From Rerandomizability to Sequential Aggregation: Efficient Signature Schemes Based on SXDH Assumption

Sanjit Chatterjee and R. Kabaleeshwaran✉

Department of Computer Science and Automation, Indian Institute of Science, Bangalore, India
{sanjit, kabaleeshwar}@iisc.ac.in

Abstract. An aggregate signature allows one to generate a short aggregate of signatures from different signers on different messages. A sequential aggregate signature (SeqAS) scheme allows the signers to aggregate their individual signatures in a sequential manner. All existing SeqAS schemes that do not use the random oracle assumption either require a large public key or the security depends upon some non-standard interactive/static assumptions. In this paper, we present an efficient SeqAS scheme with constant-size public key under the SXDH assumption. In the process, we first obtain an optimized (and more efficient) variant of Libert et al's randomizable signature scheme. While both the schemes are more efficient than the currently best ones that rely on some static assumption, they are only slightly costlier than the most efficient ones based on some interactive assumption.

Keywords: Rerandomizable signature, sequential aggregate signature, dual-form signature technique, SXDH assumption.

1 Introduction

The notion of rerandomizable signature (RRS) was introduced by Camenisch and Lysyanskaya [4]. Rerandomizability guarantees that given a signature σ on some message m under the public key PK, anybody can compute another valid signature on the same message which is indistinguishable from the original signature. The above feature makes RRS a very useful tool in building privacy-preserving protocols.

The notion of aggregate signature was introduced by Boneh et al. [3]. As the name suggests, aggregation allows one to generate a (compressed) aggregate of a collection of individual signatures on different messages generated by different signers. This notion is inspired by several applications such as certificate chains of public-key infrastructure and secure routing in the context of border gateway protocol [3]. Sequential aggregate signature (SeqAS), introduced in [24], is a special type of aggregate signature. In SeqAS each signer sequentially adds his/her signature on the aggregated-so-far signature. Lu et al. [23] presented the

first SeqAS scheme without random oracle based on the Waters signature [30] under the CDH assumption. However, their construction requires a large public key (linear in the security parameter).

In 2011, Schröder [28], showed how to construct a SeqAS scheme with constant-size public key based on the Camenisch-Lysyanskaya rerandomizable signature (CL-RRS). However, like the original CL-RRS scheme, security of Schröder's construction is based on a non-standard interactive assumption, called LRSW [25]. [28] relies upon the *randomness re-use* technique of [23], which makes use of the randomness of the so-far aggregated signature to construct the aggregate signature. Lee et al. [16] improved Schröder's construction further by introducing *public key sharing* technique, in which one of the elements from the public key of the underlying signature scheme (in this case, CL-RRS) is placed in the public parameter. Due to this new technique, they have achieved efficient verification and optimized public key size.

In 2013, Lee et al. [18] presented a SeqAS scheme with constant-size public key. Their SeqAS scheme is built on a signature scheme that supports multi-user setting and is publicly rerandomizable. In particular, the signature scheme is obtained by introducing suitable components to the signature derived from Lewko-Waters IBE [20] through Naor transformation. They have also used Gerbush et al's [7] dual-form signature technique to prove unforgeability under a previously introduced static assumption [20] along with some standard assumptions. Their follow-up work [17] improved upon the previous SeqAS in terms of signature size as well as signing/verification at the cost of a slightly larger public key under the same standard assumption along with two previously introduced static assumptions [20].

Apart from the reliance on non-standard assumption, another limitation of the CL-RRS scheme is that the signature size is linear in the number of message blocks signed. In 2016, Pointcheval and Sanders [26] presented another rerandomizable signature scheme (called, PS-RRS) where the signature size is independent of the message block length. However, unforgeability of PS-RRS scheme is proved under a new interactive assumption. Following the idea of [16], they have also presented an efficient SeqAS scheme based on the PS-RRS scheme.

In 2016, Libert et al. [21] presented a randomizable signature scheme (denoted as LMPY-RS). They suitably combined a previously proposed signature scheme [22] with a quasi-adaptive NIZK (QA-NIZK) argument [14] to obtain a constant size randomizable signature for multiple message blocks. Using the dual-form signature technique [7], they argue unforgeability of their construction under the SXDH assumption.

1.1 Our Contribution

Our first contribution is to propose an efficient rerandomizable signature scheme under the SXDH assumption. We then use the proposed RRS to realize a sequential aggregate signature scheme with constant-size public key under the SXDH assumption. The performance of the proposed schemes is very close to

that of previous proposals based on some *non-standard interactive assumption*. For detailed comparison, see Table 2 (resp. Table 4) for RRS (resp. SeqAS).

For the randomizable signature, compared to Libert et al. [21], our main novelty lies in the application of the QA-NIZK proof system of Kiltz and Wee [14]. In particular, instead of the real QA-NIZK proof component, we use the simulated one. Hence we first generate the secret exponent of the signature scheme and then define the trapdoor keys with respect to the linear subspace relation of the proof system. Then, using the dual-form signature technique [7] we argue unforgeability based on the SXDH assumption while full rerandomizability [8] is shown to follow unconditionally. See §3 for the detail.

Next, our RRS serves as a building block to construct a sequential aggregate signature (SeqAS) scheme in §4. Our construction employs both 'randomness reuse' and 'public key sharing' techniques [16]. Since the original LMPY-RS is not directly amenable to signature aggregation, we tweak the signature scheme to realize the desired functionality. As can be seen from Table 4, existing schemes in a similar setting have one of the following limitations. To have security based on a standard assumption, the scheme suffers from large public key size [23]. When public key size is constant, security relies on some non-standard assumption [19]. In contrast, we obtain an efficient construction with constant size public key (and signature) where security is argued based on the well-known SXDH assumption.

2 Preliminaries

For a prime p, $\mathbb{Z}_p^*$ denotes the set of all non-zero elements from $\mathbb{Z}_p$. We denote $a \xleftarrow{\$} A$ to be an element chosen uniformly at random from the non-empty set A. We define the bilinear group generator as follows.

Definition 1 *A bilinear group generator $\mathcal{P}$ is a probabilistic polynomial time (PPT) algorithm which takes the security parameter λ as input and outputs $\Theta = (p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, g, h)$, where p is prime, $\mathbb{G}$, $\mathbb{H}$ and $\mathbb{G}_T$ are the prime p order groups and g (resp. h) is an arbitrary generator of $\mathbb{G}$ (resp. $\mathbb{H}$) and $e : \mathbb{G} \times \mathbb{H} \longrightarrow \mathbb{G}_T$ is a bilinear map that satisfies, (i) **Bilinearity**: For all $g, g' \in \mathbb{G}$ and $h, h' \in \mathbb{H}$, one has $e(g \cdot g', h \cdot h') = e(g, h) \cdot e(g, h') \cdot e(g', h) \cdot e(g', h')$, (ii) **Non degeneracy**: If a fixed $g \in \mathbb{G}$ satisfies $e(g, h) = 1$ for all $h \in \mathbb{H}$, then $g = 1$ and similarly for elements of $\mathbb{H}$ and (iii) **Computability**: The map e is efficiently computable.*

We recall the decisional Diffie-Hellman assumption (DDH) in $\mathbb{G}$ (denoted as $\mathrm{DDH}_{\mathbb{G}}$) as follows.

Assumption 1 *Given $(\Theta = (p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, g, h), g^a, g^b)$ and $T = g^{ab+\theta}$, it is hard to decide whether $\theta = 0$ or not, for $a, b \xleftarrow{\$} \mathbb{Z}_p$.*

In the same way, we can define the DDH assumption in $\mathbb{H}$ (denoted as $\mathrm{DDH}_{\mathbb{H}}$). When $\mathcal{P}$ satisfies the DDH assumption in both $\mathbb{G}$ and $\mathbb{H}$, then we say that $\mathcal{P}$ satisfies the symmetric external Diffie-Hellman (SXDH) assumption.

We recall from [11] the double pairing assumption (DBP) in $\mathbb{H}$ (denoted as $\mathrm{DBP}_{\mathbb{H}}$) as follows.

Assumption 2 *Given* $(\Theta = (p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, g, h), h_r, h_s)$*, it is hard to compute* $(R,\ S) \neq (1,1)$ *from* $\mathbb{G}^2$ *such that* $e(R, h_r)e(S, h_s) = 1$.

In the same way, we can define the DBP assumption in $\mathbb{G}$ (denoted as $\text{DBP}_{\mathbb{G}}$). From [1, Lemma 2], it is clear that, $\text{DBP}_{\mathbb{G}}$ is reducible to $\text{DDH}_{\mathbb{G}}$ and $\text{DBP}_{\mathbb{H}}$ is reducible to $\text{DDH}_{\mathbb{H}}$. The formal definition of digital signature, rerandomizable signature and sequential aggregate signature schemes can be found in Appendix A.

3 Rerandomizable Signature

In this section, we describe an efficient rerandomizable signature scheme, whose security is proved under the SXDH assumption. Our construction is inspired from [21]. In [21], one requires to know the public key components to randomize the signature, whereas in our scheme one needs to know only the underlying group order. This feature will later play an important role in the construction of aggregate signature of §4.

3.1 Construction

Libert et al. [21] presented a randomizable signature scheme (denoted as `LMPY-RS`) based on QA-NIZK proof system. In particular, to prove that a vector of group elements belongs to some linear subspace, [12,14] showed that the argument size will be independent of the subspace dimension. [21] exploited this property to obtain a randomizable signature scheme for multiple block messages with a constant size signature. In order to prove unforgeability, they have used Gerbush et al's [7] dual-form signature technique.

In `LMPY-RS` scheme the secret key SK consists of ω from $\mathbb{Z}_p$ and the public key PK consists of $\big(g, g_0, h, U_1 = g^{u_1}, \{V_{1j} = g^{v_{1j}}\}_{j=1}^{\ell}, \Omega = g_0^{\omega}, \texttt{CRS}\big)$ with `CRS` consisting of $\big(z, U_2, \{V_{2j}\}_{j=1}^{\ell}, h_z,\ h_0 = h_z^{\delta_0}, h_{\ell+1} = h_z^{\delta_{\ell+1}}, h_{10} = h_z^{\delta_{10}}, h_{20} = h_z^{\delta_{20}}, \{h_{1j} = h_z^{\delta_{1j}}, h_{2j} = h_z^{\delta_{2j}}\}_{j=1}^{\ell}\big)$, where $g, z, g_0, U_1, U_2,\ V_{1j}, V_{2j}$ are from $\mathbb{G}$ and $h, h_z,\ h_0, h_{\ell+1}, h_{10}, h_{20}, h_{1j},\ h_{2j}$ are from $\mathbb{H}$. Note that the trapdoor information $\big(\delta_0, \delta_{\ell+1}, \delta_{10}, \delta_{20}, \{\delta_{1j}, \delta_{2j}\}_{j=1}^{\ell}\big)$ are generated in such a way that $z = g^{\delta_0} g_0^{\delta_{\ell+1}}$, $U_2 = U_1^{\delta_0} g^{\delta_{10}} g_0^{\delta_{20}}$ and $V_{2j} = V_{1j}^{\delta_0} g^{\delta_{1j}} g_0^{\delta_{2j}}$ holds. Hence one can write $U_2 = g^{u_2}, \{V_{2j} = g^{v_{2j}}\}_{j=1}^{\ell}$, where $u_2 = \delta_0 u_1 + \delta_{10} + a\delta_{20}$, $v_{2j} = \delta_0 v_{1j} + \delta_{1j} + a\delta_{2j}$ and $g_0 = g^a$. The randomizable signature on the message $\boldsymbol{m} = (m_1, \ldots, m_\ell)$ consists of $\sigma = (\sigma_1, \sigma_2, \sigma_3, \pi)$, where

$$\sigma_1 = g^{\omega}\big(U_1 \prod_{j=1}^{\ell} V_{1j}^{m_j}\big)^s,\ \sigma_2 = g^s,\ \sigma_3 = g_0^s,\ \pi = z^{\omega}\big(U_2 \prod_{j=1}^{\ell} V_{2j}^{m_j}\big)^s.$$

