Meta Ring Signature

Hiroyuki OKAZAKI* Ryuichi SAKAI† Masao KASAHARA‡

1 Introduction

In this paper, we propose a new concept "Meta Ring Signature". In Digital Signature, first a private key is picked. Next, the public key is obtained in terms of the private key. Then we can sign on to a message with the private key. In other words, a signature is calculated from the private key and a message. Suppose that a signature works as a public key, we may realize a new digital signature "Meta Signature". In Meta Signature, first a user generates a signature referred to as base signature. Next the signer of a base signature can generate another signature referred to as meta signature, to sign on to another message. A public verifier verifies the soundness of a base signature by using the signer's public key. Then the public verifier can verify the soundness of a meta signature by using the base signature without any public key. Here, the soundness of a base signature means that the base signature is generated by using the private key corresponding to the public key. Thus, a base signature is an ordinary digital signature. On the other hand, the soundness of a meta signature means that the meta signature is generated by using the same private key as generating the base signature. Naturally on practical use, Meta Signature is worthless if the base signature is an ordinary digital signature. Because we must use the public key of base signature scheme to verify the base signature. However, some well known signature schemes have the property of signer's anonymity, such as Group Signature[1], Ring Signature[2], etc. Generally in these signature schemes, any member of a group can generate a signature such that a public verifier can verify the fact that the signature is generated by someone belongs to the group or not, though the verifier cannot identify the signer. If we use one

*h_okazaki@m.ieice.org
†Osaka Electro–Communication University,sakai@isc.osakac.ac.jp
‡Osaka Gakuin University,kasahara@utc.osaka-gu.ac.jp

of such signature schemes as the base signature scheme, the signer of a base signature can generate meta signature to sign on to another message. Then a public verifier can judge that the base signature and the meta signature have generated by the same user or not without loosing the signers anonymity.

However, signature schemes with the property, "Linkability", can achieve a similar function. For example, in the schemes such as Linkable SAG signature [1][3], Linkable Group signature[4, 5], public verifier canjudge whether two different signatures are generated by the same user or not.

Suppose that we construct "Meta Ring signature on Ring signature", we refer it as "Meta Ring Signature" for short. Given the list of k base signature, a signer can generate a Meta Ring signature, if there is a base signature generated by the signer in the list. It is infeasible to identify which base signature is generated by the signer of Meta signature.

In this paper, we present a concept of Meta Ring Signature. We then show the example of the way of constructing of Meta Ring Signature from Linkable SAG Signature and its application in Appendix A.

2 Concept of Meta Ring Signature

The Meta Signature is composed of two layers of signature schemes. They are a base signature layer and a Meta signature layer. For short, we refer them as BSL and MSL, respectively. We treat signatures of BSL as the public keys of MSL. Given a list of BSL signatures L_S, a user can generate an MSL signature if and only if a BSL signature which he or she generated is included in L_S. Obviously, it is infeasible to identify the signer of a given BSL signature. In MSL, it should be infeasible to identify which BSL signature in L_S generated by the signer of a given MSL signature.

In this paper we refer to the Meta signature scheme whose MSL is A and BSL is B, as "Meta A on B" .

2.1 Base Signature Layer

We see that some signature schemes can be used as BSL. In this paper, we show the case where the BSL is Ring Signature. First, we review Ring Signature as BSL.

The Ring Signature scheme as BSL is composed of three algorithms, i.e., key generation algorithm $Gen_B()$, signing algorithm $Sig_B()$, and verification

[1]SAG Signature is A kind of Ring signature. It has the properties as same as Ring Signature has, although the construction of SAG Signature text doesn't form a ring structure.

algorithm $Ver_B()$. For $i = 1, \ldots, n$, each user $\mathcal{U}_i$ generates his or her private key x_i, and public key y_i by using $Gen_B()$. Let $L_B = \{y_1, \ldots, y_n\}$ be the list of n members' public keys.

