mathbb{Z}_q$. In the case of σ', $\tilde{x}$ and $\tilde{y}$ are sampled at random from $\mathbb{Z}_q$, and as such will also be uniformly distributed over $\mathbb{Z}_q$.
- $\{c_i\}_{i=1}^n$. For σ we have that $\{c_i\}_{\substack{i\in[n]\\ i\neq\pi}} \leftarrow\$ \mathbb{Z}_q$ and c_π is calculated as

  $$c_\pi \leftarrow \mathsf{H}_q(\mathtt{m}, L, \mathtt{ev}, t, K, K', K'') - \sum_{\substack{i\in[n]\\ i\neq\pi}} c_i \mod q$$

  So it is the result of the output of a random hash value minus a sum of random values over the same set, and therefore it is uniformly distributed over $\mathbb{Z}_q$.

  For σ' we have that $\{c_i\}_{i=2}^n \leftarrow\$ \mathbb{Z}_q$, and for c_1 we have that:

  $$c_1 = \mathsf{H}_q(\mathtt{m}, L, \mathtt{ev}, t, K_D, K'_D, K''_D) - \sum_{\substack{i\in[n]\\ i\neq 1}} c_i \pmod q$$

  and by the same argument as before, we have that c_1 is also uniformly random over $\mathbb{Z}_q$. Therefore for every index i in $[n]$ we have that $\{c_i\}_{i=1}^n$ follow the same distribution in both cases.

– $\{w_i\}_{i=1}^n$. For σ we choose every w_i at random from $\mathbb{Z}_q$. For σ' we choose at random every w_i from $\mathbb{Z}_q$ except for w_1. For w_1 we have the following:

$$w_1 = \alpha - c_1 \pmod q$$

Since α is a random element of $\mathbb{Z}_q$ we have that w_1 is also distributed uniformly over $\mathbb{Z}_q$. As a result, we see that in both cases, again as it was the case for $\{c_i\}_{i=1}^n$, for every index i, w_i follows the same distribution.

– *r and s.* In the case of σ we have that r and s are chosen at random from $\mathbb{Z}_q$. For σ' we have that:

$$r = (\beta - w_1)x_D^{-1} \pmod q, \qquad s = \gamma - ry_D \pmod q$$

Since they both contain random elements, β and γ respectively, from $\mathbb{Z}_q$ it holds that r, s are both random elements from $\mathbb{Z}_q$. Thus in both signatures these two values follow the same distribution.

By the previous observations we can conclude that a signature from a signer in the ring and a simulation from a designated verifier on the same ring message, designated verifier public key and linking tag for the same signer π both follow the same distribution and are therefore indistinguishable, even to an unbounded adversary, thus making a random choice the best, and only, option. □

Next, we state Lemma 5 which will be used in the proof of Theorem 4.

Lemma 5. *If an adversary $\mathcal{A}$ knows only one private key $\mathsf{sk}_\pi = (x_\pi, y_\pi)$ where $\pi \in [n]$ and produces a valid signature $\sigma = (t, \tilde{x}, \tilde{y}, r, s, \{c_i\}_{i=1}^n, \{w_i\}_{i=1}^n)$ for an event $\texttt{ev}$, then $t = e^{x_\pi}$, where $e = \mathsf{H}_{\mathbb{G}}(\texttt{ev})$, provided that DLOG is hard, in the random oracle model.*

Proof. Suppose that $\mathcal{A}$ produces a valid signature $\sigma_1 = (t, \tilde{x}_1, \tilde{y}_1, r_1, s_1, \{c_{i1}\}_{i=1}^n, \{w_{i1}\}_{i=1}^n)$, where $t = \mathsf{H}_{\mathbb{G}}(\texttt{ev})^{\hat{x}}$ for some $\hat{x} \in \mathbb{Z}_q$. We rewind $\mathcal{A}$ and give a different value for the random oracle query H_q, and $\mathcal{A}$ produces a second valid signature $\sigma_2 = (t, \tilde{x}_2, \tilde{y}_2, r_2, s_2, \{c_{i2}\}_{i=1}^n, \{w_{i2}\}_{i=1}^n)$.

In both runs, the query of $\mathcal{A}$ to H_q is

$$\mathsf{H}_q(\texttt{m}, L, \texttt{ev}, \mathsf{H}_{\mathbb{G}}(\texttt{ev})^{\hat{x}}, g^{\eta}h^{\eta'}, \mathsf{H}_{\mathbb{G}}(\texttt{ev})^{\kappa}, g^{\theta}h^{\theta'})$$

and in both runs the list of public keys L, the event $\texttt{ev}$, the message $\texttt{m}$, the values $\eta, \eta', \kappa, \theta, \theta', \hat{x} \in \mathbb{Z}_q$ are fixed. Suppose that in the first query of $\mathcal{A}$ to H_q we return a value c_{01} and in the second query we return a value $c_{02} \neq c_{01}$.