Note that the signature component π corresponds to the QA-NIZK proof that the statement $(\sigma_1, \sigma_2^{m_1}, \ldots, \sigma_2^{m_\ell}, \sigma_2, \sigma_3^{m_1}, \ldots, \sigma_3^{m_\ell}, \sigma_3, \Omega)$ belongs to a linear subspace generated by the matrix $M \in \mathbb{Z}_p^{(\ell+2)\times(2\ell+4)}$ as defined in Equation 1. Our `RRS` scheme is obtained from `LMPY-RS` scheme by removing the first row and the

last $\ell+2$ columns of the matrix M, which results in the matrix N as defined in Equation 1.

$$M = \left(\begin{array}{c|c|c|c} g & 1\ 1\ \ldots\ 1\ 1 & 1\ \ 1\ \ \ldots\ \ 1\ \ 1 & g_0 \\ V_{11} & g\ 1\ \ldots\ 1\ 1 & g_0\ \ 1\ \ \ldots\ \ 1\ \ 1 & 1 \\ V_{12} & 1\ g\ \ldots\ 1\ 1 & 1\ \ g_0\ \ldots\ \ 1\ \ 1 & 1 \\ \vdots & \vdots\ \vdots\ \ddots\ \vdots\ \vdots & \vdots\ \ \vdots\ \ \ddots\ \ \vdots\ \ \vdots & \\ V_{1\ell} & 1\ 1\ \ldots\ g\ 1 & 1\ \ 1\ \ \ldots\ g_0\ \ 1 & 1 \\ U_1 & 1\ 1\ \ldots\ 1\ g & 1\ \ 1\ \ \ldots\ \ 1\ \ g_0 & 1 \end{array}\right), N = \left(\begin{array}{c|c} V_{11} & g\ 1\ \ldots\ 1\ 1 \\ V_{12} & 1\ g\ \ldots\ 1\ 1 \\ \vdots & \vdots\ \vdots\ \ddots\ \vdots\ \vdots \\ V_{1\ell} & 1\ 1\ \ldots\ g\ 1 \\ U_1 & 1\ 1\ \ldots\ 1\ g \end{array}\right). \quad (1)$$

Notice that the removal of the first row of M corresponds to putting $\omega = 0 \bmod p$ which amounts to removing g^{ω} (resp. z^{ω}) from σ_1 (resp. π) and Ω from PK. The last $\ell+2$ columns of M correspond to the second generator g_0 and $\{\{h_{2j}\}_{j=1}^{\ell}, h_{20}, h_{\ell+1}\}$ in PK of `LMPY-RS`. In the signature generation, we directly use the simulated proof component instead of the real QA-NIZK proof component. This allows to set SK containing the trapdoor dependent information $u_2, \{v_{2j}\}_{j=1}^{\ell}$ along with $g, u_1, \{v_{1j}\}_{j=1}^{\ell}$.

Table 1. `RRS` scheme in the prime-order setting.

Setup(λ)

Run $\mathcal{P}(\lambda) \to (p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, g, h)$, where $g \xleftarrow{\$} \mathbb{G}, h \xleftarrow{\$} \mathbb{H}$,
Return $PP = (p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, g, h)$.

KeyGen(PP)

Choose $\delta_0, u_1, u_2, \{v_{1j}, v_{2j}\}_{j=1}^{\ell} \xleftarrow{\$} \mathbb{Z}_p$, $h_z \xleftarrow{\$} \mathbb{H}$ and set $h_0 := h_z^{\delta_0}$,
$\delta_{10} := u_2 - \delta_0 u_1$, $\delta_j := v_{2j} - \delta_0 v_{1j}$, for all $j \in [1, \ell]$.
Set $SK := \{g, u_1, u_2, \{v_{1j}, v_{2j}\}_{j=1}^{\ell}\}$, $PK := \{h_z, h_0, h_{10} := h_z^{\delta_{10}}, \{h_j := h_z^{\delta_j}\}_{j=1}^{\ell}\}$.
Return (SK, PK).

Sign($SK, \boldsymbol{m} = (m_1, \ldots, m_\ell)$)

Choose $r \xleftarrow{\$} \mathbb{Z}_p$ and set $A := g^{r(u_1 + \sum_{j=1}^{\ell} v_{1j} m_j)}, B := g^r, C := g^{r(u_2 + \sum_{j=1}^{\ell} v_{2j} m_j)}$.
Return $(\boldsymbol{m}, \sigma = (A, B, C))$.

Ver($PK, \boldsymbol{m} = (m_1, \ldots, m_\ell), \sigma = (A, B, C)$)

Parse the message and signature and check

$$B \neq 1 \text{ and } e(A, h_0) e(B, h_{10} \prod_{j=1}^{\ell} h_j^{m_j}) = e(C, h_z). \quad (2)$$

If the above two conditions hold, then return *accept*, otherwise return *reject*.

Rand($PK, \boldsymbol{m} = (m_1, \ldots, m_\ell), \sigma = (A, B, C)$)

If $\mathsf{Ver}(PK, \boldsymbol{m}, \sigma)$=1, then choose $s \xleftarrow{\$} \mathbb{Z}_p$ and compute $A' := A^s, B' := B^s, C' := C^s$.
Return $\sigma' = (A', B', C')$.
Else Return $\perp$.

The `RRS` scheme consists of four PPT algorithms, which are defined in Table 1. Notice that, we avoid the trivial forgery attack by checking $B \neq 1$. Suppose, we

do not check the above condition, then anyone can output $\sigma = (1, 1, 1)$ as a (trivial) forgery on any message $\boldsymbol{m} \in \mathbb{Z}_p^\ell$. Correctness of the scheme can be verified using the following derivation,

$$\begin{aligned} e(A, h_0)e(B, h_{10}\prod_{j=1}^{\ell} h_j^{m_j}) &= e(g^{r(u_1+\sum_j v_{1j}m_j)}, h_z^{\delta_0})e(g^r, h_z^{\delta_{10}+\sum_j \delta_j m_j}) \\ &= e(g^{r(\delta_0 u_1+\delta_{10})+r\sum_j(\delta_0 v_{1j}+\delta_j)m_j}, h_z) \\ &= e(g^{r(u_2+\sum_j v_{2j}m_j)}, h_z) = e(C, h_z). \end{aligned}$$

The first equality is obtained by substituting the values of the signature and public key components. Second equality is obtained using the bilinearity of the pairing map. The third equality is obtained by substituting the value of u_2 and v_{2j} components from Table 1.

Notice that, it is sufficient to consider the elements $U_1 = g^{u_1}, U_2 = g^{u_2}$ and $\{V_{1j} = g^{v_{1j}}, V_{2j} = g^{v_{2j}}\}_{j=1}^{\ell}$, as part of the SK. However, for better efficiency, we consider the respective exponents of U_1, U_2, V_{1j} and V_{2j} as part of the SK, which saves 2ℓ many exponentiation and multiplication in the group $\mathbb{G}$.

3.2 Randomizability

The main feature of a rerandomizable signature scheme is the so-called *randomizability property*. This feature has been utilized effectively in the construction of several other protocols, such as group signature [2] and anonymous credential scheme [4].

Theorem 1 *The RRS scheme satisfies perfect randomizability.*

Proof. We argue that our RRS scheme satisfies perfect randomizability. To establish that, it is sufficient to prove that the signature returned by Rand and Sign are identically distributed. First, we consider the signature $\sigma = (A, B, C)$ returned by the adversary $\mathcal{A}$ using Sign on the message $\boldsymbol{m} = (m_1, \ldots, m_\ell)$. In particular, we write

$$A = g^{r(u_1+\sum_j v_{1j}m_j)}, B = g^r, C = g^{r(u_2+\sum_j v_{2j}m_j)},$$

for some randomness r from $\mathbb{Z}_p$. Then we consider the signature $\sigma_1 = (A_1, B_1, C_1)$ returned by Rand on the message and signature pair $(\boldsymbol{m}, \sigma)$, where $A_1 = A^s = g^{sr(u_1+\sum_j v_{1j}m_j)}$, $B_1 = B^s = g^{rs}$ and $C_1 = C^s = g^{sr(u_2+\sum_j v_{2j}m_j)}$, for some randomness s from $\mathbb{Z}_p$. Now we consider the signature $\sigma_0 = (A_0, B_0, C_0)$ returned by Sign on the same message $\boldsymbol{m}$, where

$$A_0 = g^{z(u_1+\sum_j v_{1j}m_j)}, B_0 = g^z, C_0 = g^{z(u_2+\sum_j v_{2j}m_j)}$$

for some randomness z from $\mathbb{Z}_p$. Notice that, in the signature σ_1, the exponent s is the source of randomness whereas in the signature σ_0, the exponent z is the source of randomness. Thus it is clear that both the signatures σ_0 and σ_1 are identically distributed, as rs and z are independent and identically distributed. □

3.3 Unforgeability

We use the Gerbush et al's [7] dual-form signature technique and prove unforgeability under the SXDH assumption. Note that, unlike [21], we argue unforgeability without appealing to the security of the underlying QA-NIZK proof system.

Partition of forgery space: Let $\mathcal{V}$ be the set of all message and signature pairs such that they verify under the public key PK. We partition the forgery class $\mathcal{V}$ into two disjoint sets $\mathcal{V}_I$ and $\mathcal{V}_{II}$ which are defined as follows.

Type-I: $\mathcal{V}_I = \{(\boldsymbol{m}^*, \sigma^*) \in \mathcal{V} : S_1^* = 1 \text{ and } S_2^* = 1\}$,
Type-II: $\mathcal{V}_{II} = \{(\boldsymbol{m}^*, \sigma^*) \in \mathcal{V} : S_1^* \neq 1 \text{ and } S_2^* \neq 1\}$,

where $S_1^* := A^*(B^*)^{-u_1 - \sum_{j=1}^{\ell} v_{1j} m_j^*}$ and $S_2^* := (C^*)^{-1}(B^*)^{u_2 + \sum_{j=1}^{\ell} v_{2j} m_j^*}$. Now we argue that the Type-II forgery class is same as the complement of Type-I forgery class with respect to the forgery space $\mathcal{V}$, i.e., $\mathcal{V}_{II} = \mathcal{V} - \mathcal{V}_I$. Notice that from the verification Equation 2, we can simplify as follows.

$$
\begin{aligned}
1 &= e(A^*, h_0)e(B^*, h_{10}\prod_{j=1}^{\ell} h_j^{m_j^*})e(C^*, h_z)^{-1} \\
&= e(A^*, h_z^{\delta_0})e(B^*, h_z^{\delta_{10} + \sum_{j=1}^{\ell} \delta_j m_j^*})e(C^*, h_z)^{-1} \\
&= e(A^*, h_z^{\delta_0})e(B^*, h_z^{(u_2 - \delta_0 u_1) + \sum_{j=1}^{\ell} (v_{2j} - \delta_0 v_{1j}) m_j^*})e(C^*, h_z)^{-1} \\
&= e(A^*(B^*)^{-u_1 - \sum_{j=1}^{\ell} v_{1j} m_j^*}, h_z^{\delta_0})e((C^*)^{-1}(B^*)^{u_2 + \sum_{j=1}^{\ell} v_{2j} m_j^*}, h_z) \\
&= e(S_1^*, h_0)e(S_2^*, h_z)
\end{aligned}
$$

In the above derivation, the second equality is obtained by the values of h_{10} and h_j and then substituting the values of δ_{10} and δ_j. The third equality is obtained by using the bilinearity of the pairing map and the last equality is obtained by the definition of S_1^* and S_2^*. Suppose $S_1^* = 1$, then the above equation can be simplified as $e(S_2^*, h_z) = 1$. Then from the non-degeneracy of the pairing, we have S_2^* must be 1. In the same way, suppose $S_2^* = 1$, then S_1^* must be 1. Hence there is no valid forgery such that (i) $S_1^* = 1$ and $S_2^* \neq 1$ hold or (ii) $S_1^* \neq 1$ and $S_2^* = 1$ hold.

