A short identity-based proxy ring signature scheme from RSA

Maryam Rjabzadeh Asaar  
*Sharif University of Technology, asaar@ee.sharif.edu*

Mahmoud Salmasizadeh  
*Sharif University of Technology, salmasi@sharif.edu*

Willy Susilo  
*University of Wollongong, wsusilo@uow.edu.au*

Publication Details

A short identity-based proxy ring signature scheme from RSA

Abstract
Identity-based proxy ring signature concept was introduced by Cheng et al. in 2004. This primitive is useful where the privacy of proxy signers is required. In this paper, the first short provably secure identity-based proxy ring signature scheme from RSA assumption has been proposed. In addition, the security of the proposed scheme tightly reduces to the RSA assumption, and therefore, the proposed scheme has a proper advantage in security reduction compared to the ones from RSA. The proposed scheme not only outperforms the existing schemes in terms of efficiency and practicality, but also does not suffer from the proxy key exposure attack due to the use of the sequential aggregation paradigm.

Keywords
Identity-based proxy ring signature, Random oracle model, RSA assumption

Disciplines
Engineering | Science and Technology Studies

Publication Details

This journal article is available at Research Online: http://ro.uow.edu.au/eispapers/3263
A Short Identity-based Proxy Ring Signature Scheme from RSA

Maryam Rajabzadeh Asaar*1, Mahmoud Salmazadeh*2, and Willy Susilo**2

1 Department of Electrical Engineering,
2 Electronics Research Institute (Center),
Sharif University of Technology, Tehran, Iran.
3 Centre for Computer and Information Security Research,
University of Wollongong, Australia.
asaar@ee.sharif.ir, salmasi@sharif.edu, wsusilo@uow.edu.au

Abstract. Identity-based proxy ring signature concept was introduced by Cheng et al. in 2004. This primitive is useful where the privacy of proxy signers is required. In this paper, the first short provably secure identity-based proxy ring signature scheme from RSA assumption has been proposed. In addition, the security of the proposed scheme tightly reduces to the RSA assumption, and therefore, the proposed scheme has a proper advantage in security reduction compared to the ones from RSA. The proposed scheme not only outperforms the existing schemes in terms of efficiency and practicality, but also it does not suffer from the proxy key exposure attack due to the use of the sequential aggregation paradigm.

Keywords: identity-based proxy ring signature, random oracle model, RSA assumption.

1 Introduction

Digital signatures are widely deployed around the world and have the backing of significant international legislation to support their use in electronic environment. In this research, we are interested in exploring efficient identity-based proxy ring signature schemes.

Identity-based cryptography. Public-key cryptography has many different applications, but in its basic form, it requires extensive public-key infrastructure for practical use. In order to provide more flexible management of public keys the notion of identity-based cryptography was introduced by Shamir [1]. The main feature of identity-based cryptosystems is to remove the requirement of certification of the public keys. The public key of each party is obtained from its public identity, such as an email address, which can uniquely identify the party. Since the introduction of the notion in [1], various identity based schemes ([2–4]) have been proposed.

Identity-based cryptography has attracted a lot of interest since the elliptic curve pairings are shown to provide an elegant way for implementing identity-based encryption schemes. In the past ten years, the majority of identity-based cryptosystems proposed have relied on pairings. While extensive research has led to vast improvements in implementation of pairings, their computational cost is still higher than that of traditional public key algorithms which use the exponentiation operation in various groups. Moreover, pairing-based cryptosystems rely on newer and less analyzed computational assumptions in their security analysis compared to traditional schemes that are based on classical assumptions like the widely studied RSA assumption. There has been a proliferation of pairing-based assumptions whose difficulty is not widely understood and whose connection to established assumptions, and to each other, remains unknown [5]. Therefore, when designing new identity-based cryptographic primitives, it is desirable to diversify the computational assumptions and to use widely accepted assumptions where possible.

Proxy ring signatures. The notion of proxy signatures was introduced by Mambo et al. [6] in 1996. In a proxy signature scheme, an original signer delegates her signing right for signing messages to a proxy signer. This kind of signature supports ensuring service availability for the customers in distributed networks to avoid the dependency to a single server. Since the introduction of the notion of proxy signatures, several variants

* This research was supported in part by the Office of Vice-President for Science and Technology, I.R. Iran and by Grant No. T-500-15712 from ITRC.
** W. Susilo is supported by the Australian Research Council Discovery Project (DP130101383).
of proxy signatures such as proxy signatures from RSA and integer factorization problem ([7–12]), identity-based proxy signature schemes based on bilinear pairings ([13–19]), designated-verifier proxy signatures ([20–22]), short proxy signature [23], proxy verifiably encrypted signatures [24], proxy signature schemes without random oracles [25], identity-based multi-proxy signatures [26], proxy ring signatures ([27–31]) and identity-based proxy ring signatures from bilinear pairings ([32–39]) have been proposed.

In a proxy ring signature scheme, an original signer delegates her signing right for signing messages to a group of proxy signers with different public keys, called the proxy agent, such that they can generate proxy ring signatures on behalf of the original signer while he could be anonymous. This type of signature not only supports ensuring service availability for the customers in distributed networks to avoid the dependency to a single server but also supports preserving privacy of proxy signers. As mentioned in ([27–29]), this primitive can be used when the requirement of proxy signer’s privacy protection is necessary. For example, it is assumed that a parliament member would like to reveal an important news on behalf of the cabinet, while he wants to be anonymous. The other practical motivation for this primitive is electronic voting protocols, where just eligible voters anonymously can cast their ballots after authenticating themselves. The voting authority (an original signer) in the voting protocol authenticates eligible voters and issues certificates (generates valid delegations) for them. Voters (proxy signers) anonymously cast their ballots (generate proxy ring signatures).

On the one hand, employing identity-based proxy signatures violates the most important property of voting protocols, voter privacy. Additionally, employing blind signatures in voting protocols enables the malicious voting authority to cast its ballot instead of abstained voters, and so violates accuracy of the election. Proxy ring signatures incorporate three requirements (privacy of voters, authentication and accuracy) at the same time. We should highlight that some access control mechanisms are necessary in the voting protocol to provide uniqueness property. Indeed, this primitive is a solution to the problem of electronic voting protocols based on blind signatures in which a malicious voting authority can vote instead of abstained voters.

However, one still needs to verify public keys of proxy signers and the original signer in addition to verifying the validity of a proxy ring signature. For the first time, Cheng et al. proposed an identity-based proxy ring signature [35] to facilitate public key certificate management of these types of signatures by merely employing signer’s identities in place of the public keys and their certificates. Subsequently, there have been some follow-up works for identity-based proxy ring signatures ([32–34, 36–39]), but unfortunately, none of them supports provable security. In 2014, Asaar et al. [40] presented the first formal definition and security model for identity-based proxy ring signature schemes, and also proposed the first provably secure identity-based proxy ring signature scheme, and showed that it is secure under RSA assumption in the random oracle model. In addition, previous identity-based proxy ring signature schemes proposed in ([32–39]) are vulnerable to the proxy key exposure attack [41] presented by Schuldt et al. in 2008. In fact, in previous schemes, if temporal secret keys of proxy signers are leaked, long term secret keys of proxy signers will be compromised.

1.1 Our Contribution

In this paper, we present the first short identity-based proxy ring signature scheme from RSA. The proposed scheme is proved secure under the RSA assumption, a widely accepted assumption, in the random oracle model. The advantages of the proposed scheme are three-fold. First, it is the shortest identity-based proxy ring signature scheme without bilinear pairings. Second, it has a proper advantage in security reduction since reduction of the proposed scheme is independent of the number of members in a proxy ring. Third, it is as efficient as or more efficient than existing identity-based proxy ring signature schemes. Furthermore, the proxy key exposure attack [41] cannot be applied to our scheme since the paradigm used in designing this primitive is sequential aggregation of an identity-based signature and an identity-based ring signature scheme.

One may think that it is possible to build short identity-based proxy ring signature schemes without bilinear pairings from the idea presented in [42]. We should emphasize that if we use this idea in designing this primitive, the result is no longer identity-based since public keys according this idea are not just identities of signers, and in this primitive, each proxy signer needs to interact before signature generation to attain proxy signers’ public keys and cannot easily use identities of other proxy signers to generate an identity-based proxy ring signature.
1.2 Paper Organization

The rest of this paper is organized as follows. Section 2 presents notations and RSA complexity assumption employed as the signature foundation. The security model of identity-based proxy ring signature including outline of the identity-based proxy signature scheme and its security properties are given in Section 2. The proposed scheme and its formal security proofs are presented in Section 3. Section 4 presents the comparison and discussion. Conclusion and future work are given in Section 5.

