Efficient fault-tolerant scheme based on the RSA system

N.-Y. Lee and W.-L. Tsai

Abstract: Data security on the Internet is an area of considerable concern. A recently proposed fault-tolerant scheme for data encryption that was based on the RSA system is shown to be flawed. The flaw occurs if a malicious receiver keeps a senders message and the corresponding signature and then changes the message whilst retaining the existing signature. Under these circumstances the sender cannot deny having written the forged message since it carries his/her correct signature. An improvement to enhance the security and the performance of the existing RSA-based scheme is presented.

1 Introduction

Internet use continues to increase and thus, the protection of data transmitted through the Internet becomes increasingly important. Cryptographic systems are usually used for this purpose. However, errors can occur at high transmission speeds. In order to correct the errors, communication systems need to spend time for re-transmission and this also ties up network bandwidth. Also, errors caused by large computational processing need to be corrected. Error detection and data correction are also important issues.

Zhang [1] recently proposed a fault-tolerant scheme based on the RSA scheme [2]. The scheme can simultaneously deal with error detection and data correction. Up to three errors in the scheme can be detected and corrected. The scheme also allows the receiver to verify the sender's identity. However, we find that the Zhang scheme can suffer from an attack by a malicious receiver. If a malicious person receives a message containing a signature, then he/she can forge a different message for the existing signature. The sender cannot deny having signed the forged message. That is, the sender will be forced to take responsibility for the forged message and the signature. We now propose improvement to the Zhang scheme to repair the security flaw. Compared with the Zhang scheme, the improvement has the same advantages and is more efficient.

2 Review of the RSA scheme with fault tolerance

This Section will introduce the original RSA scheme [2] and the Zhang scheme [1].

IEE Proceedings online no. 20030085

DOI: 10.1049/ip-cdt:20030085

Paper first received 26th June 2001 and in revised form 18th September 2002

The authors are with the Information Management Department, Southern Taiwan University of Technology, Tainan County 710, Taiwan, ROC

2.1 The RSA Scheme

In the RSA scheme, a user A first has to select two large primes p and q, and then compute $N = p \times q$. A also selects a random integer e, satisfying $\gcd(e, \phi(N)) = 1$, where $\phi(\cdot)$ is the Euler's totient function [3]. Then, A computes d, where $d = e^{-1} \bmod \phi(N)$. Thus, the public key of A is (e, N) and the private key is (d, p, q). Assume that another user B wants to send a secret message m to A. B computes a ciphertext $c = m^e \bmod N$ and transmits c to A. On the receipt of c, A decrypts it by computing $m = c^d \bmod N$. On the other hand, if A wants to sign a message m, he/she can compute $s = m^d \bmod N$. Any user can check the validity of s by verifying $m \stackrel{?}{=} s^e \bmod N$. If it is true, the signature s is valid.

2.2 The scheme of Zhang [1]

According to the RSA scheme, the public keys for user A and user B and their private keys are (e_A, N_A), (d_A, p_A, q_A) and (e_B, N_B), (d_B, p_B, q_B), respectively. For the sake of simplicity, N_A and N_B are assumed to have the same length. Assume that a user B wants to send a message M to a user A. B executes the following steps.

Step 1: Translate the message M into an $n \times m$ plaintext matrix $\boldsymbol{X}$:

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}$$

Step 2: Construct another $(n+1) \times (m+1)$ matrix $\boldsymbol{X}_h$:

$$\boldsymbol{X}_h = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} & X_1 \\ x_{21} & x_{22} & \cdots & x_{2m} & X_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} & X_n \\ X^1 & X^2 & \cdots & X^m & h \end{bmatrix}$$

where $X_i = \prod_{j=1}^{m} x_{ij} \bmod N_B$, $1 \le i \le n$, and $X^j = \prod_{i=1}^{n} x_{ij} \bmod N_B$, $1 \le j \le m$, and $h = (X_1 \times X_2 \times \cdots \times X_n) \bmod N_B = (X^1 \times X^2 \times \cdots \times X^m) \bmod N_B$.

Step 3: Compute an $(n+1)\times(m+1)$ ciphered matrix $\boldsymbol{C}_h$:

$$\boldsymbol{C}_h = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1m} & C_1 \\ c_{21} & c_{22} & \dots & c_{2m} & C_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c_{n1} & c_{n2} & \dots & c_{nm} & C_n \\ C^1 & C^2 & \dots & C^m & h_c \end{bmatrix}$$

where $c_{ij} = x_{ij}^{e_A} \bmod N_A$, and $C_i = X_i^{e_A} \bmod N_A$, and $C^j = X^{j e_A} \bmod N_A$, and $h_c = h^{d_B} \bmod N_B$, for all $1 \le i \le n$, $1 \le j \le m$.