Structure of forged signature: Consider the message and signature pair $(\boldsymbol{m}^*, \sigma^*)$ satisfying the verification Equation 2, where $\boldsymbol{m}^* = (m_1^*, \ldots, m_\ell^*) \in \mathbb{Z}_p^\ell$ and $\sigma^* = (A^*, B^*, C^*) \in \mathbb{G}^3$. Suppose the forgery is Type-II, then we explain, how the signature components are written explicitly in terms of the secret exponents. Since $B^* \in \mathbb{G}$, $B^* \neq 1$ and g is the generator of $\mathbb{G}$, we can write $B^* = g^r$, for some $r \in \mathbb{Z}_p^*$. For the Type-II forgery, the condition $S_1^* \neq 1$ holds, we can write $S_1^* = g^{s_1}$, for some $s_1 \in \mathbb{Z}_p^*$. Then substituting B^* value in S_1^* we obtain that $A^* = g^{r\left(u_1 + \sum_{j=1}^{\ell} v_{1j} m_j^*\right) + s_1}$. The condition $S_2^* \neq 1$ holds for a Type-II forgery. Then we can write $S_2^* = g^{-s}$, for some $s \in \mathbb{Z}_p^*$. By substituting the value of B^* in S_2^*, we obtain that $C^* = g^{r(u_2 + \sum_{j=1}^{\ell} v_{2j} m_j^*) + s}$. Now from the verification Equation 2, the additional terms g^{s_1} and g^s must satisfy the condition $e(g^{s_1}, h_0) = e(g^s, h_z)$, so that the Type-II forgery is valid. From the above

condition, we can derive that $s_1 = s/\delta_0$. Hence a Type-II forgery can be written as,

$$A^* = g^{r(u_1+\sum_{j=1}^{\ell} v_{1j}m_j^*)+s/\delta_0}, \quad B^* = g^r, \quad C^* = g^{r(u_2+\sum_{j=1}^{\ell} v_{2j}m_j^*)+s}, \qquad (3)$$

for some $r, s \in \mathbb{Z}_p$.

Suppose the forgery is Type-I, then we see that both conditions, $S_1^* = 1$ and $S_2^* = 1$ hold. From the above explanation for the Type-II forgery case and the above conditions, it is clear that s has to be zero modulo p for a Type-I forgery. Hence by substituting $s = 0$ in Equation 3, we obtain the desired form of Type-I forgery as defined in Table 1.

Two signing algorithms: Let Sign_A be same as the Sign algorithm defined in Table 1. Next we define the following Sign_B algorithm, which is used by the simulator in the unforgeability proof. The Sign_B algorithm takes the secret key SK along with element δ_0 from $\mathbb{Z}_p$ and the message $\boldsymbol{m} \in \mathbb{Z}_p^{\ell}$ and outputs a message-signature pair.

$\mathsf{Sign}_B(SK \cup \{\delta_0\}, \boldsymbol{m} = (m_1, \ldots, m_\ell))$: Choose $r, s \xleftarrow{\$} \mathbb{Z}_p$ and compute $A := g^{r(u_1+\sum_{j=1}^{\ell} v_{1j}m_j)+s/\delta_0}, B := g^r$ $C := g^{r(u_2+\sum_{j=1}^{\ell} v_{2j}m_j)+s}$. Return $(\boldsymbol{m}, \sigma := (A, B, C))$.

From the verification Equation 2, the additional element g^{s/δ_0} in A paired with $h_z^{\delta_0}$ is same as the additional element g^s in C paired with h_z. Hence, the signature returned by Sign_B also verifies under PK.

Proof Intuition: We prove unforgeability of our RRS scheme using a hybrid argument. Let Game_R be the real EUF-CMA security game, i.e., given the public key, adversary $\mathcal{A}$ makes q many signing oracle queries which are answered using Sign_A and returns a forgery (from $\mathcal{V}$) on a new message. Next, we define a new game Game_0 which is similar to Game_R except that $\mathcal{A}$ returns a Type-I forgery. The only difference between Game_R and Game_0 is that of $\mathcal{A}$ producing a Type-II forgery. Then we prove that under the $\mathrm{DBP}_{\mathbb{H}}$ assumption, $\mathcal{A}$ cannot return a Type-II forgery, which ensures that Game_R and Game_0 are indistinguishable. In this reduction, simulator $\mathcal{B}$ embeds the DBP instance to generate the public key terms h_z and $h_z^{\delta_0}$. Then by choosing all the other secret exponents, $\mathcal{B}$ can answer for Sign_A queries. Finally, from the Type-II forgery returned by $\mathcal{A}$, $\mathcal{B}$ computes the solution for the $\mathrm{DBP}_{\mathbb{H}}$ instance.

Next, we define another game Game_k which is similar to Game_0, except that the first k signing queries are answered using Sign_B algorithm. Then we prove that Game_{k-1} and Game_k are indistinguishable under the $\mathrm{DDH}_{\mathbb{G}}$ assumption. In this reduction, simulator $\mathcal{B}$ embeds one of the terms (say g^b) from the DDH instance to define u_1 and u_2. In particular, $\mathcal{B}$ defines $u_1 = \tilde{u}_1 + tb/\delta_0$ and $u_2 = \tilde{u}_2 + tb$, for random exponents $\tilde{u}_1, \tilde{u}_2, \delta_0, t$. Then $\mathcal{B}$ uses the other term (say g^a) from the DDH instance for the k-th signature and embeds g^a and $g^{ab+\theta}$ to answer for the k-th signing query. For a given DDH tuple with $\theta = 0$ we are simulating Game_{k-1}, otherwise we are simulating Game_k.

Finally, we argue that the advantage of Game_q is negligible under the $\mathrm{DBP}_{\mathbb{H}}$ assumption. In this reduction, simulator $\mathcal{B}$ embeds the DBP instance to simulate

the public key components h_z and $h_z^{\delta_0}$. Then $\mathcal{B}$ defines the exponents u_1 and u_2 in such a way that $\mathcal{B}$ can answer for Sign_B oracle queries. In particular, $\mathcal{B}$ defines $u_1 = \tilde{u}_1 - t/\delta_0$ and $u_2 = \tilde{u}_2 - t$, for random exponents $t, \tilde{u}_1, \tilde{u}_2$. Once the adversary returns a Type-I forgery, then $\mathcal{B}$ could extract the solution for the DBP problem.

Theorem 2 *If SXDH assumption holds in $\mathcal{P}$, then the* `RRS` *scheme is EUF-CMA secure.*

Proof. First we define the following games.

- Game_R: This is the original EUF-CMA game. Recall that, after receiving the PK from the challenger, the adversary $\mathcal{A}$ makes q many signing oracle queries adaptively and then returns a forgery on a new message.
- Game_0: Same as Game_R except that $\mathcal{A}$ returns a forgery from $\mathcal{V}_I$. Let E be the event that $\mathcal{A}$ returns a forgery from $\mathcal{V}_{II}$ in Game_0. In Lemma 3, we prove that the event E happens with negligible probability under $\mathrm{DBP}_{\mathbb{H}}$ assumption. Thus we deduce that Game_R and Game_0 are computationally indistinguishable under $\mathrm{DBP}_{\mathbb{H}}$ assumption. In particular we have,
$$|Adv_{\mathcal{A}}^{\mathsf{Game_R}} - Adv_{\mathcal{A}}^{\mathsf{Game_0}}| \leq Pr[E] \leq Adv_{\mathcal{B}}^{DBP_{\mathbb{H}}}.$$
- Game_k: Same as Game_0 except that the first k signing queries are answered using Sign_B, for $k \in [1, q]$, whereas the last $q-k$ queries are answered using Sign_A. For $k \in [1, q]$, in Lemma 4, we prove that Game_{k-1} and Game_k are computationally indistinguishable under $\mathrm{DDH}_{\mathbb{G}}$ assumption. In particular we have,
$$|Adv_{\mathcal{A}}^{\mathsf{Game_{k-1}}} - Adv_{\mathcal{A}}^{\mathsf{Game_k}}| \leq Adv_{\mathcal{B}}^{DDH_{\mathbb{G}}}.$$

Finally in Lemma 5, we prove that $Adv_{\mathcal{A}}^{\mathsf{Game_q}}$ is negligible under $\mathrm{DBP}_{\mathbb{H}}$ assumption. In particular we have,

$$Adv_{\mathcal{A}}^{\mathsf{Game_q}} \leq Adv_{\mathcal{B}}^{DBP_{\mathbb{H}}}.$$

Hence by the hybrid argument and from Equations 4, 5 and 6, described below, we have,

$$\begin{aligned}
Adv_{\mathcal{A}}^{UF} = Adv_{\mathcal{A}}^{\mathsf{Game_R}} &= |Adv_{\mathcal{A}}^{\mathsf{Game_R}} - Adv_{\mathcal{A}}^{\mathsf{Game_0}} + Adv_{\mathcal{A}}^{\mathsf{Game_0}} - Adv_{\mathcal{A}}^{\mathsf{Game_1}} + \ldots + \\
&\quad Adv_{\mathcal{A}}^{\mathsf{Game_{k-1}}} - Adv_{\mathcal{A}}^{\mathsf{Game_k}} + \ldots - Adv_{\mathcal{A}}^{\mathsf{Game_q}} + Adv_{\mathcal{A}}^{\mathsf{Game_q}}| \\
&\leq |Adv_{\mathcal{A}}^{\mathsf{Game_R}} - Adv_{\mathcal{A}}^{\mathsf{Game_0}}| + \sum_{k=1}^{q} |Adv_{\mathcal{A}}^{\mathsf{Game_{k-1}}} - Adv_{\mathcal{A}}^{\mathsf{Game_k}}| + |Adv_{\mathcal{A}}^{\mathsf{Game_q}}| \\
&\leq Adv_{\mathcal{B}}^{DBP_{\mathbb{H}}} + q\, Adv_{\mathcal{B}}^{DDH_{\mathbb{G}}} + Adv_{\mathcal{B}}^{DBP_{\mathbb{H}}} \\
&\leq (q+2)\, Adv_{\mathcal{B}}^{SXDH}.
\end{aligned}$$

□

Lemma 3 *If $DBP_{\mathbb{H}}$ assumption holds in $\mathcal{P}$, then $Pr[E]$ is negligible.*

Proof. Assume that the event E happens with some non-negligible probability. Then we construct a simulator $\mathcal{B}$ to break the $\text{DBP}_{\mathbb{H}}$ assumption as follows. $\mathcal{B}$ is given Θ and h_r, h_s from $\mathbb{H}$ and his goal is to compute $(R, S) \neq (1, 1)$ from $\mathbb{G}^2$ such that $e(R, h_r)e(S, h_s) = 1$. Now $\mathcal{B}$ chooses $u_1, u_2, \{v_{1j}, v_{2j}\}_{j=1}^{\ell}$ uniformly at random from $\mathbb{Z}_p$. First $\mathcal{B}$ implicitly sets $\delta_{10} = u_2 - \delta_0 u_1$ and $\delta_j = v_{2j} - \delta_0 v_{1j}$, for $j \in [1, \ell]$. Then $\mathcal{B}$ defines the public key as,

$$PK := \big\{h_z := h_r, h_0 := h_s,\, h_{10} := h_r^{u_2} h_s^{-u_1}, \{h_j := h_r^{v_{2j}} h_s^{-v_{1j}}\}_{j=1}^{\ell}\big\}.$$

Once PK is given to $\mathcal{A}$, he makes q many signing oracle queries to $\mathcal{B}$. Since $\mathcal{B}$ knows all the SK components such as $g, u_1, u_2, \{v_{1j}, v_{2j}\}_{j=1}^{\ell}$, he can answer all the signing queries using Sign_A algorithm.