2 Background

In this section, first we give notations used throughout the paper and review the RSA assumption, and then we present the outline and security definitions for the identity-based proxy ring signature schemes [40].

2.1 Notations

If $X$ is a set, then $x \xleftarrow{} X$ denotes the operation of assigning to $x$ an element of $X$ chosen uniformly at random. If $x_1, x_2, \ldots$ are objects then $x_1||x_2||\ldots$ denotes an encoding of them as strings from which the constituent objects are effectively recoverable. Let $\perp$ be an empty string, $|x|$ be the bit length of $x$, and $\theta \leftarrow C(x_1,...)$ stands for the operation of assigning the output of the algorithm $C$ on inputs $x_1,...$ to $\theta$. Let $A$ be an algorithm which has access to $H$, $K$, KeyExtract, DelegationGen and ProxyRingSign oracles of a signature scheme, and can win a game in which a security property of the scheme is violated by $A$. If algorithm $A$ is $(t, q_h, q_k, q_e, q_d, \epsilon)$-bounded, we mean that the algorithm $A$ which runs in time at most $t$, makes at most $q_h$ queries to random oracle $H$, $q_k$ queries to random oracle $K$, $q_e$ queries to KeyExtract oracle, $q_d$ queries to DelegationGen and $\epsilon$ queries to ProxyRingSign oracle can win the game with probability at least $\epsilon$.

2.2 The RSA assumption

An RSA key generator $KG_{rsa}$ is an algorithm that generates triplets $(N, e, d)$ such that $N$ is the product of two large primes $p$ and $q$ and $ed = 1 \mod \phi(N)$, where $\phi(N) = (p-1)(q-1)$. The advantage of an algorithm $B$ in breaking the one-wayness of RSA related to $KG_{rsa}$ is defined as

$$Adv_{KG_{rsa}}^{ow-rsa}(B) = \Pr \left[ (N, e, d) \xleftarrow{} KG_{rsa}; y \xleftarrow{} \mathbb{Z}_N; \gamma \xleftarrow{} \mathbb{Z}_N; y = \gamma^e \mod N : \gamma \leftarrow B(N, e, y) \right].$$

We say that $B$, $(t', \epsilon')$-breaks the one-wayness of RSA with respect to $KG_{rsa}$ if it runs in time at most $t'$ and has advantage $Adv_{KG_{rsa}}^{ow-rsa}(B) \geq \epsilon'$. We say that the RSA function associated to $KG_{rsa}$ is $(t', \epsilon')$-one-way if there is no algorithm $B$ that can $(t', \epsilon')$-break it.

2.3 Outline of identity-based proxy ring signature schemes

Let identity of each original signer be $ID_0$, and identity set of proxy agent and each subset of that be $ID$ and $ID'$, respectively. The indices used in the signature description have no global meaning outside this protocol instance, and just serve as local pointers for original and proxy signers. An identity-based proxy ring signature scheme consists of six algorithms: Setup, KeyExtract, DelegationGen, DelegationVer, ProxyRingSign and ProxyRingVer as follows [40].

- Setup: This algorithm takes as input the system security parameter $l$ and outputs system’s parameters $Para$ and the system’s master key $(msk, mpk)$, i.e. $(Para, (msk, mpk)) \leftarrow ParaGen(l)$.

- KeyExtract: This algorithm takes as input the system’s parameter $Para$, master public key $mpk$, master secret key $msk$, and an identity $ID_u$. It outputs the corresponding secret key $x_u$ for the identity $ID_u$, i.e. $x_u \leftarrow KeyExtract(Para, mpk, msk, ID_u)$. 

DelegationGen: This algorithm takes as input the system’s parameter Para, the master public key mpk, an identity ID_o and an identity set ID, including at least two identities, for an original signer and a proxy agent, respectively. It also takes as input the secret key x_o of the original signer with identity ID_o and a message space descriptor w ⊆ \{0,1\} for which the original signer with identity ID_o delegates its signing right to a proxy agent with identity set ID, then, it outputs a delegation \(\sigma_o \leftarrow \text{DelegationGen}(\text{Para}, \text{mpk}, \text{ID}_o, \text{ID}, w, x_o)\).

DelegationVer: This algorithm takes as input the system’s parameter Para, an original signer’s identity ID_o, the proxy signers’ identity set ID, a message space descriptor w and a delegation \(\sigma_o\), then, it outputs 1 if \(\sigma_o\) is a valid delegation and outputs 0 otherwise, i.e. \(\{0, 1\} \leftarrow \text{DelegationVer}(\text{Para}, \text{mpk}, \text{ID}_o, \text{ID}, w, \sigma_o)\).

ProxyRingSign: This algorithm takes as input the system’s parameter Para, the master public key mpk, the identity set ID of proxy signers including at least two identities, a valid delegation \(\sigma_o\) for a message space descriptor w and an identity set ID of proxy signers such that \(\text{ID} \subseteq \text{ID}\) and the delegation indicates that an original signer with identity ID_o delegates its signing right on w to a proxy agent with identity set ID, a proxy signer’s secret key \(x_j\) corresponding to an identity ID_j \(\notin \text{ID} \subseteq \text{ID}\) and a message \(m \in w\), then, it outputs the identity-based proxy ring signature \(\theta\) on behalf of the original signer with identity ID_o, i.e. \(\theta \leftarrow \text{ProxyRingSign}(\text{Para}, \text{mpk}, \text{ID}_o, \text{ID}, \text{ID}_j, (m, w, \sigma_o), x_j)\).

ProxyRingVer: This algorithm takes as input the system’s parameter Para, an original signer’s identity ID_o, the proxy signers’ identity sets ID and ID, a message space descriptor w, a signed message m and a proxy ring signature \(\theta\), then, it outputs 1 if \(\theta\) is a valid identity-based proxy ring signature of the message m which means that it satisfies the verification equation, \(m \in w\) and \(\text{ID} \subseteq \text{ID}\) and outputs 0 otherwise, i.e. \(\{0, 1\} \leftarrow \text{ProxyRingVer}(\text{Para}, \text{mpk}, \text{ID}_o, \text{ID}, \text{ID}, w, m, \theta)\).

### 2.4 Security models of identity-based proxy ring signature schemes

An identity-based proxy ring signature must satisfy two independent notions of security: unforgeability and privacy of proxy signer’s identity. To achieve existential unforgeability against adaptive chosen message (chosen warrant: chosen message space descriptor and identity set of proxy signers) and chosen identity attack for identity-based proxy ring signature schemes, three types of potential adversaries as mentioned in [27] are considered as follows.

- **Type I:** This type adversary \(A_I\) only has identities of the original signer and proxy signers, and aims to forge a valid identity-based proxy ring signature w.r.t. identities of the original signer and proxy signers.
- **Type II:** This type adversary \(A_{II}\) has secret keys of some (one/all) proxy signers in a proxy group in addition to identities of the original signer and proxy signers, and aims to forge a valid identity-based proxy ring signature w.r.t. identities of the original signer and proxy signers.
- **Type III:** This type adversary \(A_{III}\) has the secret key of the original signer in addition to identities of the original signer and proxy signers, and aims to forge a valid identity-based proxy ring signature w.r.t. identities of the original signer and proxy signers.

Clearly, if an identity-based proxy ring signature scheme is secure against Type II (or Type III) adversaries then it is also secure against Type I adversary. Unforgeability against Type I, Type II and Type III adversaries \((A_I, A_{II} \text{ and } A_{III})\) is formalized using the following game between a challenger \(C\) and an adversary \(A\).

1. **Setup:** The challenger \(C\) runs the \(\text{ParaGen}\) algorithm with a security parameter \(l\) to obtain system’s parameter \(\text{para}\) and the master key \((\text{mpk}, \text{msk})\), then it sends \((\text{mpk}, \text{para})\) to \(A\).

   \(A\) issues a polynomially bounded number of queries to the following oracles adaptively:

2. **KeyExtract queries:** \(A\) can ask for the secret key corresponding to each identity \(\text{ID}_o\), then \(C\) returns the private key \(x_o\) to the adversary with running the KeyExtract algorithm.
3. DelegationGen queries: Adversary $A$ can request a delegation under the identity $ID_o$ of an original signer on a message space descriptor $w$ and an identity set $ID$ of its choice for which the original signer with identity $ID_o$ delegates its signing right on $w$ to a proxy agent with identity set $ID$. In response, $C$ runs the KeyExtract algorithm to obtain the secret key $x_o$ of the original signer, and returns $\sigma_o \leftarrow \text{DelegationGen}(Para, mpk, ID_o, ID, w, x_o)$ to $A$.

