Multiplicative Forward-Secure Threshold Signature Scheme

Sherman S. M. Chow[1,*], H. W. Go[1,*], Lucas C. K. Hui[2], and S. M. Yiu[2]
(Corresponding author: Sherman S.M. Chow)

Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, USA[1]
Department of Computer Science, The University of Hong Kong, Hong Kong[2]
(Email: schow@cs.nyu.edu, {hwgo, hui, smyiu}@cs.hku.hk)

(Received Sept. 13, 2006; revised Nov. 8, 2006; and accepted June 12, 2007)

Abstract

The devastating consequence of secret key exposure in digital signature is that any signature can be forged and cannot be trusted. To mitigate the damage of secret key exposure, *forward-secure signature schemes* and *threshold signature schemes* are devised. In this paper, we propose a robust forward-secure threshold signature scheme with the applicability to mobile ad-hoc network in mind. Our main objective is to reduce interaction among the set of signers and to reduce the dependency on broadcast as well as private point-to-point connections. We achieve this by avoiding the regular polynomial sharing and employing multiplicative sharing in a threshold structure. The security of our proposed threshold scheme is reducible to the security of a single-user scheme, which has been proven secure under the random oracle model.

Keywords: Ad-hoc network, forward secure threshold signature, forward security, multiplicative sharing

1 Introduction

In public key infrastructure (PKI), the private key of the certificate authority (CA) is mainly used for digital signature purposes: to sign on users' public key in the form of certificates, or to sign the certificate revocation lists (CRLs) that contain information about which public key is revoked prior its prescribed validity period. The public and private key pair of the CA service is normally used for rather long time, due to the difficulties in re-distributing CA's public key to all network users (who may be disconnected at any time) and re-generating all the users' certificates using CA's new private key.

The devastating consequence of secret key exposure in digital signature is that any signature can be forged and cannot be trusted. The damage of secret key exposure of CA is particularly severe, as it implies all the chains of certificates generated using the CA's private key are no longer trusted and need to be revoked, and there is no way to tell if a signature is generated before or after the key compromise. Along with the revocation of CA's private key come the revocations of all the certificates in the certification chains, and the re-generation of new certificates using the newly generated CA's key pair. The new public key needs to be distributed to all users as well, which may also pose a big administrative overhead. Strong protection of this secret is thus needed.

To mitigate the damage of secret key exposure, *forward-secure signature schemes* and *threshold signature schemes* are devised. Forward-secure signature schemes address the key exposure problem by dividing the usage lifetime of the public key into discrete time periods and using different secret keys in different periods, such that all signatures generated in previous time periods with old secret keys are still considered valid even if the current secret key is compromised. Threshold signature schemes lower the chance of complete exposure of the secret by sharing it among different entities. No single entity will get hold of the complete secret. Threshold signature schemes also address the problem of unavailability, in which any t out of n ($t \leq n$) signing entities can give a valid signature.

Forward-secure threshold signature schemes combine threshold signatures with forward security. Using a forward secure threshold signature scheme, CA's private key can be evolved via the key updating protocol with at least t participating servers, and shares may be refreshed between key updates as well. Such a class of schemes is an important technology in mobile ad-hoc network (MANET) as mobile nodes are more vulnerable to compromise. Moreover, the transient and volatile nature of MANETs makes the idea of single CA not practical. The problems of low security level and low availability of a single CA can be addressed with the use of forward-secure threshold signature schemes.

[*]The research was done when the authors were with Department of Computer Science, The University of Hong Kong.

1.1 Our Contributions

In this paper, we propose a robust forward-secure threshold signature scheme based on the technique of multiplicative secret sharing. Each signer only needs to perform a modular exponentiation to obtain a new share value for the new time period from the old share value, i.e. no interaction among the signers is required, while interaction is necessary for previous constructions [1, 4, 7]. For signing, our scheme uses fewer communication rounds when compared with existing work. Our scheme is also the first multiplicative threshold signature scheme that does not require the presence of all n signers to perform signing and key update, as compared with [1], which requires n signers to perform signing and key update. The security of our threshold scheme is reducible to the security of the scheme in single user setting, which has been already proven secure under the random oracle model in [2].

