FAST IMPLEMENTATION OF DIGITAL SIGNATURE ALGORITHMS ON SMARTCARDS WITHOUT COPROCESSOR

Chung-Huang Yang, Ph.D.
National Kaohsiung Normal University
116, Ho Ping 1st Road, Kaohsiung, TAIWAN 802
chyang@computer.org

Hikaru Morita, Dr. Eng.
NTT Service Integration Laboratories
3-9-11 Midori-cho, Musashino-shi, Tokyo, JAPAN 180-8585
morita.hikaru@lab.ntt.co.jp

Tatsuaki Okamoto, Dr. Eng.
NTT Information Sharing Platform Laboratories
1-1 Hikarino-Oka, Yokosuka, Kanagawa, JAPAN 239-0847
okamoto@sucaba.isl.ntt.co.jp

Abstract

The advancement of information and communications technology, especially the Internet, has created an opportunity to improve the administrative efficiency and service quality in governments of many nations. However, due to the lack of communication security services, sensitive documents could not be transmitted securely over open networks using off-the-shelf software. Digital signature is by far one of the most important cryptographic techniques used in the e-government and e-commerce applications. It provides authentication of senders or receivers (they are who they claim to be) and offers non-repudiation of transmission (senders can't deny their digital signature in the signed documents and the document cannot be altered in transmission without being detected). This paper presents our efforts in the implementation of digital signature algorithms on smartcards without coprocessor. New arithmetic algorithms are proposed and implemented for this research. Besides, we evaluated the performance of well-known RSA and ESIGN digital signature algorithms on an 8-bit smart card.

Smartcard [1-2], or smart card, is a plastic card with an embedded microcomputer chip. The dimension of a smartcard is the size of an ordinary credit card, which can be easily carried around in a wallet. Smartcards are much more difficult to duplicate than magnetic strip cards and cryptographic functions can be implemented inside these cards. With substantial cost reductions and the ability to handle multiple applications on a single card, the smartcards are about to enter a period of the rapid growth. An individual bearing a single smartcard will be able to electronically and securely interact with several servers or service provides. As a consequence, an entirely new type of commercial and educational landscape is being created. However, smartcard itself has limited computing power and memory capacity. This makes it a difficult job to efficiently implement the cryptographic functions inside smartcard.

One of the e-government's [3] objectives is to facilitate the exchange and integration of information between different agencies and the Internet is being used as the communication channel to exchange information between all sectors of society. However, due to the lack of communication security services and the export control of U.S.A. and many nations, sensitive information could not be securely transferred between and within governmental agencies over the Internet using off-the-shelf software. Here, the security services [4] mean data confidentiality, authentication, access control, data integrity, and non-repudiation.

Cryptography [5-6] is the only practical means for providing security services over an insecure channel such as Internet. The increasing use of electronic means of data communications, coupled with the growth of computer usage, has extended the need to protect information. Considerable progress has been made in the techniques for encryption, authentication, and fending off attacks from intruders over the last decade. Nevertheless, the impose of export controls on those computer software or hardware devices has precluded the use of secure products or has made such imported products very expensive.

In the public-key cryptography [4] or public-key scheme, each entity has a pair of public key and corresponding private key; private key shall be kept secretly at every entity and could be individually stored at the built-in EEPROM area of the smartcard while the public key is openly available to each other entity. Public-key infrastructures (PKI) [6] are comprised of supporting services that are needed for using public-key technologies on a large scale and are widely adopted in the e-government projects worldwide. Digital signature is a core technique in the PKI.

Digital signature [5-6] is by far one of the most important cryptographic techniques employed in the e-government and e-commerce applications. A digital signature algorithm allows an entity to use its private key to electronically sign a message and generate a signature that is dependent on both message itself and the entity's private-key information. The computed signature is then attached to the message and sent with the message. The signature verification process could be performed by any receiving party and cannot be repudiated. Recipient could verify the signature by doing some computation involving the received message, the attached signature, and sender's public key. If the results properly hold in a predefined mathematical relation, the signature is accepted as genuine. Otherwise, the signature may be fraudulent or the message altered.

In this paper, we carry out the implementation of digital signature algorithms on smartcards. New arithmetic algorithms are proposed and implemented and we also evaluated the performance of well-knows RSA [7] and ESIGN [8] digital signature algorithms on the Hitachi 8-bit smart cards. In order to understand the cost-effectiveness of different smart cards, we evaluated digital signature schemes on the low-cost 8-bit chip without using the coprocessor.