Finally, $\mathcal{A}$ returns a forgery $(\boldsymbol{m}^*, \sigma^*)$, where $\boldsymbol{m}^* = (m_1^*, \ldots, m_\ell^*) \in \mathbb{Z}_p^\ell$ and $\sigma^* = (A^*, B^*, C^*) \in \mathbb{G}^3$. Then $\mathcal{B}$ checks (i) the forgery $(\boldsymbol{m}^*, \sigma^*)$ is valid and (ii) the message $\boldsymbol{m}^*$ is not queried earlier. If any of these checks fail to hold, then $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ tries to solve the $\text{DBP}_{\mathbb{H}}$ assumption as follows. First $\mathcal{B}$ computes $S = A^*(B^*)^{-u_1 - \sum_{j=1}^{\ell} v_{1j} m_j^*}$ and $R = (C^*)^{-1}(B^*)^{u_2 + \sum_{j=1}^{\ell} v_{2j} m_j^*}$. Since the forgery is valid, hence it satisfies the verification Equation 2 which can be re-written and then simplified as follows.

$$\begin{aligned}
1 &= e(C^*, h_z)^{-1} \cdot e(A^*, h_0) \cdot e(B^*, h_{10} \prod_j h_{1j}^{m_j^*}) \\
&= e(C^*, h_r)^{-1} e(A^*, h_s) \cdot e(B^*, h_r^{u_2} h_s^{-u_1} \prod_j (h_r^{v_{2j}} h_s^{-v_{1j}})^{m_j^*}) \\
&= e((C^*)^{-1}(B^*)^{u_2 + \sum_j v_{2j} m_j^*}, h_r) e(A^*(B^*)^{-u_1 - \sum_j v_{1j} m_j^*}, h_s) \\
&= e(R, h_r) e(S, h_s).
\end{aligned}$$

In the above derivation, the second equality follows from the structure of public key components and the third equality follows from the bilinearity of the pairing map. The last equality follows from the definition of R and S.

In order to break the $\text{DBP}_{\mathbb{H}}$ problem, it is sufficient to argue that $(R, S) \neq (1, 1)$. From our contradiction assumption, $\mathcal{A}$ returns a Type-II forgery, then it must satisfy $S = A^*\ (B^*)^{-u_1 - \sum_j v_{1j} m_j^*} \neq 1$ and $R = (C^*)^{-1}(B^*)^{u_2 + \sum_j v_{2j} m_j^*} \neq 1$. Then $\mathcal{B}$ returns (R, S) as a non-trivial solution for the $\text{DBP}_{\mathbb{H}}$ instance. Thus we have,

$$Pr[E] \leq Adv_{\mathcal{B}}^{DBP_{\mathbb{H}}}. \tag{4}$$

□

Lemma 4 *If $DDH_{\mathbb{G}}$ assumption holds in $\mathcal{P}$, then $\mathsf{Game}_{k-1} \approx_c \mathsf{Game}_k$, for $k \in [1, q]$.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$, who distinguishes Game_{k-1} from Game_k with some non-negligible probability under the condition that $\mathcal{A}$

returns a Type-I forgery. Then we construct a simulator $\mathcal{B}$ to break the $\mathrm{DDH}_{\mathbb{G}}$ assumption as follows. $\mathcal{B}$ is given $\Theta, g^a, g^b, g^{ab+\theta}$ and his goal is to decide whether $\theta = 0 \bmod p$ or not. Now $\mathcal{B}$ chooses $\delta_0, \tilde{u}_1, \tilde{u}_2, t$ uniformly at random from $\mathbb{Z}_p$ and implicitly sets $u_1 = \tilde{u}_1 + tb/\delta_0$ and $u_2 = \tilde{u}_2 + tb$. $\mathcal{B}$ also chooses h_z uniformly at random from $\mathbb{H}$ and defines $h_0 = h_z^{\delta_0}$. Then $\mathcal{B}$ defines $\delta_{10} := u_2 - \delta_0 u_1 = \tilde{u}_2 - \delta_0 \tilde{u}_1$ and hence s/he simulates h_{10} as $h_z^{\tilde{u}_2 - \delta_0 \tilde{u}_1}$. $\mathcal{B}$ also chooses v_{1j}, v_{2j} uniformly at random from $\mathbb{Z}_p$ and defines h_j as $h_z^{v_{2j} - \delta_0 v_{1j}}$. Here PK consists of $\{h_z, h_0, h_{10}, \{h_j\}_{j=1}^{\ell}\}$, which is then sent to $\mathcal{A}$. Notice that $\mathcal{B}$ can simulate the SK components $g, \{v_{1j}, v_{2j}\}_{j=1}^{\ell}$ along with δ_0. However, $\mathcal{B}$ can simulate $U_1 := g^{\tilde{u}_1}(g^b)^{t/\delta_0}$ and $U_2 = g^{\tilde{u}_2}(g^b)^t$, so that he can answer for the signing queries. $\mathcal{B}$ computes $V_{1j} := g^{v_{1j}}, V_{2j} := g^{v_{2j}}$, for $j \in [1, \ell]$.

After receiving PK, $\mathcal{A}$ makes signing queries on some message $\boldsymbol{m}_i = (m_{i1}, \ldots, m_{i\ell})$. For the first $k-1$ (resp. last $q-k$) queries, $\mathcal{B}$ uses Sign_B (resp. Sign_A) algorithm to answer for signing queries, as he knows the components $g, U_1, U_2, \{V_{1j}, V_{2j}\}_{j=1}^{\ell}$ as well as δ_0. In particular, Sign_B queries are answered by computing $\sigma_i = (A_i, B_i, C_i)$, where

$$A_i = (U_1 \prod_j V_{1j}^{m_{ij}})^r g^{s/\delta_0}, \quad B_i = g^r, \quad C_i = (U_2 \prod_j V_{2j}^{m_{ij}})^r g^s$$

and r, s are chosen uniformly at random from $\mathbb{Z}_p$. However, Sign_A queries are answered by letting $s = 0$ in the above signature obtained using Sign_B algorithm. For the k-th query, $\mathcal{B}$ embeds the DDH instance to construct the signature $\sigma_k = (A_k, B_k, C_k)$, where $B_k := g^a$ and

$$\begin{aligned} A_k &:= (g^a)^{\tilde{u}_1}(g^{ab+\theta})^{t/\delta_0}(g^a)^{\sum_j v_{1j} m_{kj}} = g^{a\left((\tilde{u}_1 + tb/\delta_0) + \sum_j v_{1j} m_{kj}\right)} g^{t\theta/\delta_0} \\ &= g^{a(u_1 + \sum_j v_{1j} m_{kj}) + t\theta/\delta_0}, \\ C_k &:= (g^a)^{\tilde{u}_2}(g^{ab+\theta})^{t}(g^a)^{\sum_j v_{2j} m_{kj}} = g^{a\left((\tilde{u}_2 + tb) + \sum_j v_{2j} m_{kj}\right)} g^{t\theta}, \\ &= g^{a(u_2 + \sum_j v_{2j} m_{kj}) + t\theta}. \end{aligned}$$

In the above derivation, we re-arrange the terms appropriately and use the definition of u_1 and u_2. Note that the exponent a from the DDH instance is used to simulate the signature randomness whereas $s = t\theta \bmod p$. Suppose $\theta = 0 \bmod p$, then $s = 0 \bmod p$, i.e., the signature σ_k is distributed as an output of Sign_A. If $\theta \neq 0 \bmod p$, then $s \neq 0 \bmod p$, i.e., the signature σ_k is distributed as an output of Sign_B with non-zero exponent $s = t\theta \bmod p$.

Finally, $\mathcal{A}$ returns a forgery $(\boldsymbol{m}^*, \sigma^*)$. As before, $\mathcal{B}$ checks (i) the forgery is valid and (ii) the message $\boldsymbol{m}^* = (m_1^*, \ldots, m_\ell^*)$ is not queried earlier. Note that σ_k is generated using the DDH instance. Since $\mathcal{B}$ knows δ_0, U_1, U_2 and all the other secret key components, $\mathcal{B}$ can generate the k-th signature of any type properly. However, $\mathcal{B}$ cannot on her/his own decide the type of the signatures generated using the problem instance, as s/he cannot compute S_1^* and S_2^* which uses the exponents u_1 and u_2. In other words, $\mathcal{B}$ needs to rely on the advantage of $\mathcal{A}$.

From Lemma 3, under DBP assumption, $\mathcal{A}$ only returns a Type-I forgery. Also from our initial contradiction assumption, $\mathcal{A}$ distinguishes between Game_{k-1}

and Game_k with some non-negligible probability. So $\mathcal{B}$ leverages $\mathcal{A}$ to break the DDH assumption. Thus we have,

$$\left|Adv_{\mathcal{A}}^{\mathsf{Game}_{k-1}} - Adv_{\mathcal{A}}^{\mathsf{Game}_k}\right| \leq Adv_{\mathcal{B}}^{DDH_{\mathbb{G}}}. \tag{5}$$

□

Lemma 5 *If $DBP_{\mathbb{H}}$ assumption holds in $\mathcal{P}$, then Adv^{Game_q} is negligible.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$, who wins in Game_q and produces a Type-I forgery with some non-negligible probability. Then we construct a simulator $\mathcal{B}$ to break the $\mathrm{DBP}_{\mathbb{H}}$ assumption as follows. $\mathcal{B}$ is given Θ and h_r, h_s from $\mathbb{H}$ and his goal is to compute $(R, S) \neq (1, 1)$ from $\mathbb{G}^2$ such that $e(R, h_r)e(S, h_s) = 1$. Now $\mathcal{B}$ sets $h_z := h_r$, $h_z^{\delta_0} := h_s$ and implicitly sets $u_1 := \tilde{u}_1 - t/\delta_0$ and $u_2 := \tilde{u}_2 - t$, for randomly chosen $\tilde{u}_1, \tilde{u}_2, t$ from $\mathbb{Z}_p$. Thus $\mathcal{B}$ can simulate $U_1' := g^{\tilde{u}_1} = g^{u_1 + t/\delta_0}$ and $U_2' := g^{\tilde{u}_2} = g^{u_2+t}$. Next $\mathcal{B}$ chooses $\{v_{1j}, v_{2j}\}_{j=1}^{\ell}$ uniformly at random from $\mathbb{Z}_p$ and implicitly sets $\delta_{10} := u_2 - \delta_0 u_1 = \tilde{u}_2 - \delta_0 \tilde{u}_1$ and $\delta_j := v_{2j} - \delta_0 v_{1j}$, for $j \in [1, \ell]$. Then $\mathcal{B}$ computes $h_{10} = h_z^{\delta_{10}} = h_r^{\tilde{u}_2} h_s^{-\tilde{u}_1}$ and $h_j = h_z^{\delta_j} = h_r^{v_{2j}} h_s^{-v_{1j}}$, for $j \in [1, \ell]$. Now $\mathcal{B}$ defines the PK as $\big(h_z, h_0, h_{10}, \{h_j\}_{j=1}^{\ell}\big)$. Notice that $\mathcal{B}$ knows the secret exponents $\{v_{1j}, v_{2j}\}_{j=1}^{\ell}$. Hence $\mathcal{B}$ computes $V_{1j} = g^{v_{1j}}, V_{2j} = g^{v_{1j}}$, for $j \in [1, \ell]$.