4. ProxyRingSign queries: Adversary $A$ can request the proxy ring signature of $m$ w.r.t. $\tilde{ID}$ to $C$.

In addition, adversary $A$ provides a delegation $\sigma_o$ of an original signer with identity $ID_o$ for a message space descriptor $w$ and an identity set $ID$ of proxy signers. This delegation was obtained from DelegationGen algorithm or was generated by adversary $A$.

Algorithm $C$ checks that $\sigma_o$ is a valid delegation in which the original signer with identity $ID_o$ delegates its signing right for the message space descriptor $w$ to the proxy agent with identity set $ID$; that $\tilde{ID} \subseteq ID$; and that $m \in w$. If any of these fail to hold, returns ⊥. Otherwise, $C$ runs the KeyExtract algorithm to obtain the secret key $x_j$ corresponding to one of the proxy signers with identity $ID_j$ such that $ID_j \not\subseteq \tilde{ID}$. Next, $C$ runs ProxyRingSign algorithm $\theta \leftarrow \text{ProxyRingSign}(Para, mpk, ID_o, ID, \tilde{ID}, (m, w, \sigma_o), x_j)$ to generate the proxy ring signature $\theta$ and returns it to the adversary $A$.

5. Finally, $A$ outputs a valid identity-based proxy ring signature $(m^*, w^*, \theta^*)$ w.r.t. original signer’s identity $ID_o^*$ and proxy signers’ identity sets $ID^*$ and $\tilde{ID}^* \subseteq ID^* \setminus \tilde{ID}^*$, where $\tilde{ID}^*$ is the set of corrupted proxy signers, and wins the game if the following conditions hold.

For $A = A_I$:
- $E_0$: $ID_o^*$ and all identities in $\tilde{ID}^*$ have not been requested to the KeyExtract oracle which means that $A_I$ does not have secret keys corresponding to them.
- $E_1$: The pair $(w^*, ID^*)$ has not been requested as one of the DelegationGen queries under the identity $ID_o^*$.
- $E_2$: $m^*$ has not been requested as one of the ProxyRingSign queries under the identity set $\tilde{ID}^*$.

The formal definition of existential unforgeability against adversary $A_I$ [40] is expressed in Definition 1.

**Definition 1.** An identity-based proxy ring signature is $(t, q_h, q_e, q_d, q_{prs}, \epsilon)$-existentially unforgeable against adaptive chosen message (warrant) and chosen identity attack if there is no $(t, q_h, q_e, q_d, q_{prs}, \epsilon)$-bounded adversary $A$ which wins the aforementioned game.

For $A = A_{II}$:
- $E_0$: $ID_o^*$ has not been requested as one of the KeyExtract queries which means $A_{II}$ does not have the secret key corresponding to $ID_o^*$.
- $E_1$: The pair $(w^*, ID^*)$ has not been requested as one of the DelegationGen queries under the identity $ID_o^*$.

The formal definition of existential unforgeability against adversary $A_{II}$ [40] is expressed in Definition 2.

**Definition 2.** An identity-based proxy ring signature is $(t, q_h, q_e, q_d, \epsilon)$-existentially unforgeable against adaptive chosen message (warrant) and chosen identity attack if there is no $(t, q_h, q_e, q_d, \epsilon)$-bounded adversary $A$ which wins the aforementioned game.

For $A = A_{III}$:
- $E_0$: Each identity in $\tilde{ID}^*$ has not been requested as one of the KeyExtract queries which means that $A_{III}$ does not have the secret keys corresponding to identities in $\tilde{ID}^*$.
- $E_1$: $m^*$ has not been requested as one of the ProxyRingSign queries under identity set $\tilde{ID}^* \subseteq ID^*$.

The formal definition of existential unforgeability against adversary $A_{III}$ [40] is expressed in Definition 3.

**Definition 3.** An identity-based proxy ring signature is $(t, q_h, q_e, q_{prs}, \epsilon)$-existentially unforgeable against adaptive chosen message (warrant) and chosen identity attack if there is no $(t, q_h, q_e, q_{prs}, \epsilon)$-bounded adversary $A$ which wins the aforementioned game.
Privacy of proxy signer’s identity (PPSI) in an identity-based proxy ring signature means that it should be infeasible for any probabilistic polynomial time (PPT) distinguisher $D$ to tell which proxy signer in a proxy group generates $\theta$ on a message $m$. To have a formal definition for this property consider the following game between a challenger $C$ and a distinguisher $D$.

1. Setup: The challenger $C$ runs the $\text{ParaGen}$ algorithm with a security parameter $l$ to obtain system’s parameter $\text{para}$ and the master key $(\text{mpk}, \text{msk})$, then it sends $(\text{mpk}, \text{para})$ to $D$.

The distinguisher $D$ issues a polynomially bounded number of $\text{KeyExtract}$, $\text{DelegationGen}$ and $\text{ProxyRingSign}$ queries adaptively as explained in the forgery game.

2. the distinguisher $D$ chooses two honest identities $ID_1$ and $ID_2$ ($D$ never make $\text{KeyExtract}$ query for these two identities), and makes a $\text{DelegationGen}$ and $\text{ProxyRingSign}$ query on $(w, \text{ID})$ under an identity $ID_o$ and on the message $m \in w$ under the identity set $\text{ID} = \{ID_0, ID_1\} \subseteq \text{ID}$, respectively. In response, $C$ chooses $j \xleftarrow{\$} \{0, 1\}$, runs $\text{KeyExtract}$ for $ID_o$ and $ID_j$ to obtain their corresponding secret keys, and runs $\text{DelegationGen}$ on $(w, \text{ID})$ under the identity $ID_o$ to obtain $\sigma_o$ and returns $\theta \leftarrow \text{ProxyRingSign}(\text{Para}, \text{mpk}, ID_o, \text{ID}, \theta, \text{ID}_i, w, m, \sigma_i) \to D$.

3. Finally, the distinguisher $D$ outputs $j'$ and wins the game if $j' = j$.

The formal definition for privacy of proxy signer’s identity [40] is given in definition 4.

**Definition 4.** (Privacy of the proxy signer’s identity). An identity-based proxy ring signature scheme is $(t, q_t, q_e, q_d, q_{prs}, \epsilon + \frac{1}{2})$-PPSI-secure if there is no $(t, q_t, q_e, q_d, q_{prs}, \epsilon + \frac{1}{2})$-bounded adversary $D$ which can win the aforementioned game.

If the probability is equal to $\frac{1}{2}$, the scheme satisfies privacy of the proxy signer’s identity perfectly.

### 3 Our identity-based proxy ring signature scheme

In this section, we present an identity-based proxy ring signature scheme using sequential aggregation of GQ identity-based signature [43] and the identity-based ring signature scheme [44]. Our scheme generates an identity-based proxy ring signature scheme in a way that a delegation is original signer’s GQ identity-based signature on a message space descriptor and proxy signers’ identities concatenated with 11, and a proxy ring signature is sequential aggregation of a delegation and a ring signature generated by one of the proxy signers on a message, belonged to the message space descriptor concatenated with 11. Indeed, concatenation with 11 prevents trivial attacks as suggested by Boldyreva et al. [45].

#### 3.1 Details of identity-based proxy ring signature scheme

In this section, we present the details of our scheme. When describing the signature scheme, let identity of each original signer be $ID_o$, and identity set of each proxy agent and each subset of that be $\text{ID}$ and $\text{ID}_j$ respectively. The indices used in the signature description have no global meaning outside this protocol instance which means that there is no certified relationship between indices and identities, and just serve as local pointers for original and proxy signers.

It is assumed that $n \geq 2$ and $z \geq 2$ are the number of identities for proxy signers in the proxy agent and the size of each subset $\text{ID}_j$ of $\text{ID}$, respectively. Our scheme consists of $\text{Setup}$, $\text{KeyExtract}$, $\text{DelegationGen}$, $\text{DelegationVer}$, $\text{ProxyRingSign}$ and $\text{ProxyRingVer}$ algorithms as described below.

1. **Setup:** The system parameters are as follows. Let $l_0$, $l_1$ and $l_N \in \mathbb{N}$, and let $K_0 : \{0, 1\}^* \to \{0, 1\}^{l_0}$, $K_1 : \{0, 1\}^* \to \{0, 1\}^{l_1}$ and $H : \{0, 1\}^* \to \mathbb{Z}_N^*$ be random oracles. Let $KG_{rsa}$ be a RSA key pair generator that outputs triplets $(N, e, d)$ such that $\varphi(N) > 2^{l_0}$ and with prime encryption exponents $\epsilon$ of length
strictly greater than \( l_0 \) and \( l_1 \) bits. The key distribution center runs \( KG_{rsa} \) to generate RSA parameters \((N, e, d)\). It publishes \( mpk = (N, e) \) as the master public key, and keeps the master secret key \( msk = d \) secret. Therefore, public parameters are \( Para = \{K_0, K_1, H\} \) and \( mpk \).