Step 4: Transmit $\boldsymbol{C}_h$ to A.

Note that all X_i, X^j, C_i, C^j, $1 \le i \le n$, $1 \le j \le m$ and h_c are checksums and ciphered checksums, and they are error free. When A received the ciphered matrix $\boldsymbol{C}_h$, he/she decrypts $\boldsymbol{C}_h$ by using his/her private key d_A. So A will get the matrix $\boldsymbol{X}_h^*$:

$$\boldsymbol{X}_h^* = \begin{bmatrix} x_{11}^* & x_{12}^* & \dots & x_{1m}^* & X_1^* \\ x_{21}^* & x_{22}^* & \dots & x_{2m}^* & X_2^* \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n1}^* & x_{n2}^* & \dots & x_{nm}^* & X_n^* \\ X^{1*} & X^{2*} & \dots & X^{m*} & h^* \end{bmatrix}$$

where $x_{ij}^* = c_{ij}^{d_A} \bmod N_A$, and $X_i^* = C_i^{d_A} \bmod N_A$, and $X^{j*} = C^{j d_A} \bmod N_A$, and $h^* = h_c^{e_B} \bmod N_B$, for all $1 \le i \le n$, $1 \le j \le m$.

If $X_i^* = \prod_{j=1}^{m} x_{ij}^* \bmod N_B$, $X_j^* = \prod_{i=1}^{n} x_{ij}^* \bmod N_B$, and $h^* = (X_1^* \times X_2^* \times \dots \times X_n^*) \bmod N_B = (X^{1*} \times X^{2*} \times \dots \times X^{m*}) \bmod N_B$, the message is authenticated as having been sent by B. Otherwise, for the sake of simplicity, we assume that a single error has occurred in x_{kl} during the computation phase or data transmission phase, for some k, l, where $1 \le k \le n$ and $1 \le l \le m$. Therefore, we have the following two formulas:

$$X_k^* \ne \prod_{j=1}^{m} x_{kj}^* \bmod N_B \tag{1}$$

$$X^{l*} \ne \prod_{i=1}^{n} x_{il}^* \bmod N_B \tag{2}$$

Using the data correction method in the Zhang scheme, the data can be corrected by computing either one of the following two formulas:

$$x_{kl}^* = X_k^* \times \left(\prod_{\substack{j=1 \\ j \ne l}}^{m} x_{kj}^* \right)^{-1} \bmod N_B \tag{3}$$

$$x_{kl}^* = X^{l*} \times \left(\prod_{\substack{i=1 \\ i \ne k}}^{n} x_{il}^* \right)^{-1} \bmod N_B \tag{4}$$

If multiple errors have occurred, the data correction method is similar to the above discussions. Readers can refer to Zhang [1] for the details.

3 Security flaw in the scheme of Zhang [1]

Let the ciphered matrix $\boldsymbol{C}_h$ be sent to A from B. A can decrypt it to retrieve the plaintext matrix $\boldsymbol{X}_h$ and B's signature h_c. Assume that the malicious receiver A wants to forge another message $\tilde{M}$ for h_c, he/she executes the following steps.

Step 1: Transform the new message $\tilde{M}$ into the following elements:

$$\tilde{x}_{11}, \tilde{x}_{12}, \dots, \tilde{x}_{1m}, \tilde{x}_{21}, \tilde{x}_{22}, \dots, \tilde{x}_{2m}, \dots, \tilde{x}_{n1}, \dots, \tilde{x}_{nm-1}$$

Step 2: Compute $\tilde{x}_{nm} = ((\tilde{x}_{11} \times \tilde{x}_{12} \times \dots \times \tilde{x}_{nm-1})^{-1} \times h) \bmod N_B$, where $h = h_c^{e_B} \bmod N_B$. So, the new plaintext matrix is:

$$\tilde{\boldsymbol{X}} = \begin{bmatrix} \tilde{x}_{11} & \tilde{x}_{12} & \dots & \tilde{x}_{1m} \\ \tilde{x}_{21} & \tilde{x}_{22} & \dots & \tilde{x}_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{x}_{n1} & \tilde{x}_{n2} & \dots & \tilde{x}_{nm} \end{bmatrix}$$

Step 3: Compute $\tilde{X}_i = \prod_{j=1}^{m} \tilde{x}_{ij} \bmod N_B$, $1 \le i \le n$, and $\tilde{X}^j = \prod_{i=1}^{n} \tilde{x}_{ij} \bmod N_B$, $1 \le j \le m$. Let $\tilde{h}_c = h_c$. Now, the new plaintext matrix $\tilde{X}_h$ with data correcting ability is constructed as:

$$\tilde{\boldsymbol{X}}_h = \begin{bmatrix} \tilde{x}_{11} & \tilde{x}_{12} & \dots & \tilde{x}_{1m} & \tilde{X}_1 \\ \tilde{x}_{21} & \tilde{x}_{22} & \dots & \tilde{x}_{2m} & \tilde{X}_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \tilde{x}_{n1} & \tilde{x}_{n2} & \dots & \tilde{x}_{nm} & \tilde{X}_n \\ \tilde{X}^1 & \tilde{X}^2 & \dots & \tilde{X}^m & \tilde{h}_c \end{bmatrix}$$

Anyone can verify the validity of B's signature by checking the following equations:

$$\tilde{X}_i \stackrel{?}{=} \prod_{j=1}^{m} \tilde{x}_{ij} \bmod N_B, \quad \text{for } 1 \le i \le n$$

$$\tilde{X}^j \stackrel{?}{=} \prod_{i=1}^{n} \tilde{x}_{ij} \bmod N_B, \quad \text{for } 1 \le j \le m$$

$$\tilde{h} \stackrel{?}{=} \tilde{h}_c^{e_B} \bmod N_B$$

where $\tilde{h} = (\tilde{X}_1 \times \tilde{X}_2 \times \dots \times \tilde{X}_n) \bmod N_B = (\tilde{X}^1 \times \tilde{X}^2 \times \dots \times \tilde{X}^m) \bmod N_B$. If these equations are true, B cannot deny having signed the forged plaintext matrix $\tilde{\boldsymbol{X}}_h$. Thus, the Zhang scheme does indeed have a security flaw.

Remark: We also note that the malicious receiver's attack could be recognised by using some kind of context checks or tests. This is because the value $\tilde{x}_{nm}$ cannot be randomly chosen by the malicious receiver. Even so, we will continue to give an improvement to repair the security flaw in the following Section.

4 Improvement of the Zhang scheme

This Section will show a heuristic way to repair the aforementioned security flaw. Similar to the Zhang scheme, we assume that the user B wants to send a message to a user A. B executes the following steps.

Step 1: Translate the message into an $n \times m$ plaintext matrix $\boldsymbol{X}$:

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1m} \\ x_{21} & x_{22} & \dots & x_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nm} \end{bmatrix}$$

Step 2: Construct another $(n+1)\times(m+1)$ matrix $\boldsymbol{X}_h$:

$$\boldsymbol{X}_h = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1m} & X_1 \\ x_{21} & x_{22} & \dots & x_{2m} & X_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nm} & X_n \\ X^1 & X^2 & \dots & X^m & Z \end{bmatrix}$$

where $X_i = \prod_{j=1}^{m} x_{ij} \bmod N_B$, $1 \le i \le n$, and $X^j = \prod_{i=1}^{n} x_{ij} \bmod N_B$, $1 \le j \le m$, $Y_i = \sum_{j=1}^{m} x_{ij} \bmod N_B$, $1 \le i \le n$, and $Y^j = \sum_{i=1}^{n} x_{ij} \bmod N_B$, $1 \le j \le m$, and

$Z = H(Y_1, \ldots, Y_n, Y^1, \ldots, Y^m, X_1, \ldots, X_n, X^1, \ldots, X^m)$, and $H(\,)$ is a one-way hash function [4].

Step 3: Generate the signature Z_c,

$$Z_c = Z^{d_B} \bmod N_B$$

Step 4: Construct an $(n+1) \times (m+1)$ ciphered matrix $\boldsymbol{C}_h^*$. B first computes the following ciphertexts.

$$\begin{aligned} C_i^* &= X_i^{e_A} \bmod N_A \quad 1 \le i \le n \\ C^{j*} &= (X^j)^{e_A} \bmod N_A \quad 1 \le j \le m \\ Z_c^* &= Z_c^{e_A} \bmod N_A \end{aligned}$$

Then, B constructs $\boldsymbol{C}_h^*$ as follows:

$$\boldsymbol{C}_h^* = \begin{bmatrix} x_{11} \oplus H(Z_c, 1, 1) & x_{12} \oplus H(Z_c, 1, 2) & \ldots \\ x_{21} \oplus H(Z_c, 2, 1) & x_{22} \oplus H(Z_c, 2, 2) & \ldots \\ \vdots & \vdots & \vdots \\ x_{n1} \oplus H(Z_c, n, 1) & x_{n2} \oplus H(Z_c, n, 2) & \ldots \\ C^{1*} & C^{2*} & \ldots \end{bmatrix} \begin{bmatrix} x_{1m} \oplus H(Z_c, 1, m) & C_1^* \\ x_{2m} \oplus H(Z_c, 2, m) & C_2^* \\ \vdots & \vdots \\ x_{nm} \oplus H(Z_c, n, m) & C_n^* \\ C^{m*} & Z_c^* \end{bmatrix}$$

$$= \begin{bmatrix} c_{11} & c_{12} & \ldots & c_{1m} & C_1^* \\ c_{21} & c_{22} & \ldots & c_{2m} & C_2^* \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ c_{n1} & c_{n2} & \ldots & c_{nm} & C_n^* \\ C^{1*} & C^{2*} & \ldots & C^{m*} & Z_c^* \end{bmatrix}$$

where $\oplus$ denotes XOR operation.

Step 5: Transmit $\boldsymbol{C}_h^*$ to A.

Once the user A receives the ciphered matrix $\boldsymbol{C}_h^*$, he/she decrypts Z_c^* and all checksums by using his/her own private key as follows:

$$\begin{aligned} Z_c &= Z_c^{*d_A} \bmod N_A \\ X_i &= (C_i^*)^{d_A} \bmod N_A \quad 1 \le i \le n \\ X^j &= (C^{j*})^{d_A} \bmod N_A \quad 1 \le j \le m \end{aligned}$$

Then, A proceeds the bitwise XOR operations of c_{ij} with the hash values of Z_c and the indexes of the matrix to obtain the plaintext matrix $\boldsymbol{X}_h$:

$$\boldsymbol{X}_h = \begin{bmatrix} x_{11} \oplus H(Z_c, 1, 1) & x_{12} \oplus H(Z_c, 1, 2) & \ldots \\ x_{21} \oplus H(Z_c, 2, 1) & x_{22} \oplus H(Z_c, 2, 2) & \ldots \\ \vdots & \vdots & \vdots \\ x_{n1} \oplus H(Z_c, n, 1) & x_{n2} \oplus H(Z_c, n, 2) & \ldots \\ X^1 & X^2 & \ldots \end{bmatrix} \begin{bmatrix} x_{1m} \oplus H(Z_c, 1, m) & X_1 \\ x_{2m} \oplus H(Z_c, 2, m) & X_2 \\ \vdots & \vdots \\ x_{nm} \oplus H(Z_c, n, m) & X_n \\ X^m & Z_c \end{bmatrix}$$

$$= \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1m} & X_1 \\ x_{21} & x_{22} & \ldots & x_{2m} & X_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nm} & X_n \\ X^1 & X^2 & \ldots & X^m & Z_c \end{bmatrix}$$

Then, A verifies the validity of B's signature by computing:

$$\begin{aligned} Y_i &= \sum_{j=1}^{m} x_{ij} \bmod N_B \quad 1 \le i \le n \\ Y^j &= \sum_{i=1}^{n} x_{ij} \bmod N_B \quad 1 \le j \le m \end{aligned}$$

and checking

$$\begin{aligned} X_i &\stackrel{?}{=} \prod_{j=1}^{m} x_{ij} \bmod N_B \quad 1 \le i \le n \\ X^j &\stackrel{?}{=} \prod_{i=1}^{n} x_{ij} \bmod N_B \quad 1 \le j \le m \end{aligned}$$

If they are true, A computes $Z' = H(Y_1, \ldots, Y_n, Y^1, \ldots, Y^m, X_1, \ldots, X_n, X^1, \ldots, X^m)$, and checks $Z' \stackrel{?}{=} Z_c^{e_B} \bmod N_B$. If it is true, the signature is valid.

Similar to the scheme of Zhang [1], the improved scheme can be used not only for error detection but also for the correction of up to three errors. If errors have occurred in either the computation phase or the data transmission phase, then data correction method is the same as in the Zhang scheme.

5 Discussions

5.1 Security analysis

In this Section, we will discuss the security of the improved scheme and show that the improved scheme is secure from the attack outlined in Section 3.

