An Efficient and Flexible Forward-Secure Signature Scheme

Hui-Feng Huang* and Chin-Chen Chang
*Department of Information Management
National Taichung Institute of Technology, Taichung, Taiwan, 404, R.O.C.

Department of Information Engineering and Computer Science
Feng Chia University, Taichung, Taiwan, 40724, R.O.C.

Abstract

In 2001, Itkis and Reyzin proposed the first forward-secure signature scheme in which both the signing and the verifying are as efficient as the underlying ordinary signature scheme. However, their method will require to generate many larger primes for keys, and key update algorithm is very slow, making it impractical for some applications. This paper proposes a new efficient method to implement the forward-secure signature. The proposed method can solve the problem raised by Itkis and Reyzin's scheme of finding many larger primes and the efficiency of this method is the same as the underlying signature scheme. In the proposed scheme, no modular exponentiation and inverse computations are required for the key update algorithm. Moreover, only four modular multiplications, three modular additions and two hashing computations are performed by the key update procedure. That is more efficient than Itkis and Reyzin's key update algorithm. Comparing with the existing forward-secure signature schemes in the literatures, the proposed scheme reduces a large of computations for the key update procedure. Hence, the fast key update algorithm could be used for electronic checkbook application. Furthermore, in our scheme, the total number of time periods T could be extended to T+1 or more periods, and the storage space to store its keys and signatures is almost the same as in those ordinary signature schemes. Thus, the proposed scheme is more practical and has lower limitations.

Keywords:

forward-secure signature, cryptography

1. Introduction

The purpose of the digital signature is to establish the identity of the document's signer. If the secret key of a signer is compromised, then all of the past signatures become worthless. It is possible for a signer to deny ever signing the message by claiming that the private key has been compromised. The ordinary digital signatures have this limitation. The forward-secure signature schemes were first proposed by Anderson [1] in 1997 and formalized by Bellare and Miner [2] in 1999. The goal of the forward-secure signature scheme is to preserve the validity of past signatures, and the forger cannot forge signatures for past time periods even if the current secret key has been compromised. This is achieved by dividing total time into T periods, and by using a different secret key in

each time period. That is, the number of the time period is part of the signature. Therefore, in the signature verification, a signature with incorrect time periods should not be verified. Some schemes require more storage to maintain secret keys for future certificates. Since the verifier needs to verify a chain of certificates to verify the actual signature on a message, a longer verification time is needed than in ordinary signatures. In fact, these schemes offer a trade-off between storage requirements and verification time. Later, some proposed schemes keep the master public key fixed but they do not need such certificates while per-period secret (public) keys are presented. Basically, forward-secure signature schemes are the same as ordinary signature schemes, the only difference is that there are time period and key update algorithm in forward-secure signatures. Because the key update algorithm is public in these schemes, the verifier does not need to maintain the certificates of the secret key in each time period. However, all of these previous forward-secure signature schemes took significantly longer to sign and verify than ordinary signature schemes.

In 2001, Itkis and Reyzin [3] proposed a forward-secure signature scheme based on the scheme presented by Guillou and Quisquater [4], which is one of the most efficient ordinary signature schemes. Itkis and Reyzin's method is the first forward-secure signature scheme in which both the signing and the verifying are as efficient as in the ordinary signature scheme. Their method also keeps the master public key fixed and the key update algorithm is public so that no certificates are required for per-period secret key. In their method, the signer will use the key (n, s_i, e_i) and the verifier will use the public key (n, V, e_i), where V is a fixed master public key; e_i and s_i are the public key and the secret key for the current time period i respectively; and n is a product of two large secret primes then publishes n. Both signing and verifying procedures will take two modular exponentiations, one modular multiplication, and one hash computation. However, the (per-period) secret key update has to perform $O(T)$ modular exponentiations which is inefficient. On the other hand, for the security of their scheme, these e_i's should be larger primes and need to ensure that each of them should be relatively prime to $\phi(n)$, where $\phi(n)$ is Euler's phi-function. Then, it is very time-consuming to find larger prime e_i's and to make sure that e_i is relatively prime to $\phi(n)$. In the key update algorithm, the signer needs some storage requirements to store these e_i's so that it can compute these secret keys s_i's. Moreover, in Itkis and Reyzin's scheme, the total number of periods T should be fixed from the very beginning, for their subsequence secret keys are relatively connected. Then, T periods cannot be easily extended to T+1 or more periods. In fact, all previous known schemes (with the exception of the very inefficient "long signature scheme" described in [2]) require the total number of periods T to be fixed in advance and passed as a parameter to the key generation algorithm. This is short of flexibility.