After receiving PK, $\mathcal{A}$ makes signing oracle queries on the message $\boldsymbol{m}_i = (m_{i1}, \ldots, m_{i\ell})$. Then $\mathcal{B}$ answers the Sign_B queries by computing $\sigma_i = (A_i, B_i, C_i)$, where $B_i := g^r$ and

$$A_i := \big(U_1' \prod_j V_{1j}^{m_{ij}}\big)^r = \big(g^{u_1 + t/\delta_0} \prod_j (g^{v_{1j}})^{m_{ij}}\big)^r = g^{r(u_1 + \sum_j v_{1j} m_{ij}) + rt/\delta_0},$$
$$C_i := \big(U_2' \prod_j V_{2j}^{m_{ij}}\big)^r = \big(g^{u_2 + t} \prod_j (g^{v_{2j}})^{m_{ij}}\big)^r = g^{r(u_2 + \sum_j v_{2j} m_{ij}) + rt},$$

for r randomly chosen from $\mathbb{Z}_p$. From the above derivation, it is clear that signature σ_i is properly distributed as an output of Sign_B with $s = rt$ modulo p. Finally $\mathcal{A}$ returns a forgery $(\boldsymbol{m}^*, \sigma^*)$, where $\boldsymbol{m}^* = (m_1^*, \ldots, m_\ell^*)$ and $\sigma^* = (A^*, B^*, C^*)$. As before, $\mathcal{B}$ checks (i) the forgery is valid and (ii) $\boldsymbol{m}^*$ is not queried earlier. If any of these checks fail to hold then $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ breaks the DBP assumption as follows.

From the contradiction assumption, $\mathcal{A}$ returns a Type-I forgery with some non-negligible probability. Then, $\mathcal{B}$ can write the Type-I forgery components as, $A^* = g^{r(u_1 + \sum_{j=1}^{\ell} v_{1j} m_j^*)}$, $B^* = g^r$ and $C^* = g^{r(u_2 + \sum_{j=1}^{\ell} v_{2j} m_j^*)}$, for some $r \in \mathbb{Z}_p$. Now $\mathcal{B}$ computes

$$S = A^* (B^*)^{-\tilde{u}_1 - \sum_{j=1}^{\ell} v_{1j} m_j^*} = g^{r(\tilde{u}_1 - t/\delta_0) - r\tilde{u}_1} = g^{-rt/\delta_0},$$
$$R = (C^*)^{-1} (B^*)^{\tilde{u}_2 + \sum_{j=1}^{\ell} v_{2j} m_j^*} = g^{-r(\tilde{u}_2 - t) + r\tilde{u}_2} = g^{rt}.$$

In the above derivation, we have used the definition of $u_1 = \tilde{u}_1 - t/\delta_0$ and $u_2 = \tilde{u}_2 - t$. Then one can verify that $e(R, h_r)e(S, h_s) = e(g^{rt}, h_z)e(g^{-rt/\delta_0}, h_z^{\delta_0}) = 1$,

here we have used the values of h_r and h_s. From the verification Equation 2, $B^* \neq 1$ holds and hence $R \neq 1$ holds. In other words, (R, S) is a non-trivial solution of $\text{DBP}_{\mathbb{H}}$ problem instance. Thus we have,

$$Adv_{\mathcal{A}}^{\mathsf{Game_q}} \leq Adv_{\mathcal{B}}^{DBP_{\mathbb{H}}}. \tag{6}$$

□

3.4 Comparison

In Table 2, we compare our rerandomizable signature scheme with some existing schemes in the prime-order pairing setting. We use the following metrics: public key size (denoted as $|PK|$), signature size (denoted as $|\sigma|$), signing cost, verification cost and the computational assumption required to prove unforgeability.

Table 2. Comparing rerandomizable signatures for multiple block messages.

	$\lvert PK\rvert$	$\lvert\sigma\rvert$	Signing Cost	Verification Cost	Assum.
PS-RRS	$(\ell+2)\lvert\mathbb{H}\rvert$	$2\lvert\mathbb{G}\rvert$	$2E_{\mathbb{G}}$	$2\mathbb{P} + \ell(E_{\mathbb{H}} + M_{\mathbb{H}})$	PS
`LMPY-RS`	$(2\ell+5)\lvert\mathbb{G}\rvert + (2\ell+6)\lvert\mathbb{H}\rvert$	$4\lvert\mathbb{G}\rvert$	$6E_{\mathbb{G}} + (2\ell+2)M_{\mathbb{G}}$	$5\mathbb{P} + 3M_{\mathbb{G}_T} + 2\ell M_{\mathbb{H}}$	SXDH
`RRS` §3.1	$(\ell+3)\lvert\mathbb{H}\rvert$	$3\lvert\mathbb{G}\rvert$	$3E_{\mathbb{G}}$	$3\mathbb{P} + M_{\mathbb{G}_T} + \ell(E_{\mathbb{H}} + M_{\mathbb{H}})$	SXDH

For any group $X \in \{\mathbb{G}, \mathbb{H}, \mathbb{G}_T\}$, E_X, M_X respectively denote the cost of exponentiation, multiplication in X and $|X|$ is the bit size of X whereas $\mathbb{P}$ denotes pairing computation cost. PS denote the interactive assumption used in [26].

We denote PS-RRS to be the rerandomizable signature scheme described in [26]. As we can see, PS-RRS is an efficient scheme in terms of the size of the public key and signature as well as the running time of signing and verification algorithms. However, unforgeability of the PS-RRS scheme is proved under an interactive assumption [26, Assumption 1].

Libert et al's [21] randomizable signature scheme (`LMPY-RS`) is currently the most efficient one under the SXDH assumption. Here the size of the public key and running time of the signing algorithm are at least three times that of the PS-RRS scheme whereas, the signature size is double and the verification time is two and a half times that of the PS-RRS scheme.

The performance of the `RRS` scheme proposed in this paper is roughly two times better than that of `LMPY-RS` scheme, in terms of public key size and running time of the signing and verification algorithms, whereas our scheme has three signature components instead of four in `LMPY-RS`. Compared to the PS-RRS scheme, our scheme requires just one additional group element $\mathbb{H}$ in the public key and only one additional exponentiation in $\mathbb{G}$ for signing and one additional pairing plus a single multiplication in $\mathbb{G}_T$ for signature verification. However, in contrast to the interactive assumption used in [26], security of our scheme requires only the SXDH assumption.

4 Sequential Aggregate Signature

In this section, we present a sequential aggregate signature (SeqAS) scheme with constant-size public key and signature and prove its unforgeability under the SXDH assumption. Like [16], our construction uses both 'randomness re-use' and 'public key sharing' techniques.

4.1 Construction

The starting point of the SeqAS is our RRS scheme. We observe that the LMPY-RS scheme does not allow signature aggregation. For that, it seems necessary to make one element of the trapdoor key of the underlying knowledge system, namely δ_0, publicly available. However, in that case, the security reduction no longer works. We resolve this issue, by first letting the setup authority choose u_1, u_2, δ_0 from $\mathbb{Z}_p$ and define $\delta_{10} = u_2 - \delta_0 u_1$. Then the j-th signer chooses v_{1j}, v_{2j} from $\mathbb{Z}_p$ and implicitly defines $\delta_j = v_{2j} - \delta_0 v_{1j}$. Note that the above changes are possible as, unlike [21], our scheme uses simulated QA-NIZK proof component.

To construct the `SeqAS` scheme, we thus extend the `RRS` structure to the multi-user setting. Recall that our `RRS` signature is of the form $g^{r(u_1+\sum_j v_{1j}m_j)}, g^r$ and $g^{r(u_2+\sum_j v_{2j}m_j)}$. As mentioned above we treat v_{1j} and v_{2j} as the j-th signer's secret key. However, to apply the 'public key sharing' technique [16], g, g^{u_1} and g^{u_2} need to be made public for aggregation and h_z, h_0, h_{10} are also needed for verification. This implies that the public parameter $AS.PP$ of the `SeqAS` scheme consists of $\{g, U_1 = g^{u_1}, U_2 = g^{u_2}, h_z, h_0 = h_z^{\delta_0}, h_{10} = h_z^{u_2} h_0^{-u_1}\}$. Then, the j-th signer implicitly sets $\delta_j = v_{2j} - \delta_0 v_{1j}$, where the corresponding secret key SK_j contains $\{v_{1j}, v_{2j}\}$ and the verification key PK_j contains $\{h_j = h_z^{\delta_j} = h_z^{v_{2j}} h_0^{-v_{1j}}\}$.

Now we explain, how to aggregate the signature using 'randomness re-use' technique. Consider a message m_1 which is signed by the first user using $SK_1 = \{v_{11}, v_{21}\}$ by computing $\sigma_1 = (A_1, B_1, C_1)$, where $A_1 = (U_1 g^{v_{11}m_1})^{t_1}$, $B_1 = g^{t_1}, C_1 = (U_2 g^{v_{21}m_1})^{t_1}$, for some t_1 randomly chosen from $\mathbb{Z}_p$. Then, given (m_1, σ_1), the second user uses $SK_2 = \{v_{12}, v_{22}\}$ to compute the aggregate signature $\sigma_2 = (A_2, B_2, C_2)$ on the message m_2, where $A_2 = (A_1 B_1^{v_{12}m_2})^{t_2}, B_2 = B_1^{t_2}, C_2 = (C_1 B_1^{v_{22}m_2})^{t_2}$, for t_2 randomly chosen from $\mathbb{Z}_p$. In the same way, we can extend the above procedure for polynomial many aggregation. We present our `SeqAS` construction in Table 3.

Correctness Note that the verification Equation 7 of the `SeqAS` scheme is same as the verification Equation 2 of our `RRS` scheme. Hence, it is sufficient to ensure the resulting aggregate signature can be written explicitly as in the `RRS` scheme. This will guarantee the correctness of the `SeqAS` scheme. Now we establish the structure of the signature returned by $\mathsf{AS.Sign}$ using mathematical induction. If $s = 0$, then $\mathsf{AS.Sign}$ sets $\sigma = (A, B, C) = (g^{u_1}, g, g^{u_2})$ and computes the signature $\sigma_1 = (A_1, B_1, C_1)$ on the message m_1, where $A_1 = (AB^{v_{11}m_1})^t =$

Table 3. SeqAS scheme in the prime-order setting.