Let an integer π be $1 \leq \pi \leq n$, a user $\mathcal{U}_\pi$ can generate a signature $\sigma(m, L_B) = Sig_B(m, x_\pi, L_B)$ to sign on to a given message m. A public verifier verifies a given signature text $\sigma_B(m, L_B)$ with $Ver_B(\sigma(m, L_B), m, L_B)$. Ver_B outputs "accept" if the signature $\sigma(m, L_B)$ is generated with the private key x_i corresponding to a public key y_i in L_B. Otherwise, Ver_B outputs "reject". Generally, it is infeasible for public verifier to identify which private key was actually used in generating a given signature $\sigma(m, L_B)$. In other words, it is infeasible to identify the signer of a given signature. However, in some signature schemes with signer's anonymity, such as Group Signature, the signer can be identified by the special authority.

2.2 Meta Signature Layer

2.2.1 BSL Signature as MSL Public Key

In MSL, we treat BSL as the key generation algorithm. Let

$$L_S = \{\sigma^1(m_1, L_B), \ldots, \sigma^j(m_j, L_B), \ldots, \sigma^k(m_k, L_B)\}$$

be the given list of k signatures of BSL. Note that all signatures in L_S are generated with the same public key list, L_B. In MSL, L_S works as the list of public keys.

2.2.2 Signing

Let $Sig_M()$ be the signing algorithm of MSL. Let $\mathcal{U}_\Sigma$ be the signer of BSL signature $\sigma^\Sigma(m_\Sigma, L_B)$. If $\sigma^\Sigma(m_\Sigma, L_B)$ is included in L_S, such that

$$L_S = \{\sigma^1(m_1, L_B), \ldots, \sigma^\Sigma(m_\Sigma, L_B), \ldots, \sigma^k(m_k, L_B)\},$$

$\mathcal{U}_\Sigma$ can generate the MSL signature $\sigma_M(m', L_S) = Sig_M(m', x_\Sigma, L_S)$ to sign on to a given message m'. At this time, $\mathcal{U}_\Sigma$ uses his or her private key of BSL, x_Σ, as the private key of MSL, and uses BSL signature $\sigma^\Sigma(m_\Sigma, L_B)$ as the public key of MSL corresponding to the private key x_Σ.

Note that, depending on the MSL and/or BSL, we may have to use the same random integer r to generate MSL signature and BSL signature as follows:

$$\sigma(m, L_B) = Sig_B(m, x_\pi, L_B, r), \sigma_M(m', L_S) = Sig_M(m', x_\Sigma, L_S, r).$$

2.2.3 Verification

Let $Ver_M()$ be the verification algorithm of MSL. A public verifier verifies a given MSL signature $\sigma_M(m', L_S)$ with $Ver_M(\sigma_M(m', L_S), m', L_S)$. $Ver_M()$ outputs "accept" if the MSL signature $\sigma_M(m', L_S)$ is generated with the private key x_j corresponding to a BSL signature $\sigma^j(m_j, L_B)$ in L_S. Otherwise, $Ver_M()$ outputs "reject".

2.3 Security Requirements

The security requirements for a BSL depends on the signature scheme used. The security requirements for an actual MSL may depend on the BSL. We propose here the minimum requirements that the MSL should satisfy.

Meta Unforgeability Given the BSL signatures with no corresponding private keys, an adversary cannot forge an MSL signature for any message.

Base Signature Anonymity It is infeasible to identify which BSL signature corresponds to the private key used to generate a given MSL signature.

In Appendix A, we present Meta LSAGS on LSAGS as an example of Meta Ring Signature.

3 Conclusion

We have presented the concept of Meta Signature. Particularly we discussed on Meta Ring Signature.

We shall discuss other kinds of Meta Signatures, Meta Group Signature on Ring Signature, Meta Ring Signature on Group Signature, and so on. In addition, we shall discuss Meta public cryptography, the scheme using signature as public key for encoding. We think that the concept of the Meta Signature can be one of the efficient way to enhance various signature schemes systematically.

References

[1] D. Chaum , E. van Heijst , "Group Signature" , Advances in Cryptology - EUROCRYPTO '91, pp.257–265,Springer-Verlag, 1991.

[2] R. Rivest, A. Shamir, and Y. Tauman , “How to leak a secret” , Advances in Cryptology - EUROCRYPTO '91, pp.257–265,Springer-Verlag ASIACRYPTO 2001, pp.552–565, Springer-Verlag, 2001. LNCS Vol. 2248.

[3] J. K. Liu, V. K. Wei, and D. S. Wong , “Linkable Spontaneous Anonymous Group Signature for Ad Hoc Groups” , ACISP 2004, pp.325–335, Springer-Verlag Berlin, 2004. LNCS Vol. 3108.

