Efficient Forward Secure Signature from Bilinear Pairings

Duc-Liem Vo* and Kwangjo Kim*

*International Research center for Information Security (IRIS),

Information and Communications University (ICU)

Abstract

In this work, we have proposed an efficient forward secure signature based on bilinear pairings. Although our forward secure signature takes the total number of time periods as a input parameter, performance of the scheme is independent to the total number of time periods. The scheme maintains sizes of keys and signature fixed. In addition, the signing algorithm is very efficient with the simple verification algorithm. We also provide a formal definition along with a detailed security proof of our signature scheme under the assumption of Computational Diffie-Hellman problem.

I. Introduction

Key exposure problem happens when secret keys of a cryptosystem are compromised and it may cause serious devastation to applications which require security. Once a secret key is exposed, all protected information by that key is not ensure, both the past and future information. To deal with this problem, many solutions have been introduced including secret sharing, proactive cryptography, timestamp service, and so on. Recently, the constructing cryptosystems supporting *forward security* gets interests of researchers. Anderson [2] first suggested forward security for a signature scheme in which the private key is updated periodically while the public key is kept unchanged. This idea then was formalized by Bellare and Miner [4] with the concepts of the forward secure signature using a key evolution technique in which compromising the private key of a certain time period will not harm previously issuing signatures signed by keys prior to the losing key. From this work, a variety of the forward secure signature schemes has been proposed [1, 8, 9, 10].

Although the forward secure signature provides stronger security than the traditional signature, it requires additional computation as well as storage for the key updating processes. In most of the schemes, besides general security parameters, schemes' computational complexity depends on the total time periods parameter too. To overcome this limitation, we have proposed a pairing based forward secure signature scheme in which computational complexity does not depend on the total time periods. In addition, our signature scheme achieves efficiency in terms of the size in key and signature due to operating over an elliptic curve.

Organization: In Section II, we will provide the mathematical treatment in bilinear

pairings and definitions about the forward secure signature scheme. The details of our signature scheme are presented in Section III and its security analysis is discussed in Section IV. The evaluation of the scheme is discussed in Section V and concluding remarks are given in Section VI.

II. Backgrounds

1. Definitions

Forward Secure Signature Scheme. Like almost other forward secure signature schemes, the definitions of our forward secure signature scheme follow the formal definition from [1, 4].

Definition 1. (Key-evolving Signature Scheme). *A key-evolving digital signature scheme is a quadruple of algorithms, FSIG =(FSIG.KeyGen; FSIG.KeyUp; FSIG.Sign; FSIG.Verify), where:*

- *FSIG.KeyGen, the (probabilistic) Key Generation algorithm, takes as input a security parameter $k \in N$ (given in unary as 1^k) and returns a pair (SK_0; PK), the initial secret key and the public key;*

- *FSIG.Sign, the (possibly probabilistic) Signing algorithm, takes as input the secret key SK_i of the current time period i and a message M, and returns a pair $\langle i,\sigma \rangle$, the signature of M for time period i;*

- *FSIG.KeyUp, the (possibly probabilistic) Secret Key Update algorithm, takes the secret key for the current period SK_i as input and returns the new secret key SK_{i+1} for the next time period;*

- *FSIG.Verify, the (deterministic) Verification algorithm, takes the public key PK, a message M, and a candidate signature $\langle i,\sigma \rangle$ as input, and returns 1 if σ is a valid signature of M or 0, otherwise. It is required that $FSIG.Verify_{PK}(M; FSIG.Sign_{SKi}(M))=1$ for every M and time period i.*

Security Analysis Using Random Oracle Model. We analyze our signature scheme in the random oracle model [5] and use the security model introduced by Bellare and Miner [4].

To assess the success probability of F breaking the forward security of *FSIG*, consider the following experiment. Throughout this paper, $k,...,T$ indicates that the arguments of the key generation algorithm could be more than k and T.