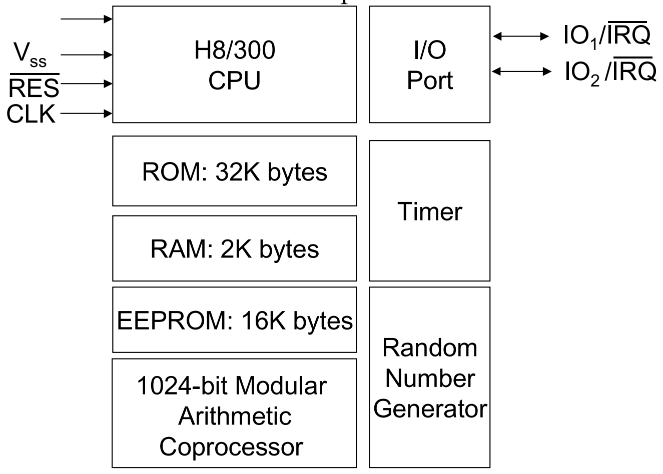
The 8-bit H8/300 Smartcards

The smartcard IC chip we adopted for implementing and evaluating digital signature algorithms is the Hitachi H8/3113 [9]. It contains an 8-bit H8/300 microprocessor, 32-Kbyte ROM, 16-Kbyte EEPROM, 2.5K-byte RAM, an arithmetic coprocessor, a random number generator, and a watchdog timer. The on-chip coprocessor could be used to calculate $A^B \; mod \; N$ modular exponentiation at a high speed. Figure 1 shows block diagram of the Hitachi H8/3113 chip.

The ROM area of the chip is used for chip operating system while the EEPROM is used for personalized data and for storing cryptographic keys with data retention time of 10 years and rewrite endurance of 100,000 times. The chip also has many built-in protection measures to prevent physical attack. These measures include high/low frequency detector, high/low voltage detector, concealment of ROM code, scattered layout, multi-metal layer, removal of test pin, etc.

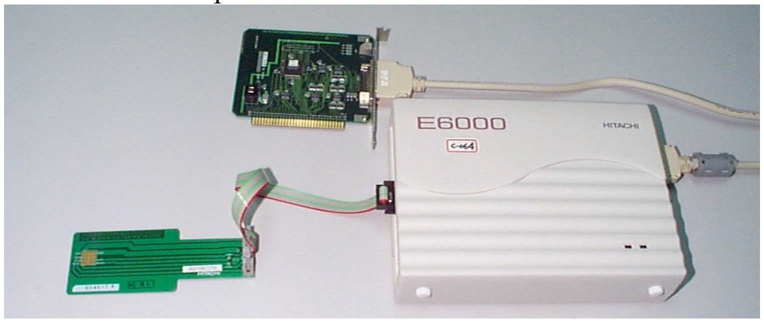
The H8/3113 is a powerful 8-bit microcomputer suitable for the implementation of both symmetric and asymmetric cryptographic algorithms [5]. With built-in random number generator and coprocessor, one could do high-speed key generation for most public-key cryptographic algorithms. Its functional drawback is in low input/output rate of smart card since only two I/O pins are available. The device can operate at a maximum of 10 MHz external clock rate or a cycle time of 200 ns.

Figure 1. Block diagram of H8/3113 smartcard chip



The H8/300 CPU contains 14 general-purpose registers of 8-bit each. It has 8 addressing modes and 55 basic instructions with most instructions are executed in 2 to 4 states. Basic instruction set contains multiplication, division, and some instructions, such as read/write from RAM, which could handle 16-bit operands as fast as 8-bit operands. The vendor provides a real-time in-circuit emulator under Windows environment that can support the Hitachi H8/3113 smart card devices. It may be used totally self-contained for software development and debug, or connected to a smart card reader for real time I/O debugging. Figure 2 shows the hardware emulator.