1.2 Organization

The rest of the paper is organized as follows. The next section shows how existing schemes cannot satisfy the requirements in MANETs. Followed by some preliminaries in Section 3, Section 4 presents our proposed multiplicative forward-secure threshold signature scheme. The proposed scheme's efficiency and security analysis are given in Section 5. Finally, Section 6 concludes our paper.

2 Existing Schemes

In a threshold CA setting, the private key of CA is distributed among a number of servers. To apply forward security to the threshold CA key, a number of requirements related to communication issues are introduced:

1) The number of communication rounds required for the operations should be minimized;

2) Only loose synchronization is required;

3) The scheme should not depend on the network topology. For example, the scheme should rely on the existence of broadcast or point-to-point private channels as little as possible.

To the best of authors' knowledge and an extensive survey in [6], existing forward-secure threshold signature schemes include [1, 4, 7]. The comparisons of them are summarized in Table 1. It can be seen that although these schemes are quite optimal in the security aspects (*e.g.* some of them achieve optimal resilience of $t = (n-1)/2$), they are far from efficient in terms of communication overhead and do not fit in MANET very well. All of them require a quite tight synchronization, and the uses of a broadcast channel and point-to-point private channels.

Scheme 1 in [1] requires all n signing servers to be online for signing and private key updates, hence is not truly a threshold scheme, and it is not considered in our comparisons. All the three schemes in Table 1 are or can be extended to be secure against mobile malicious attackers. They require at least $t+1$ uncorrupted servers to perform signing (which represents more than half of n). In the case some of the partial signatures being invalid (produced by corrupted servers) or lost, recovery procedures need to be performed. However, the communication overheads required to sign is rather unsatisfactory. All of them require a broadcast and point-to-point private channels to exchange messages and require quite a number of rounds. We refer one round of communication as the flow of messages from node $\mathcal{P}_i$ to node $\mathcal{P}_j$ and then back to node $\mathcal{P}_i$. The numbers in Table 1 only represent the minimum number of rounds required under 'perfect' condition. If recovery procedures are needed, the numbers will be even higher. To perform private key update, scheme 2 in [1] requires at least two-third of online servers, while the schemes in [4] and [7] both requires all the n servers to participate which may not be feasible in MANETs as nodes may be unreachable at any time.

There are difficulties to apply the above three schemes to threshold CA servers in MANET environments, due to the restrictive assumptions:

1) Presence of broadcast channels and/or point-to-point private channels;

2) Synchronization between the servers with a global clock;

3) Signing and key updating requires a majority or even all the servers to participate and they need to connect to each other.

These cannot be easily achieved due to high node mobility and poor connectivity in MANETs. In order to relax the broadcast and point-to-point assumptions, some secure routing protocols and Byzantine agreement may be required. More practical and efficient forward secure threshold schemes which are round optimal and useful in MANET environments is a viable research direction.

3 Preliminaries

3.1 Framework

A standard framework of a forward-secure threshold signature scheme [1, 4, 7] is adopted. For a (t, n) forward-secure threshold signature scheme, there are a total of n signers each with a share of the secret key, where any t out of n signers can function together correctly to implicitly "reproduce" the secret key and generate signatures, but any $t-1$ signers cannot. A (t, n) forward-secure threshold signature scheme consists of the following four components: key generation (KeyGen), distributed signing (Sign), distributed key evolution (Update), and verification (Verify). The functions of these algorithms are formalized as below.

- KeyGen: On input of the total number of signers n, the threshold number of signers t, the total number of

Schemes	**AMN01 [1] Scheme 2**	**TT01 [7]**	**CLT03 [4]**
Security-Related Aspects			
Sharing Principle	Polynomial-based	Multiplicative and polynomial-based	Polynomial-based
Number of Compromised Servers Tolerated	$t = (n-1)/3$	$t = (n-1)/2$	$t = (n-1)/2$
Number of *Uncorrupted* Servers for Signing	$2t+1$	$t+1$	$t+1$
Type of Adversaries Tolerated	Mobile Halting	Mobile Malicious	Adaptive Malicious
Communication-Related Aspects			
Rounds of Communication in Signing	$2L$ [1]	At least 3 broadcasts and 1 private exchange	At least 5 broadcasts
Synchronization Model	Synchronous Communication	Synchronous by Rounds	Synchronous Communication
Broadcast and Private Channel Requirement	Both[2]	Both[3]	Both

Table 1: Comparisons of different forward-secure threshold signature schemes

time periods T and an unary string input 1^k where k is a security parameter, it produces the common public parameters *params*, which include the public key PK, a description of a finite message space together with a description of a finite signature space. Each signer $\mathcal{P}_i$ also gets $SK_0^{(i)}$ as the share of the secret key value SK_0 for period 0.