Experiment F-Forge-RO($FSIG$, F)

Select $H: \{0,1\}^* \rightarrow \{0,1\}^l$ at random

$(SK_0, PK) \xleftarrow{R} FSIG.KeyGen^H(k,...,T)$;
$i \leftarrow 0$

Repeat

$d \leftarrow F^{H, FSIG.Sign^H_{SK_i}(\cdot)}(cma, PK)$;
$SK_{i+1} \leftarrow FSIG.KeyUp^H(SK_i); i \leftarrow i+1$

Until $(d = breakin)$ or $(i = T)$

If $(d \neq breakin)$ and $(i = T)$ then $i \leftarrow T+1$

$i \leftarrow i-1$;
$(M, \langle b, \sigma \rangle) \leftarrow F^H(forge, SK_i)$

If $FSIG.Verify^H_{SK_i}(M, \langle b, \sigma \rangle) = 1$ &$0 \leq b < i$

and M was not queried to $FSIG.Sign^H_{SK_i}$

in period b

then return 1 else return 0

With the above forger, we can define the notion of security of the forward secure signature scheme in the random oracle model.

Definition 2 (Forward-security in the Random Oracle Model). *Let FSIG =(FSIG.KeyGen; FSIG.KeyUp; FSIG.Sign; FSIG.Verify) be a key-evolving signature*

scheme, H be a random oracle and F be an adversary as described above. We let $\mathbf{Succ}^{fwsig}(FSIG[k,...,T];F)$ *denote the probability that the experiment F-Forge-RO(FSIG[k,...,T];F) returns 1. Then the insecurity of FSIG is the function*

$$\mathbf{InSec}^{fwsig}(FSIG[k,...,T];t;q_{sig};q_{hash})=\max_{F}\{\mathbf{Succ}^{fwsig}(FSIG[k,...,T];F)\}$$

where the maximum here is taken over all adversaries F making a total of at most q_{sig} *queries to the signing oracles across all the stages and for which the running time of the above experiment is at most t and at most* q_{hash} *queries are made to the random oracle H.*

2. Bilinear Pairings

We summarize some concepts of bilinear pairings using similar notations used by Zhang and Kim [11] which was used to design ID-based blind signature and ring signature based on pairings.

Let G_1 and G_2 be additive and multiplicative groups of the same prime order q, respectively. Let P is a generator of G_1. Assume that the discrete logarithm problems in both G_1 and G_2 are hard. Let $e:G_1\times G_1\rightarrow G_2$ be a pairing which satisfies the following properties:

1. *Bilinear*: $e(aP,bP') = e(P,P')^{ab}$ for all P, $P'\in G_1$ and all $a,b\in Z$.

2. *Non-degenerate*: If $e(P,P') = 1$ for all $P'\in G_1$ then $P = O$.

3. *Computable*: There is an efficient algorithm such as [3] to compute $e(P,P')$ for any P, $P'\in G_1$.

Group G_1 is called Gap Diffie-Hellman (GDH) group if in this group, the Computational Diffie-Hellman (CDH) problem is hard but the Decision Diffie-Hellman (DDH) problem is easy. For the sake of comparison, we assume that, as in [8], there is a parameter generator IG takes input k, and outputs G_1, G_2 of order q, and pairing e. The computational complexity of IG is $O(k^n)$. Also the computational complexity in groups G_1, G_2, and pairings e are at most $O(k^{n_1})$, $O(k^{n_2})$, and $O(k^e)$, respectively. We have n, n_1, n_2, $e\in N$ are order of the polynomial time algorithm.

The definition of the CDH assumption used for our security analysis is as follows:

Definition 3 (CDH Assumption). *A probabilistic algorithm A is said to be* (t,ε)*-break-CDH in a cyclic group G if A runs at most time t, computes the Diffie-Hellman function* $DH_{P,q}(aP,bP)=abP$*, with input (P, q) and (aP,bP), with a probability of at least* ε*, where the probability is over the coins of A and (a, b) is chosen uniformly from* $Z_q\times Z_q$*. The group G is a* (t,ε)*-CDH group if no algorithm* (t,ε)*-break-CDH in this group.*

III. Proposed Scheme

Our forward secure signature scheme *FSIG* consists of four algorithms.

Key Generation Algorithm *FSIG.KeyGen.* Let $H_1:\{0,1\}^*\rightarrow Z_q^*$ and $H_2:\{0,1\}^*\rightarrow G_1$ be collision-free hash functions.

FSIG.KeyGen$(1^k,T)$

Run IG to get groups G_1, G_2 (prime order q), bilinear map e.