Figure 2. H8/3113 hardware development tools



The RSA and ESIGN Digital Signature Algorithms

The RSA [7] digital signature scheme was first published in 1978. In this scheme, each entity first chooses two large prime numbers. Let's denote as P and Q, as *private key*. Then a public exponent E is chosen such that GCD(E,(P-1)(Q-1))=1 is satisfied, which means the greatest common divisor (GCD) of positive integer E with (P-1)(Q-1) equals to one. Now, we apply the extended Euclidean algorithm [10] to find a positive integer D satisfies

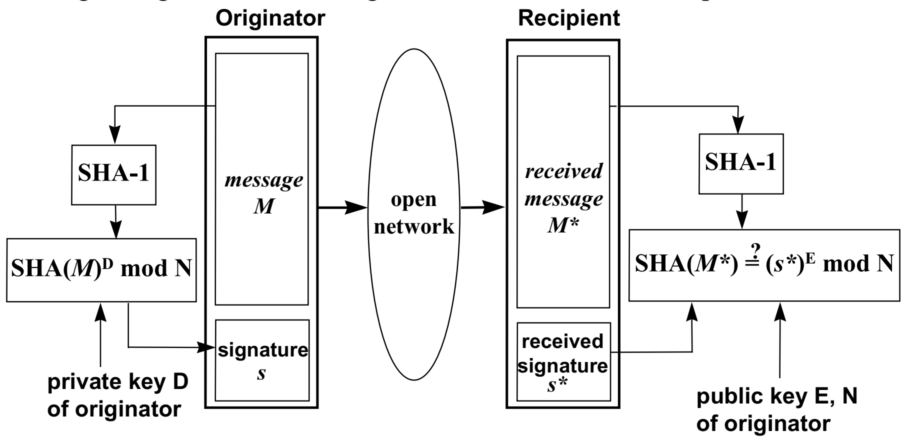
$$D \times E = 1 \bmod (P-1)\times(Q-1) \quad or \quad D = E^{-1} \bmod (P-1)\times(Q-1)$$

where *mod* denotes the modular arithmetic [10]. The *public key* of entity is E and N $(= P\,.\,Q)$

while the private key is *D and P and Q;* private key might be stored inside smartcards and is needed for generating digital signature while public key is needed for verifying digital signature. Security strength of RSA scheme is related to the discrete logarithm problem and the factorization problem [5]. The size of private key must be properly chosen. If it is too small, it will enable attackers to crack the system easily [11]. If it is too large, it will degrade the signature generation and verification performance.

In practice, a message digest algorithm, such as the NIST's SHA-1 [12], is used with the RSA algorithm for signature generation and signature verification, as shown in Figure 3. The *message digest* [5] algorithm is an irreversible function, so-called *one-way hash function*, which takes an arbitrary sized message and output a fixed length hash value. Therefore, instead of directly applying RSA digital signature scheme on a long message, we could efficiently sign on the message's hash value that is 160 bits for SHA-1 hash function, signature $S = SHA(M)^D \bmod N$.

Figure 3. RSA digital signature for message authentication and non-repudiation



ESIGN (abbreviation for Efficient digital SIGNature) digital signature scheme was first proposed in 1990 and is now a part of the ISO 14888 international standard [13]. In this scheme, each entity chooses two large prime numbers. Let's also denote as *P* and *Q* but the *public key* of entity is P^2Q (= N) while the private key is *P* and *Q*. A public security parameter *E* also is needed, $8 \leq E$ with recommendation of $E=1024$. To generate a signature on a message *M*; first you need to pick a random number *R*, $0 \leq R \leq PQ-1$. Let the bit length of *P* be *k*, then we compute the signature *S* by the following equations:

$$W_0 = \left\lceil \frac{\left(0\|H(M)\|0^{2k}\right) - R^E \bmod N}{PQ} \right\rceil$$

$$T = W_0 \times \left(E \times R^{E-1}\right)^{-1} \quad \bmod P \quad ,$$

$$S = R + T \times PQ \bmod N$$

where $\lceil \ \rceil$ denotes the ceiling function, *H(M)* is a *k-1* bit hash value, and $\left(0\|H(M)\|0^{2k}\right)$ is a *3k*-bit value obtained by putting zero as the most-significant bit and 2k-bit zeroes as the least significant bits. Signature verification process involves the computation of $S^E \bmod N$ and check for the equality of the most significant k+1 bits of $S^E \bmod N$ with *(0||H(M)||0)*. That is, you check for whether the equation $(0\|H(M)\|0) \overset{?}{=} \left[S^E \bmod N\right]^{k+1}$ holds or not.