2. KeyExtract: On input master secret key \( msk = d \) and the user identity \( ID_u \), the key distribution center computes \( x_u = H(ID_u)^d \mod N \), and sends the user secret key \( x_u \) over a secure and authenticated channel to the user with identity \( ID_u \).

3. DelegationGen: Let \( w \) be a message space descriptor for which an original signer with identity \( ID_o \) would like to delegate her signing right to a group of proxy signers with an identity set \( ID \), the delegation is \( \sigma_o = (R_0, s_o) = (r_o^e \mod N, r_0 x_o^e \mod N) \), where \( r_o \leftarrow Z_N^* \) and \( c_o = K_0(R_o|w|ID||ID|11) \). Then, the original signer publishes the delegation \( \sigma_o \) on \((w, ID)\).

4. DelegationVer: Given the identity \( ID_o \) of an original signer and identity set \( ID \) of the proxy signers, a message space descriptor \( w \) and a delegation \( \sigma_o \), a verifier checks if the relation \( s_o^e = R_o H(ID_o)^e \) holds, where \( c_o = K_0(R_o|w|ID||ID|11) \). If so, the delegation is valid; otherwise, it is not valid.

5. ProxyRingSign: A proxy signer with identity \( ID_j \) \( \notin \) \( \tilde{ID} \subseteq ID \) \((j \in \{0, ..., z - 1\}) \) can sign a message \( m \in w \) anonymously on behalf of the original signer with the identity \( ID_o \) with his secret key \( x_j \) and a valid delegation \( \sigma_o \) as follows.

- The proxy signer \( ID_j \) chooses \( r \leftarrow Z_N^* \), computes \( R = r^e \mod N \) and \( c_{j+1} = K_1(R||ID||\tilde{ID}||ID_j||w||m||11) \).
- For \( j + 1 \leq u \leq j - 1 \) (let the index \( u \) be module \( z \)), the proxy signer \( ID_j \) chooses \( r_u \leftarrow Z_N^* \), computes \( R_u = r_u^e \mod N \) and \( c_{u+1} = K_1(R_u H(ID_u)^e R_o H(ID_o)^e ||ID||\tilde{ID}||ID_u||w||m||11) \).
- The proxy signer \( ID_j \) computes \( c_j = r_{j+1} s_j^e \mod N \).
- The proxy ring signature is \( \theta = (R_0, r_0, ..., r_{z-1}, c_0) \) on the message \( m \) and the message space descriptor \( w \) under original signer’s identity \( ID_o \) and an identity subset \( \tilde{ID} \subseteq ID \) of proxy signers.

6. ProxyRingVer: Given the identity \( ID_o \) of an original signer and the identity sets \( ID \) and \( \tilde{ID} \) of the proxy signers, a message space descriptor \( w \), a message \( m \), and a proxy ring signature \( \theta \), a verifier operates as follows:

- Checks if \( m \in w \), otherwise, it stops.
- Checks if \( \tilde{ID} \subseteq ID \), otherwise, it stops.
- For \( 0 \leq u \leq z - 1 \), computes \( R_u = r_u^e \) and \( c_{u+1} = K_1(R_u H(ID_u)^e R_o H(ID_o)^e ||ID||\tilde{ID}||ID_u||w||m||11) \), and accepts the signature if and only if \( c_z = c_0 \), where \( c_o = K_0(R_o|w|ID||ID|11) \).

3.2 Analysis of the scheme

In this section, we verify the correctness, and prove the privacy of the proxy signer’s identity and existential unforgeability of the proposed scheme in the random oracles model (see [46] for the background). In order to prove unforgeability of the proposed scheme, we need to show that it is unforgeable against adversaries of types II and III (as defined in Section 2.4).

To prove unforgeability of our proposed scheme, and by contradiction, assuming an adversary \( A_{II+1-\zeta} \), \( \zeta \in \{0, 1\} \), (the parameter \( \zeta \) makes difference between adversaries \( A_{II} \) and \( A_{III} \)) we show that there is a solver (algorithm \( B \)) that can solve a random instance of the RSA problem with a non-negligible probability. To do this, we show that there exists a simulator \( C_{A_{II+1-\zeta}} \) that can simulate the signature scheme without knowing the secret key(s) of the honest signer(s), and runs the adversary \( A_{II+1-\zeta} \) as its sub-routine. In this regard, we compute the run-time and a lower-bound for the success probability of this simulator in terms of the run-time and success probability of the adversary and the number of queries to the oracles. Then, \( B \) uses the oracle replay technique [47] to solve an instance \((N, e, y)\) of the RSA problem, using a useful pair that \( C_{A_{II+1-\zeta}} \) outputs when the random string used in both simulations are the same. In this case, we compute a lower bound for the probability of producing such a useful pair and solving the RSA instance as the main body of the solver algorithm \( B \).
To start let us verify the correctness of the proposed scheme, and we use the fact that $u$ is module $z$. Note that, all computations are done modulo $N$, but we omit this for simplicity.

\[
\begin{align*}
\frac{r_{\gamma}^c}{s_{\gamma}^c} H(ID_{\gamma})^{c_{\gamma}} R_\gamma H(ID_{\gamma})^{c_{\gamma}} &= \frac{(r_{\gamma}^c)^e}{s_{\gamma}^c e} H(ID_{\gamma})^{c_{\gamma}} R_\gamma H(ID_{\gamma})^{c_{\gamma}} \\
&= \left(\frac{r_{\gamma}^c}{s_{\gamma}^c} H(ID_{\gamma})^{c_{\gamma}} R_\gamma H(ID_{\gamma})^{c_{\gamma}}\right)^2 \\
&= \frac{R_\gamma H(ID_{\gamma})^{c_{\gamma}} H(ID_{\gamma})^{c_{\gamma}}}{R_\gamma H(ID_{\gamma})^{c_{\gamma}} H(ID_{\gamma})^{c_{\gamma}}} \\
&= R_\gamma.
\end{align*}
\]

Also, in what follows we will be needing the following Splitting lemma.

**Lemma 1.** [47]. Let $A \subset X \times Y$ such that $\Pr[(x, y) \in A] \geq \delta$. For any $\alpha < \delta$, define $B = \{(x, y) \in X \times Y | \Pr_{y' \in Y}[(x, y') \in A] \geq \delta - \alpha\}$ and $\tilde{B} = (X \times Y) \setminus B$, then the following statements hold:

- $\Pr[B] \geq \alpha$ \\
- $\forall (x, y) \in B, \Pr_{y' \in Y}[(x, y') \in A] \geq \delta - \alpha$ \\
- $\Pr[B|A] \geq \frac{\alpha}{\delta}$.

**Theorem 1.** The proposed scheme is $(t, q_H, q_{K_0}, q_E, q_d, \epsilon)$-secure against an adversary $A_{II}$ if the RSA function associated to $K_{grsa}$ is $(t', \epsilon')$-one-way, and

\[
\epsilon' \geq \frac{e_2 (1 - 2^{1-t_0})}{4(q_{K_0} T q_0)}, \\
t' \leq 2(t + (q_H + q_E + 2q_d) t_0),
\]

where $e_2 \geq \frac{\epsilon_0}{2^{t_0}} - q_d (2q_d + q_{K_0}) 2^{-1N} - 2^{t_0}$, $t_0$ is the time of an exponentiation in $\mathbb{Z}_N$, and $q_H, q_{K_0}, q_E$ and $q_d$ are the number of queries to the oracles $H$, $K_0$, KeyExtract and DelegationGen, respectively.

**Proof.** Given the adversary $A_{II}$, we construct another algorithm $B$ which runs $C_{A_{II}}$ on inputs $(N, e, y = \gamma^x \mod N)$. The $B$’s goal is to output $\gamma = y^{\bar{x}} \mod N$. Algorithm $C_{A_{II}}$ runs $A_{II}$, which breaks existential unforgeability of the proposal, on inputs $mpk = (N, e)$ and answers $A_{II}$’s oracle queries. Since $A_{II}$ has secret keys of all proxy signers, it can generate valid proxy ring signatures if and only if it forges a valid delegation. It is assumed that algorithm $C_{A_{II}}$ maintains initially empty associative arrays $T[\cdot]$ and $K_{0}[\cdot]$, and answers $A_{II}$’s oracle queries as follows.