A malicious attack receiver A will try to generate a different message for the existing signature. A will first choose $x_{11}, x_{12}, \ldots, x_{1,m-1}$ and then find a x_{1m}, which must satisfy the following two formulas:

$$\begin{aligned} x_{1m} &= X_1 \times \left(\prod_{j=1}^{m-1} x_{1j} \right)^{-1} \bmod N_B \\ x_{1m} &= Y_1 - \left(\sum_{j=1}^{m-1} x_{1j} \right) \bmod N_B \end{aligned}$$

However, it is very hard to find a x_{1m} to simultaneously satisfy the above two formulas. Similarly, it is hard for A to find $x_{2m}, x_{3m}, \ldots, x_{n-1,m}$. In addition, A also has to find a x_{n1}, so that:

$$\begin{aligned} x_{n1} &= X^1 \times \left(\prod_{i=1}^{n-1} x_{i1} \right)^{-1} \bmod N_B \\ x_{n1} &= Y^1 - \left(\sum_{i=1}^{n-1} x_{i1} \right) \bmod N_B \end{aligned}$$

This is also very difficult for A to find. Similarly, $x_{n2}, \ldots, x_{nm}$ have the same conditions. Thus, the attack outlined in Section 3 is infeasible under the improved scheme.

All elements in the plaintext matrix must proceed an XOR operation with the hash values of Z_c and the indexes of the matrix. Thus, if an attacker wants to view the content of the plaintext matrix, he/she has to first get Z_c. However, Z_c is enciphered by the receiver's RSA public key. To get Z_c is as difficult as breaking RSA.

If an attacker wants to generate a valid signature for any message, he/she follows the signature generation process to construct a plaintext matrix and checksums. Then, he/she uses those checksums and the hash function to compute a Z. To compute Z_c from Z, the attacker has to either know the sender's private key or break the RSA system. So, it is hard to succeed.

In order to prevent known-plaintext attack and chosen-plaintext attack, we encrypt the plaintext matrix by using the hash values of Z_c and the indexes of the matrix. A malicious attacker may guess a part of the message, x_{11}, and use it to find some hash values, like $H(Z_c, 1, 1) = c_{11} \oplus x_{11}$. However, it cannot find other elements in the plaintext matrix. This is because the malicious attacker cannot reveal Z_c from $H(Z_c, 1, 1)$ or find the values of $H(Z_c, 1, 2)$ and $H(Z_c, 2, 1)$. We also note that the length of N_B may be larger than the length of the hash value. For example, N_B is 1024 bits and the hash value is 64 bits long. Then, the hash value is iterated 16 times to form a full 1024-bit block. The ciphertext will be computed as $c_{11} = x_{11} \oplus (H(Z_c, 1, 1) \| H(Z_c, 1, 1) \| \cdots \| H(Z_c, 1, 1))$, where $\|$ means concatenation.

5.2 Performance analysis

In this Section, we will analyse the performance of the Zhang scheme and the improved scheme. Table 1 shows the number of modular exponentiations and hash function computations required by the Zhang scheme and the improved scheme.

According to Table 1, the improved scheme is obviously more efficient than the Zhang scheme.

Table 1: Computational complexity of the Zhang scheme and the improved scheme

	The Zhang scheme	The improved scheme
Number of modular exponentiations (sender)	$(n+1) \times (m+1)$	$n+m+2$
Number of modular exponentiations (receiver)	$(n+1) \times (m+1)$	$n+m+2$
Number of hash function computations (sender)	0	$n \times m+1$
Number of hash function computations (receiver)	0	$n \times m+1$

6 Conclusions

Zhang combined the concepts of data correction and digital signature to propose a fault-tolerant scheme that is not only for error detection but is also capable of correcting up to three errors. It also allows the receiver to check the sender's identity. However, we have shown that a malicious receiver can forge a different message for an existing signature. The sender thus cannot deny having signed the forged message. We also proposed an improved scheme to withstand the attack. The efficiency of the improved scheme is much better than the Zhang scheme.

7 Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. This work was supported in part by the National Science Council of the Republic of China under contract NSC91-2213-E218-012.

8 References

1 ZHANG, C.N.: 'Integrated approach for fault tolerance and digital signature in RSA', *IEE Proc., Comput. Digit. Tech.*, 1999, **146**, (3), pp. 120–126
2 RIVEST, R.L., SHAMIR, A., and ADLEMAN, L: 'A method for obtaining digital signature and public-key cryptosystems', *Commun. ACM*, 1978, **21**, (2), pp. 120–126
3 STALLINGS, W.: 'Cryptography network security: principles and practice' (Alan Apt, New Jersey, 1999, 2nd edn.)
4 MERKLE, R.C.: 'A fast software one-way hash function', *J. Cryptol.*, 1990, **3**, (1), pp. 43–58