AS.Setup(λ)

Run $\mathcal{P}(\lambda) \to (p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, g, h)$, where $g \xleftarrow{\$} \mathbb{G}, h \xleftarrow{\$} \mathbb{H}$.

Choose $u_1, u_2, \delta_0 \xleftarrow{\$} \mathbb{Z}_p, h_z \xleftarrow{\$} \mathbb{H}$, define $\delta_{10} := u_2 - \delta_0 u_1$ and compute $U_1 = g^{u_1}, U_2 = g^{u_2}, h_0 = h_z^{\delta_0}, h_{10} = h_z^{\delta_{10}}$.

Return $AS.PP := \{p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, g, h, U_1, U_2, h_z, h_0, h_{10}\}$.

AS.KeyGen($AS.PP$**)**

Choose $v_{1j}, v_{2j} \xleftarrow{\$} \mathbb{Z}_p$ and compute $h_j := h_z^{v_{2j}} h_0^{-v_{1j}}$.

Return $(SK_j := \{v_{1j}, v_{2j}\}, PK_j := \{h_j\})$.

AS.Sign($SK, \boldsymbol{m} = (m_1, \ldots, m_s), \sigma, (PK_1, \ldots, PK_s), m$**)**

Check the following,

If $s = 0$, then set $\sigma = (U_1, g, U_2)$,

If $s > 0$ and $\mathsf{AS.Ver}((PK_1, \ldots, PK_s), (m_1, \ldots, m_s), \sigma) = 0$, then it halts,

If $m = 0$ or any of $m_i = 0$, then it halts,

If for some $j \in [1, s]$ such that $PK_j = PK$, then it halts,

Suppose the algorithm did not halt, then

parse SK as $\{v_{1\tau}, v_{2\tau}\}$, PK as $\{h_\tau\}$, σ as (A, B, C).

Select $t \xleftarrow{\$} \mathbb{Z}_p$ and compute $A' = (AB^{v_{1\tau} m})^t, B' = B^t$ and $C' = (CB^{v_{2\tau} m})^t$.

Return $((m_1, \ldots, m_s, m), \sigma' = (A', B', C'))$.

AS.Ver($AS.PP, (PK_1, \ldots, PK_s), (m_1, \ldots, m_s), \sigma = (A, B, C)$**)**

Parse $PK_j = \{h_j\}$, for all $j \in [1, s]$ and check whether $m_i = 0$ or $PK_i = PK_j$, for any $i \neq j$ and $i, j \in [1, s]$. If any of the above conditions hold, then abort, otherwise check

$$B \neq 1 \text{ and } e(A, h_0) e(B, h_{10} \prod_{j=1}^{s} h_j^{m_j}) = e(C, h_z). \tag{7}$$

If the above two conditions hold, then return *accept*, otherwise return *reject*.

$g^{t(u_1 + v_{11} m_1)}$, $B_1 = B^t = g^t$ and $C_1 = (CB^{v_{12} m_1})^t = g^{t(u_2 + v_{12} m_1)}$. Let's assume that after k many aggregation, the aggregate signature $\sigma_k = (A_k, B_k, C_k)$ on the messages $(m_1, \ldots, m_k)$ under the public keys $(PK_1, \ldots, PK_k)$ can be written as $A_k = g^{r(u_1 + \sum_{j=1}^{k} v_{1j} m_j)}$, $B_k = g^r$, $C_k = g^{r(u_2 + \sum_{j=1}^{k} v_{2j} m_j)}$, for some $r \in \mathbb{Z}_p$. Now we prove that the aggregate signature $\sigma_{k+1} = (A_{k+1}, B_{k+1}, C_{k+1})$ of the messages $(m_1, \ldots, m_k, m_{k+1})$ under $(PK_1, \ldots, PK_k, PK_{k+1})$ can be written explicitly as in Table 3. Observe that from the definition of AS.Sign, we can write $A_{k+1} = (A_k B_k^{v_{1k+1} m_{k+1}})^t = g^{rt(u_1 + \sum_{j=1}^{k+1} v_{1j} m_j)}$, $B_{k+1} = B_k^t = g^{rt}$ and $C_{k+1} = (C_k B_k^{v_{2k+1} m_{k+1}})^t = g^{rt(u_2 + \sum_{j=1}^{k+1} v_{2j} m_j)}$, for $t \xleftarrow{\$} \mathbb{Z}_p$. Thus the resulting signature σ_{k+1} is distributed as similar to the RRS scheme of §3.1. Hence correctness of the SeqAS scheme follows from that of RRS scheme.

4.2 Security

We argue the unforgeability of SeqAS scheme in the certified public key setting [23], see Appendix A.3 for the formal security definition. Informally, given the public key and access to join and signing oracle queries, the adversary cannot produce a valid forgery.

We give a reduction from the security of our RRS scheme. In the certified public key model, the adversary gives both public and secret keys to the simulator. Hence simulator knows all the secret keys except the secret key of the underlying RRS scheme (for single message). The simulator responds to an aggregate signature query by first obtaining a signature of the underlying RRS from its challenger and then constructing the aggregate signature. From the aggregate signature structure, which can be viewed as a linear function of the secret exponents, the aggregation is oblivious to the order in which the messages are signed. Once the adversary returns a non-trivial forgery, the simulator extracts the forgery for the underlying RRS scheme by using the secret keys of all the other signers.

Theorem 6 *If RRS scheme is EUF-CMA secure, then SeqAS scheme is EUF-CMA secure in the certified public key setting.*

Proof. Suppose there exists a PPT adversary $\mathcal{A}$ who breaks the EUF-CMA security of SeqAS scheme in the certified public key setting, with some non-negligible probability. Then we construct a simulator $\mathcal{B}$ that breaks the EUF-CMA security of the RRS scheme as follows.

Setup: First, $\mathcal{B}$ initializes the key list KeyList as empty. Next $\mathcal{B}$ obtains the public parameter PP as $(p, \mathbb{G}, \mathbb{H}, \mathbb{G}_T, e, \tilde{g}, h)$ along with the public-key PK as $\{h_z, h_0, h_{10}, h_\tau\}$, from his challenger $\mathcal{C}$. Then $\mathcal{B}$ sets $PK^* = h_\tau$. Now $\mathcal{B}$ requests a signature query on zero message to $\mathcal{C}$. Then $\mathcal{C}$ returns $\sigma_0 = (A_0, B_0, C_0) = (\tilde{g}^{ru_1}, \tilde{g}^r, \tilde{g}^{ru_2})$, for some $r \in \mathbb{Z}_p$. Now $\mathcal{B}$ assigns $g := B_0, U_1 := A_0, U_2 := C_0$ and computes the public parameter for our SeqAS scheme as, $AS.PP := \{g, h, U_1, U_2, h_z, h_0, h_{10}\}$. Then $\mathcal{B}$ forwards $AS.PP$ and PK^* to $\mathcal{A}$.

Join Query: $\mathcal{A}$ makes the join queries by sending the key pair $SK_j = \{v_{1j}, v_{2j}\}$ and $PK_j = \{h_j\}$ to $\mathcal{B}$. $\mathcal{B}$ checks whether (SK_j, PK_j) are generated correctly by checking

$$h_j \stackrel{?}{=} h_z^{v_{2j}} h_0^{-v_{1j}}. \tag{8}$$

For correctly generated key pair (SK_j, PK_j), $\mathcal{B}$ adds it in the KeyList.

Signing Query: $\mathcal{A}$ makes a sequential aggregate signature query on the message m_i to $\mathcal{B}$ by sending an aggregate signature σ_i on the messages $(m_{i1}, \ldots m_{is_i})$ under the public keys $(PK_{i1}, \ldots, PK_{is_i})$. $\mathcal{B}$ aborts, if any of the following condition holds, (i) if $s_i > 0$ and the aggregate signature σ_i is invalid, (ii) if there exists $j \in [1, s_i]$ such that PK_{ij} does not belong to KeyList, (iii) if there exists some $j \in [1, s_i]$ such that $PK_{ij} = PK^*$. Otherwise, $\mathcal{B}$ requests a signature query on m_i to his challenger $\mathcal{C}$, which returns $\sigma = (A, B, C)$.

Note that, all the public keys PK_{ij} involved in the signing queries are certified before. Hence, $\mathcal{B}$ knows the associated key pairs (SK_{ij}, PK_{ij}), for all $j \in [1, s_i]$. Then, $\mathcal{B}$ parses SK_{ij} as $\{v_{1ij}, v_{2ij}\}$. Next, $\mathcal{B}$ chooses t uniformly at random from $\mathbb{Z}_p$ and computes the aggregate signature $\sigma'_i = (A'_i, B'_i, C'_i)$, where $A'_i = \left(AB^{\sum_{j=1}^{s_i} v_{1ij} m_{ij}}\right)^t, B'_i = B^t, C'_i = \left(CB^{\sum_{j=1}^{s_i} v_{2ij} m_{ij}}\right)^t$. It is easy to see that the signature generated above is properly distributed. From the linear structure of the signature exponents, the order of aggregation does not matter. Hence, σ'_i is a valid aggregate signature on $(m_{i1}, \ldots, m_{is_i}, m_i)$ under the public keys $(PK_{i1}, \ldots, PK_{is_i}, PK^*)$.

Output: After q many number of aggregate signature queries, $\mathcal{A}$ returns a forgery $\sigma^* = (A^*, B^*, C^*)$ on the messages $(m_1^*, \ldots, m_s^*)$ under the public keys $(PK_1, \ldots, PK_s)$. Now $\mathcal{B}$ ensures the validity of the forgery, if the following conditions are satisfied,

i. $\mathsf{AS.Ver}((PK_1, \ldots, PK_s), \sigma^*, (m_1^*, \ldots, m_s^*)) = 1$,
ii. For all $PK_j \neq PK^*$, $PK_j \in \mathtt{KeyList}$,
iii. There exists one $j^* \in [1, s]$, $PK^* = PK_{j^*}$ and $m_{j^*}^* \neq m_i$, for all $i \in [1, q]$.