- Sign: On input $(i, j, m, SK_j^{(i)})$, where m denotes the message to be signed and $SK_j^{(i)}$ denotes the ith share of the secret key SK_j for the current time period j $(1 \le j \le T)$, the signer $\mathcal{P}_i$ outputs the partial signature $\sigma^{(i)}$. A third party or any signer is able to construct the final signature σ given a set of t partial signatures $\{\sigma^{(i)}\}$.

- Update: On input $(i, j, SK_j^{(i)})$, where $SK_j^{(i)}$ denotes the ith share of the secret key SK_j for the current period j $(1 \le j \le T)$, the signer $\mathcal{P}_i$ gets $SK_{j+1}^{(i)}$ for period $j+1$ and deletes $SK_j^{(i)}$. As a result, the system's secret key is implicitly evolved to SK_{j+1}.

- Verify: On input (σ, j, PK, m), it outputs 1 for "true" or 0 for "false", depending on whether σ is a valid signature of message m signed by the secret key SK_j at period j of the corresponding PK or not.

3.2 Multiplicative Secret Sharing

A multiplicative secret sharing scheme for secret in non-Abelian group was proposed in [5]. In this scheme, there are n parties where n is a power of 2, and a threshold of t parties to recover the secret. The scheme is based on *Vandermonde's convolution* equation, *i.e.*

$$\sum_{\ell} \binom{r}{\ell}\binom{s}{t-\ell} = \binom{r+s}{t},$$

which has a combinatorial meaning as the number of ways to choose ℓ red balls among a total of r red balls, and to choose $t-\ell$ green balls among a total of s green balls, for each possible ℓ, is equal to the number of ways to choose t balls among r red balls and s green balls.

To share a secret y to n parties in which t of them can recover it, we can put $r = s = n/2$ and spilt the secret in a recursive manner. First randomly pick a y_2 and compute $y_1 = y(y_2)^{-1}$, then share y_1 in $(n/2,\ \ell)$ threshold scheme and share y_2 in $(n/2,\ t-\ell)$ threshold scheme, where ℓ is in the range $\max(0, t-n/2) \le \ell \le \min(t, n/2)$.

Our scheme use two procedures from [5]: Share and Recover. Share$(S_0, (1,n), t, list)$ distributes the initial secret key S_0 to the signers (indexed by 1 to n) such that t of them can recover the secret, using a multiplicative sharing method. Data structure for storing the shares is named SUB-SHARE, a global associative array which is indexed by *list* hinting which share to use given the list of online signers. After executing the procedure Share, each signer is given a number of sub-shares. To perform signing, a chosen signer $\mathcal{P}_i$ (t of n signers) will execute Recover$(i, (1,n), \mathcal{B}, list)$ to determine which sub-share to use, where $\mathcal{B}$ is the set of the selected signers. An algorithm Share to distribute the secret y to n parties is described as followed:

```
Share($y; (i,j); t; list = empty$) {
  if $(i = j)$ and $(t = 1)$ then
    give the pair $(list, y)$ to party $\mathcal{P}_i$;
    ($\mathcal{P}_i$ then stores $y$ at virtual address $list$)
    SUB-SHARE[$list$] $\leftarrow y$;
  if $(i < j)$ then
    $m \leftarrow j - i + 1$;
    $m_1 \leftarrow \max(0, t - m/2)$;
    $m_2 \leftarrow \min(t, m/2)$;
    for $\ell = m_1, \cdots, m_2$ inclusive
      $list_1 \leftarrow list \cup ((i, i-1+m/2), \ell)$;
      $list_2 \leftarrow list \cup ((i+m/2, j), t-\ell)$;
```

```
        if ($\ell \neq 0$) then
          if ($\ell \neq t$) then
            uniformly choose $y_{1,\ell}$ in the group $\mathbb{Z}$;
            $y_{2,\ell} \leftarrow (y_{1,\ell})^{-1} \cdot y$;
            $\mathsf{Share}(y_{1,\ell}; (i, i-1+m/2); \ell; list_1)$;
            $\mathsf{Share}(y_{2,\ell}; (i+m/2, j); t-\ell; list_2)$;
          else $\mathsf{Share}(k; (i, i-1+m/2); t; list_1)$;
        else $\mathsf{Share}(k; (i+m/2, j); t; list_2)$;
}
```

A (t, n) sharing of secret s can be done by executing $\mathsf{Share}(s; (1, n); t; list = empty)$.