Select random generator $P\in G_1$; s, t, $r_0\in_R Z_q^*$

Compute: $Q = sP$, $X = tP$

Set $PK=(G_1, G_2, e, P, Q, X, T)$

Compute:

$s_0 = s + r_0H_1(0)$; $t_0 = t - r_0H_1(0)$;

$Q_0 = r_0H_1(0)P$;

$V_0 = t_0 Q_0$;

Erase s, t, r_0, t_0

Set $SK_0=(0, s_0, V_0, Q_0, T)$

Output $(SK_0; PK)$

Key Update Algorithm *FSIG.KeyUp.*

FSIG.KeyUp(SK_{i-1})

Parse SK_{i-1} as $(i-1, s_{i-1}, V_{i-1}, Q_{i-1}, T)$

If $i=T-1$

$SK_i=\oslash$; delete SK_{i-1}

else

Pick a random element $r_i \in_R Z_q^*$

Compute:

$s_i = s_{i-1} + r_i H_1(i)$;

$Q_i = Q_{i-1} + r_i H_1(i) P$;

$V_i = V_{i-1} + r_i H_1(i)(X - Q_{i-1} - Q_i)$;

Erase s_{i-1}, r_i, V_{i-1}, Q_{i-1}

Set SK_i (i, s_i, V_i, Q_i, T)

Output SK_i

Signing Algorithm *FSIG.Sign.*

FSIG.Sign(M, SK_i)

Parse SK_i as (i, s_i, V_i, Q_i, T)

Set $U = Q_i$;

Compute $\alpha = s_i Q_i + V_i$; and $\beta = s_i H_2(i,M,U)$;

Set $\sigma = (U, \alpha, \beta)$

Output signature for M as $\langle i, \sigma \rangle$

Verification Algorithm *FSIG.Verify.*

FSIG.Verify$_{PK}$(M, i, σ):

Parse σ as (U, α, β) and Verify

$$e(\alpha, P) = e(U, X + Q) \quad (1)$$

$$e(\beta, P) = e(H_2(i, M, U), U + Q) \quad (2)$$

Output 1 if Eqs (1) and (2) are correct, otherwise output 0.

Correctness The correctness of the proposed signature scheme comes from the correctness of Eqs (1) and (2). The correctness of Eq (1) is shown bellow:

$$\begin{aligned}
&e(\alpha,P) = e(s_i Q_i + V_i, P) = e(\left[s_{i-1} + r_i H_1(i)\right] Q_i + V_i, P) \\
&= e\left(\left[s + \sum_0^i r_k H_1(k)\right] Q_i + V_i, P\right) \\
&= e\left(s Q_i + \sum_0^i r_k H_1(k) Q_i + V_i, P\right) \\
&= e(s Q_i + \sum_0^i r_k H_1(k) Q_i + V_{i-1} \\
&+ r_i H_1(i)\left[X - Q_{i-1} - Q_i\right], P) \\
&= e(s Q_i + \sum_0^{i-1} r_k H_1(k) Q_i + V_{i-1} + r_i H_1(i)\left[X - Q_{i-1}\right], P) \\
&= e(s Q_i + \sum_0^{i-1} r_k H_1(k)\left[Q_{i-1} + r_i H_1(i) P\right] + \\
&+ V_{i-1} + r_i H_1(i)\left[X - Q_{i-1}\right], P) \\
&= e(s Q_i + \sum_0^{i-1} r_k H_1(k)\left[Q_{i-1} + r_i H_1(i) P\right] + \\
&+ V_{i-1} + r_i H_1(i)\left[X - Q_{i-1}\right], P) \\
&= e(s Q_i + \sum_0^{i-1} r_k H_1(k) Q_{i-1} + r_i H_1(i) \sum_0^{i-1} r_k H_1(k) P + \\
&+ V_{i-1} + r_i H_1(i)\left[X - Q_{i-1}\right], P) \\
&= e(s Q_i + \sum_0^{i-1} r_k H_1(k) Q_{i-1} + V_{i-1} + r_i H_1(i) X, P) \\
&\vdots \\
&= e(s Q_i + r_0 H_1(0) Q_0 + V_0 + r_1 H_1(1) X + + \ldots + r_i H_1(i) X, P) \\
&= e(s Q_i + r_0 H_1(0) Q_0 + t_0 Q_0 + r_1 H_1(1) X + + \ldots + r_i H_1(i) X, P) \\
&= e(s Q_i + t Q_0 + r_1 H_1(1) X + \ldots + r_i H_1(i) X, P) \\
&= e(s Q_i + r_0 H_1(0) X + \ldots + r_i H_1(i) X, P) \\
&= e(s Q_i + \sum_0^i r_k H_1(k) X, P) = e(s Q_i + t \sum_0^i r_k H_1(k) P, P) \\
&= e(\left[s + t\right] Q_i, P) = e(Q_i, P)^{s+t} = e(Q_i, X + Q)
\end{aligned}$$