Multiple-precision modular arithmetic [10] is required at both RSA and ESIGN digital signature algorithms. However, most general-purpose smartcards feature limited RAM/ROM and slow 8-bit CPU that make them traditionally unsuitable for public-key cryptosystems or digital signature schemes aimed at real-time applications. While there have been enormous publications (see, for example, [10]) on modular arithmetic algorithms, few of them considering the smartcard environment. On smartcards, RAM area is usually only 2K bytes or less and this represents a major implementation constrain.

In the following, we describe three modular arithmetic algorithms: modular exponentiation, modular multiplication, and modular inverse, which are required for implementing RSA and ESIGN digital signature scheme on the H8/300 general-purpose smartcards. Figure 4 shows the modular exponentiation algorithm. We scan 2 bits of exponent at once in Figure 4 and more performance improvement can be achieved if more bits are processing at once but more program ROM size and data RAM size would be required.

Figure 4. Modular exponentiation algorithm for 8-bit smartcards

```
Given:   $M, E, N$        $0 \leq M < N < R = 256^n$
         Let PROD(A, B) indicate the operation of modular multiplication,
         PROD(A, B)= $A * B \mod N$
Find: $C$,  $C = M^E \mod N$
         Assume that binary presentation of exponent $E$ is
                  $e_{2t-1}e_{2(t-1)}\cdots e_{2j+1}e_{2j}e_{2j-1}\cdots e_1e_0$ ,
Solution:
   Step 1. Pre-compute    $M^3$ = PROD(PROD (M, M), M)
           Initialize     flag $\leftarrow 0$
   Step 2. for i = $t$ -1 to 0 do
              case ($e_{2i+1}e_{2i}$)
              '00': if flag = 1 do   $C \leftarrow$PROD($C$, $C$); $C \leftarrow$PROD ($C$, $C$)
              '01': if flag = 1 do   $C \leftarrow$PROD ($C$, $C$); $C \leftarrow$PROD ($C$, M)
                    else             $C \leftarrow$PROD ($C$, M); flag $\leftarrow 1$
              '10': if flag = 1 do   $C \leftarrow$PROD ($C$, M); $C \leftarrow$PROD ($C$, $C$)
                    else             $C \leftarrow$PROD ($C$, M); $C \leftarrow$PROD ($C$, $C$); flag $\leftarrow 1$
              '11': if flag = 1 do   $C \leftarrow$PROD ($C$, $C$); $C \leftarrow$PROD ($C$, $M^3$)
                    else             $C \leftarrow$PROD ($C$, $M^3$); flag$\leftarrow 1$
```

The modular multiplication algorithm [14] is shown in Figure 5. Instead of using conventional method of multiplication following by division approach, a *lookahead determination* technique [14] was developed so as to reduce the RAM usage while it still provides excellent performance.

Figure 5. Modular multiplication algorithm for 8-bit smartcards

Given: $A, B, N,\ A = \sum_{j=0}^{n-1} A_j 2^{8j}\ ,\ B = \sum_{j=0}^{n-1} B_j 2^{8j}\ ,\ N = \sum_{j=0}^{n-1} N_j 2^{8j}$

$$0 \leq A, B < N\ ,\ \frac{2^8}{2} \leq N_{n-1} \leq 2^8 - 1$$

Find: $C = A * B \bmod N$

Solution:

```
Step 1.   $C \leftarrow 0$     (Byte length of $C$ is $n$+2)
          $i \leftarrow n$ -1
Step 2.   $C \leftarrow 2^8 C + A * B_i$
Step 3.   $q \leftarrow \left\lfloor \frac{2^8 C_{n+1} + C_n}{N_{n-1} + 1} \right\rfloor$
Step 4.   If $q = 0$, then   goto Step 9
          else $C \leftarrow C$ - $2^8 * q$
Step 5.   $q \leftarrow \left\lfloor \frac{2^8 C_{n+1} + C_n}{N_{n-1} + 1} \right\rfloor$
Step 6.   If $q = 1$, then   $C \leftarrow C$ - $2^8 * N$
          else if $q = 2$, then   $C \leftarrow C$ - $2^9 * N$
Step 7.   If $2^8 C_n + C_{n-1} \geq (N_{n-1} + 1)(2^8 - 1)$, then $C \leftarrow C$ - $(2^8\text{-}1) * N$
Step 8.   $i \leftarrow i$ - $1$
          If $i \geqq 0$, then   goto Step 2
Step 9.   If $C < N$, then return $C = \sum_{j=0}^{n-1} C_j 2^{8j}$
Step 10.  $q \leftarrow \left\lfloor \frac{2^8 C_n + C_{n-1}}{N_{n-1} + 1} \right\rfloor$
Step 11.  $C \leftarrow C$ - $q * N$
Step 12.  If $C < N$, then return $C = \sum_{j=0}^{n-1} C_j 2^{8j}$
Step 13.  $C \leftarrow C$ - $N$
          Goto Step 12
```

