A Forward-Secure Signature with Backward-Secure Detection

Dai-Rui Lin and Chih-I Wang
Department of Computer Science and Engineering
National Sun Yat-sen University, Kaohsiung 804, Taiwan

June 17, 2007

Abstract

This paper enhances the security of Abdalla and Reyzin's forward-secure signature scheme with backward-secure detection. In the proposed scheme, we embeded the hash-chain into the forward-secure signature scheme. It achieves not only forward-secure but also backward-secure for the digital signature.

Keywords : Forward-Secure, Backward-Secure, Digital signatures, GQ signatures.

1 Introduction

The key exposures carries very serious problems, it is regarded a total break of the system. In order to avoid this undesirable situation, the goal of forward security is to protect against this kind of threat.

A protocol is said to provide forward secrecy if the compromise of long-term keys does not compromise past session keys that have been established before the compromise of the long-term key [3]. Bellare and Miner first proposed signatures with forward-security properties [3]. In 2001, Abdalla and Reyzin improved the Bellare-Miner's forward-secure GQ signature schemes with a shorter public key [2]. Since then, many works about forward-secure related schemes has been proposed [8, 5, 6, 1, 9, 10, 7].

The main concept of forward-secure signature scheme is that: the public key is fixed but the secret signing key is updated at regular intervals. Each secret signing key are use for sign messages only during a particular time period. At the end of each time period, a new secret key is produced and

old one is erased. This can be useful to mitigate the damage caused by key exposure without requiring distributions of keys.

However, once the key is exposed, the security of past uses of the keys can be protected, but the future uses of it are compromised. That is, once the future secret keys can be forge, even the original signer cannot detect the secret keys have been used, it is regarded a total crash of the system. Therefore, the perfect solution is that intrusion is detected and we change keys.

In this paper, we improve the Abdalla-Reyzin's forward-secure signature scheme with backward-secure detection. Our constructions achieve not only forward-secure but also backward-secure for the digital signature. We employ the concept of hash-chain to the digital signature. This can ensure the security of the signature be more robust and achieve that the future signature cannot be forge even the secret key are exposed.

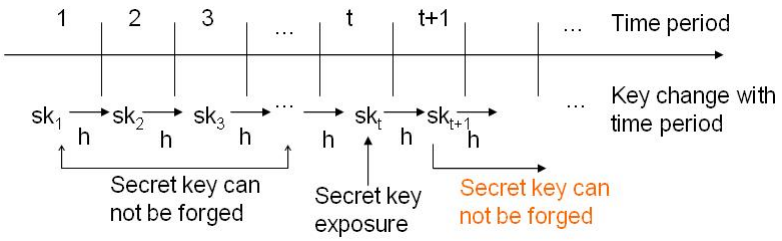

Figure 1: Forward-security and backward-security

2 Review of Abdalla-Reyzin's Forward-secure GQ signature scheme

In this section, we will brief review of forward-secure GQ signature scheme that proposed by Abdalla and Reyzin [2]. Abdalla-Reyzin's was improve the Bellare-Miner's forward-secure GQ signature schemes [3]. Abdalla-Reyzin's schemes has significantly shorter keys that more practical then Bellare-Miner's schemes and depicted in Figure 1. Their scheme has also be proven forward secure in the random oracle model, assuming factoring is hard. The scheme is divided into four phases, (1) *key generation*, (2) *signature generation*, (3) *signature verification* and (4) *key updating*.

- *Key generation phase*:
  Let $p \equiv q \equiv 3(\mod 4)$ be two primes and $N = pq$ be a k-bit integer. The signer chooses a random number $s_0 \in Z_N^*$ as his/her signing

key, and computes the corresponding verification key $u = 1/s_0^{2^{l(T+1)}} \mod N$, where T denotes as the total number of periods, and l denotes as the length of binary string. The signer publishes the verification key set $VK = (N, u, T)$.

- *Signature generation phase*:
  Then the signer generate the signature for period j by the jth period secret key s_j as follows. First, he/she chooses a random number $r \in Z_N^*$ and computes $a = H(j, y, M)$ and $z = r(s_j)^a \mod N$, where $y = r^{2^{l(T+1-j)}} \mod N$, and M is the message to be signed. The signature for period j is (j, a, z) and publish (j, a, z, M) to verifier.

- *Signature verification phase*:
  Upon receiving (j, a, z, M), the verifier computes y' as $y' = z^{2^{l(T+1-j)}} u^a \mod N$. Then, he computes a' as $a' = H(j, y', M)$. The signature is valid if $a = a'$.

- *Key updating phase*:
  Via key-updating mechanism, in each current period j, the signer updating the jth period signing key s_j into s_{j+1} as $s_{j+1} = s_j^{2^l} \mod N$. The public key stays the same.

