How to Implement Cost-Effective and Secure Public Key Cryptosystems

Pil Joong Lee[1], Eun Jeong Lee[2], and Yong Duk Kim[3]

[1] Dept. of Electronics and Electrical Eng., POSTECH, Pohang, Korea
[2] POSTECH Information Research Laboratories, Pohang, Korea
[3] Penta Security Systems Inc., 23-3 Yoido-dong, Yongdeungpo-ku, Seoul, Korea
{pjl,ejlee}@postech.ac.kr, kyds@penta.co.kr

Abstract. The smart card has been suggested for personal security in public key cryptosystems. As the size of the keys in public key cryptosystems is increased, the design of crypto-controllers in smart cards becomes more complicated. This paper proposes a secure device in a terminal, "Secure Module", which can support precomputation technique for Schnorr-type cryptosystems such as Schnorr [7], DSA [1], KCDSA [5]. This gives a simple method to implement secure public key cryptosystems without technical efforts to redesign a cryptographic controller in $25mm^2$ smart card ICs.

Keywords : Secure module in terminal, smart cards, precomputation, digital signature/identification, public-key cryptosystem

1 Introduction

Public key cryptography proposed by Diffie and Hellman in 1976 allowed users to communicate securely without sharing secret information beforehand. These techniques can provide secure communication in today's open systems with privacy and message authentication. With public key techniques, a party has pairs of keys: a private key that is known only to the party and a public key available to all other users. A certificate for a public key is issued to a user so that any other party can verify the owner of the public key.

The user's personal security depends on the management of the private key. Not only must the private key be stored in a secure memory, but also the computation using the private key must be performed in a secure device. Smart cards, chip-embedded plastic cards with an MCU, make this possible with their tamper-resistant silicon chips. To store user-specific data securely against adversary, the EEPROM (Electrically Erasable Programmable Read Only Memory) in the chip is used.

The crypto-controller that uses the MCU and an additional arithmetic coprocessor for modular arithmetic provides the secure computation using the private key.

However, it is difficult to implement a secure and efficient public key cryptosystem due to the size of chip allowed in the smart card which is specified by

Ç.K. Koç and C. Paar (Eds.): CHES'99, LNCS 1717, pp. 73–79, 1999.

International Organization for Standardization (ISO) standard 7816. The parameters such as modulus and base integer in Schnorr-type protocol and the private key in RSA-type protocol are at least 128 bytes long, and may be more than 300 bytes, depending on the algorithm and security required [8]. Therefore, the smart card needs an additional arithmetic coprocessor optimized for modular exponentiation of long operands working on an 8-bit MCU [6]. Since the mechanical stability requirements of ISO standard 7816-1 limit the maximum chip size of the smart card ICs to 25 mm^2, the additional coprocessor for cryptographic algorithms must take up the space typically occupied by nonvolatile memory, specifically EEPROM.

Since the modular exponentiation of long operands is a much time-consuming job, Schnorr proposes to use precomputation for exponentiation using secret random numbers in idle time [7]. This idea can be applied to every schemes based on the discrete logarithm problem such as DSA [1], KCDSA [5], etc. With this precomputation, the signature generation can be computed by only one modular multiplication. However, since the ISO standard 7816 specifies the smart card is not supplied with electric power during its idle time, the precomputation technique cannot be applied to the smart card. In this paper, we propose a concept of “Secure Module (SM)” to design an efficient and secure public key cryptosystem without additional crypto-coprocessor. A normal smart card can be used as a SM when located in a terminal. With this SM and precomputation technique, we can efficiently generate Schnorr-type signatures.

Section 2 explains the environment of the proposed system and section 3 gives an example on how to implement the proposed system with Schnorr scheme. Then we consider the security of our proposed system in section 4.

2 Environment of the Proposed System

A public key cryptosystem needs a secure place to store the user’s private key. The nonvolatile memory, EEPROM, of the smart card is an appropriate space for the lengthy private key. To compute the exponentiation of a secret random number in signature/identification, we propose to outfit the terminal with a tamper-resistant device. We will call this device a Secure Module (SM).

The hardware specification is depicted in Fig. 1 and the requirements of the smart card (SC) and the terminal with secure module are as follows:

Smart Card (SC)

A normal IC card with 8-bit MCU, ROM, RAM, EEPROM, and COS (Card Operating System) according to ISO standard 7816 satisfies the hardware specification of the SC. The user data and cryptographic functions to be stored in EEPROM are as follows:

encryption algorithm, $E(\,)$ **:** symmetric key ciphering algorithm used in sending encrypted private key to SM.

random number generator, $r_1(\,)$ **:** algorithm to generate a secure random number used in mutual authentication with SM.

identifier of the SC, ID_{SC} **:** identifier of the smart card.
secret key, K_{SC} **:** symmetric key for $E(\ )$ used in mutual authentication. This is issued when the card is issued for a user and depends on the ID_{SC} and the master key K_M stored only in SM.
private key, X **:** user's private key for public-key cryptosystem.
certificate for public key, C **:** certificate for user's public key corresponding to the private key X.