The algorithm Recover to recover a secret y from a subset of size t out of a total of n is described as followed:

Let $\mathcal{B}$ be a subset of t parties to recover the secret y. Set $\mathcal{B}_i = 1$ if and only if party $\mathcal{P}_i$ is in $\mathcal{B}$. Each party $\mathcal{P}_i$ invokes $\mathsf{Recover}(i; (1, n); (\mathcal{B}_1, \mathcal{B}_2, \cdots, \mathcal{B}_n); list = empty)$ to recover his sub-share y_i. It is noted that the hamming weight of $(\mathcal{B}_1, \mathcal{B}_2, \cdots, \mathcal{B}_n) = t$ and $y = \prod_{\mathcal{P}_i \in \mathcal{B}} y_i$.

```
$\mathsf{Recover}(\ell; (i, j); (\mathcal{B}_1, \mathcal{B}_2, \cdots, \mathcal{B}_n); list)$ {
  set $m = j - i + 1$;
  if ($m = 1$) then
    set $list = list \cup ((i, i); \mathcal{B}_i)$;
    output $y_\ell$ = SUB-SHARE[$list$];
  if ($\ell \leq m/2$) then
    set $j' = i - 1 + m/2$;
    set $list = list \cup ((i, j'); \sum_{z=i}^{j'} \mathcal{B}_z)$;
    $\mathsf{Recover}(\ell; (i, j'); (\mathcal{B}_i, \cdots, \mathcal{B}_{j'}); list)$;
  else
    set $i' = i + m/2$;
    set $list = list \cup ((i', j); \sum_{z=i'}^{j} \mathcal{B}_z)$;
    $\mathsf{Recover}(\ell; (i', j); (\mathcal{B}_{i'}, \cdots, \mathcal{B}_j); list)$;
}
```

The number of duplicated sub-shares will depend on the threshold t and the total number of parties n. When t is small, more sub-shares (given to different nodes) will be the same to provide the needed redundancy. Larger t means the parties have higher chance to get unique share.

4 Proposed Signature Scheme

Our proposed scheme extends the forward-secure signature scheme in [2] to a (t, n) threshold version, in which n signers each with a share of the secret key, and any t of them can function correctly to generate signatures. The secret key is not *explicitly* constructed to prevent the secret from being exposed. The scheme in [2] is a non-threshold signature scheme provably forward-secure under the random oracle model. In our scheme, the multiplicative secret sharing technique from [5] is used in sharing the secret as well as in signing messages.

Let $H : \{0, 1\}^* \rightarrow \{0, 1\}^l$ be a cryptographic hash that is modeled by a random oracle. The total number of periods is $T = 2^r < 2^l < 2^k$. In general, the notation $X^{(i)}$ indicates the share of secret X held by signer $\mathcal{P}_i$.

4.1 Key Generation

With the input of the security parameter k and the total number of periods T, the trusted dealer $\mathcal{D}$ performs the following.

1) $\mathcal{D}$ computes $N = p \cdot q$, where N is k bit, p and q are two random distinct primes such that $p \equiv q \equiv 3 \bmod 4$.

2) $\mathcal{D}$ randomly selects S_0 from the group $\mathbb{Z}_N^*$.

3) $\mathcal{D}$ runs $\mathsf{Share}(S_0, (1, n), t, list)$ to distribute the secret S_0 so that the i^{th} signer gets the share $S_0^{(i)}$.

4) $\mathcal{D}$ computes $U \leftarrow 1/S_0^{2^{l(T+1)}} \bmod N$.