In order to improve these disadvantages as above, we propose a new efficient and flexible forward-secure signature scheme in which both the signing and the verifying are as efficient as

the underlying ordinary signature scheme. The concept behind the proposed method is similar to the concept behind Itkis and Reyzin's scheme. We use two generators of distinct order to implement so that all public keys and secret keys would not be limited by larger primes. Our scheme also keeps the master public key fixed, but each period secret key can be randomly produced so the key update algorithm will be more efficient than all previous known schemes. And the T periods could be easily extended to T+1 or more periods. That means our forward-secure signature scheme which does not need the total number of time periods to be fixed in advance. In addition, the total storage required even with additional secrets is still less than in Itkis and Reyzin's scheme as well as in most of the previous schemes. In our scheme both the signing and the verifying will also require two modular exponentiations, one modular multiplication, and one hash computation. It is the same as in Itkis and Reyzin's method. But in our key update algorithm, no modular exponentiation and inverse computations are required. Moreover, only four modular multiplications, three modular additions and two hashing computations are performed by the key update procedure. This is very fast during each key update. Comparing with Itkis and Reyzin's scheme and the best previous schemes that are mentioned in Malkin et al.'s paper [7], our scheme reduces a large of computations for the key update in each time period. In fact, our key update algorithm is more efficient than the existing forward-secure signature schemes. A fast key update is more useful for many applications such as an electronic checkbook (e-check) application where the time period corresponds to the check serial number, rather than physical time. If your electronic wallet is stolen or the current (e-check) secret key is exposed, you want to revoke all compromised checks without invalidating the checks which were legitimately issued. This could be achieved if the secret key is updated after signing every check. In this case, the fast key update algorithm is as important as fast signature generation. Hence, our forward-secure signature scheme possesses this property. From above, the proposed method is more practical and flexible. Furthermore, the security of our scheme is based on the difficulty of computing discrete logarithms and factorization problems.

The rest of this paper is organized as follows. In the next section, we will review Itkis and Reyzin's forward-secure signature scheme. In Section 3, we present our forward-secure signature scheme. In Section 4, we analyze the security of our proposed method and discuss the efficiency of our scheme. Finally, a brief conclusion is presented in Section 5.

2. Related Work

Itkis and Reyzin's forward-secure signature scheme is based on RSA [5]. There are four stages in their scheme: key generation, secret key update algorithm, signature generation, and signature verification. For the initialization, let $n=pq$, where p and q are two large primes and $\phi(n) = (p-1)(q-1)$. Later the signer publishes n, and keeps p, q, and $\phi(n)$ secret. Divide total time into T time periods. Then, we briefly review these four stages as follows.

Key generation:

1. Choose a random number t_1 and generate larger primes e_i for $i = 1,2,...,T$, where each e_i is relatively prime to $\phi(n)$.
2. Let $f = e_2 \times e_3 \times \cdots \times e_T \pmod{\phi(n)}$ and $s_1 = t_1^{\,f} \pmod{n}$.
3. Compute a master public key $V = s_1^{\,-e_1} \pmod{n}$ and compute $t_2 = t_1^{\,e_1} \pmod{n}$.
4. Finally, the public key is $pk = \{n, V\}$ and the signer will use the secret key s_1, and the verifier will use the public key (n, V, e_1) for the initial time period signature.