The certificate C is optional, because, in some system, other users can obtain C from the trusted third party (TTP).

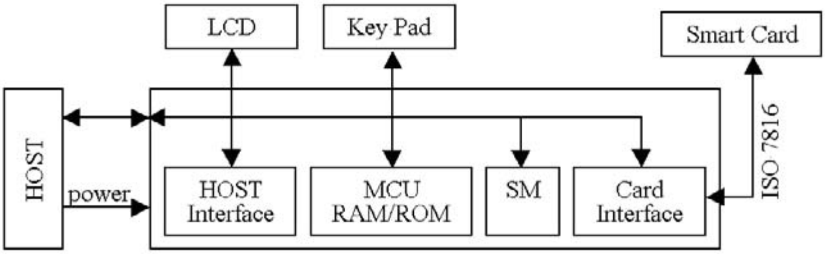

Fig. 1. The hardware specification of terminal with SM

Hardware Specification of Terminal

interface : LCD, keypad, host interface (RS232, modem, floppy, etc), and card interface (ISO 7816)
microprocessor
memory : RAM, ROM
secure module

Secure Module (SM)

A normal IC card with 8-bit MCU, ROM, RAM, EEPROM, and COS satisfies the hardware specification of SM. The user data and cryptographic functions to be stored in EEPROM are as follows:

encryption algorithm, $E(\ )$ **:** symmetric key ciphering algorithm.
random number generator $r_2(\ )$ **:** algorithm generating a secure random number used in mutual authentication with the SC and signature/identification schemes.
modular arithmetic functions : executable programs of modular multiplication, modular addition, and modular reduction.
precomputation functions : executable programs to precompute modular exponentiations for secret random numbers.
master key, K_M **:** symmetric key for $E(\ )$ used in mutual authentication. This is issued securely when the terminal is initialized.

3 Implementations of a Cryptographic Protocol

Every signature/identification scheme based on discrete log problem computes $W = g^k \pmod p$ for some random number $k \in \{1, 2, ..., q-1\}$, where p is a large prime and g is an element in $\{2, ..., p-1\}$ with order q, large prime divisor of $p-1$. Since k does not depend on the message to be signed or the identifier to be authenticated, these values can be precomputed in idle time, i.e. when the card is not in use. However, since the smart card is not supplied with power when the card is not in use, if there is the executable program of precomputation in smart card then we can not preprocess the exponentiation. Thus, it is the only time when the program of this algorithm is stored in the SM located in the terminal that we can preprocess the exponentiation.

An explanation for the Schnorr scheme is described below [7]:

Schnorr Signature Scheme for Message M:

(a) Choose a random number k in $\{1, ..., q-1\}$ and compute $W = g^k \pmod p$.
(b) Compute the first signature R, where $R = h(W||M)$, where $h(\ )$ is a collision-resistant hash function.
(c) Compute the second signature S, where $S = k + X \cdot R \pmod q$.

Since (a) does not depend on the message, this step permits precomputation of the quantities needed to sign the next messages. We propose to precompute (k, W) values in the SM during idle time of the terminal. Since the SM is tamper-resistant, the random k's can be stored securely and the private key X saved in SC cannot be exposed. When the message is signed, with W precomputed, only one modular multiplication and one modular addition are needed. Since (c) uses the user's private key, the SM must bring the private key securely from the SC. A protocol of the Schnorr scheme using the proposed system is described below. This protocol is depicted in Fig. 2.

Protocol to Compute a Signature Using Schnorr Scheme:

Step 0. *Precompute values of* (k, W) *in SM.*
If there is any empty place in storage, pick a random number k in $\{1, ..., q-1\}$ and compute $W = g^k \bmod p$ and store the pair (k, W). Otherwise, wait for an interrupt signal that is sent by the terminal when a message arrives. If an interrupt signal arrives during the precomputation, SM quits the precomputation and dumps the intermediate result.

Step 1. *Establish mutual authentication and share a session key between the SC and the SM.*
The SC and the terminal must verify whether the other party is legitimate by carrying out mutual authentication algorithms. If the SC sends the ID_{SC} to the SM, then the SM generates the K_{SC} using ID_{SC} and K_M in a specified method. With this shared symmetric key K_{SC} and encryption algorithm $E(\ )$, the SC and the SM mutually authenticate and share a session key K_S using random numbers as in [2,3].

Step 2. *Transfer the encrypted private key from the SC to the SM.*
The SC encrypts the private key X with the session key K_S, denote $E_{K_S}(X)$, and sends the ciphertext to the SM. The SM decrypts the ciphertext $E_{K_S}(X)$ with K_S, and gets the private key X.

Step 3. *Compute signature values in SM.*
The SM computes signature R and S using X, secret random number k, precomputed value W and message M as follows:

Step 3-1. Retrieving a precomputed pair (k, W).
Step 3-2. Compute $R = h(W||M)$.
Step 3-3. Compute $S = k + X \cdot R \pmod q$.
Step 3-4. Output (R, S) and erase the used pair (k, W).