[4] T. Nakanishi,T. Fujiwara,H. Watanabe, ”A Linkable Group Signature and Its Application to Secret Voting”, Trans. IPSJ, Vol.40,No.7, pp.3085-3096, 1999.

[5] H. Okazaki, R. Sakai, K.Shibayama, M. Kasahara, “A Linkable Group Signature Scheme Based on Weil Pairing” , IEICE Trans. Vol.J87-A No.4, pp.563–568, 2004. (in Japanese)

[6] H. Okazaki, R. Sakai, M. Kasahara, “Meta Ring Signature” , IEICE Technical Report, ISEC2005–36, pp.201–204, 2005. (Japanese version of this manuscript)

A Example of Meta Ring Signature

Recently, Liu, Wei, and Wong presented an interesting signature scheme, Linkable Spontaneous Anonymous Group Signature[3], LSAGS for short. We show an example of the construction of Meta Ring signature, Meta LSAGS on LSAGS.

A.1 Linkable Spontaneous Anonymous Group Signature

We introduce the Liu et al.'s LSAGS briefly. Refer to [3] for details. The LSAGS satisfies the following security requirements.

Unforgeability Given the public keys of all group members with no corresponding private keys, an adversary cannot forge a signature for any message.

Signer Anonymity It is infeasible to identify which private key was actually used in generating a given LSAGS signature to find out which member has generated a signature.

Linkability Two LSAGS signature with the same public key list L are linked if they are generated with the same private key.

Let $G = \langle g \rangle$ be a group of prime order q. Let $H_1 : \{0,1\}^* \to \mathbb{Z}_q$ and $H_2 : \{0,1\}^* \to G$ be hash functions. For $i = 1, \ldots, n$, each user $\mathcal{U}_i$ picks his or her private key x_i and then calculates the public key $y_i = g^{x_i}$. Let $L = \{y_1, \ldots, y_n\}$ be the list of n members' public keys.

A.1.1 Signing

To sign on to a given message $m \in \{0,1\}^*$, a user $\mathcal{U}_\pi$ calculates the signature $\sigma(m, L) = LsagSig(m, x_\pi, L)$. $LsagSig()$ is the signing procedure given as follows:

Step 1 : Calculate $h = H_2(L)$ and $\tilde{y} = h^{x_\pi}$.

Step 2 : Pick $u \in \mathbb{Z}_q$ randomly, and calculate

$$c_{\pi+1} = H_1\left(L, \tilde{y}, m, g^u, h^u\right). \tag{1}$$

Step 3 : For $i = \pi+1, \ldots, n, 1, \ldots, \pi-1$, pick $s_i \in \mathbb{Z}_q$ randomly, and calculate

$$c_{i+1} = H_1\left(L, \tilde{y}, m, g^{s_i} y_i^{c_i}, h^{s_i} \tilde{y}^{c_i}\right). \tag{2}$$

Step 4 : Calculate $s_\pi = u - x_\pi c_\pi \mod q$, and then output the signature

$$\sigma(m, L) = \{c_1, s_1, \ldots, s_n, \tilde{y}\}. \tag{3}$$

A.1.2 Verification

A public verifier verifies a given signature $\sigma(m, L)$ with the verification procedure $LsagVer(\sigma(m, L), m, L)$ as follows:

Step 1 : Calculate $h = H_2(L)$.

Step 2 : For $i = 1, \ldots, n$, calculate $z_i' = g^{s_i} y_i^{c_i}$, $z_i'' = h^{s_i} \tilde{y}^{c_i}$, and

$$c_{i+1} = H_1\left(L, \tilde{y}, m, z_i', z_i''\right). \tag{4}$$

Step 3 : Accept $\sigma(m, L)$ if $c_1 = H_1\left(L, \tilde{y}, m, z_n', z_n''\right)$, otherwise reject it.

A.1.3 Linking

Let $\sigma'(m', L) = \{c'_1, s'_1, \ldots, s'_n, \tilde{y}'\}$ and $\sigma''(m'', L) = \{c'_1, s''_1, \ldots, s''_n, \tilde{y}''\}$ be valid signatures signed with the same public key list L. If $\tilde{y}' = \tilde{y}''$, $\sigma'_L(m')$ and $\sigma''_L(m'')$ are generated by the same signer. Otherwise, the two signatures are generated by two different signers.