Secret key update algorithm:

The $(j+1)$th time period secret key $s_{j+1} = t_{j+1}^{\,e_{j+2} \times \cdots \times e_T} \pmod{n}$ for $j = 1,2,...,T-1$. Compute $t_{j+2} = t_{j+1}^{\,e_{j+1}} \pmod{n}$.

Then, the signer will use the secret key s_{j+1} and the verifier will use the public key (n, V, e_{j+1}) for the current time period $j+1 \le T$.

Signature generation: For a message *m*, suppose a signer *A* uses secret key s_i to compute σ and *Z* as *A*'s signature in the time period *i*.

The signature is computed as follows:

1. Choose a random integer r and compute $y = r^{e_i} \pmod{n}$.
2. Compute $\sigma = h(i, e_i, y, m)$ and $Z = r s_i^{\sigma} \pmod{n}$.

Here *h*( ) denotes a publicly known one-way hash function.

Signature verification: This process makes sure that (σ, Z) is *A*'s signature for message *m* in time period i. The verifier computes $y' = Z^{e_i} V^{\sigma} \pmod{n}$.

If the equation $\sigma = h(i, e_i, y', m)$ is satisfied, meaning that $y' = y$ the verifier concludes that (σ, Z) is a valid signature for message *m* in the time period i.

The main idea of Itkis and Reyzin's scheme involves choosing $e_1, e_2, ..., e_T$ to be distinct primes such that $s_i^{e_i} = V^{-1} \pmod{n}$ for $1 \le i \le T$. For the security of their scheme, e_i's should be primes. In key generation and secret key update stages, the reader can easily obtain $s_i^{e_i} = V^{-1} \pmod{n}$ for $1 \le i \le T$. And their method needs some storage requirements for e_i's to maintain the secret key update. However, it is very time-consuming to find *T* larger primes [3]. Moreover, when the *T* time periods are designed, then they generate T distinct primes, for example, $e_1, e_2, ..., e_T$. And the master public key *V* can be computed as follows:

$$V = s_i^{\,-e_i} = t_1^{\,-(e_1 \times e_2 \times \cdots \times e_T)} \pmod{n}.$$

Therefore, these secret keys s_i's are mutually dependent. So *T* cannot be extended to *T*+1 or more, unless it is redefined. Hence, it is not flexible.

3. Our Proposed Scheme

The main idea behind our proposed method is based on the following statements, which are close to the subgroup membership problem [6]. We will discuss its security in Section 4.

First, choose a random large prime *N* of the

form $N = 2n+1$, where $n = p_1 p_2$ with p_1, p_2 primes. And $K = |p_1| = |p_2|$, the size of the binary representation of both p_1 and p_2. Next, select two elements g_1 and g_2 such that g_1 of order p_1 modulo *N* and g_2 of order p_2 modulo *N*, i.e., $g_i^{p_i} = 1 \pmod N$ for $i = 1,2$. Thus, the set $G_1 = \{1, g_1, g_1^2, ..., g_1^{p_1-1}\}$ modulo *N* has p_1 distinct elements, and the set $G_2 = \{1, g_2, g_2^2, \cdots, g_2^{p_2-1}\}$ modulo *N* has p_2 distinct elements. Two sets G_1 and G_2 are the subgroups of Z_N^* of order p_1 and p_2, respectively. And let g_1 and g_2 are generators of G_1 and G_2 respectively. We note that $G_1 \cap G_2 = \{1\}$. Here p_1 and p_2 are distinct primes, so p_1 has a multiplicative inverse q_1 modulo p_2. Also, p_2 has a multiplicative inverse q_2 modulo p_1. That is $p_1 q_1 = 1 \pmod{p_2}$ and $p_2 q_2 = 1 \pmod{p_1}$. Let $\alpha_1 = p_2 q_2$ and $\alpha_2 = p_1 q_1$. Then, we have the following proposition.

Proposition 1: If $V = k_1\alpha_1 + k_2\alpha_2 \pmod n$, for some positive integers k_1 and k_2, then $g_1^V = g_1^{k_1}$ and $g_2^V = g_2^{k_2} \pmod N$ hold.