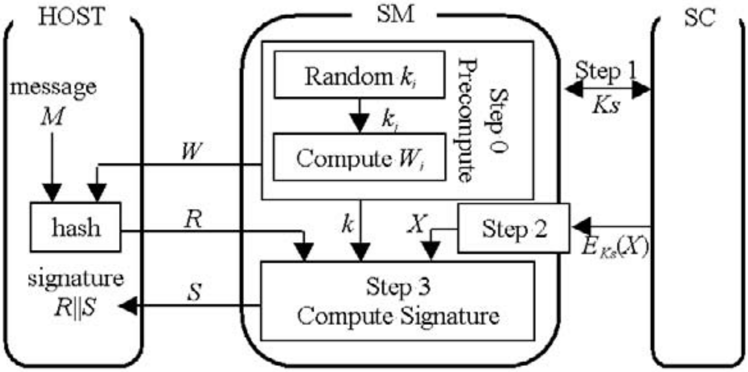

Fig. 2. The protocol to compute Schnorr-signature using SM

When the message is long, the hashing in Step 3-2 may be operated in the HOST like Fig. 2, because the channel between the terminal and the host has generally limited bandwidth.

We implemented this protocol with the digital signature algorithm, KCDSA [5] using the SM in a terminal with MCU 80C320 and the SC with Hitachi H8/3102 (8Kbyte EEPROM, 16Kbyte ROM, 512byte RAM). The cryptographic programs (modular operations, precomputation functions and random number generator) were coded using an 8051 assembler and their executable programs were stored in the EEPROM of the SM. We adapted algorithm [4] for exponent computation and this algorithm takes 30 seconds for one exponentiation, which is rather slow. However, thanks to the precomputation of this time-consuming job, the elapsed time to compute a digital signature is one second. Therefore, we could efficiently get a Schnorr-type digital signature of without using highly complex techniques for ICs.

4 Security Consideration

In our proposed system, each terminal has its own SM and all SM have the same master key K_M in common. The master key is issued when the terminal is initialized. Let us assume that an adversary knows the master key K_M. Since every one can get the identifier ID_{SC} of any smart card easily, the adversary can compute the session key K_{SC} shared between the smart card and the SM. In addition, if he succeeds in tapping or monitoring the transferred bit stream between the smart card and the terminal during mutual authentication and session key (K_S) establishment, he can get the session key K_S. Eventually he can get the user's private key X from the encrypted private key $E_{K_S}(X)$. Unfortunately, this reason will allown an adversary who knows the master key K_M to get the private key X stored in any smart card even though the private key is stored in the tamper resistant region of the smart card. Consequently, preventing the master key form being revealed is very important in our proposed system.

However, if there is no problem in initializing a terminal, i.e, the master key is stored in the tamper resistant region of the terminal without being revealed, and if computing of K_{SC} from the smart card identifier ID_{SC} and master key K_M is performed in tamper resistant region of SM, then we believe there will be no serious security hole in our system.

Even if an adversary knows the smart card identifier ID_{SC} but not the master key K_M, the attacks using differential analysis, timing attack or fault attack nearly do not work because the adversary knows only the values ID_{SC}, $E_{K_S}(X)$, $W = g^k \bmod p$, $R = h(W||M)$ and $S = k + X \cdot R \bmod q$.

5 Conclusion

In this paper, we proposed to use a secure device in a terminal, "Secure Module", which can support precomputation technique for Schnorr-type cryptosystem. This gives an easy method to implement personal security without technical efforts to design cryptographic controller in $25mm^2$ smart card ICs. We could get KCDSA digital signature in one second using the proposed system.

References

1. NIST, Digital Signature standard, FIPS PUB 186, `http://csrc.nist.gov/cryptval/dss.htm`, 1994.
2. ISO/IEC IS 9798-2 Information technology - Security techniques - Entity Authentication, part 2: Mechanisms using a symmetric encipherment algorithm, 1995.
3. ISO/IEC IS 11770-2 Information technology - Security techniques - Key Management, part 2: Mechanisms using a symmetric techniques, 1996.
4. Chae Hoon Lim and Pil Joong Lee, "More flexible exponentiation with precomputation," *Advances in Cryptology-Crypto'94*, LNCS 839, Springer-Verlag, pp.95-107, 1994.

5. Chae Hoon Lim and Pil Joong Lee, "A study on the proposed Korean digital signature algorithm," *Advances in Cryptology-Asiacrypt'98*, LNCS 1514, pp.175-186, 1998.
6. David Naccache and David M'Raïhi, "Cryptographic Smart Cards," *IEEE Micro*, Vol. 16, No. 3, pp.14-24, 1996.
7. C. P. Schnorr, "Efficient signature generation by smart card," *J. of Cryptology*, vol. 4, pp.161-174, 1991.
8. A.M. Odlyzko, "The future of integer factorization," *Cryptobytes*, Vol. 1, No.2, pp. 5- 12, 1995.