Sender		Receiver
$s_j = s_{j-1}^{2^l} \mod N$		$y' = z^{2^{l(T+1-j)}} u^a \mod N$
$r \in_R Z_N^*$		$= (r{s_j}^a)^{2^{l(T+1-j)}} u^a \mod N$
$y = r^{2^{l(T+1-j)}} \mod N$	$\xrightarrow{j, a, z, M}$	$= (r{s_j}^a)^{2^{l(T+1-j)}} 1/s_0^{2^{l(T+1)}a} \mod N$
$a = H(j, y, M)$		$= r^{2^{l(T+1-j)}} \mod N$
$z = r{s_j}^a \mod N$		$a \stackrel{?}{=} H(j, y', M)$

Fig. 2. The Abdalla-Reyzin's forward-secure GQ signature.

3 Forward-secure signature with backward-secure detection

In this section, we consider the backward-secure of Abdalla-Reyzin's forward-secure signature. We implement a backward-secure on the condition that the verification formula of the underlying signature scheme is kept unchanged.

3.1 The one-way hash chain

Definition 1. *Let $h(\cdot)$ be a collision-resistant one-way hash function, we denote the $h^T(\cdot)$ be a one-way hash chain, where $h^T(\cdot) = h(h^{T-1}(\cdot)) = h(h(h^{T-2}(\cdot))) = \overbrace{h(h...(h(}^{T}\cdot))...)$ and $h^0(\cdot) = \cdot$. The T be the total number of time periods.*

In order to achieve the forward-secure signature with detection of backward-secure, during the period j, by employee the hash function h, the jth random number r_j can define as $r_j = h^{T-j}(x)$, where x is the random number or the personal secret password.

3.2 Definitions

We denoted the propose schemes by **FSBD**, is integrate hash chain technique into forward-secure signature. The propose schemes consists of five procedures.

Definition 2. A **FSBD** consists of five algorithms that **FSBD**=(**FSBD**$_{\cdot\mathbf{keyGen}}$, **FSBD**$_{\cdot\mathbf{sign}}$, **FSBD**$_{\cdot\mathbf{verify}}$, **FSBD**$_{\cdot\mathbf{backDet}}$, **FSBD**$_{\cdot\mathbf{keyUpdate}}$):

- Key generation algorithm **FSBD**$_{\cdot\mathbf{keyGen}}$: it is a probabilistic polynomial-time algorithm that takes as input a security parameter k and possibility other parameters and returns a public key u and corresponding private key s_0. That is, **FSBD**$_{\cdot\mathbf{keyGen}}(1^k) = (u, s_0, l)$, where l is the system secret.

- Signature generation algorithm **FSBD**$_{\cdot\mathbf{sign}}$: the signing algorithm, takes as input the secret key s_j for the time period j ($j \leq T$) and the message M to be signed and returns the signature $< j, sign >$ of M for time period j. That is, **FSBD**$_{\cdot\mathbf{sign}}(j, s_j, M) = (sign)$.

- Signature verification algorithm **FSBD**$_{\cdot\mathbf{verify}}$: the verification algorithm, takes as input the public key u, a message M, and a candidate signature $< j, sign >$. The algorithm will return 1 if $< j, sign >$ is a valid signature of M. Otherwise, it will return 0. That is, **FSBD**$_{\cdot\mathbf{verify}}(j, u, M, sign) = 1$.

- Backward detection algorithm **FSBD**$_{\cdot\mathbf{backDet}}$: the backward detection algorithm, takes as input the $j + 1$-th value r_{j+1}, computes r_j from $h(r_{j+1})$ as $r_j = h(r_{j+1}) = h^{T-j}(x)$. The algorithm will return 1 if **FSBD**$_{\cdot\mathbf{backDet}}(j, r_{j+1}, y'_j) = 1$, where $y'_j = r_j^{2^{l(T+1-j)}} \mod N$. Otherwise, it will return 0.

- Key updating algorithm $\mathbf{FSBD}_{\cdot\mathbf{keyUpdate}}$: the secret key update algorithm, takes as input the secret key s_j for the current period $j < T$ and returns the new secret key s_{j+1} for the next period $j+1$. That is, $\mathbf{FSBD}_{\cdot\mathbf{keyUpdate}}(j, s_j) = (s_{j+1})$.

Definition 3. *A forward secure signature with the properties of backward-secure and backward-secure detection if the compromise of secret key s_i will not compromise the s_j for all $j > i$, and the verifier can detect the signature has been forge or not by the hash chain value $h^{i+1}(x)$ for each time period $i > 0$.*

Definition 4. (The Blum Factorization Problem) *Given n, where n is the product of two distinct large primes p and q with roughly the same length $p \equiv q \equiv 3(\mod 4)$, find p or q.*

3.3 The proposed scheme

Our construction of the proposed scheme depicted in Figure 2. The proposed scheme divided into five phases, (1) *key generation*, (2) *signature generation*, (3) *signature verification*, (4) *backward-secure detection* and (5) *key updating*. We describe the details of the five phases as follows.

- *Key generation phase*:
  The proposed scheme consists two keys and one hash chain number, the signing key(s), the verification key(u), and the hash chain number(r_0). Let $p \equiv q \equiv 3(\mod 4)$ be two primes and $N = pq$ be a k-bit integer. The signer chooses a random number $s_0 \in Z_N^*$ as his/her signing key, and computes the corresponding verification key $u = 1/s_0^{2^{l(T+1)}} \mod N$, where T denotes as the total number of periods, and l denotes as the length of binary string. The signer publishes the verification key set $VK = (N, u, T)$. Finally, the signer random choose a random number $x \in Z_N^*$ and computes $r_0 = h^T(x)$.

- *Signature generation phase*:
  Then the signer generate the signature for period j by the jth period secret key s_j as follows. First, he/she computes the jth and the j+1th hash chain number r_j and r_{j+1} as $r_j = h^{T-j}(x)$ and $r_{j+1} = h^{T-j+1}(x)$, then he/she computes $a_j = H(j, r_{j+1}, y_j, M_j)$ and $z_j = r_j(s_j)^{a_j} \mod N$, where $y_j = r_j^{2^{l(T+1-j)}} \mod N$, and M_j is the jth message to be signed. The signature for period j is $(j, r_{j+1}, a_j, z_j, M_j)$ and publish $(j, r_{j+1}, a_j, z_j, M_j)$ to verifier.

- *Signature verification phase*:
  Upon receiving $(j, r_{j+1}, a_j, z_j, M_j)$, the verifier computes y'_j as $y'_j = z_j^{2^{l(T+1-j)}} u^{a_j} \pmod N$. Then, he computes a'_j as $a'_j = H(j, y'_j, M_j)$. The signature is valid if $a_j = a'_j$.

- *Backward detection phase*:
  After verify the signature in regular process, he/she can use the r_{j+1} to detect the secret key has been expose or not as follows. First, the verifier computes r_j as $r_j = h(r_{j+1})$, where $r_j \equiv h(h^{T-(j+1)}(x)) \equiv h^{T-j}(x)$, then the verifer check that $y'_j \stackrel{?}{=} (h^{T-j}(x))^{2^{l(T+1-j)}}$

- *Key updating phase*:
  Via key-updating mechanism, in each current period j, the signer updating the jth period signing key s_j into s_{j+1} as $s_{j+1} = s_j^{2^l} \pmod N$. The public key stays the same.

Signer		Verifier
$s_0 \in_R Z_N^*$		$y'_j = z_j^{2^{l(T+1-j)}} u^{a_j} \pmod N$
$u = 1/s_0^{2^{l(T+1)}} \pmod N$		$= (z_j)^{2^{l(T+1-j)}} (1/s_0^{2^{l(T+1)}})^{a_j}$
$s_j = s_{j-1}^{2^l} \pmod N$		$= (r_j (s_j)^{a_j})^{2^{l(T+1-j)}} (1/s_0^{2^{l(T+1)}})^{a_j}$
$r_j = h^{T-j}(x)$		$= r_j^{2^{l(T+1-j)}} {s_0}^{a_j 2^{l(T+1)}} (1/s_0^{2^{l(T+1)}})^{a_j}$
$r_{j+1} = h^{T-(j+1)}(x)$	$\xrightarrow{j, r_{j+1}, a_j, z_j, M_j}$	$= r_j^{2^{l(T+1-j)}} \pmod N$
$y_j = r_j^{2^{l(T+1-j)}} \pmod N$		$a_j \stackrel{?}{=} H(j, y'_j, M_j)$
$a_j = H(j, y_j, M_j)$		backward-secure detection
$z_j = r_j (s_j)^{a_j}$		$r_j = h(r_{j+1}) = h^{T-j}(x)$
		$(h^{T-j}(x))^{2^{l(T+1-j)}} \stackrel{?}{=} y'_j$

Fig. 3. The proposed scheme

4 Security Analysis and Discussions

It is straightforward that the security of the proposed scheme as equal to the forward-secure signature scheme as describe in [4] by Abdalla and Reyzin. They improve the forward-secure signature scheme in the random oracle model. In this section, we will show that the proposed scheme is robust that satisfy not only the forward secure, but also backward secure. The properties comparisons between our scheme and previous schemes [3, 10, 8, 9, 2] are shown in Table 1.

4.1 Correctness

Theorem 2. (Correctness) *Let $j, r_{j+1}, a_j, z_j, M_j$ be a signature values produced by the scheme of Section 3. The execution of our proposed scheme between the signer and the verifier is always successful.*

Proof. From figure 2, we can see the correctness of our schemes is sound. First, the verifier computs y'_j from (j, z_j, u, a_j, N) as $y'_j = z_j^{2^{l(T+1-j)}} u^{a_j} (\mod N)$. Then, the verifier check that $a_j \stackrel{?}{=} H(j, y'_j, M_j)$. Finally, the verifier detect the backward-security as $(r_j)^{2^{l(T+1-j)}} \stackrel{?}{=} y'_j$. If both of the two express is hold, then the signature is valid.

4.2 Forward-seucre protection

Forward-secrecy refers to that the compromise of one or several secret keys does not compromise previous secret keys. Each secret signing key is adopted to sign messages only during a particular time period. Assume that the $j+1$-*th* secret key exposed, an adversary A success forged a *j-th* valid signature for a message m, however, A can not forged the past uses of the secret keys, since computs s_{j-1} from $s_j^{2^{-l}} \mod N$ is irreversible, this is equivalent to factor N. Therefore, the security of past uses of the keys can be protected.

4.3 Backward-secure protection and detection

Backward-secrecy refers to that the compromise of one or several secret keys does not compromise future secret keys.

Theorem 1. (The Backward-secure Detection) *If there exists an attacker A who can break the j-th period in the random oracle model, and success gain a valid signature (j, a_j, z_j) for message M^*. Then the* $\mathbf{FSBD}_{\cdot\mathbf{backDet}}$ *procedure is backward-secure detection that against the malicious attacker.*

Proof. Suppose an attacker A success forged a valid signature (j, a_j, z_j) for message M^*, however, A can not success break $\mathbf{FSBD}_{\cdot\mathbf{backDet}}$ procedure, since the A has to provide the $j+1$-*th* hash value r_{j+1} for computs $r_j (r_j = h(r_{j+1}) = h^{T-j}(x))$, and the verifier has to check the $(r_j)^{2^{l(T+1-j)}} \stackrel{?}{=} y'_j$. The attacker A computs r_j from r_{j+1} is irreversible. Therefore, the backward-security of future uses of the keys can be protected and detected, even the Blum Factorization Problem totally break.

Table 1. The comparisons between [8, 3, 2, 5, 10, 9] and our scheme.

	[8]	[3]	[2]	[5]	[10]	[9]	Our scheme
Forward secure	√	√	√	√	√	√	√
Backward secure	×	×	√	√	√	√	√
Backward detection	×	×	×	×	×	√	√

5 Conclusions

In this manuscript we have enhances the security of Abdalla and Reyzin's forward-secure digital signature scheme with backward-secure detection. Once the secret key is exposed, both of the security of past/future uses of the keys can be protected. We achieved that the intrusion of the future signature can be detected as we desired.

References

[1] M. Abdalla and M. Bellare. Increasing the lifetime of a key: a comparative analysis of the security of re-keying techniques. In *Proc. ASIACRYPT '00, LNCS 1976*, pages 431–448, 1999.

[2] M. Abdalla and L. Reyzin. A new forward-secure digital signature scheme. In *Proc. ASIACRYPT '00, LNCS 2139*, pages 116–129. Springer-Verlag, 2001.

[3] M. Bellare and S. K. Miner. A forward-secure digital signature scheme. In *Proc. CRYPTO '99, LNCS 1666*, pages 431–448. Springer-Verlag, 1999.

[4] S. Goldwasser, S. Micali, and R.L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. Technical report, MIT Lab., Computer Science, Cambridge, Mass., March, 1995.

[5] G. Itkis and L. Reyzin. Forward-secure signatures with optimal signing and verifying. In *Proc. CRYPTO '01, LNCS 2139*, pages 332–354. Springer-Verlag, 2001.

[6] H. Krawczyk. Simple forward-secure signatures from any signature scheme. In *Proceedings of the 7th ACM Conference on Computer and Communications Security*, pages 108–115, 2000.

[7] C. F Lu and S. Shieh. Secure key-evolving protocols for discrete logarithm schemes. In *CT-RSA 2002, LNCS 2271*, pages 300–309, 2002.

[8] C. F. Lu and S. Shieh. Efficient key-evolving protocol for the gq signature. *Journal of information science and engineering*, 20:763–769, 2004.

[9] W. G Tzeng and Z. J Tzeng. Robust forward-secure signature schemes with proactive security. In *Proc. PKC '00, LNCS 1992*, pages 264–276, 2001.

[10] W. G Tzeng and Z. J Tzeng. Robust key-evolving public key encryption schemes. In *ICICS 2002, LNCS 2513*, pages 61–72, 2002.