The proposed modular inverse algorithm, see Figure 6, is based on the extended Euclidean algorithm, but we adopt *approximation division* [10] instead of direct long division (see Step 2). Our experiences indicate that such an approach provides a 10 to 20 times performance improvement over the original extended Euclidean algorithm.

Figure 6. Modular inverse algorithm for 8-bit smartcards

```
    Given:    $A, N$
    Find:     $T$ such that $T * A \equiv 1 \bmod N$
  Solution:
    Step 1.   C $\leftarrow N$
              D $\leftarrow A$
              X' $\leftarrow 0$
              X $\leftarrow 1$
              counter $\leftarrow 0$
    Step 2.   If $D$ is more than on byte, then $Q \leftarrow \left\lfloor \frac{C}{D_{top} + 1} \right\rfloor$
                  ($D_{top}$ is the non-zero most-significant digit of $D$)
```

```
                else $Q \leftarrow \left\lfloor \frac{C}{D} \right\rfloor$
                If $Q = 0$, then $Q \leftarrow 1$
                $C$ (new)    $\leftarrow D$ (old)
                $D$ (new)    $\leftarrow C$ (old) - $Q * D$ (old)
Step 3.         $D = 0$, then    goto Step 5
Step 4.         $X'$ (new)   $\leftarrow$   $X$ (old)
                $X$ (new)    $\leftarrow$   $X'$ (old) + $Q * X$ (old)
                If ($C \leqq D$),    then swap $C$ with $D$, swap $X$ with $X'$
                                     else $counter \leftarrow counter$ +1
                Goto Step 2
Step 5.   If (counter = 1 mod 2), then return $N - X$
                                       else return $X$
```

Implementation of the Digital Signature Algorithms

The RSA and ESIGN algorithms were implemented on Hitachi's H8/3113 smartcards running at internal 5 MHz. Assembly codes for modular arithmetic algorithms and RSA and ESIGN digital signature algorithms were written and emulated using Hitachi's E6000 hardware development system. Total program size is about 3K bytes for each digital signature algorithm. Performance of our modular multiplication and modular inverse algorithms are summarized in Table 1 and Table 2.

Table 1. Performance of modular multiplication on H8/300 8-bit smartcards running at 5MHz

modular multiplication	
576 bits	90 ms
768 bits	150 ms
1152 bits	360 ms

Table 2. Performance of modular inverse on H8/300 8-bit smartcards

modular inverse	
192 bits	120 ms
256 bits	200 ms
384 bits	470 ms

Besides, the implementation complexity (in terms of code size) of an algorithm is as important as the algorithm performance on the smart card environment. In other words, we have to carefully evaluate the tradeoff between program size and performance. Table 3 shows the performance of 1152-bit RSA with E=65537 and ESIGN with E=1024 on H8/300 CPU running at 5MHz. The total RAM usage in the implementation is 466 bytes.

Table 3. Performance of RSA and ESIGN on H8/300 8-bit smartcards

Digital Signature Algorithm	1152-bit RSA	1152-bit ESIGN
Pre-computation	No applicable	6.0 s
Signature Generation	155 s	0.15 s
Signature Verification	6.12 s	3.7 s

For ESIGN, a pre-computation technique could be adopted to speed up signature generation. In this approach, we calculate two variables T_1 and T_2 in advance that are dependent on random number R while is independent on message M.

$$T_1 = R^E \bmod N$$

$$T_2 = \left(E \times R^E\right)^{-1} \times R \quad \bmod P$$

Then when message M is available (for example, when a document needed to be signed), we generate signature S by the following equations,

$$W_0 = \left\lceil \frac{\left(0 \| H(M) \| 0^{2k}\right) - T_1 \quad \bmod N}{PQ} \right\rceil$$

$$T = W_0 \times T_2 \quad \bmod P$$

$$S = R + T \times PQ \bmod N$$

By pre-computation, ESIGN could generate 1152-bit signature in less than 0.5 second without using coprocessor in 8-bit smartcards.