The public key PK of the whole system is (N, U, T), and the (initial) secret key is shared among n signers where each signer $\mathcal{P}_i$ is given the share $SK_0^{(i)} = (N, T, 0, S_0^{(i)})$. In the key generation process, the trusted dealer $\mathcal{D}$ needs to establish n point-to-point private communication channels to distribute the respective secret shares to the n signers. After that, $\mathcal{D}$ can go offline.

4.2 Key Update

At the beginning of each time period j $(1 \leq j \leq T)$, each signer $\mathcal{P}_i$ computes his new secret share $S_j^{(i)}$ by raising 2^l power on the existing share $S_{j-1}^{(i)}$ used in period $j - 1$, *i.e.* $S_j^{(i)} = [S_{j-1}^{(i)}]^{2^l} \bmod N$. The secret key share for signer $\mathcal{P}_i$ at period j is $(N, T, j, S_j^{(i)})$. When j equals $T + 1$, each signer simply does nothing and returns ϵ (the empty string). This step requires no interaction among the signers, except for the signaling to perform the key update operation. If a signer is not available or reachable during the key update time, he can get the most current time index from another signer later, so that he can perform the key update by himself to 'catch up' the time index. For example, if his own time index is T_a and his neighbour is $T_a + 1$, then he knows he has missed one update and needs to perform the update by one exponentiation.

4.3 Signing

In the signing protocol, a user $\mathcal{U}$ tries to contact at least any t out of the n signers to get a signature. For example, $\mathcal{U}$ may broadcast to his neighbours and wait for responses (together with the time index j) from any t signers. $\mathcal{U}$ then acts as a combiner for the partial signatures generated by the t signers. This is a common scenario in MANETs where $\mathcal{U}$ is the user requesting certificate from the distributed CA servers in a (t, n) threshold setting. Only $\mathcal{U}$ is interested in getting the final signature from the servers' partial signatures, but the signers may not be interested at all in the generated partial signatures. $\mathcal{U}$ will perform the following to get a signature:

1) After $\mathcal{U}$ succeeds in contacting at least t signers with consistent time index j, he sends the coalition information to the t selected signers, in the form of an n-bits string $(\mathcal{B}_1, \cdots, \mathcal{B}_n)$, where $\mathcal{B}_i$ equals 1 if signer $\mathcal{P}_i$ is selected (i.e. in the coalition), 0 otherwise. This requires at least t messages n bits each.

2) Each signer $\mathcal{P}_i$ in the coalition individually picks a random number R_i from the group $\mathbb{Z}_N^*$, computes $Y_i = R_i^{2^{l(T+1-j)}} \bmod N$, and sends Y_i backs to $\mathcal{U}$. This requires t messages k bits each.

3) After $\mathcal{U}$ gets all Y_i where $\mathcal{B}_i = 1$, he can reconstruct $Y = \prod_{\mathcal{B}_i=1} Y_i \bmod N$. Then $\mathcal{U}$ performs hashing on time index j, the computed Y, and message M, to get $\sigma = H(j, Y, M)$. $\mathcal{U}$ sends σ to signer $\mathcal{P}_i$ where $\mathcal{B}_i = 1$. This requires t messages l bits each.

4) Each signer $\mathcal{P}_i$ in the coalition invokes $\mathsf{Recover}(i, (1, n), (\mathcal{B}_1, \cdots, \mathcal{B}_n), list)$ to decide which sub-shares to use in signing. Each of them then computes $Z_i = R_i[S_j^{(i)}]^\sigma \bmod N$, sends Z_i back to $\mathcal{U}$. This requires t messages k bits each.

5) $\mathcal{U}$ constructs $Z = \prod_{\mathcal{B}_i=1} Z_i \bmod N$.

The signature produced for message M is $\{j, Z, \sigma\}$, which is $(r + k + l)$ bits long. All signers then safely erase their R_i's. Hence, in the signing protocol, there are only two rounds of communication required for constructing the signature after user $\mathcal{U}$ has located at least t signers.

4.4 Verification

The signature verification is simply an algorithm which can be executed by any party who possesses the public key PK of the signer to verify the signature without any interaction with the signers possessing the secret key shares. Suppose a user $\mathcal{U}$ possesses the signer's public key $PK = (N, U, T)$, he can verify the signature $\{j, Z, \sigma\}$ of a message M by the following steps.

1) If $Z \equiv 0 \bmod N$, then return 0.