From the two valid signatures σ_1 and σ_2 by using the same reasoning as in Theorem 1 (Eq. 2 and Eq. 3) we get the following equations:

$$c_{01} = c_{11} + \cdots + c_{n1},$$
$$c_{02} = c_{12} + \cdots + c_{n2},$$

$$\eta = \tilde{x}_1 + \sum_{i=1}^{n} x_i(c_{i1} + w_{i1}) = \tilde{x}_2 + \sum_{i=1}^{n} x_i(c_{i2} + w_{i2}) \tag{5}$$

$$\kappa = \tilde{x}_1 + \hat{x}\sum_{i=1}^{n}(c_{i1} + w_{i1}) = \tilde{x}_2 + \hat{x}\sum_{i=1}^{n}(c_{i2} + w_{i2}) \tag{6}$$

$$\theta = r_1 x_D + \sum_{i=1}^{n} w_{i1} = r_2 x_D + \sum_{i=1}^{n} w_{i2} \tag{7}$$

From (Eq. 7) we conclude that either $\mathcal{A}$ knows x_D, which is a contradiction, or $\sum_{i=1}^{n} w_{i1} = \sum_{i=1}^{n} w_{i2}$. So, similarly to the proof of Theorem 1 we can conclude that there exists at least one $j \in [n]$ such that $(c_{j1} + w_{j1}) - (c_{j2} + w_{j2}) \neq 0$.

We now evaluate the possible values of $\hat{x}$ so that $\mathcal{A}$ can generate two such signatures having only one private key. We split into two cases:

1. Suppose that $(c_{i1} + w_{i1}) = (c_{i2} + w_{i2})$ for all $i \in [n]$ except when $i = j$ for some $j \in [n]$, that is, $(c_{j1} + w_{j1}) \neq (c_{j2} + w_{j2})$.
   Then from (Eq. 5) we have
   $$\tilde{x}_1 + x_j(c_{j1} + w_{j1}) = \tilde{x}_2 + x_j(c_{j2} + w_{j2})$$
   and from (Eq. 6) we have
   $$\tilde{x}_1 + \hat{x}(c_{j1} + w_{j1}) = \tilde{x}_2 + \hat{x}(c_{j2} + w_{j2})$$
   Thus $\hat{x} = x_j = \frac{\tilde{x}_2 - \tilde{x}_1}{c_{j1} + w_{j1} - c_{j2} - w_{j2}}$. The value $\hat{x}$ is known by $\mathcal{A}$, and since $\mathcal{A}$ is assumed to know only one private key, we have $j = \pi$.
2. Suppose that $(c_{i1} + w_{i1}) = (c_{i2} + w_{i2})$ for all $i \in [n]$ except when $i = j$ for $j \in \{j_1, j_2\}$. That is, $(c_{j_1 1} + w_{j_1 1}) \neq (c_{j_1 2} + w_{j_1 2})$ and $(c_{j_2 1} + w_{j_2 1}) \neq (c_{j_2 2} + w_{j_2 2})$. Also $\tilde{x}_1 \neq \tilde{x}_2$. From (Eq. 5) we have
   $$\tilde{x}_1 + x_{j_1}(c_{j_1 1} + w_{j_1 1}) + x_{j_2}(c_{j_2 1} + w_{j_2 1}) = \tilde{x}_2 + x_{j_1}(c_{j_1 2} + w_{j_1 2}) + x_{j_2}(c_{j_2 2} + w_{j_2 2}) \tag{8}$$
   - If $\pi \in \{j_1, j_2\}$ then from Eq. 8, $\mathcal{A}$ knows both x_{j_1} and x_{j_2}, which is a contradiction since $\mathcal{A}$ knows only one private key.
   - Else, if $\pi \notin \{j_1, j_2\}$ then we have the following:
     $$x_{j_1} + x_{j_2}\phi_2 = \phi_1,$$
     where $\phi_2 = \frac{c_{j_2 1} + w_{j_2 1} - c_{j_2 2} - w_{j_2 2}}{c_{j_1 1} + w_{j_1 1} - c_{j_1 2} - w_{j_1 2}}$ and $\phi_1 = \frac{\tilde{x}_2 - \tilde{x}_1}{c_{j_1 1} + w_{j_1 1} - c_{j_1 2} - w_{j_1 2}}$.
     This implies that $\mathcal{A}$ can solve the MDLR problem. From Proposition 1, this problem is computationally equivalent to DLOG, and since we assumed DLOG is hard in $\mathbb{G}$, this case should not exist.