Proof: For $\alpha_1 = p_2 q_2 = 1 \pmod{p_1}$ and $\alpha_2 = p_1 q_1 = 1 \pmod{p_2}$, there exist two integers b_1 and b_2 such that $\alpha_1 = b_1 p_1 + 1$ and $\alpha_2 = b_2 p_2 + 1$. Since $g_1^{p_1} = 1$ and $g_2^{p_2} = 1 \pmod N$, then we have

$$\begin{aligned} g_1^V &= g_1^{k_1\alpha_1 + k_2\alpha_2} \\ &= g_1^{k_1(b_1 p_1 + 1)} \cdot g_1^{k_2 p_1 q_1} \\ &= g_1^{k_1} \pmod N. \end{aligned}$$

Similarly,

$$\begin{aligned} g_2^V &= g_2^{k_1\alpha_1 + k_2\alpha_2} \\ &= g_2^{k_1 p_2 q_2} \cdot g_2^{k_2(b_2 p_2 + 1)} \\ &= g_2^{k_2} \pmod N. \end{aligned}$$

Hence, the proposition is proved.

From the Proposition 1, one can also see the workings of the Chinese Remainder Theorem (CRT) here. Thus we can write $V = k_1 \pmod{p_1}$, $V = k_2 \pmod{p_2}$.

In our forward-secure signature scheme, we divide the total time into *T* periods with fixed "master" public key, and we use different secret key in each time period so as to provide a forward-security property. Our scheme consists of four stages: key generation, key update algorithm, signature generation, and signature verification. Since the key update algorithm is public, nothing can be certified for per-period secret (public) key. For the proposed method, it uses two fixed "master" public keys and generates two secret keys in each time period, which differs from other schemes. Then, we depict these four stages as follows.

Key generation:

1. Choose a large prime *N* of the form $N = 2n+1$ with $n = p_1 p_2$, where p_1, p_2 are distinct secret large primes. Select two integers g_i of order p_i modulo *N*, for $i = 1, 2$. The signer publishes *N*, g_1, and g_2; and keeps p_1 and p_2 secret.
2. Compute two master public keys *V* and *U*. For some positive integers k_1, k_2, and *r*, let $V = k_1\alpha_1 + k_2\alpha_2 \pmod n$ and $U = g_1^{rk_1} g_2^{rk_2} \pmod N$, where $\alpha_1 = p_2 q_2 = 1 \pmod{p_1}$ and $\alpha_2 = p_1 q_1 = 1 \pmod{p_2}$, and k_1, k_2, r, α_1, and α_2 are secret parameters. // $U = g_1^{rk_1} g_2^{rk_2} = (g_1 g_2)^{rV} \pmod N$. //
3. Select two secret random integers t_1 and t_2, initial two secret keys $s_{11} = h_1(t_1, 1) \pmod N$ and $s_{12} = h_1(t_2, 1) \pmod N$

Then, compute the current public key $e_1 = k_1(s_{11}+r)\alpha_1 + k_2(s_{12}+r)\alpha_2 \pmod n$, where $h_1(\,)$ is a public one-way hashing function.

4. Finally, the public key is $pk = \{N, g_1, g_2, V, U\}$ and the signer will use the secret keys (s_{11}, s_{12}) and the verifier will use the public key (N, V, U, e_1) for the initial time period signature.

Key update algorithm:

The j th time period secret keys $s_{j1} = h_1(t_{1_1}, j) \pmod N$, $s_{j2} = h_1(t_2, j) \pmod N$ where for $j = 2,3,\ldots,T$. And compute $e_j = k_1(s_{j1}+r)\alpha_1 + k_2(s_{j2}+r)\alpha_2 \pmod n$.

Then, the signer will use the secret key (s_{j1}, s_{j2}) and the verifier will use the public key (N, V, U, e_j) for the current time period $j \le T$.

Signature generation: For a message *m*, suppose a signer *A* uses secret keys s_{i1} and s_{i2} to compute σ and *Z* as *A*'s signature in the time period i.