2) Compute $Y' = Z^{2^{l(T+1-j)}} \cdot U^\sigma \bmod N$.

3) If $\sigma = H(j, Y', M)$, then return 1; else return 0.

The correctness of the scheme is as followed. For a message M and its signature $\{j, Z, \sigma\}$, given the signer's public key $PK = (N, U, T)$,

$$\begin{aligned}
Y' &= Z^{2^{l(T+1-j)}} \cdot U^\sigma \bmod N \\
&= [\prod_{\mathcal{B}_i=1} Z_i^{2^{l(T+1-j)}}] \cdot [1/S_0^{2^{l(T+1)}}]^\sigma \bmod N \\
&= [\prod_{\mathcal{B}_i=1} R_i^{2^{l(T+1-j)}} (S_j^{(i)})^{\sigma 2^{l(T+1-j)}}]/S_0^{\sigma 2^{l(T+1)}} \bmod N \\
&= (\prod_{\mathcal{B}_i=1} Y_i) \cdot S_j^{\sigma 2^{l(T+1-j)}}/S_0^{\sigma 2^{l(T+1)}} \bmod N \\
&= (\prod_{\mathcal{B}_i=1} Y_i) \cdot S_0^{\sigma 2^{l(T+1)}}/S_0^{\sigma 2^{l(T+1)}} \bmod N \\
&= (\prod_{\mathcal{B}_i=1} Y_i) \bmod N \\
&= Y \bmod N.
\end{aligned}$$

Hence, we have $H(j, Y', M) = H(j, Y, M) = \sigma$.

4.5 Robustness

If some of the signers uses an incorrect share to produce the partial signature, the final signature generated will be invalid and hence malicious adversaries cannot be tolerated. We can overcome this problem by doubling the size of non-secure storage: generating the "public keys" for the shared private keys by $U^{(i)} \leftarrow 1/{S_0^{(i)}}^{2^{l(T+1)}} \bmod N$, then the validity of the partial signature can be verified as follows.

1) After $\mathcal{U}$ gets $\{Y_i\}$ from all participating signers, $\mathcal{U}$ computes $Y = \prod_{\mathcal{B}_i=1} Y_i \bmod N$ and $\sigma = H(j, Y, M)$.

2) $\mathcal{U}$ sends σ to all participating signers and receives partial signature Z_i in turns.

3) Then $\mathcal{U}$ computes $Y'_i = Z_i^{2^{l(T+1-j)}} \cdot {U^{(i)}}^\sigma \bmod N$.

4) If $Y_i = Y'_i$, then the signature combination proceeds, else it is concluded that the signer $\mathcal{P}_i$ has produced an invalid partial signature.

5 Analysis of Our Proposal

5.1 Efficiency

For initial key generation, a trusted dealer is assumed. He will establish n point-to-point private channels with the signers to distribute the secret shares. If a PKI is already deployed, this is easily done by encrypting the shares with each signer's public key to ensure data confidentiality. Besides, after the distribution of shares is done, the trusted dealer no longer needs to stay online and the private channels are no longer needed. i.e. our scheme do not rely on the restrictive assumption #1 in Section 2.

Regarding the restrictive assumption #2 in Section 2; only loose synchronization (the 'catch up' mechanism for signers who are unavailable during the key update time) is required in our scheme for key update. The 'catch up' mechanism enables the key update to be done by any t signers instead of all n signers. By our construction of the signing protocol, any t signers can give a valid signature. To conclude, the update and signing algorithm of our proposed scheme do not reply on the restrict assumption #3.

Now comes to the computational efficiency. For key update, the proposed scheme requires only one exponentiation operation with no interaction among signers, i.e. $O(k^3)$ where k being the security parameter, while [4] runs in $O(\log^5 T)$ time where T is the total number of time periods, or equivalently, $O(k^5)$ time if $T = 2^r < 2^l < 2^k$. For signing, the proposed scheme also uses fewer communication rounds, namely two rounds, than previous schemes. In fact, the proposed scheme trades the efficiency of key update by the size of the secret shares. Verification is just as efficient as other schemes.