   The same argument can be generalized to have three or more $c_{i1} + w_{i1}$ be not equal to the corresponding $c_{i2} + w_{i2}$.

Concluding, only case 1(a) is possible. That is, if $\mathcal{A}$ knows only one private key (x_π, y_π), we have $t = \mathsf{H}_{\mathbb{G}}(\mathsf{ev})^{x_\pi}$. $\square$

Theorem 4 (Linkability). *Our UDVLRS scheme is linkable in the random oracle model if* DLOG *is hard in* $\mathbb{G}$*.*

Proof. We assume that $\mathcal{A}$, knowing only one private key, can produce two valid signatures that are unlinked, that is, for the linking tags of the two signatures t_1 and t_2 it holds that $t_1 \neq t_2$. Then, by Lemma 5, it should hold that $t_i = e^{x_i}, i = 1, 2$ where $e = H_{\mathbb{G}}(\mathtt{ev})$. This means that $\mathcal{A}$ knows two different private keys or that it can solve the DLOG problem, both of which are a contradiction. □

Theorem 5 (Non-Slanderability). *Our UDVLRS scheme is non-slanderable in the random oracle model if* DLOG *is hard in* $\mathbb{G}$*.*

Proof. Suppose that there exists a PPT adversary $\mathcal{A}$ that given a valid signature $\sigma_1 = (t_1, \cdot)$ manages to produce a valid signature $\sigma_2 = (t_2, \cdot)$ such that $t_1 = t_2$ without knowing the private key used to produce σ_1. We will construct a PPT adversary $\mathcal{B}$ that given two DLOG instances can solve one of them using $\mathcal{A}$.

The input of $\mathcal{B}$ consists of $\mathbb{G}, g, q, X_\pi, X_D$. The adversary $\mathcal{B}$ simulates the environment for $\mathcal{A}$ as in the proof of Theorem 1. At some point in the execution, $\mathcal{A}$ chooses a public key $\mathsf{pk}_{\pi'}$ of a user π' and gives it to $\mathcal{B}$.

– If $\pi' = \pi$ then $\mathcal{B}$ programs the oracles as in the proof of Theorem 1 and creates a signature $\sigma_1 = (t_1, \cdot)$ where $t_1 = \mathsf{H}_{\mathbb{G}}(\mathtt{ev})^{x_\pi}$ and x_π is the discrete logarithm of X_π. As in the proof of Theorem 1, $\mathcal{B}$ manages to create such a signature without knowing x_π by setting $t_1 = X_\pi^a$, where $e = g^a$ was the answer to the query $\mathsf{H}_{\mathbb{G}}(\mathtt{ev})$. The adversary $\mathcal{B}$ returns σ_1 to $\mathcal{A}$.
– If $\pi' \neq \pi$ then $\mathcal{B}$ halts.

The adversary $\mathcal{A}$ manages after interactions with the oracles $\mathcal{RO}, \mathcal{JO}, \mathcal{CO}, \mathcal{SO}, \mathcal{MO}$ controlled by $\mathcal{B}$ to output a signature $\sigma_2 = (t_2, \tilde{x}_2, \tilde{y}_2, r_2, s_2, \{c_{i2}\}_{i=1}^n, \{w_{i2}\}_{i=1}^n)$. $\mathcal{B}$ rewinds $\mathcal{A}$ and for the same input for the random oracle query H_q returns a different value to $\mathcal{A}$, and $\mathcal{A}$ produces another signature $\sigma_2^* = (t_2^*, \tilde{x}_2^*, \tilde{y}_2^*, r_2^*, s_2^*, \{c_{i2}^*\}_{i=1}^n, \{w_{i2}^*\}_{i=1}^n)$. Similarly to the proof of Theorem 1 and since $\mathsf{H}(\mathtt{ev})^{x_\pi} = t_1 = t_2 = t_2^*$ and $K_2' = {K_2^*}'$, we obtain the following equation:

$$\tilde{x}_2 + x_\pi \sum_{i=1}^{n}(c_{i2} + w_{i2}) = \tilde{x}_2^* + x_\pi \sum_{i=1}^{n}(c_{i2}^* + w_{i2}^*) \tag{9}$$

We split into two cases:

1. If $\sum_{i=1}^n (c_{i2} + w_{i2}) = \sum_{i=1}^n (c_{i2}^* + w_{i2}^*)$ and since $\sum_{i=1}^n c_{i2} \neq \sum_{i=1}^n c_{i2}^*$, we have $\sum_{i=1}^n w_{i2} \neq \sum_{i=1}^n w_{i2}^*$. By using a similar relation to Eq. 3 of Theorem 1, $\mathcal{B}$ can find x_D, the discrete logarithm of X_D.
2. If $\sum_{i=1}^n (c_{i2} + w_{i2}) \neq \sum_{i=1}^n (c_{i2}^* + w_{i2}^*)$ then $\mathcal{B}$ can solve Eq. 9 and obtain x_π.