The signature is computed as follows:

1. Choose a random integer $d = (g_1 g_2)^x \pmod N$ for some integer x and compute $y = d^V \pmod N$.
2. Compute $\sigma = h(i, e_{i,}, y, m)$ and $Z = d(g_1^{s_{i1}} g_2^{s_{i2}})^{\sigma} \pmod N$.

Here *h*( ) denotes a publicly known one-way hash function.

Signature verification: This process makes sure that (σ, Z) is *A*'s signature for message m in time period i. The verifier computes $y' = Z^V (g_1 g_2)^{-e_i \sigma} U^{\sigma} \pmod N$, or $y' = Z^V [(g_1 g_2)^{-e_i} U]^{\sigma} \pmod N$. If the equation $\sigma = h(i, e_i, y', m)$ is satisfied, meaning that $y' = y$ the verifier concludes that (σ, Z) is a valid signature for message *m* in the time period i. The verifier can store pre-computation $(g_1 g_2)^{-e_i}$ for each period *i*.

Then, we give the following theorem to examine the correctness of the proposed method:

Theorem 1: If $y = d^V$,b where $d = (g_1 g_2)^x$, and $U = g_1^{rk_1} g_2^{rk_2} \pmod N$, for some σ and $Z = d(g_1^{s_{i1}} g_2^{s_{i2}})^{\sigma} \pmod N$, then $Z^V (g_1 g_2)^{-e_i \sigma} U^{\sigma} = y \pmod N$ holds.

Proof: Since $V = k_1\alpha_1 + k_2\alpha_2 \pmod n$ and $e_i = k_1(s_{i1}+r)\alpha_1 + k_2(s_{i2}+r)\alpha_2 \pmod n$, from Proposition 1, we have $g_1^V = g_1^{k_1}$ and $g_2^V = g_2^{k2} \pmod N$; also $g_1^{e_i} = g_1^{k_1(s_{i1}+r)}$ and $g_2^{e_i} = g_2^{k_2(s_{i2}+r)} \pmod N$. Then,

$$
\begin{aligned}
& Z^V (g_1 g_2)^{-e_i \sigma} U^{\sigma} \\
&= [d(g_1^{s_{i1}} g_2^{s_{i2}})^{\sigma}]^V (g_1 g_2)^{-e_i \sigma} U^{\sigma} \\
&= d^V (g_1^{s_{i1}\sigma V} g_2^{s_{i2}\sigma V})(g_1^{-e_i \sigma} g_2^{-e_i \sigma}) g_1^{rk_1\sigma} g_2^{rk_2\sigma} \\
&= d^V \\
&= y \pmod N.
\end{aligned}
$$

Hence, the theorem is proved.

According to Theorem 1, our proposed scheme will allow a verifier to verify a valid signature in each time period, and the keys e_i, s_{i1} and s_{i2} need not relatively prime. In fact, these s_{i1} 's and s_{i2} 's can be independent and the number of periods *T* could be extended to *T*+1 or more periods. This means the proposed method could be used in unbounded number of time periods. But in Itkis and Reyzin's method and some schemes that are mentioned in Malkin et al.'s paper [7], the number of periods *T* should be fixed from the very beginning, for their secret keys are relatively connected. Therefore, the proposed method is proven to be more flexible and friendly than those above mentioned schemes.