5.2 Security

We first give some intuitive idea about the security of the scheme. User $\mathcal{U}$ collects all Y_i's and Z_i's, but will not know R_i due to the RSA assumption. Even R_i is known, $\mathcal{U}$ will not know $S_j^{(i)}$ from $Z_i R_i^{-1} \bmod N$ as long as the exponentiation by σ is non-trivial, again due to the RSA assumption. Hence, $\mathcal{U}$ cannot recover the secret.

For the distributed random number generation, each signer $\mathcal{P}_i$ selects its own random number R_i. Dishonest signers may choose a Y_i' according to the Y_i's generated from other signers. However, it is not possible to derive the corresponding R_i' from Y_i' by the RSA assumption.

Finally, the forward security relies on the intractability of the Blum integer factorization problem [1] (an integer N is called a Blum integer [3] if $N = pq$ where p and q are both primes and $p \equiv q \equiv 3 \bmod 4$).

The following theorem proves our proposed scheme is secure against adaptive chosen message attack [1, 4, 7].

Theorem 1. *Let FS-DS denote the single-user signature scheme in [2]. Our proposed scheme is a forward-secure threshold signature scheme secure against adaptive chosen message attack as long as FS-DS is a forward-secure signature scheme in the single-user sense.*

Proof. Let $\mathcal{A}$ be an adversary who controls up to $t-1$ signers during execution of our scheme before the j^{th} time period and controls up to t signers (i.e. knowing the secret key of the system) at the j^{th} time period. $\mathcal{A}$ is allowed to launch adaptive chosen-message attack, i.e. it can obtain valid signatures for message $M_1, M_2, \cdots$ on its wishes, in an adaptive manner. If $\mathcal{A}$ can produce with non-negligible probability a valid signature for an un-queried message M, (i.e. $M \neq M_i$ for $i \geq 1$) for time period j', where $j' < j$, we can construct a forger $\mathcal{F}$ to forge a signature of FS-DS using the procedure $\mathcal{A}$ and the signing oracle $\mathcal{O}_{sig}$ of FS-DS.

Let (N, U, T) be the public key of FS-DS. We set U to be the public key corresponding to the secret share obtained by one signer $\mathcal{U}_k$ of our simulated scheme, while we can execute the original key generation algorithm for generating the secret share for the rest of $n-1$ signers. The public key of our simulated scheme will be (N, U', T), where $U' = \prod_{i \neq j} U^{(i)} \cdot U$.

For signing requests issued by $\mathcal{A}$, for a signature of M_i, $\mathcal{F}$ queries $\mathcal{O}_{sig}$ to obtain a signature $\{j, Z, \sigma\}$. $\mathcal{F}$ simulates the generation of partial signature by $\mathcal{U}_k$ as follows. Firstly, Y is recovered by $Y = Z^{2^{l(T+1-j)}} \cdot U^{\sigma} \bmod N$, then a random number R_k is chosen from $\mathbb{Z}_N^*$ and $\mathcal{F}$ will set $\mathcal{U}_k$'s first message to be $Y \cdot R_k^{2^{l(T+1-j)}}$. For the rest of the participating signers other than $\mathcal{U}_k$, the random number R_i used during partial signature generation will be randomly chosen except an arbitrary one of them, which is set to be the inverse of the product of these R_is and R_k, i.e. $[R_k \cdot \prod_{\mathcal{B}_i=1} R_i]^{-1}$. This is to ensure that the Y will appear again in the final signature. Specifically, $\mathcal{F}$ simulates the transcript of the signature generation by returning the following.

1) $Y \cdot R_k^{2^{l(T+1-j)}}$ as the intermediate value produced by $\mathcal{U}_k$ during partial signature generation.

2) $\{j, Z \cdot R_k, \sigma\}$ as the partial signature produced by $\mathcal{U}_k$.

3) $\{j, Z \cdot \prod_{\mathcal{B}_i=1} [S_j^{(i)}]^{\sigma} \bmod N, \sigma\}$ as the final signature.

It is easy to see that the transcripts for the partial signature generation of all signers are valid and the final signature is valid as well. Moreover, $\mathcal{A}$ will not find the transcripts are deviated from the real situation as all R_i are generated from a random distribution.