This means that $\mathcal{B}$ can find the discrete logarithm of X_π or X_D, which is a contradiction since we assumed that DLOG is hard in $\mathbb{G}$. □

6 Conclusion and Future Work

In this work we defined UDVLRS, a signature construction that provides public verifiability, unconditional anonymity and non-transferability, unforgeability, non-sladerability, and linkability - the latter conditional to the hardness of DLOG. This is an improvement on its predecessor DVLRS, the anonymity of which was computational. We formally defined its security model and proved its properties. While a UDVLR signature consists of fewer elements than a DVLR signature, the size of both schemes is linear to the number of ring members. As a result, the goal of shorter signatures set in [2] remains an open problem.

References

[1] Handan Kilinç Alper and Jeffrey Burdges. "Two-Round Trip Schnorr Multi-signatures via Delinearized Witnesses". In: *CRYPTO 2021*. Vol. 12825. LNCS. Springer, 2021, pp. 157–188.

[2] Pourandokht Behrouz, Panagiotis Grontas, Vangelis Konstantakatos, Aris Pagourtzis, and Marianna Spyrakou. "Designated-Verifier Linkable Ring Signatures". In: *24th International Conference on Information Security and Cryptology - ICISC 2021*. Preprint: `https://ia.cr/2022/470`. 2021.

[3] Mihir Bellare and Phillip Rogaway. "Random Oracles Are Practical: A Paradigm for Designing Efficient Protocols". In: CCS '93. ACM, 1993, 62–73.

[4] Adam Bender, Jonathan Katz, and Ruggero Morselli. "Ring Signatures: Stronger Definitions, and Constructions without Random Oracles". In: *J. Cryptol.* 22.1 (2009), pp. 114–138.

[5] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. "Designated Verifier Proofs and Their Applications". In: *EUROCRYPT 96*. Vol. 1070. LNCS. Springer, 1996, pp. 143–154.

[6] Ji-Seon Lee and Jik Hyun Chang. "Strong Designated Verifier Ring Signature Scheme". In: *Innovations and Advanced Techniques in Computer and Information Sciences and Engineering*. Springer, 2007, pp. 543–547.

[7] Jin Li and Yanming Wang. "Universal Designated Verifier Ring Signature (Proof) Without Random Oracles". In: *Emerging Directions in Embedded and Ubiquitous Computing*. Springer, 2006, pp. 332–341.

[8] Helger Lipmaa, Guilin Wang, and Feng Bao. "Designated Verifier Signature Schemes: Attacks, New Security Notions and a New Construction". In: *ICALP*. Vol. 3580. LNCS. Springer, 2005, pp. 459–471.

[9] Joseph K. Liu, Man Ho Au, Willy Susilo, and Jianying Zhou. "Linkable Ring Signature with Unconditional Anonymity". In: *IEEE Trans. Knowl. Data Eng.* 26.1 (2014), pp. 157–165.

[10] Joseph K. Liu, Victor K. Wei, and Duncan S. Wong. "Linkable Spontaneous Anonymous Group Signature for Ad Hoc Groups (Extended Abstract)". In: *ACISP*. Vol. 3108. LNCS. Springer, 2004, pp. 325–335.

[11] Joseph K. Liu and Duncan S. Wong. "Enhanced security models and a generic construction approach for linkable ring signature". In: *Int. J. Found. Comput. Sci* 17.06 (2006), pp. 1403–1422.

[12] Joseph K. Liu and Duncan S. Wong. "Linkable Ring Signatures: Security Models and New Schemes". In: *ICCSA 2005*. Vol. 3481. LNCS. Springer, 2005, pp. 614–623.

[13] David Pointcheval and Jacques Stern. "Security Arguments for Digital Signatures and Blind Signatures". In: *J. Cryptol.* 13.3 (2000), pp. 361–396.

[14] Ronald L. Rivest, Adi Shamir, and Yael Tauman. "How to Leak a Secret". In: *ASIACRYPT 01*. Vol. 2248. LNCS. Springer, 2001, pp. 552–565.

[15] Ron Steinfeld, Laurence Bull, Huaxiong Wang, and Josef Pieprzyk. "Universal Designated-Verifier Signatures". In: *ASIACRYPT 03,* vol. 2894. LNCS. Springer, 2003, pp. 523–542.

[16] Patrick P. Tsang and Victor K. Wei. "Short Linkable Ring Signatures for E-Voting, E-Cash and Attestation". In: *Information Security Practice and Experience*. Vol. 3439. LNCS. Springer, 2005, pp. 48–60.