The validity of the signature is guaranteed by the correctness of Eq (2).

$$\begin{aligned}
&e(\beta,P) = e(s_i H_2(i,M,U), P) \\
&= e(\left[s_{i-1} + r_i H_1(i)\right] H_2(i,M,U), P) \\
&\vdots \\
&= e\left(\left[s + \sum_{k=0}^i r_k H_1(k)\right] H_2(i,M,U), P\right) \\
&= e(s H_2(i,M,U), P)\, e(H_2(i,M,U), P)^{\sum_{k=0}^i r_k H_1(k)} \\
&= e(H_2(i,M,U), Q)\, e\left(H_2(i,M,U), \sum_{k=0}^i H_1(k) r_k P\right) \\
&= e(H_2(i,M,U), Q)\, e(H_2(i,M,U), Q_i) \\
&= e(H_2(i,M,U), Q)\, e(H_2(i,M,U), U) \\
&= e(H_2(i,M,U), Q + U)
\end{aligned}$$

IV. Security Analysis

We analyze the security of our forward secure signature scheme used technique like in [1, 4, 8]. In addition, we assume that, partial key exposure also leads to key

exposure problem. The following theorem shows the security of our scheme.

Theorem 1. *If there exists a forger F that runs in time at most t, asking at most qhash hash queries and qsig signing queries, such that* $\mathbf{Succ}^{fwsig}(FSIG[k,...,T];F)>\varepsilon$ *then there exists a adversary A that* (t',ε')*-break CDH in group* G_1 *where:*

$t'=t+O(k^{n_1})$; *and* $\varepsilon'=\varepsilon/[T(q_{hash}+q_{sig}+1)]$

Proof (Sketch). To break CDH problem in the additive group G_1 of the order q, an adversary A is given P (a random generator of G_1), $P'=aP$, $Q'=bP$, where $a,b\in_R Z_q^*$ remaining unknown to A. The task of A is to derive $S'=abP$ with the help of the forger F. A provides the public key to F and answers its hash queries, signing queries, and *breakin* query. First, A guesses a random i at which F will ask for the break-in query. Then A set the public key $PK=(G_1, G_2, e, P, Q, X, T)$, where $Q=Q'$. A provides PK to F and runs it. A can answer the hash queries and the signing queries since it controls the hash oracle. During execution, A guesses a random index g' and hopes the forgery will base on g'-th hash query. A makes this hash value special, i.e., P'. Suppose F outputs a signature on message $M_{g'}$ for time period $i'<i$. From this signature, A derives $S'=abP$, hence solves CDH problem.

■

Theorem 2. *Let FSIG[k,...,T] represent our key-evolving signature scheme with modulus size k. Then for any t,* q_{hash} *and* q_{sig},

$\mathbf{InSec}^{fwsig}(FSIG[k,...,T];t;q_{sig};q_{hash})\leq$

$T(q_{hash}+q_{sig}+1)\mathbf{InSec}^{cdh}(k,t')$

where $t'=t+O(k^{n1})$

Proof. From Definition 2 and Theorem 1, the insecurity function is computed simply by solving function in Theorem 1 and express ε' in terms of ε we have:

$\varepsilon'=\varepsilon/[T(q_{hash}+q_{sig}+1)]\Rightarrow \varepsilon' T(q_{hash}+q_{sig}+1)=\varepsilon$

This completes the proof of Theorem 2. ■

V. Evaluation

In this section, we compare our proposed signature scheme with the previous signature scheme [8] which has the same computational assumption.

In Table 1, T is total number of time periods and in Table 2, l is a security parameter of conventional cryptographic operation as explained in [10].