- $K_0[R_{o} || w || \text{ID}[11]$ queries: If $T[K_{0}[R_{o} || w || \text{ID}[11]$ is defined then $C_{A_{II}}$ returns its value; otherwise, $C_{A_{II}}$ chooses $T[K_{0}[R_{o} || w || \text{ID}[11] \leftarrow \{0,1\}^t$, and returns $T[K_{0}[R_{o} || w || \text{ID}[11]$ to $A_{II}$.

- $H(ID_{w})$ queries: If $T[ID_{w}] = (b, x_{u}, X_{u})$ then $C_{A_{II}}$ returns $X_{u}$. If this entry is not yet defined, it chooses $x_{u} \leftarrow \mathbb{Z}_{N}^{*}$ and tosses a biased coin $b$ so that $b = 0$ with probability $\beta$ and $b = 1$ with probability $1 - \beta$. If $b = 0$, then $C_{A_{II}}$ sets $X_{u} = x_{u}^{0} \mod N$; if $b = 1$, it sets $X_{u} = x_{u}^{0} y \mod N$. It stores $T[ID_{u}] \leftarrow (b, x_{u}, X_{u})$ and returns $X_{u}$ to $A_{II}$.

- KeyExtract queries for $ID_{w}$: Algorithm $C_{A_{II}}$ looks up $T[ID_{w}] = (b, x_{u}, X_{u})$, if this entry is not yet defined, it performs a query $H(ID_{w})$. If $b = 0$, then $C_{A_{II}}$ returns $x_{u}$; otherwise, it sets $bad_{KE} \leftarrow true$ and aborts the execution of $A_{II}$.

- DelegationGen queries for $(w, \text{ID})$ under identity $ID_{o}$: Algorithm $C_{A_{II}}$ performs a query $H(ID_{o})$ and looks up $T[ID_{o}] = (b, x_{o}, X_{o})$. If $b = 0$, then $C_{A_{II}}$ simulates the delegation of $ID_{o}$ with the DelegationGen algorithm $\sigma_{o} \leftarrow \text{DelegationGen}(\text{Para}, mpk, x_{o}, w, \text{ID})$ since $C_{A_{II}}$ knows $x_{o}$, the original signer’s secret key. If $b = 1$, $C_{A_{II}}$ first chooses $c_{\sigma} \leftarrow \{0,1\}^t$ and $s_{\sigma} \leftarrow \mathbb{Z}_{N}^{*}$, and computes $R_{o} \leftarrow s_{\sigma}^{c_{\sigma}} mod N$. If $T[K_{0}[R_{o} || w || \text{ID}[11]$ has already been defined, then $C_{A_{II}}$ sets $bad_{DG} \leftarrow true$ and halts; otherwise, it sets $T[K_{0}[R_{o} || w || \text{ID}[11] \leftarrow c_{\sigma}$, and returns $\sigma_{o} = (R_{o}, s_{\sigma}, c_{\sigma})$ to $A_{II}$. 


Claim 1. \( \Pr[\neg \text{bad}_{KE}] \geq \beta E \).

Proof. \( \Pr[\neg \text{bad}_{KE}] \) is the probability that \( C_{A_{II}} \) does not abort as a result of \( A_{II} \)'s KeyExtract queries. The algorithm \( C_{A_{II}} \) aborts at answering to a KeyExtract query when \( \text{bad}_{KE} \) is set to true which means that \( b = 1 \) for a given identity. The probability of this event is \( 1 - \beta \), so the probability that \( C_{A_{II}} \) does not abort for one KeyExtract query is \( \beta \). Since \( A_{II} \) makes at most \( q_E \) KeyExtract queries, the probability that \( C_{A_{II}} \) does not abort as a result of \( q_E \) KeyExtract queries is at least \( \beta^{q_E} \).

Claim 2. \( \Pr[\neg \text{bad}_{DG}|\neg \text{bad}_{KE}] \geq (1-q_{d}(q_d+q_{K_0}))2^{-l_N}) - q_d2^{-l_N}. \)

Proof. Events \( \neg \text{bad}_{KE} \) and \( \neg \text{bad}_{DG} \) are independent, so \( \Pr[\neg \text{bad}_{DG}|\neg \text{bad}_{KE}] = \Pr[\neg \text{bad}_{DG}] \). The value of \( \Pr[\neg \text{bad}_{DG}] \) is the probability that \( C_{A_{II}} \) does not abort as a result of DelegationGen queries. The algorithm \( C_{A_{II}} \) aborts at answering to a DelegationGen query if \( \text{bad}_{DG} \) is set to true which means that there is a conflict in the table \( T_{K_0}[.] \). The probability of finding a conflict in \( T_{K_0}[.] \) for one DelegationGen query \( (w||ID, ID_o) \) equals the probability that \( (R_o||w||ID)[11] \) generated in a DelegationGen simulation has been occurred by chance in a previous query to the oracle \( K_0 \). Since there are at most \( q_{K_0} + q_d \) entries in the table \( T_{K_0}[.] \) and the number of \( R_o \), uniformly distributed in \( Z_N \), is \( 2^{l_N} \), the probability of this event for one DelegationGen query is at most \( (q_d+q_{K_0})2^{-l_N} \). Hence, the probability of this event for \( q_d \) queries is at most \( q_d(q_d+q_{K_0})2^{-l_N} \). In addition, this probability includes the probability that \( C_{A_{II}} \) previously used the same randomness \( R_o \), uniformly distributed in \( Z_N \), in one DelegationGen simulation. Since there are at most \( q_d \) DelegationGen simulations, this probability is at most \( q_d2^{-l_N} \). Therefore, for \( q_d \) DelegationGen queries, the probability of this event is at most \( q_d2^{-l_N} \).

Therefore, the probability that \( C_{A_{II}} \) does not halt in signature simulation is at least \( \eta \geq \beta^{q_E} - q_d(2q_d + q_{K_0})2^{-l_N} \).

Since the forgery is valid, we have \( s'_c = R_o(H(ID_o))^{c_o} \), \( A_{II} \) has not asked the warrant \( (w||ID) \) from DelegationGen algorithm under original signer’s identity \( ID_o \) and \( ID_o \) has not asked as a KeyExtract query. Also, the probability of having \( H(ID_o) = x_o^{c_o} \) is \( 1 - \beta \). Then, \( C_{A_{II}} \) looks up \( T[.] \) for \( ID_o \) to obtain the value \( x_o \), and returns a useful output \( (R_o, s_o, c_o, x_o) \) with probability \( \epsilon_1 \geq \epsilon(1-\beta)\eta \geq \epsilon(1-\beta)^{q_E} - q_d(2q_d + q_{K_0})2^{-l_N} \).

The value of \( \beta^{q_E}(1-\beta) \) is maximized for \( \beta = \frac{q_{K_0} + q_d}{q_{K_0} + q_d + 1} \). With substituting the value of \( \beta \), we obtain \( \beta^{q_E}(1-\beta) = \left(\frac{q_{K_0} + q_d}{q_{K_0} + q_d + 1}\right)^{q_E} = \frac{1}{q_{K_0} + q_d + 1}^{q_E}. \) If \( q_E = 0 \), this value is 1 and \( (1-\frac{1}{q_{K_0} + q_d + 1})^{1+q_E} \) is a monotonically increasing sequence for \( q_E \geq 1 \). Therefore, the lower bound of \( \beta^{q_E}(1-\beta) \) is \( \frac{1}{q_{K_0} + q_d + 1} \). As a consequence, \( C_{A_{II}} \) returns a useful output \( (R_o, s_o, c_o, x_o) \) with probability \( \epsilon_1 \geq \frac{1}{q_{K_0} + q_d} - q_d(2q_d + q_{K_0})2^{-l_N} \).

Since \( K_0 \) is a random oracle, the probability of the event that \( c_o = K_0(R_o||w||ID)[11] \) is less than \( 2^{-l_o} \), unless it is asked during the attack. Hence, in what follows it is likely that query \( (R_o||w||ID)[11] \) has been asked during a successful attack. The lower bound of probability of producing a valid forgery after making query to \( K_0 \) oracle is \( \epsilon_2 \geq \epsilon_1 - 2^{-l_o} \). Then, \( B \) uses the oracle replay technique [47] to solve the RSA problem.