At the j^{th} time period, $\mathcal{F}$ provides the correct secret shares of t signers to $\mathcal{A}$ by choosing j as the break-in period of FS-DS. On input of these correct shares and previous transcripts, $\mathcal{A}$ produces a valid signature $\{j', Z', \sigma'\}$ for a new message M, $M \neq M_i$, at time period j, where $j' < j$, then $\{j', Z', \sigma' / \prod_{\mathcal{B}_i=1} [S_{j'}^{(i)}]^{\sigma'} \bmod N\}$ is the forgery for FS-DS. Thus, FS-DS is forgeable under a chosen message attack, which is a contradiction. □

6 Concluding Remarks

Existing forward-secure threshold signature schemes may not fulfill the low communication round requirement of mobile ad-hoc network. A new robust forward-secure threshold signature scheme is proposed. Our scheme is based on multiplicative secret sharing, which allows non-interactive proofs in the form of partial signatures, and avoids a lot of interactive proof of knowledge required in the polynomial-based scheme. In this way, key update and signing are efficient in terms of communication rounds when compared with existing schemes.

We note that polynomial sharing gives an optimum information theoretic secure secret sharing scheme, with the size of the shares given to each player is equal to the size of the dealt secret; while it is not the case for multiplicative threshold structure. In spite of the low communication attained, we need a comparatively larger secure storage, specifically, a $\log n$ expansion. Studying the trade-off between bandwidth and storage requirement is a viable research direction.

References

[1] M. Abdalla, S. Miner, and C. Namprempre, "Forward-secure threshold signature schemes," *Topics in Cryptology - CT-RSA '01*, LNCS 2020, pp. 441-456, David Naccache, Editor, Springer-Verlag, 2001.

[2] M. Abdalla, and L. Reyzin, "A new forward-secure digital signature scheme," *Advances in Cryptology - Asiacrypt '00, 6th International Conference on the Theory and Application of Cryptology and Information Security*, LNCS 1976, pp. 116-129, Tatsuaki Okamoto, Editor, Springer-Verlag, Kyoto, Japan, Dec. 3-7, 2000.

[3] M. Blum, "Coin flipping by telephone: A protocol for solving impossible problems," *Proceedings of 24th IEEE Computer Conference (CompCon)*, pp. 133-137, 1982.

[4] C. K. Chu, L. S. Liu, and W. G. Tzeng, "A threshold GQ signature scheme," *Cryptology ePrint Archive*, Report 2003/016, 2003, http://eprint.iacr.org.

[5] Y. Desmedt, G. D. Crescenzo, and M. Burmester, "Multiplicative non-abelian sharing schemes and their application to threshold cryptography," *Advances in Cryptology - Asiacrypt '94*, LNCS 917, pp. 21-32, Josef Pieprzyk and Reihaneh Safavi-Naini, Editors, Springer-Verlag, 1995.

[6] G. Itkis, *Forward Security – Adaptive Cryptography: Time Evolution*, Handbook of Information Security, John Wiley and Sons, 2005.

[7] W. G. Tzeng, and Z. J. Tzeng, "Robust forward-secure signature schemes with proactive security," *Public Key Cryptography '01*, LNCS 1992, pp. 264-276, Kwangjo Kim, Editor, Springer-Verlag, 2001.

Sherman S.M. Chow is currently a PhD candidate in the Courant Institute of Mathematical Sciences at New York University. He obtained his BEng degree in Computer Engineering (with first class honors) and MPhil degree in Computer Science from the University of Hong Kong, and his MS degree in Computer Science from New York University. He is a member of the IACR. His research interests include Applied Cryptography and Distributed System Security.

H. W. Go obtained her M.Phil. degree in Computer Science from the University of Hong Kong. Her research interest is in Information Security and Cryptography, specifically in Forward Security.

Lucas Chi-Kwong Hui is the founder and Honorary Director of the Center for Information Security & Cryptography, and concurrently an associate professor in the Department of Computer Science, The University of Hong Kong. His research interests include Information Security, Computer Crime, Cryptographic Systems, and Electronic Commerce Security. Dr Hui received his B.Sc. and M.Phil. degrees in computer science from the University of Hong Kong, and his M.Sc. and Ph.D. degrees in computer science from the University of California, Davis. He is a member of HKIE and a senior member of IEEE.

S.M. Yiu obtained his PhD in Computer Science from the University of Hong Kong and is currently a Research Assistant Professor in the Department of Computer Science of the same university. His research interests include bioinformatics, information security, and cryptography.