[Table 1] *Complexity comparison*

	Hu et al.[8]	Ours
Key Gen.	$O(k^n+k^{n_1}+k^{n_1}\log T)$	$O(k^n+k^{n_1})$
Signing	$O(k^{n_1})$	$O(k^{n_1})$
Verification	$O(k^e\log T+k^{n_1}\log T)$	$O(k^e+k^{n_1})$
Key Update	$O(k^{n_1})$	$O(k^{n_1})$
Public Key	$O(k)$	$O(k)$
Private Key	$O(k\log T+k)$	$O(k)$
Signature	$O(k\log T+k)$	$O(k)$

[Table 2] *Complexity of other schemes*

	BM[4]	IR[9]
Key Gen.	lk^2T	$k^5+(k+l^3)lT$
Signing	$(T+l)k^2$	k^2l
Verification	$(T+l)k^2$	k^2l
Key Update	lk^2	$(k^2+l^3)lT$
Public Key	lk	k
Private Key	lk	k
Signature	k	k

As we can see from Tables 1 and 2, our signature scheme is very efficient in terms of computation as well as performance. The signature and key sizes do not depend on the total number of time periods. Moreover,

comparing with the schemes [1, 4, 9], the signature size is shorter for the same security level since our scheme is operating over an elliptic curve (so the security parameter k is different). The signing algorithm will be similar to that of [6] if we store the fixed part in the signature for later use.

The verification of the signature just requires four pairing operations. For verifying multiple signatures of the same time period, the verification result of Eq (1) can be saved for later use. In this case, the verifying process remains just two pairing operations. Utilizing good pairing implementations [3, 7], our scheme can be efficient in performance. Considering above features, our signature scheme can be applied in the application where storage and computation power are limited like mobile devices while providing stronger security feature.

VI. Concluding Remarks

We have proposed a pairing based efficient forward signature scheme in which performance does not depend on the total number of time periods. Under the assumption of the hardness of Computational Diffie-Hellman problem, we have presented the security proof of the signature scheme in the random oracle model. Moreover, the proposed signature scheme is very efficient in terms the signature size and performance compared to the previous schemes. With a good pairing computation algorithm, we can have an efficient signature verifying algorithm. For further work, we consider integrating our scheme with other cryptographic techniques to have new applications.

References

[1] M. Abdalla and L. Reyzin, "A New Forward-Secure Digital Signature Scheme," Advances in Cryptology - Asiacrypt'00, LNCS 1976, pp. 116 - 129, Springer-Verlag, 2000.

[2] R. Anderson. "Two Remarks on Public-Key Cryptology From Invited Lecture," Fourth ACM Conference on Computer and Communications Security, 1997.

[3] P.S.L.M. Barreto, H.Y. Kim, B. Lynn and M. Scott, "Efficient Algorithms for Pairing-Based Cryptosystems", Advances in Cryptology - Crypto'2002, LNCS 2442, pp. 354-369, Springer-Verlag, 2002.

[4] M. Bellare and S. K. Miner, "A Forward-Secure Digital Signature Scheme," Advances in Cryptology - Crypto'99, LNCS 1666, pp. 431 - 448, Springer-Verlag, 1999.

[5] M. Bellare and P. Rogaway, "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols", ACM Conference on Computer and Communications Security, pp. 62 - 73, 1993.

[6] D. Boneh, B. Lynn, and H. Shacham, "Short Signatures from the Weil Pairing", Advances in Cryptology - Asiacrypt'01, LNCS 2248, pp. 514 - 532, Springer-Verlag, 2001.

[7]S. Galbraith, K. Harrison and D. Soldera, "Implementing the Tate Pairing," Algorithm Number Theory Symposium - ANTS V, LNCS 2369, pp. 324 - 337, 2003.

[8]F. Hu, C. Wu, and J.D. Irwin, "A New Forward Secure Signature Scheme using Bilinear Maps," http://eprint.iacr.org/2003/188.pdf.

[9]G. Itkis and L. Reyzin. "Forward-secure signatures with optimal signing and verifying," Advances in Cryptology - Crypto'01, LNCS 2139, pp. 332-354. Springer-Verlag, 2001.

[10]T. Maklin, D. Micciancio and S. Miner, "Efficient Generic Forward-Secure Signatures with an Unbounded Number of Time Periods," Advances in Cryptology - Eurocrypt 2002, LNCS 2332, pp. 400 - 417, Springer-Verlag, 2002.

[11]F. Zhang and K. Kim, "ID-Based Blind Signature and Ring Signature from Pairings", Advances in Cryptology - Asiacrypt'02, LNCS 2501, Springer-Verlag,

pp. 533 - 547, 2002.