A.2 Meta LSAGS on LSAGS

We show here the way of the construction of Meta LSAGS on LSAGS. We use LSAGS both as the BSL and the MSL.

Let $L_S = \{\sigma^1(m_1, L), \ldots, \sigma^j(m_j, L), \ldots, \sigma^k(m_k, L)\}$ be the list of k BSL signature such that the BSL signature signed on to the message m_j is $\sigma^j(m_j, L) = \left\{c_{(j,1)}, s_{(j,1)}, \ldots, s_{(j,n)}, \tilde{y}_j\right\}$. All BSL signatures in the list L_S are signed with the same public key list L. Let $L'_S = \{\tilde{y}_1, \ldots, \tilde{y}_j, \ldots, \tilde{y}_k\}$ be the list of $\tilde{y}_j$, an element taken out of the BSL signature $\sigma^j(m_j, L)$in L_S. In MSL, L'_S work as the list of public keys.

A.2.1 Signing

To sign on to a given message $m_M \in \{0,1\}^*$, a user $\mathcal{U}_\Sigma$, the signer of $\sigma^\Sigma(m_\Sigma, L)$ in L_S, calculates the MSL signature $\sigma_M(m_M, L'_S) = LsagSig(m_M, x_\Sigma, L'_S)$.

A.2.2 Verification

A public verifier verifies a given MSL signature $\sigma_M(m_M, L'_S)$ with the verification procedure of the LSAGS such as $LsagVer(\sigma_M(m_M, L'_S), m_M, L'_S)$.

A.3 Security

Meta Unforgeability Meta LSAGS on LSAGS satisfies Meta Unforgeability. Because the signature scheme of MSL is the LSAGS. If there is an algorithm against Meta Unforgeability, an adversary can forge the signature of LSAGS though LSAGS satisfies Unforgeability.

Base Signature Anonymity Meta LSAGS on LSAGS satisfies Base Signature Anonymity. Because the signature scheme of MSL is the LSAGS. If there is an algorithm against base signature anonymity, an adversary can identify the signer of a given signature of LSAGS though LSAGS satisfies signer anonymity.

A.4 Application of Meta LSAGS, Linkable Group Signature without Authority

We suggest the outline of an application of Meta LSAGS on LSAGS , though we have not yet discussed its security in a formal manner. We shall soon discuss this application formally in near future. We achieve a linkable group signature scheme by adding a signer identification procedure to the LSAGS. In group signature, it is possible for Special Authority , such as group manager, to identify the signer of a given signature. The LSAGS has no such function. We give the functions of group signature, key generation , signing, and verification by using the LSAGS, the BSL of Meta LSAGS on LSAGS. We then give the signer identification function by using the MSL. Recall that the LSAGS has Linkability. Let $\sigma(m, L)$ and $\sigma'(m', L)$ be two signatures of the LSAGS. Let U_i be the signer of $\sigma(m, L)$. U_i will be able to prove that U_i is not the signer of $\sigma'(m', L)$, by proving that he or she generated $\sigma(m, L)$ with some methods, for example signing on to $\sigma(m, L)$ by a usual digital signature scheme. If all users except the signer of $\sigma'(m', L)$ do so, the signer can be specified.

However, even if the private keys of users are kept secret, the Signer Anonymity is lost in such a method. Then, the user U_i proves that he or she is not the signer of $\sigma'(m', L)$ in the following method. First, U_i gathers the signatures were not generated by the signer of $\sigma'(m', L)$, and then makes them into the list L'_S. Next, U_i generates MSL signature $\sigma^i_M(m_i, L'_S)$. Finally, U_i proves that he or she generated $\sigma^i_M(m_i, L'_S)$ by some methods, for instance, using U_i's normal digital signature signed on $\sigma^i_M(m_i, L'_S)$ etc. A public verifier verify that $\sigma^i_M(m_i, L'_S)$ is generated by U_i and linking all BSL signatures in L_S and $\sigma'(m', L)$. Thus, U_i can prove to the public verifier that U_i is not the signer of $\sigma'(m', L)$. When all innocent users finish proving, the signer of $\sigma'(m', L)$ is identified. It should be noted that there is no authority in this method.