Condition (i) ensures that the forgery σ^* satisfies the verification Equation 7. Whereas, the condition (ii) ensures that $\mathcal{B}$ knows all the secret key SK_j components such that Equation 8 holds, for the associated keys $PK_j \neq PK^*$. Recall that the public keys are generated independently by each of signer. Also the condition (iii) ensures that the aggregate signature forgery includes the challenge public key PK^* as part of the aggregation. Now $\mathcal{B}$ computes the rerandomizable signature forgery $\sigma_m^* = (A_m^*, B_m^*, C_m^*)$, where $A_m^* = A^* \left(B^*\right)^{-\sum_{j \neq j^*} v_{1j} m_j^*}$, $B_m^* = B^*$ and $C_m^* = C^* \left(B^*\right)^{-\sum_{j \neq j^*} v_{2j} m_j^*}$. Next we prove that the above constructed rerandomizable signature σ_m^* satisfies the verification Equation 2 as follows,

$$
\begin{aligned}
e(C_m^*, h_z) &= e(C^* \left(B^*\right)^{-\sum_{j \neq j^*} v_{2j} m_j^*}, h_z) = e(C^*, h_z) e((B^*)^{-\sum_{j \neq j^*} v_{2j} m_j^*}, h_z) \\
&= \left(e(A^*, h_0) e(B^*, h_{10} h_\tau^{m_{j^*}^*} \prod_{j \neq j^*} h_j^{m_j^*})\right) e((B^*)^{-\sum_{j \neq j^*} v_{2j} m_j^*}, h_z) \\
&= e(A^*, h_0) e(B^*, h_{10} h_\tau^{m_{j^*}^*}) e(B^*, \prod_{j \neq j^*} h_z^{(v_{2j} - \delta_0 v_{1j}) m_j^*}) e(B^*, \prod_{j \neq j^*} h_z^{-v_{2j} m_j^*}) \\
&= e(A^* (B^*)^{-\sum_{j \neq j^*} v_{1j} m_j^*}, h_0) e(B^*, h_{10} h_\tau^{m_{j^*}^*}) \\
&= e(A_m^*, h_0) e(B_m^*, h_{10} h_\tau^{m_{j^*}^*}).
\end{aligned}
$$

The first equality is obtained by using the value of C_m^* and the second equality is obtained by using the bilinearity of the pairing. Third equality is obtained from the verification Equation 7 and the fourth equality is obtained by using the definition of δ_j and bilinearity. The fifth equality is obtained by canceling the v_{2j} terms. The final equality is obtained using the values of A_m^* and B_m^*.

The condition (iii) of aggregate signature forgery ensures the non-trivial forgery with respect to the queried messages. Hence, the resulted forgery σ_m^* on the message $m_{j^*}^*$ under PK^* is clearly a valid forgery for the RRS scheme. □

4.3 Comparison

We compare our `SeqAS` scheme with the existing schemes in Table 4. We consider all the sequential aggregate signature schemes in asymmetric pairing setting [6] whose unforgeability is proved in the certified public key setting (see Appendix A.3) in the standard model. Here we use the following metrics: public key size (denoted as $|PK|$), aggregate signature size (denoted as $|AS|$), signing and verification cost and the computational assumption required to prove unforgeability of the aggregate signature scheme.

Table 4. Comparing sequential aggregate signature schemes using certified public key setting in the standard model.

Scheme	$\|PK\|$	$\|AS\|$	Signing cost	Verification cost	Assumption
LOSSW-3a	$\lambda\|\mathbb{G}\| + \lambda\|\mathbb{H}\| + 1\|\mathbb{G}_T\|$	$2\|\mathbb{G}\|$	$2\mathbb{P} + \ell M_{\mathbb{G}_T} + (\ell-1)\lambda M_{\mathbb{H}} + 6E_{\mathbb{G}} + (2\ell\lambda+6)M_{\mathbb{G}}$	$2\mathbb{P} + \ell M_{\mathbb{G}_T} + \ell\lambda M_{\mathbb{H}}$	CDH
SAS1	$2\|\mathbb{G}\| + 8\|\mathbb{H}\| + 1\|\mathbb{G}_T\|$	$8\|\mathbb{G}\|$	$8\mathbb{P} + (4\ell+14)E_{\mathbb{H}} + 10E_{\mathbb{G}} + 1E_{\mathbb{G}_T}$	$8\mathbb{P} + 1E_{\mathbb{G}_T} + (4\ell+14)E_{\mathbb{H}}$	SXDH, DBDH LW2
SAS2	$6\|\mathbb{G}\| + 6\|\mathbb{H}\| + 1\|\mathbb{G}_T\|$	$6\|\mathbb{G}\|$	$6\mathbb{P} + (3\ell+6)E_{\mathbb{H}} + (3\ell+18)E_{\mathbb{G}} + 1E_{\mathbb{G}_T}$	$6\mathbb{P} + 1E_{\mathbb{G}_T} + (3\ell+6)E_{\mathbb{H}}$	LW1, DBDH LW2
LLY-SeqAS	$1\|\mathbb{H}\|$	$3\|\mathbb{G}\|$	$5\mathbb{P} + M_{\mathbb{G}_T} + \ell E_{\mathbb{H}} + 2\ell M_{\mathbb{H}} + 5E_{\mathbb{G}} + 2M_{\mathbb{G}}$	$5\mathbb{P} + M_{\mathbb{G}_T} + \ell E_{\mathbb{H}} + 2\ell M_{\mathbb{H}}$	LRSW
PS-SeqAS	$1\|\mathbb{H}\|$	$2\|\mathbb{G}\|$	$2\mathbb{P} + \ell(E_{\mathbb{H}} + M_{\mathbb{H}}) + 3E_{\mathbb{G}} + 1M_{\mathbb{G}}$	$2\mathbb{P} + \ell(E_{\mathbb{H}} + M_{\mathbb{H}})$	PS
`SeqAS`	$1\|\mathbb{H}\|$	$3\|\mathbb{G}\|$	$3\mathbb{P} + M_{\mathbb{G}_T} + \ell(E_{\mathbb{H}} + M_{\mathbb{H}}) + 5E_{\mathbb{G}} + 2M_{\mathbb{G}}$	$3\mathbb{P} + M_{\mathbb{G}_T} + \ell(E_{\mathbb{H}} + M_{\mathbb{H}})$	SXDH

For any group $X \in \{\mathbb{G}, \mathbb{H}, \mathbb{G}_T\}$, we denote E_X, M_X and $|X|$ be the exponentiation, multiplication in X and bit size of X and $\mathbb{P}$ denotes asymmetric pairing computation time. λ denotes the security parameter and ℓ denotes the number of signature aggregated so far. PS and LRSW denote the interactive assumptions in [26] and [25].

Chatterjee et al. [5] presented aggregate signature variants of [23] denoted as LOSSW-3a, whose security is proved under the CDH assumption. However, the size of the public key is some multiple of the security parameter λ, where λ takes 256 for the 128 bit-level security. This results in a sequential aggregate signature with a large public key size.

In 2012, Lee et al. [15, Section 3.5] extended the idea from [28] and presented a sequential aggregate signature scheme based on the Camenisch-Lysyanskaya (CL) signature in the asymmetric pairing setting. The resulted scheme is denoted as LLY-SeqAS and it has a constant size public key. The security of the LLY-SeqAS scheme is based on the security of the CL-signature scheme, which is proved under an interactive assumption. In 2015, Lee et al. [19] presented two SeqAS schemes, namely SAS1 and SAS2 schemes. The security of SAS1 is proved under SXDH, DBDH and LW2 assumptions and SAS2 is proved under DBDH, LW1 and LW2 assumptions described in [20]. Note that, both LW1 and LW2 assumptions are non-standard static assumptions whose hardness is established in the generic group model, which provides only the lower bound [13, 29]. They have used the dual system encryption technique to prove the security of their schemes. However, their public key size (resp. aggregate signature size) increases by a factor of 9 (resp. 2) with respect to the LLY-SeqAS construction.

In 2016, [26] presented a sequential aggregate signature (denoted as PS-SeqAS) scheme based on their rerandomizable signature construction. One can see that PS-SeqAS is the most efficient scheme among all the SeqAS schemes presented in Table 4. However, the security of the PS-SeqAS scheme is proved under an interactive assumption. The performance of our `SeqAS` scheme is very close to PS-SeqAS even though we argue security under the SXDH assumption. In particular, public key size remains the same in both schemes, whereas signature size increases by one group element in our scheme as compared to the PS-SeqAS scheme. Also, we require one additional pairing and one target group multiplication to verify the signature.

5 Concluding Remark

We proposed the first construction of a sequential aggregate signature scheme with constant-size public key in the standard model based on the SXDH assumption in the prime order bilinear pairing setting. This is achieved by suitably modifying a randomizable signature scheme from [21]. The performance of both the rerandomizable signature scheme and sequential aggregate signature scheme comes quite close to prior proposals where security is based on some interactive assumption.

References

1. Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. *J. Cryptology*, 29(2):363–421, 2016.
2. Patrik Bichsel, Jan Camenisch, Gregory Neven, Nigel P. Smart, and Bogdan Warinschi. Get shorty via group signatures without encryption. In Juan A. Garay and Roberto De Prisco, editors, *SCN*, volume 6280, pages 381–398, Springer, 2010.
3. Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *EUROCRYPT*, volume 2656, pages 416–432, Springer, 2003.
4. Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152, pages 56–72, Springer, 2004.
5. Sanjit Chatterjee, Darrel Hankerson, Edward Knapp, and Alfred Menezes. Comparing two pairing-based aggregate signature schemes. *Des. Codes Cryptography*, 55(2-3):141–167, 2010.
6. Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. Pairings for cryptographers. *Discrete Applied Mathematics*, 156(16):3113–3121, 2008.
7. Michael Gerbush, Allison B. Lewko, Adam O'Neill, and Brent Waters. Dual form signatures: An approach for proving security from static assumptions. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT*, volume 7658, pages 25–42, Springer, 2012.
8. Essam Ghadafi. Short structure-preserving signatures. In Sako [27], pages 305–321.
9. Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A "paradoxical" solution to the signature problem (extended abstract). In *FOCS*, pages 441–448. IEEE Computer Society, 1984.

10. Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.
11. Jens Groth. Homomorphic Trapdoor Commitments to Group Elements. *IACR Cryptology ePrint Archive*, 2009:007, 2009.
12. Charanjit S. Jutla and Arnab Roy. Switching lemma for bilinear tests and constant-size NIZK proofs for linear subspaces. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO*, volume 8617, pages 295–312, Springer, 2014.
13. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965, pages 146–162, Springer, 2008
14. Eike Kiltz and Hoeteck Wee. Quasi-adaptive NIZK for linear subspaces revisited. In Elisabeth Oswald and Marc Fischlin, editors, *EUROCRYPT*, volume 9057, pages 101–128, Springer, 2015.
15. Kwangsu Lee, Dong Hoon Lee, and Moti Yung. Aggregating cl-signatures revisited: Extended functionality and better efficiency. *IACR Cryptology ePrint Archive*, 2012:562, 2012.
16. Kwangsu Lee, Dong Hoon Lee, and Moti Yung. Aggregating cl-signatures revisited: Extended functionality and better efficiency. In Ahmad-Reza Sadeghi, editor, *FC*, volume 7859, pages 171–188, Springer, 2013.
17. Kwangsu Lee, Dong Hoon Lee, and Moti Yung. Sequential aggregate signatures made shorter. In Michael J. Jacobson Jr., Michael E. Locasto, Payman Mohassel, and Reihaneh Safavi-Naini, editors, *ACNS*, volume 7954, pages 202–217, Springer, 2013.
18. Kwangsu Lee, Dong Hoon Lee, and Moti Yung. Sequential aggregate signatures with short public keys: Design, analysis and implementation studies. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC*, volume 7778, pages 423–442, Springer, 2013.
19. Kwangsu Lee, Dong Hoon Lee, and Moti Yung. Sequential aggregate signatures with short public keys without random oracles. *Theor. Comput. Sci.*, 579:100–125, 2015.
20. Allison B. Lewko and Brent Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In Daniele Micciancio, editor, *TCC*, volume 5978, pages 455–479, Springer, 2010.
21. Benoît Libert, Fabrice Mouhartem, Thomas Peters, and Moti Yung. Practical "signatures with efficient protocols" from simple assumptions. In Xiaofeng Chen, XiaoFeng Wang, and Xinyi Huang, editors, *AsiaCCS*, pages 511–522. ACM, 2016.
22. Benoît Libert, Thomas Peters, and Moti Yung. Short group signatures via structure-preserving signatures: Standard model security from simple assumptions. In Rosario Gennaro and Matthew Robshaw, editors, *CRYPTO*, volume 9216, pages 296–316, Springer, 2015.
23. Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. Sequential aggregate signatures and multisignatures without random oracles. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004, pages 465–485, Springer, 2006.
24. Anna Lysyanskaya, Silvio Micali, Leonid Reyzin, and Hovav Shacham. Sequential aggregate signatures from trapdoor permutations. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027, pages 74–90, Springer, 2004.
25. Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In Howard M. Heys and Carlisle M. Adams, editors, *SAC*, volume 1758, pages 184–199, Springer, 1999.