4. Security and Performance Analysis

In this section, we discuss the security and analyze the efficiency of our forward-secure signature scheme. The security of our scheme lies on the difficulty of computing discrete logarithms and factorization problems. If an intruder can easily factor the integers p_1 and p_2 from $n = p_1 p_2$, then he can derive those secret parameters $k_1 = V \pmod{p_1}$, $k_2 = V \pmod{p_2}$, $\alpha_1 = p_2 q_2 = 1 \pmod{p_1}$ and $\alpha_2 = p_1 q_1 = 1 \pmod{p_2}$. And suppose some private session keys s_{j1} and s_{j2} are compromised, then the intruder can compute another secret parameter *r* from the public key $e_j = k_1(s_{j1} + r)\alpha_1 + k_2(s_{j2} + r)\alpha_2 \pmod{n}$. Since he knows all the secret values k_1, k_2, *r*, α_1, and α_2, he can derive each session secret keys s_{i1} and s_{i2} when the signer has published the session public key $e_i = k_1(s_{i1} + r)\alpha_1 + k_2(s_{i2} + r)\alpha_2 \pmod{n}$. In this situation, all session's secret keys can be obtained. Then, the proposed scheme is insecure. For the security of our scheme, we have $N = 2n+1$ with $n = p_1 p_2$, where N, p_1, p_2 are primes, thus $N-1$ should be large enough to make finding its factorization difficult. Therefore, the same modulo length recommendation as for RSA applies. One can suggest that $|N| \approx 1024$. Moreover, the master public key $V = k_1\alpha_1 + k_2\alpha_2 \pmod{n}$, where k_1 and k_2 are positive integers. That is $V = k_1 \pmod{p_1}$, and $V = k_2 \pmod{p_2}$. So *V* is relatively prime to n, this imply that an adversary could not factor n by means of *V*.

On the other hand, given a value y in Z_N and some integer *V* such that $d^V = y \pmod{N}$, it should be intractable to compute *d*. Therefore, for our scheme to be secure, the value *V* needs to be larger so that computing *d* is infeasible. If *V* is small enough that an adversary can easily compute *d*, then the adversary can easily obtain $(g_1^{s_{i1}} g_2^{s_{i2}})^{\sigma}$ from $Z = d(g_1^{s_{i1}} g_2^{s_{i2}})^{\sigma}$. In this condition, suppose that (σ_1, Z_1) and (σ_2, Z_2) are the signatures of m_1 and m_2 in the i th time period respectively. Now, if $\gcd(\sigma_1, \sigma_2) = 1$ then the attacker can obtain $(g_1^{s_{i1}} g_2^{s_{i2}})$ by applying the Euclidean algorithm. Thus, he can compute $\sigma^{\sim} = h(i, e_{i,} y, m^{\sim})$ and $Z^{\sim} = d(g_1^{s_{i1}} g_2^{s_{i2}})^{\sigma^{\sim}} \pmod{N}$ to forge the signer's signature for message $m^{\sim}$. Therefore, the attacker can forge the signer's signature without knowing the current secret keys s_{i1} and s_{i2}. Hence, the value *V* needs to be larger so that computing *d* from $d^V = y$ is infeasible. Similarly, $U = g_1^{rk_1} g_2^{rk_2} = (g_1 g_2)^{rV} \pmod{N}$, so that computing *r* is infeasible. Since $|N| \approx |p_1| + |p_2|$ and $|p_1| = |p_2|$, it would also be sure that the discrete logarithm problem is hard in G_1 and G_2. The security of our scheme is very similar to Gonzalez Nieto *el al.*'s scheme [6] that proposed a public key cryptosystem based on the subgroup membership problem. To learn more about the treatment of the subgroup membership problem, the reader can refer to Gonzalez Nieto *el al.*'s paper.

Let's consider the key update algorithm, the secret keys are $s_{j1} = h_1(t_{1_1}, j) \pmod{N}$, and $s_{j2} = h_1(t_2, j) \pmod{N}$ for each time period *j*, where t_1 and t_2 are secret parameters. Suppose some private keys s_{i1} and s_{i2} have been exposed, but it does not help the attacker

to compute the values t_1 and t_2 because they are protected under the hashing function $h_1(\ )$. Then, the attacker cannot derive other secret keys s_{j_1} and s_{j_2} for the time period $j \neq i$. Therefore, the past sessions are safe, and the future sessions are safe. That is, forward security is provided.