Algorithm \( B \) employs \( q \) copies of \( C_{A_{II}} \), guesses a fixed index \( 1 \leq \kappa \leq (q_{K_0} + q_d) \) and hopes that \( \kappa \) be the index of query \( (R_o||w||ID)[11] \) to oracle \( K_0 \) for which \( A_{II} \) forges a delegation, and the probability of a good guess by chance is \( \frac{1}{q_{K_0} + q_d} \). Algorithm \( B \) gives the same system parameters, the same identities and the same sequence of random bits to the two copies of \( C_{A_{II}} \), and responds with the same random answers to their queries for the oracles until they ask the oracle \( K_0 \) for \( \kappa \)th query. At that point (the \( \kappa \)th query to the oracle \( K_0 \)), \( B \) gives two random answers \( c_o \) and \( c'_o \) such that \( c_o \neq c'_o \) to the hash queries \( K_\kappa \). Hence, \( B \) obtains two useful outputs (a useful pair) \( (R_o, s_o, c_o, x_o) \) and \( (R_o, s'_o, c'_o, x_o) \) after \( A_{II} \) asks the same query.
(R_o||w||ID||11) from K_o. We employ Splitting Lemma to compute the probability of B in returning a useful pair.

It is assumed that \( \Gamma \) denotes the set of successful executions of \( C_{A_{II}} \), and its success probability of \( C_{A_{II}} \) in returning a useful output is taken over the space (X, Y), where X is the set of random bits and random oracle responses that \( C_{A_{II}} \) takes up except for randomness related to the oracle \( K_o \) and Y is the set of random oracle responses to the oracle \( K_o \). Hence, we have \( \Pr[(X, Y) \in \Gamma] = \epsilon_2 \). With Splitting Lemma, we split the randomness \( Y \) related to \( K_o \) to \( (Y', c_o) \), where \( Y' \) is the set of all random responses to different queries of \( K_o \) except for \( \gamma \)th query whose answer is denoted as \( c_o \). The Splitting Lemma ensures the existence of a subset of executions \( \Omega \) such that \( \Pr[\Omega | \Gamma] \geq \frac{\gamma}{2} = \frac{1}{2} \), and for each \((X, Y') \in \Omega\), \( \Pr_{c_o'}[(X, Y', c_o') \in \Gamma] \geq \delta - \gamma = \frac{\epsilon_2}{2(q_{K_o} + q_d)} \).

If B replays the attack with fixed \((X, Y')\) and a randomly chosen \( c_o' \in \{0, 1\}^{l_0} \), it gets another successful pair \(((X, Y'), c_o') \) such that \( c_o \neq c_o' \) with probability \( \frac{\epsilon_2(1 - 2^{-l_0})}{4(q_{K_o} + q_d)} \).

After two successful executions of \( C_{A_{II}} \), B obtains \(((X, Y'), c_o)\) and \(((X, Y'), c_o')\), \( c_o \neq c_o' \), which means that it obtains a useful pair \((R_o, s_o, c_o, x_o)\) and \((R_o, s_o', c_o', x_o)\) with probability \( \epsilon' \geq \frac{\epsilon_2(1 - 2^{-l_0})}{4(q_{K_o} + q_d)} \), where \( \epsilon_2 \geq \epsilon_1 - 2^{-l_0} \).

From the useful pair \((R_o, s_o, c_o, x_o)\) and \((R_o, s_o', c_o', x_o)\), B computes the RSA inversion of \( y \) as follows. Since these useful outputs are derived from valid forgeries, we have

\[
s_o^e = R_o(x_o^e y)^{c_o}
\]
and

\[
s_o'^e = R_o(x_o'^e y)^{c_o}'.
\]

By dividing the two aforementioned equations, we obtain \((x_o^{(c_o' - c_o)} y) = y^{(e - c_o' \epsilon)} \mod N \). Since \( c_o \neq c_o' \in \{0, 1\}^{l_0} \) and \( e \) is a prime of length strictly greater than \( l_0 \), we have \( e > (c_o - c_o') \) and therefore \( gcd(e, c_o - c_o') = 1 \). Using the extended Euclidean algorithm, one can find \( a, b \in \mathbb{Z} \) such that \( ae + b(c_o - c_o') = 1 \). Hence, we have \( y = y^{ae + b(c_o - c_o')} = (y^a x_o^{(c_o' - c_o) \frac{s_a}{t}})^b \) \( \mod N \). Therefore, algorithm B outputs \((y^{a}(x_o^{(c_o' - c_o) \frac{s}{t}}))^b \) as the RSA inversion of \( y \) with probability \( \epsilon' \).

Algorithm B’s run-time \( t' \) is twice of \( A_{III} \)’s run-time, \( t \), plus the time required to respond to hash queries, \( q_E \) KeyExtract and \( q_d \) DelegationGen queries. To estimate the required time of signature generation, it is assumed that a (multi-) exponentiation in \( \mathbb{Z}_N \) takes \( t_o \) time while all other operations take zero time. Since each random oracle \( H \) or KeyExtract query takes at most one exponentiation, a delegation simulation takes 2 exponentiations, B’s run-time is \( t' \leq 2(t + (q_H + q_E + 2q_d)t_e) \). This completes the proof.

**Theorem 2.** The proposed scheme is \((t, q_H, q_{K_i}, q_E, q_{prs}, \epsilon)\)-secure against an adversary \( A_{III} \) if the RSA function associated to \( K_{g_{rsa}} \) is \((t', \epsilon')\)-one-way, and

\[
\epsilon' \geq \frac{(\epsilon_1 - 2^{-l_0})^2(1 - 2^{-l_1})}{8(q_{K_o} + q_{prs})(q_{K_o} + q_{K_i} + 1)},
\]

\[
t' \leq 2(t + (q_H + q_E + (2z + 1)q_{prs})t_e),
\]

where \( \epsilon_1 \geq \frac{(2^{2t_o} - 2q_{prs}^2 + q_{prs}a)^2}{2^{2t_o}} - (2q_{prs}^2 q_{K_i})2^{2t_o} \), \( t_e \) is the time of an exponentiation in \( \mathbb{Z}_N \), \( q_H, q_{K_i}, q_E \) and \( q_{prs} \) are the number of queries to the oracles \( H, K_o, K_i, KeyExtract \) and ProxyRingSign, respectively.

**Proof.** Given the adversary \( A_{III} \), we construct another algorithm B which runs \( C_{A_{III}} \) on inputs \((N, e, y = \gamma^e \mod N)\). The B’s goal is to output \( \gamma = y^{\frac{a}{t}} \mod N \). Algorithm \( C_{A_{III}} \) runs \( A_{III} \), which breaks existential unforgeability of the proposal, on inputs \( mpk = (N, e) \) and answers \( A_{III} \)’s oracle queries. Since \( A_{III} \) has the secret key of the original signer, it can simulate delegations by itself, and the oracle access to DelegationGen is not necessary. It is assumed that algorithm \( C_{A_{III}} \) maintains initially empty associative arrays \( T[.] \), \( T_{K_o}[.] \) and \( T_{K_i}[.] \), and answers \( A_{III} \)’s oracle queries as follows.

- \( K_o(R_o||w||ID||11) \) queries: If \( T_{K_o}[R_o||w||ID||11] \) is defined then \( C_{A_{III}} \) returns its value, otherwise \( C_{A_{III}} \) chooses \( T_{K_o}[R_o||w||ID||11] \leftarrow \{0, 1\}^{l_0} \), and returns \( T_{K_o}[R_o||w||ID||11] \) to \( A_{III} \).
Claim 3. $\Pr[-\text{bad}_{KE}] \geq \beta^{q_e}$.

Proof. The proof is similar to the proof of Claim 1.

Claim 4. $\Pr[-\text{bad}_{PS}|-\text{bad}_{KE}] \geq 1 - q_{prs}(q_{prs} + q_{K_1})^{2^{-l_N}} - q_{prs}^{2}2^{-l_N}$.

Proof. Events $-\text{bad}_{KE}$ and $-\text{bad}_{PS}$ are independent, so $\Pr[-\text{bad}_{PS}|-\text{bad}_{KE}] = \Pr[-\text{bad}_{PS}]$. The value of $\Pr[-\text{bad}_{PS}]$ is the probability that $C_{A_{111}}$ does not abort as a result of ProxyRingSign queries. The algorithm $C_{A_{111}}$ aborts at answering to a ProxyRingSign query if $\text{bad}_{PS}$ is set to true which means that there is a conflict in table $T_{K_1}[.]$ for these kinds of queries. The probability of finding a conflict in $T_{K_1}[.]$ for one ProxyRingSign query equals the probability that $(R_aH(ID_u)^cR_oH(ID_o)^c||ID||ID_u||w||m||11)$ generated in ProxyRingSign simulation has been occurred by chance in a previous query to the oracle $K_1$. Since there are at most $q_{K_1} + q_{prs}$ entries in the table $T_{K_1}[.]$ for these kinds of queries and the number of $R_a$, uniformly distributed in $\mathbb{Z}_N$, is $2^{l_N}$, the probability of this event for one ProxyRingSign is at most $(q_{prs} + q_{K_1})^{2^{-l_N}}$. Hence, the probability of this event for $q_{prs}$ queries is at most $q_{prs}(q_{prs} + q_{K_1})^{2^{-l_N}}$. In addition, this probability includes the probability that $C_{A_{111}}$ previously used the same randomness $R_a$, uniformly distributed in $\mathbb{Z}_N$, in one ProxyRingSign simulation. Since there are at most $q_{prs}$ Prox-
yRingSign simulations, this probability is at most $q_{prs}2^{-l_N}$. Therefore, for $q_{prs}$ ProxyRingSign queries
the probability of this event is at most $q_{prs}^22^{-l_N}$.