Conclusion and Future Work

Digital signature scheme is the core component of the public-key infrastructure (PKI) and is essential for the viability of e-government or e-commerce. In this paper, we have described our implementation efforts for 1152-bit RSA and ESIGN digital signature algorithms on general-purpose 8-bit smartcards. The results show that it takes less than 0.5 second to generate 1152-bit ESIGN signature on H8/3113 smartcards without a coprocessor while 1152-bit RSA signature takes more than 150 seconds.

Future work is to extend our current results to elliptic curve cryptosystems (ECCs). The use of ECCs was suggested independently by Neal Koblitz and Victor Miller in 1985. ECCs are new generation of cryptographic algorithms and have been accepted in the standardization bodies and recently adopted as ANSI X9.62, ISO 11770-3, IEEE P1363, and FIPS 186-2 standards. Compared to the RSA digital signature algorithm, ECC digital signature algorithm yields higher implementation speed in software since high security is realized even with shorter length keys. This makes ECC ideal for constrained environments such as smart cards.

References

[1] ISO 7816 Part 1 to 10: *Identification Cards – Integrated Circuit(s) Cards with Contacts*, 1987 to 2000.

[2] W. Rankl and W. Effing, *Smart Card Handbook*, 2nd edition, John Wiley & Sons, 2000.

[3] N. Adam, F. Artigas, V. Atluri, S. Chun, S. Colbert, M. Degeratu, A. Ebeid, V. Hatzivassiloglou, R. Holowczak, O. Marcopolus, P. Mazzoleni, W. Rayner and Y. Yesha, "E-Government: Human Centered Systems for Business Services," *The Proceedings of The First National Conference on Digital Government*, May 21-23, 2001.

[4] ISO 7498-2, "Information Processing Systems - Open Systems Interconnection - Basic Reference Model - Part 2: Security Architecture," 1989.

[5] A. J. Menezes, P.C. van Oorschot, and S.A. Vanstone, *Handbook of Applied Cryptography*, CRC Press Series on Discrete Mathematics and Its Applications, 1996. http://www.cacr.math.uwaterloo.ca/hac/

[6] Warrick Ford and Michael S. Baum, *Secure Electronic Commerce: Building the Infrastructure for Digital Signatures and Encryption*, Prentice Hall, 1997.

[7] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", *Communications of the ACM*, Feb. 1978, Vol. 21, No. 2, pp. 120-126.

[8] Tatsuaki Okamoto, "A Fast Signature Scheme Based on Congruential Polynomial Operations", *IEEE Trans. Info. Theory*, Vol. 36, pp. 47-53, January 1990. See also

http://info.isl.ntt.co.jp/esign/ or https://www.cosic.esat.kuleuven.ac.be/nessie/updatedPhase2Specs/esign/esign5.pdf

[9] *Hitachi Single-Chip Microcomputer H8/3113 Hardware Manual*, Hitachi Ltd., 1998. See also http://www.hitachisemiconductor.com/sic/jsp/japan/eng/solutions/ic_cards_solution/image/english/h8_3114e.pdf.

[10] Donald E. Knuth, *The Art of Computer Programming - Seminumerical Algorithms*, Vol. 2, Section 4.3, Addison-Wesley, 1981.

[11] The RSA Challenge Numbers, http://www.rsasecurity.com/rsalabs/challenges/factoring/numbers.html.

[12] National Institute of Standards and Technology, “Secure Hash Standard,” *Federal Information Processing Standard, FIPS PUB 180-1*, April 1995.

[13] ISO/IEC 14888-3, Information technology - Security techniques - Digital Signatures with Appendix-Part 3: Certificate-Based Mechanisms, Dec. 1998.

[14] Hikaru Morita and Chung-Huang Yang, "A Modular-Multiplication Algorithm using Lookahead Determination," *IEICE Trans. Fundamentals,* Vol. E76-A, No. 1, January 1993, pp. 70-77.

[15] ANSI X9.62-1998, Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA).