On the Security of Signature Scheme with Message Recovery and Its Application

Eun-Jun Yoon and Kee-Young Yoo
(Corresponding author: Eun-Jun Yoon)

Department of Computer Engineering, Kyungpook National University
Daegu 702-701, Republic of Korea. (Email: ejyoon@infosec.knu.ac.kr, yook@knu.ac.kr)

(Received Aug. 9, 2005; revised and accepted Oct. 1, 2005)

Abstract

In 2004, Sekhar proposed a new signature scheme with message recovery. Based on this signature scheme with message recovery, they also proposed a designated verifier signature scheme with non-repudiation of origin and a convertible designated verifier signature scheme with non-repudiation of origin. This paper, however, presents a security analysis where Sekhar's signature schemes are vulnerable to some forgery attacks. This means that any attacker can forge a signature for any message.

Keywords: Cryptanalysis, digital signature, forgery attack, message recovery, security

1 Introduction

In 1995, Nyberg and Rueppel [2, 3] proposed message recovery signatures based on the discrete logarithm problem. But their methods have the advantage of smaller signature data for short messages. Therefore, it is effective in applications like public-key certifying protocols and key exchange protocols.

Recently, Sekhar [4] proposed a new signature scheme with message recovery [2, 3] and without using a one-way hash function. In his scheme, a third party can verify the signature without divulging any private keys. It, however, needs a great deal of authentication for many other applications. With a slight modification, they also proposed a designated verifier signature scheme [1] with non-repudiation of origin and a convertible designated verifier signature scheme with non-repudiation of origin. This allows a designated verifier, specified by the signer of the message, to verify the authenticity of the message and non-repudiation of the origin [5] provides the receiver of the message with evidence of the message origin. This will protect against any attempt by the signer to falsely deny having sent the message. Both these schemes are based on their proposed signature scheme with message recovery. This paper presents a security analysis whereby Sekhar's signature schemes are vulnerable to some forgery attacks, in that any attacker can forge the signature for any message.

This paper is organized as follows: In Section 2, we briefly review Sekhar's signature schemes. In Section 3, we show some forgery attacks on Sekhar's schemes. Finally, Section 4 offers concluding remarks.

2 Review of Sekhar's Signature Schemes

This section briefly reviews Sekhar's signature scheme with message recovery and its application. Let p and q be large primes such that $q|p-1$ and α be an element of Z_p^* of order q. Let *Alice*'s public key be $y_A = \alpha^{x_A} (\text{mod } p)$, where $x_A \in Z_q^*$ is her private key. Let *Bob*'s public key be $y_B = \alpha^{x_B} (\text{mod } p)$, where $x_B \in Z_q^*$ is his private key.

2.1 Signature Scheme with Message Recovery

Suppose that *Alice* wants to send a message m to *Bob*. *Alice* executes the following steps:

1) Chooses two random numbers $k_1, k_2 \in Z_q^*$ and computes $U = \alpha^{k_1 - k_2} (\text{mod } p)$ and $Z = y_B^{k_1} (\text{mod} p)$.
2) Computes the ciphertext $r = ZUm (\text{mod } p)$.
3) Computes the signature $s = (k_2 x_A^{-1} - r) (\text{mod } q)$.
4) Sends (r, s, U) to *Bob*.

After receiving (r, s, U) from *Alice*, *Bob* recovers the message by computing

$$m = r(y_A^{(r+s)} U)^{-x_B} U^{-1} (\text{mod } p).$$

If *Bob* wants to prove to the third party that it is a valid signature and was signed by *Alice*, then he can execute the following protocol with the third party. Suppose *Alice* sends the signature (r, s, U, m) on a message m to *Bob*.

1) The third party chooses randomly $a, b \in Z_q^*$, computes $V = (y_A^{(r+s)}U)^a \alpha^b (\text{mod} p)$, and sends V to *Bob*.

2) *Bob* chooses randomly $t \in Z_q^*$, computes $h_1 = V\alpha^t (\text{mod} p)$, $h_2 = h_1^{x_B} (\text{mod } p)$, and sends h_1, h_2 to the third party.

3) The third party sends (a, b) to *Bob*.

4) *Bob* verifies $V = (y_A^{(r+s)}U)^a \alpha^b (\text{mod } p)$. If correct, then *Bob* sends t to the third party. If not, then (a, b) is wrong and *Bob* stops the protocol.

5) After receiving t, the third party verifies $h_1 = V\alpha^t (\text{mod } p)$. If correct, then the third party computes $T = rU^{-1}m^{-1} (\text{mod } p)$ and verifies $h_2 = T^a y_B^{(b+t)} (\text{mod } p)$. If correct, the third party confirms that the signature is valid.

2.2 Designated Verifier Signature Scheme with Non-repudiation of Origin

In Shkhar's designated verifier signature scheme, *Alice* signs a document, of which only *Bob* can verify the validity of the signature. With non-repudiability, if valid, *Bob* can prove its validity to a third party.

Suppose that *Alice* wants to sign a message m so that only *Bob* can verify it. The signature generation is the same as Shkhar's signature scheme with message recovery. After signing, *Alice* sends the signature (r, s, U, m) on a message m to *Bob*. *Bob* verifies and accepts the signature by checking

$$rm^{-1}U^{-1} = (y_A^{(r+s)}U)^{x_B} (\text{mod } p). \quad (1)$$

In the case of a dispute, *Bob* can execute similar verification protocol with the third party as Shkhar's signature scheme with message recovery of Subsection 2.1. If valid, this is to convince the third party that it is *Alice*'s valid signature.

2.3 Convertible Designated Verifier Signature Scheme with Non-repudiation of Origin

Suppose that *Alice* wants to send amessage m to *Bob*. *Alice* executes the following steps:

1) Chooses two random numbers $k_1, k_2 \in Z_q^*$ and computes $j = h(k_1 - k_2)$, where $h(\cdot)$ is a one-way hash function.

2) Computes $U = \alpha^{k_1 - k_2} (\text{mod } p)$, $Z = y_B^{k_1 + j} (\text{mod } p)$ and $J = \alpha^j (\text{mod } p)$.

3) Computes the ciphertext $r = ZUm (\text{mod } p)$.

4) Computes the signature $s = (k_2 x_A^{-1} - r) (\text{mod } q)$.

5) Sends (r, s, U, J, m) to *Bob*.

After receiving (r, s, U, J, m) from *Alice*, *Bob* verifies the signature by checking

$$rm^{-1}U^{-1} = (y_A^{(r+s)}UJ)^{x_B} (\text{mod } p). \quad (2)$$

To convert the signature, *Alice* releases u such as $u = (r + s)x_A + k_1 - k_2 + j$. Anyone can verify the signature after releasing u as in the following:

1) *Alice* publishes $u = (r + s)x_A + k_1 - k_2 + j$.

2) *Bob* checks $\alpha^u = y_A^{(r+s)}UJ$. If not, then the signature is invalid.

3) *Bob* recovers the message by computing

$$rm^{-1}U^{-1} = y_B^u (\text{mod } p).$$

3 Forgery Attacks on Sekhar's Signature Schemes

This section demonstrates that Sekhar's signature schemes are vulnerable to some forgery attacks, in that any attacker can forge a signature for any message.

Theorem 1. *Sekhar's signature scheme with message recovery is insecure in terms of a forgery attack.*

Proof. Suppose that an attacker knows two public keys of the *Alice* and *Bob* and wants to send a forged message m' to *Bob*. Then, any attacker can forge a valid signature of any message m' by the following:

1) Selects forged message m'.

2) Computes $U' = \alpha (\text{mod } p)$, $r' = m'U'y_B (\text{mod } p)$ and $s' = -r' (\text{mod } p)$.

3) Sends the forged signature (r', s', U') on a message m' to *Bob*.

After receiving (r', s', U') from an attacker, *Bob* recovers the message by computing

$$\begin{aligned} r'(y_A^{(r'+s')}U')^{-x_B}U'^{-1} &= r'(y_A^{(r'+(-r'))}\alpha)^{-x_B}U'^{-1} \\ &= m'U'y_B(\alpha)^{-x_B}U'^{-1} \\ &= m' (\text{mod } p). \end{aligned}$$

If *Bob* wants to prove to the third party that it is a valid signature and that it was signed by *Alice*, then he can execute a similar protocol with the third party, as in Shkhar's signature scheme with message recovery of the above Subsection 2.1, in order to convince the third party that it is *Alice*'s valid signature. Unfortunately, since the third party cannot distinguish the forged signature and *Alice*'s valid signature, the third party can easily confirm that the forged signature is valid by performing the verification protocol of Subsection 2.1. Therefore, an attacker can successfully inflict a forgery attack. □

Theorem 2. *Sekhar's designated verifier signature scheme with non-repudiation of origin is insecure in terms of a forgery attack.*

Proof. Any attacker can forge a valid signature as follows:

1) Selects forged message m'.

2) Computes $U' = \alpha(\text{mod } p)$, $r' = m'U'y_B(\text{mod } p)$ and $s' = -r'(\text{mod } p)$.

3) Sends the forged signature (r', s', U', m') on a message m' to *Bob*.

After receiving a forged signature (r', s', U', m') from an attacker, *Bob* verifies the signature by checking

$$r'm'^{-1}U'^{-1} = (y_A^{(r'+s')}U')^{x_B}(\text{mod } p).$$

Obviously, *Bob* cannot determine that the messages (r', s', U', m') are forged, since the verification Equation (1) holds. Its validity is easy to see in that the left-hand side is

$$r'(m')^{-1}(U')^{-1} = y_B(\text{mod } p)$$

and the right-hand side is

$$\begin{aligned}(y_A^{(r'+s')}U')^{x_B} &= (y_A^{(r'+(-r'))}\alpha)^{x_B} \\ &= (\alpha)^{x_B} \\ &= y_B(\text{mod } p).\end{aligned}$$

Since the third party cannot distinguish the forged signature and *Alice*'s valid signature, the third party also can easily confirm that the forged signature is valid by performing the verification protocol of Subsection 2.1, such as a forgery attack on a signature scheme with message recovery. □

Theorem 3. *Sekhar's convertible designated verifier signature scheme with non-repudiation of origin scheme is insecure in terms of a forgery attack.*

Proof. Any attacker can forge a valid signature as follows:

1) Selects forged message m' and random number $j' \in Z_q^*$.

2) Computes $U' = \alpha(\text{mod } p)$, $J' = \alpha^{j'}(\text{mod } p)$, $r' = m'U'y_B^{(1+j')}(\text{mod } p)$ and $s' = -r'(\text{mod } p)$.

3) Sends the forged signature (r', s', U', J', m') on a message m' to *Bob*.

After receiving a forged signature (r', s', U', J', m') from the attacker, *Bob* verifies the signature by checking

$$r'(m')^{-1}(U')^{-1} = (y_A^{(r'+s')}U'J')^{x_B}(\text{mod } p).$$

Obviously, *Bob* cannot determine that the messages (r', s', U', J', m') are forged, since the verification Equation (2) holds. Its validity is easy to see in that the left-hand side is

$$\begin{aligned}r'(m')^{-1}(U')^{-1} &= m'U'y_B^{(1+j')}(m')^{-1}(U')^{-1} \\ &= y_B^{(1+j')}(\text{mod } p)\end{aligned}$$

and the right-hand side is

$$\begin{aligned}(y_A^{(r'+s')}U'J')^{x_B} &= (y_A^{(r'+(-r'))}\alpha\alpha^{j'})^{x_B} \\ &= (\alpha\alpha^{j'})^{x_B} \\ &= y_B^{(1+j')}(\text{mod } p).\end{aligned}$$

Furthermore, if *Bob* wants to prove to the third party that it is a valid signature and that it was signed by *Alice*, then anyone can verify the signature by an attacker's man-in-the-middle attack as in the following:

1) When *Alice* publishes $u = (r+s)x_A + k_1 - k_2 + j$, an attacker intercepts u and then publishes $u' = (1+j')$.

2) Then, the third party can easily be convinced that this holds since the verification equation $\alpha^{u'} = y_A^{(r'+s')}U'J'$ holds. It is easy to see that

$$\begin{aligned}\alpha^{u'} &= y_A^{(r'+s')}U'J' \\ &= y_A^{(r'+(-r'))}\alpha\alpha^{j'} \\ &= \alpha^{(1+j')} \\ &= \alpha^{u'}(\text{mod } p).\end{aligned}$$

□

4 Conclusion

Recently, Sekhar proposed a new signature scheme with message recovery. Based on this scheme, they also proposed a designated verifier signature scheme with non-repudiation of origin and a convertible designated verifier signature scheme with non-repudiation of origin. This paper, however, demonstrated that Sekhar's signature schemes are vulnerable to an attack in that any attacker can forge a signature for any message.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments in improving our manuscript. This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment).

References

[1] M. Jakobsson, K. Sako, and R. Impagliazzo, "Designated verifier proofs and their applications," in *Eurocrypt'96*, LNCS 1070, pp. 143-154, 1996.

[2] K. Nyberg, and R. A. Rueppel, "A new signature scheme based on the DSA giving message recovery," in *Proceedings of the First ACM Conference on Computer and Communication Security*, pp. 58-61, 1993.

[3] K. Nyberg, and R. A. Rueppel, "Message recovery for signature schemes based on the discrete logarithm problem," in *Eurocrypt'94*, LNCS 950, pp. 182-193, 1995.

[4] M. Sekhar, "Signature scheme with message recovery and its application," *International Journal of Computer Mathematics*, vol. 81, no. 3, pp. 285-289, 2004.

[5] J. Zhou, and K. Y. Lam, "Securing digital signatures for non-repudiation," *Computer Communications*, vol. 22, no. 8, pp. 710-716, 1999.

Eun-Jun Yoon received his BS in the School of Textile and Fashion Technology from the Kyung Il University, South Korea in 1995, and his MS in the Computer Engineering from the same University in 2003. He is now working toward the Ph.D. degree in the Kyungpook National University, South Korea. His current research interests are cryptography and network security.

Kee-Young Yoo received his BS degree in education of mathematics from Kyungpook National University in 1976; the MS degree in Computer Engineering from Korea Advanced Institute of Science and Technology in 1978 and the PhD degree in the Computer Science from Rensselaer Polytechnic Institute, New York, U.S.A., in 1992. He is now a Professor at the Department of Computer Engineering, Kyungpook National University. His current research interests are wireless security and cryptography.