Since the forgery is valid, we have $R_u = r_u^1, c_{u+1} = K_1(R_u H(I D_u)^{c_u} R_o H(I D_o)^{c_o} || I D || I D_u || w || m || 11)$
for $0 \leq u \leq z - 1$, and $c_z = c_0$, and also $A_{III}$ has not asked the message $m$ from ProxyRingSign algorithm
under proxy signer's identity set $I D \subseteq I D$ and it contains $z$ uncorrupted identities with probability at least
$(1 - \beta)^z$. Algorithm $C_{A_{III}}$ performs additional random oracle queries $H(I D_u)$ for identities in the forgery to
find $T(I D_u) = \{b, x_u, X_u\}$ for them, and returns $(R_o, r_{0u}, r_{1u}, c_{o}, \{x_u\}_{0 \leq u \leq z - 1}, x_o, m, u)$. As a result, the
probability of returning a useful output is at least $\epsilon_1(1 - \beta)^z \eta \geq \epsilon_1(1 - \beta)^z \beta^q E - q_{prs}(2q_{prs} + q_{Kr}) 2^{-l_N}$. The value of $\beta^q E (1 - \beta)^z$ is maximized for $\beta = \frac{q_{prs}}{q_{prs} + 1}$. With substituting the value of $\beta$, we have $\beta^q E (1 - \beta)^z \geq \frac{1}{2^2 q_E}$.

Since $K_1$ is a random oracle, the probability of the event
\[
c_{u+1} = K_1(R_u H(I D_u)^{c_u} R_o H(I D_o)^{c_o} || I D || I D_u || w || m || 11)
\]
for $0 \leq u \leq z - 1$ is less than $\epsilon_1 z 2^{-l_N}$, unless they are asked during the attack. Hence, it is likely that
questions $(R_u H(I D_u)^{c_u} R_o H(I D_o)^{c_o} || I D || I D_u || w || m || 11)$ for $0 \leq u \leq z - 1$ are asked during a successful
attack. The lower bound of probability of producing a useful output after making queries to $K_1$ oracle is $\epsilon_2 \geq \epsilon_1 - z 2^{-l_N}$.

It is assumed that $T$ denotes the set of successful executions of $C_{A_{III}}$, and its success probability of $C_{A_{III}}$
in returning a useful output after making query to $K_1$ is taken over the space $(X, Y)$, where $X$ is the set of random
bits and random oracle responses that $C_{A_{III}}$ takes up except for the randomness related to the oracle $K_1$, and $Y$ is the set of random oracle responses to the oracle $K_1$. Hence, we have $Pr[(X, Y) \in T] \geq \epsilon_2$. There is at least one index $\nu \in (0, ..., z - 1)$ such that the query $Q_\nu = (R_o H(I D_o)^{c_o} R_o H(I D_o)^{c_o} || I D || I D_u || w || m || 11)$ was made to the oracle $K_1$ before query $Q_\nu = (R_o H(I D_o)^{c_o} R_o H(I D_o)^{c_o} || I D || I D_u || w || m || 11)$. The pair $(u, v)$ is called a gap index. If there are more than one gap index for a forged proxy ring signature, only the pair with smallest value for $u$ is considered.

The cardinality of the set $T_{u, v}$ as a subset of $T$ with gap index $(u, v)$ is $\pi = \frac{(q_{Kr} + q_{prs})(q_{Kr} + q_{prs} + 1)}{2}$. This gives us a partition of $T$ in exactly $\pi$ classes. Let $I$ be the set of most likely gap indices, $I = \{(u, v)| Pr[T_{u, v}] \geq \frac{1}{2 \pi}\}$. Hence, for each $(u, v) \in I$, $T_{u, v}$ is denoted as $T_{u, v}^I$, we have $Pr[T_{u, v}^I] = Pr[T_{u, v}] \geq \frac{1}{2 \pi}$.

With Splitting Lemma, we split the randomness $Y$ related to $K_1$ to $(Y', c_0)$, where $Y'$ is the set of all random
responses to different queries of $K_1$ except for query $Q_\nu$ whose answer is denoted as $c_0$. This lemma ensures the existence of a subset $T_{u, v, o}$ of executions $(X, Y)$ such that $Pr[O_{u, v} \in T_{u, v, o}] \geq \frac{1}{2}$ and for each $(X, Y) \in O_{u, v}, Pr_{c_o}[(\omega, (\nu', c'_o)) \in T_{u, v, o}] \geq \delta = \frac{1}{2\pi}$.

Since $T_{u, v}$ are disjoint, and we have $Pr[(X, Y) \in I s.t. T_{u, v} \cap T^I_{u, v, o}] = \sum_{(u, v) \in I} Pr[O_{u, v} \in T_{u, v}] = \sum_{(u, v) \in I} Pr[T_{u, v}^I \cap T_{u, v}] \geq \frac{1}{2 \pi}$. Therefore, with probability at least $\frac{1}{2}, (u, v) \in I$ and $(X, Y) \in O_{u, v, o}$.

If we replay the attack with fixed $(X, Y')$ and a randomly chosen $c'_o$, we get another successful pair $(X, (Y', c'_o))$ such that $c_o \neq c'_o$ with probability $2(1 - \frac{1}{2 \pi})$.

Hence, after two successful executions of $C_{A_{III}}$, the algorithm $B$ obtains $((X, Y'), c_o)$ and $((X, Y'), c'_o)$ with probability $\epsilon' \geq \frac{1}{16} q_{Kr} + q_{prs} (q_{Kr} + q_{prs} + 1)$, which means that $B$ obtains a useful pair $(R_o, r_{0u}', r_{1u}', c_{o}', \{x_u\}_{0 \leq u \leq z - 1}, x_o, m, w)$ and $(R_o, r_{0u}', r_{1u}', c_{o}', \{x_u\}_{0 \leq u \leq z - 1}, x_o, m, w)$. Since the query $Q_u = (R_o H(I D_o)^{c_o} R_o H(I D_o)^{c_o} || I D || I D_u || w || m || 11)$ was made to the oracle $K_1$ before query $Q_v = (R_{o-1} H(I D_{o-1})^{c_o} R_o H(I D_o)^{c_o} || I D || I D_{o-1} || w || m || 11)$, we have
\[
R_o H(I D_o)^{c_o} R_o H(I D_o)^{c_o} = R_{o-1} H(I D_{o-1})^{c_o} R_o H(I D_o)^{c_o},
\]
where $c_v \neq c'_o$ and $R_{o-1} = r_{o-1}'$. With rearranging the above equation, we obtain $(x_{c_o}^{c'_o - c_o} R_{o-1}^{r_{o-1}')})^e = y^{(c'_o - c_o)} mod N$.

Since $c_v \neq c'_o \in \{0, 1\}^l_1$ and $e$ is a prime of length strictly greater than $l_1$, we have $e > (c'_o - c_o)$ and therefore gcd($e, (c'_o - c_o)$) = 1. Using the extended Euclidean algorithm, one can find $a, b \in Z$ such that
$ae + b(c_x - c_y) = 1$. Hence, we have $y = y^{ae+b(c_x - c_y)} = (y^a (x_c^{c_x} c_y^{c_y}))^b \mod N$. Therefore, algorithm $B$ can output $(y^a (x_c^{c_x} c_y^{c_y}))^b$ as the RSA inversion of $y$ with probability $e'$. Algorithm $B$’s run-time $t'$ is twice of $A_{II}$’s run-time, $t$, plus the time required to respond to hash queries, $q_E$ KeyExtract and $q_{prs}$ ProxyRingSign queries. To estimate the required time of signature simulation, it is assumed that a (multi-) exponentiation in $\mathbb{Z}_N$ takes $t_x$ time, while all other operations take zero time. Since each random oracle $H$ or KeyExtract query takes at most one exponentiation, a ProxyRingSign simulation takes $(2x + 1)$ exponentiations, $B$’s run-time is $t' = 2(t + (q_h + q_E + (2x + 1)q_{prs})t_x)$. This completes the proof.