With regard to the efficiency of our scheme, both the signing and the verifying take two modular exponentiations, one modular multiplication and one hash computation. It is the same as in Itkis and Reyzin's method. However, Itkis and Reyzin's key update algorithm has to perform *O*(*T*) modular exponentiations in each time period. But in the proposed scheme, no modular exponentiation and inverse computations are required for the key update. Moreover, only four modular multiplications, three modular additions and two hashing computations are performed by the key update procedure. That is very efficient during each key update. Comparing with Itkis and Reyzin's scheme and the best previous schemes which are mentioned in Malkin et al.'s paper [7], the proposed scheme reduces a large of computations for the key update in each time period. In fact, our key update algorithm method is more efficient than the existing forward-secure signature schemes. Moreover, if we take away the time period and the key update algorithm, our forward-secure signature scheme could still be used in the same way as the ordinary signature schemes. And it only needs two extra space requirements for two random numbers t_1 and t_2 to update the secret key for each time period. The increase storage is less than in most previous schemes. On the other hand, the proposed method does not certify the current public (secret) key for each time period. Thus, in our forward-secure signature scheme, both the signing and the verifying are as efficient as in the ordinary signature schemes. We summarize the comparisons of our forward-secure signature scheme with Itkis and Reyzin's scheme in Table 1. We define related notations to analyze the computational complexity. T_e is the time for one exponentiation computation; T_m denotes the time for one modular multiplication computation; T_h means the time for executing the adopted one-way hash function in one's scheme; Note that the time for computing modular addition is ignored, since it is much smaller than T_e, T_m, and T_h.

Table 1. Comparisons for two forward-secure signature schemes

	Our scheme	Itkis-Reyzin's scheme
Computations for the verifier	$2T_e + T_m + T_h$	$2T_e + T_m + T_h$
Computations for the signer	$2T_e + T_m + T_h$	$2T_e + T_m + T_h$
Computations for the key update in each time period	$4T_m + 2T_h$	$O(T) \cdot T_e$

As shown in Table 1, our scheme only requires four modular multiplications and two hashing function for the key update in each period. It is obvious that the proposed scheme

provides the fast key update algorithm.

5. Conclusions

In this paper, we solve the problem of generating many of the larger primes in Itkis and Reyzin's method. In our scheme, both the signing and the verifying are as efficient as underlying the ordinary signature scheme, and the storage space for keys and signatures is almost the same as in those ordinary signature schemes. Moreover, only four modular multiplications, three modular additions and two hashing computations are required for our key update procedure. That is more efficient than the existing forward-secure signature schemes. And the total number of periods T could be extended to T+1 or more periods. It means that our forward-secure signature scheme does not need its maximal number of time periods to be fixed in advance. So, the fast key update algorithm and the flexible time periods make our scheme very attractive in many electronic applications.

References

[1] R. Anderson, Invited Lecture, *Fourth Annual Conference on Computer and Communications Security, ACM*, 1997.

[2] M. Bellare and S. Miner, "A Forward-Secure Digital Signature Scheme," *Advances in Cryptology-CRYPTO'99,* LNCS 1666, Aug. 1999, pp. 431-448.

[3] G. Itkis and L. Reyzin, "Forward-Secure Signatures with Optimal Signing and Verifying," *Advances in Cryptology-CRYPTO 2001*, LNCS 2139, Aug. 2001, pp. 332-354.

[4] L. C. Guillou and J. J. Quisquater, "A Paradoxical Identity-Based Signature Scheme Resulting from Zero-Knowledge," *Advances in Cryptolgy-CRYPTO'88*, Vol. 403, pp. 216-231.

[5] R. L. Rivest, A. Shamir, and L. Adlemsan, "A Method for Obtaining Digital Signatures and Public-key Cryptosystem," *Communications of ACM*, Vol. 21, No. 2, Feb. 1978, pp. 33-39.

[6] J. M. Gonzalez Nieto, C. Boyd, and Ed Dawson, "A Public Key Cryptosystem Based on the Subgroup Membership Problem," *Information and Communications Security,* LNCS 2229, 2001, pp. 352-363.

[7] T. Malkin, D. Micciancio, and S. Miner, " Efficient Generic Forward-Secure Signatures with an Unbounded Number of Time Periods," *Advances in Cryptology-EUROCRYPT 2002*, LNCS 2332, 2002, pp. 400-417.