26. David Pointcheval and Olivier Sanders. Short randomizable signatures. In Sako [27], pages 111–126.
27. Kazue Sako, editor. *CT-RSA*, volume 9610, Springer, 2016.
28. Dominique Schröder. How to aggregate the CL signature scheme. In Vijay Atluri and Claudia Díaz, editors, *ESORICS*, volume 6879, pages 298–314, Springer, 2011.
29. Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *EUROCRYPT*, volume 1233, pages 256–266, Springer, 1997.
30. Brent Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494, pages 114–127, Springer, 2005.

A Definitions

A.1 Digital Signature

The formal definition of digital signature scheme and its security model were introduced in [9,10]. We recall the definition of digital signature scheme from [8], which consists of four PPT algorithms.

Setup(1^λ) Given the security parameter λ, it returns a public parameter PP. Here PP includes the description of the message space $\mathcal{M}$ and the signature space Σ.

KeyGen(PP) Given the public parameter PP, it returns a key pair (PK, SK). We assume that SK contains PK and PK contains PP.

Sign(SK, m) Given the message $m \in \mathcal{M}$ and SK, it returns a signature σ on the message m.

Ver(PK, m, σ) Given the message and signature pair along with the public key PK, it returns 1 only if σ is a valid signature on the message m under PK.

The digital signature scheme is *correct*, if for all security parameter λ, all $PP \leftarrow \mathsf{Setup}(1^\lambda)$, all $(PK, SK) \leftarrow \mathsf{KeyGen}(PP)$, all messages $m \in \mathcal{M}$ and $\sigma \leftarrow \mathsf{Sign}(SK, m)$, it holds that $\mathsf{Ver}(PK, m, \sigma)$=1.

The security of digital signature scheme is captured using the existential unforgeability under chosen message attack (EUF-CMA) model [10] which is defined using the following experiment between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup $\mathcal{C}$ runs the Setup and KeyGen to obtain (PK, SK). $\mathcal{A}$ is given with PK.

Queries $\mathcal{A}$ adaptively requests the signature on the message m_i, for $i \in [1, q]$. $\mathcal{C}$ answers each query by computing $\sigma_i = \mathsf{Sign}(SK, m_i)$.

Output Finally, $\mathcal{A}$ returns a message and signature pair (m^*, σ^*) and wins if $\mathsf{Ver}(PK, m^*, \sigma^*)$=1 with $m^* \neq m_i$, for all $i \in [1, q]$.

The advantage of $\mathcal{A}$ (denoted as $Adv_{\mathcal{A}}^{UF}$) is defined to be the probability that $\mathcal{A}$ wins in the above game. A signature scheme is said to be (t, q, ϵ)-secure against existential unforgeability under chosen message attack ((t, q, ϵ)-EUF-CMA secure), if for any t-time adversary $\mathcal{A}$ that makes at most q many signing oracle queries, $Adv_{\mathcal{A}}^{UF} \leq \epsilon$, where t and q are the polynomial functions of λ and ϵ is a negligible function in λ.

A.2 Rerandomizable Signature

We recall the definition of rerandomizable signature scheme from [8]. A rerandomizable signature scheme consists of five PPT algorithms (Setup, KeyGen, Sign, Ver, Rand), in which the first four algorithms are defined as in the digital signature scheme, whereas the Rand algorithm is defined as follows.

Rand(PK, m, σ) Given a PK along with a valid message and signature pair, it returns a new signature σ' on the same message m under PK.

A rerandomizable signature scheme is said to be secure if the scheme satisfies randomizability and unforgeability. Unforgeability is captured using EUF-CMA model as similar to the digital signature scheme described in Appendix A.1. Informally, randomizability ensures that given both PK and SK, $\mathcal{A}$ cannot decide whether he is given a signature returned by Sign or by Rand algorithm. Formally, it is defined in [8] using the following experiment between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup $\mathcal{C}$ runs the Setup and KeyGen to obtain (PK, SK). $\mathcal{A}$ is given with PK and SK.

Challenge Phase $\mathcal{A}$ returns a message and signature pair (m^*, σ^*) to $\mathcal{C}$ who chooses a random bit $b \in \{0, 1\}$. If $b = 0$, then $\mathcal{C}$ computes $\sigma_0 = \mathsf{Sign}(\mathsf{SK}, \mathsf{m}^*)$, else computes $\sigma_1 = \mathsf{Rand}(\mathsf{PK}, \mathsf{m}^*, \sigma^*)$. Then, $\mathcal{C}$ sends σ_b to $\mathcal{A}$.

Output Finally $\mathcal{A}$ returns a bit b' and wins if $\mathsf{Ver}(PK, m^*, \sigma^*)$=1 and $b' = b$.

The advantage of $\mathcal{A}$ is defined as $Adv_{\mathcal{A}}^{Rand} = |Pr[b' = b] - 1/2|$. A randomizable signature scheme is said to be (t, ϵ)-secure against randomizability, if for any t-time adversary $\mathcal{A}$, $Adv_{\mathcal{A}}^{Rand} \leq \epsilon$, where t is a polynomial function of λ and ϵ is a negligible function in λ. If $\epsilon = 0$, then we say that the scheme satisfies *perfect randomizability*.

A.3 Sequential Aggregate Signature

The notion of sequential aggregate signature scheme was introduced by Lysyanskaya et al. [24]. Here we recall the definition of sequential aggregate signature scheme from [23]. A *sequential aggregate signature* scheme is defined using four PPT algorithms which are described below.

AS.Setup(1^λ) Given the security parameter λ, it returns a public parameter PP.

AS.KeyGen(PP) Given the public parameter PP, it returns a key pair (PK, SK). We assume that SK contains PK and PK contains PP.

AS.Sign$(SK, (m_1, \ldots, m_s), \sigma, (PK_1, \ldots, PK_s), m)$ Given an aggregate signature σ on the messages $(m_1, \ldots, m_s)$ under the public keys $(PK_1, \ldots, PK_s)$, a message m and a secret key SK such that the associated public key $PK \notin \{PK_j\}_{j=1}^s$, it returns a new aggregate signature σ' on the messages $(m_1, \ldots, m_s, m)$ under the public keys $(PK_1, \ldots, PK_s, PK)$.

AS.Ver$((PK_1, \ldots, PK_s), (m_1, \ldots, m_s), \sigma)$ Given the public keys $(PK_1, \ldots, PK_s)$, messages $(m_1, \ldots, m_s)$ along with the signature σ, it returns 1 only if σ is a valid aggregate signature on $(m_1, \ldots, m_s)$ under $(PK_1, \ldots, PK_s)$.

The sequential aggregate signature scheme is *correct*, if for all security parameter λ, all $PP \leftarrow \mathsf{AS.Setup}(1^\lambda)$, all $(PK_j, SK_j) \leftarrow \mathsf{AS.KeyGen}(PP)$ for all $j \in [1, s]$, all messages $(m_1, \ldots, m_{s-1}) \in \mathcal{M}$, all aggregated-so-far signature σ on the messages $(m_1, \ldots, m_{s-1})$ under $(PK_1, \ldots, PK_{s-1})$ and $\sigma' \leftarrow \mathsf{AS.Sign}(SK_s, (m_1, \ldots, m_{s-1}), \sigma, (PK_1, \ldots, PK_{s-1}), m)$, it holds that $\mathsf{AS.Ver}($ $(PK_1, \ldots, PK_s,), (m_1, \ldots, m_s), \sigma')$=1.

We consider the security of sequential aggregate signature scheme in the *certified public key setting* [23]. Informally, given a public key and polynomial many aggregate signing oracle access along with polynomial many join oracle access, it is hard for an adversary to produce an aggregate signature forgery on a new message, which is not queried to the aggregate signing oracle. While making a join oracle query, adversary requests a certification of a public key by sending public and secret key pair to the challenger. The challenger returns a certification of the requested public key, if the given key pair is valid. Formally, security is defined using the following experiment between the challenge $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup: $\mathcal{C}$ initializes a key list $\mathsf{KeyList}$ as empty. Next, it runs the $\mathsf{AS.Setup}$ to get PP and the $\mathsf{AS.KeyGen}$ to get (PK^*, SK^*). Then $\mathcal{C}$ sends PK^* to $\mathcal{A}$.

Join Query: $\mathcal{A}$ adaptively requests to authenticate the public key PK_i by sending (PK_i, SK_i) to $\mathcal{C}$, for $i \in [1, q_J]$. If the key pair is valid, then $\mathcal{C}$ authenticates the public key PK_i and adds to $\mathsf{KeyList}$.

Signing Query: $\mathcal{A}$ adaptively requests a sequential aggregate signature under the challenge public key PK^* on the message m_i, for $i \in [1, q]$. In addition, it supplies an aggregated-so-far signature σ_i on the messages $(m_{i,1}, \ldots, m_{i,s_i})$ under the public keys $(PK_{i,1}, \ldots, PK_{i,s_i})$. $\mathcal{C}$ checks (i) σ_i verifies, (ii) $PK_{i,j}$ belongs to $\mathsf{KeyList}$, for all $j \in [1, s_i]$ and (iii) $PK^* \neq PK_{i,j}$, for all $j \in [1, s_i]$. Once all the above conditions satisfy, then $\mathcal{C}$ answers by computing $\sigma'_i = \mathsf{AS.Sign}(SK^*, (m_{i,1}, \ldots, m_{i,s_i}), \sigma_i, (PK_{i,1}, \ldots, PK_{i,s_i}), m_i)$.

Output: $\mathcal{A}$ outputs an aggregate signature σ^* on the messages $(m_1^*, \ldots, m_s^*)$ under the public keys $(PK_1, \ldots, PK_s)$ and wins the game if the following conditions are all satisfied:

1. $\mathsf{AS.Ver}((PK_1, \ldots, PK_s), (m_1^*, \ldots, m_s^*), \sigma^*)$=1,
2. For all $PK_j \neq PK^*$, $PK_j \in \mathsf{KeyList}$,
3. For some $j^* \in [1, s]$, $PK^* = PK_{j^*}$ and $m_{j^*}^* \neq m_i$, for all $i \in [1, q]$.

The advantage of $\mathcal{A}$ (denoted as $Adv_{\mathcal{A}}^{AS.UF}$) is defined to be the probability that $\mathcal{A}$ wins in the above game. A sequential aggregate signature scheme is said to be (t, q, q_J, ϵ)-secure against existential unforgeability under chosen message attack ((t, q, q_J, ϵ)-EUF-CMA secure), if for any t-time adversary $\mathcal{A}$ that makes at most q many aggregate signing oracle queries and at most q_J many join oracle queries, $Adv_{\mathcal{A}}^{AS.UF} \leq \epsilon$, where t, q and q_J are the polynomial functions of λ and ϵ is a negligible function in λ.