**Theorem 3.** The identity-based proxy ring signature scheme is $(t, q_H, q_{K_0}, q_{K_1}, q_c, q_d, q_{prs}, \frac{1}{2})$-PSI-secure since the probability of $D$ in guessing the identity of the proxy signer for a given signature $\theta$, $Pr[D(\theta) = 1 | ID]$ (where $ID_j \in \tilde{ID} = \{ID_0, ID_1\}$), is $\frac{1}{2}$ against $(t, q_H, q_{K_0}, q_{K_1}, q_c, q_d, q_{prs}, \epsilon)$-bounded adversary $D$.

**Proof.** The distinguisher $D$ issues a polynomially bounded number of random oracle, KeyExtract, DelegationGen and ProxyRingSign queries adaptively as explained in the proof of Theorems 1 and 2.

Then, $D$ chooses two honest identities $ID_0$ and $ID_1$ for proxy ring ($D$ never make KeyExtract query for these two identities), and makes a DelegationGen and ProxyRingSign query on $(w, ID)$ under an identity $ID_0$, and on the message $m \in w$ under the identity set $\tilde{ID} = \{ID_0, ID_1\} \subseteq ID$, respectively. In response, $C$ chooses $j \leftarrow \{0, 1\}$, runs DelegationGen on $(w, ID)$ under an identity $ID_0$ to obtain $\sigma_0$ and returns $\theta \leftarrow ProxyRingSign(Para, mpk, ID_0, ID, (w, m, \sigma_0), x_j)$ to $D$. Finally, the distinguisher $D$ outputs $j' = j$ with probability $\frac{1}{2}$. To show the value of this probability, we compute the probability that $ID_j$ generates valid values for $R_0$ and $R_1$ of $\theta$ which are pairwise different. The probability of choosing different values for $R_0$ and $R_1$ is $\frac{1}{2N} - \frac{1}{2N}$. Then, $\theta$ is computed from random numbers $r_u$ for $u \neq j$ in $R_u$ and $r$ employed in $R_j$. The probability of generation of the proxy ring signature $\theta = (R_o, r_0, r_1, c_0)$ is independent from the identity of the real signer $ID_j$, then, this probability is the same for two members in the set of proxy signers. Therefore, the probability of $D$ in guessing the real signer is $\frac{1}{2}$.

### 3.3 On achieving identity-based proxy ring signatures without bilinear pairings in the standard model

Although our scheme is the first short identity-based proxy ring signature which is efficient (due to the not relying on bilinear pairings), it is proved secure in the random oracle model. In fact, an identity-based proxy ring signature without bilinear pairings is sequential aggregation of an identity-based standard signature without bilinear pairings (a non-interactive proof of knowledge of the secret key of an original signer) and a non-interactive proof of knowledge of the secret key of one of the proxy signers in the proxy ring. On the other hand, to the best of our knowledge, Fiat-Shamir heuristic is used to implement a non-interactive proof of knowledge, and therefore its security relies on the randomness of the underlying hash function. So far there is no scheme for identity-based proxy ring signatures without bilinear pairings with provable security in the standard model, and hence this leads to the difficulty of achieving identity-based proxy ring signatures without bilinear pairings in the standard model, which is an interesting future research problem.

### 4 Comparison

The comparison for some provably secure (identity-based) proxy ring signature schemes is summarized in Table 1. The comparison is in terms of $DelegGen-Cost$, $DelegVer-Cost$, $PRS-Sign-Cost$ and $PRV-Cost$, dominating computational cost in delegation generation, delegation verification, proxy ring signature generation and proxy ring signature verification, respectively. In Table 1, $exp$ denotes exponentiation in $\mathbb{Z}_N$. For the sake of comparison, it is assumed that other operations take zero time and $z = n$ which means that $\tilde{ID} = ID$.

Since previous identity-based proxy ring signature schemes ([32–39]) do not support provable security, they are not considered in comparison. As shown in Table 1, our scheme compared to Asaar et al.’s provably secure proxy ring signature scheme [40] has a proper advantage in signature-size. In a nutshell, since $l_1 \ll |\mathbb{Z}_N|$ (for example, $l_1$ is about 160 bits, while $|\mathbb{Z}_N| = 1024$), the signature-size is improved by a factor $|\mathbb{Z}_N| - l_1$. 

$\text{DelegGen-Cost}$ | $\text{DelegVer-Cost}$ | $\text{PRS-Sign-Cost}$ | $\text{PRV-Cost}$
--- | --- | --- | ---
$\text{Asaar et al.}$ | $\text{Our scheme}$ | $\text{Our scheme}$ | $\text{Our scheme}$

$\text{PRSign-Cost}$ | $\text{DeleVer-Cost}$ | $\text{DeleVer-Cost}$ | $\text{DeleVer-Cost}$
--- | --- | --- | ---
$\text{Asaar et al.}$ | $\text{Our scheme}$ | $\text{Our scheme}$ | $\text{Our scheme}$

$\text{DelegGen-Cost}$ | $\text{DelegVer-Cost}$ | $\text{PRS-Sign-Cost}$ | $\text{PRV-Cost}$
--- | --- | --- | ---
$\text{Asaar et al.}$ | $\text{Our scheme}$ | $\text{Our scheme}$ | $\text{Our scheme}$

$\text{PRSign-Cost}$ | $\text{DeleVer-Cost}$ | $\text{DeleVer-Cost}$ | $\text{DeleVer-Cost}$
--- | --- | --- | ---
$\text{Asaar et al.}$ | $\text{Our scheme}$ | $\text{Our scheme}$ | $\text{Our scheme}$
Table 1. Comparison between our proposal and provably secure schemes

<table>
<thead>
<tr>
<th>Scheme</th>
<th>DeleGen Cost</th>
<th>DeleVer Cost</th>
<th>PRSign Cost</th>
<th>PRVer Cost</th>
<th>Sign Size</th>
<th>ID-based</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td>2exp</td>
<td>2exp</td>
<td>(n + 2)exp</td>
<td>(2n + 1)exp</td>
<td>(n + 1)Z_N + l</td>
<td>✓</td>
</tr>
<tr>
<td>Asaar et al. [40]</td>
<td>2exp</td>
<td>2exp</td>
<td>(2n + 1)exp</td>
<td>(n + 2)exp</td>
<td>(n + 2)Z_N</td>
<td>✓</td>
</tr>
</tbody>
</table>

On one hand, since the probability of solving the RSA problem as shown in Theorem 4 is nearly $\epsilon^2/(qK_1 + qprs)^z$, and so is independent of the number of proxy signers.

On the other hand, in [40], the probability of solving the RSA problem is $\epsilon' \geq \frac{\epsilon^2(1-2^{-z})}{8 \sum_{j=1}^{qK_1+qprs-1} \prod_{i=0}^{z-1} (qK_1 + qprs - i - j)}$ where $\epsilon_1 \geq \frac{\epsilon^2}{qK_1 + qprs} - (2qprs + qprsqK_1)2^{-z-1} - (z+1)2^{-l}$. With approximately estimation, the probability is $\epsilon^2/(qK_1 + qprs)^z$, and so it is a function of the number of proxy signers in a proxy ring, $z$. Therefore, the security reduction is improved.

Furthermore, since the total number of exponentiations in proxy ring signature generation and verification for the two schemes are the same, the new scheme is as efficient as the old one.

5 Conclusion and future work

In this paper, we proposed a provably secure proxy ring signature scheme from RSA assumption. This scheme is the first short one for this type of signature from RSA. In addition, the security reduction of the proposal has been improved compared to the ones from RSA since it is independent of the number of proxy signers in the proxy ring. We should highlight that the proposed scheme has a proper advantage in efficiency due to the avoiding pairing computations since the cost of each pairing computation is roughly that of 2.3 exponentiations. Furthermore, the proxy key exposure attack is not applicable to our scheme since it is generated based on sequential aggregation paradigm.

Although our scheme is the first short identity-based proxy ring signature scheme which is efficient (due to the not relying on bilinear pairings), it is proved secure in the random oracle model. According to Section 4, there is no identity-based proxy ring signature scheme without bilinear pairing with provable security in the standard model. As a result, presenting an identity-based proxy ring signature scheme from traditional assumptions such as RSA and discrete logarithm with provable security in the standard model is an open problem, and proposing a scheme with the aforementioned features will be considered as a future work.

References


