LETTER

Security and Performance Evaluation of ESIGN and RSA on IC Cards by Using Byte-Unit Modular Algorithms

Chung-Huang YANG[†a)], Hikaru MORITA[††], *and* Tatsuaki OKAMOTO[††], *Members*

SUMMARY 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 and offers non-repudiation of transmission (senders cannot deny their digital signature in the signed documents and the document cannot be altered in transmission without being detected). This paper presents our implementation results of digital signature algorithms on IC cards by using byte-unit modular arithmetic algorithm method [13], [20]. We evaluated the performance of well-known ESIGN and RSA digital signature algorithms on the dedicated IC card chips and showed that ESIGN is more efficient than RSA.

key words: *digital signature, ESIGN, PKI, smartcards, modular arithmetic*

1. Introduction

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.

IC cards [7], [17] 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 IC cards are about to enter a period of the rapid growth. An individual bearing a single IC card 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, IC card itself has limited computing power and memory capacity. This makes it a difficult job to efficiently implement the cryptographic functions inside IC card.

One of the e-government's [1] 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 [8] mean data confidentiality, authentication, access control, data integrity, and non-repudiation.

Cryptography [2], [11] 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 [11] 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 IC card while the public key is openly available to each other entity. Public-key infrastructures (PKI) [2] 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 [2], [11] 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 IC cards using byte-unit modular arithmetic algorithm method [13], [20]. New arithmetic algorithms are proposed and implemented and we also evaluated the performance of well-knows ESIGN [3], [14] and RSA [18] digital signature algorithms on the Hitachi

Manuscript received October 27, 2003.
Manuscript revised August 6, 2004.

[†] The author is with the Graduate Institute of Information and Computer Education, National Kaohsiung Normal University, 116, Ho-Ping 1st Road, Kaohsiung, 802 Taiwan.

[††] The authors are with NTT Laboratories, NTT Corporation, Yokosuka-shi, 239-0847 Japan.

a) E-mail: chyang@computer.org

DOI: 10.1093/ietcom/e88–b.3.1244

H8/300 8-bit IC cards. In order to understand the cost-effectiveness of different IC cards, we evaluated digital signature schemes on the low-cost 8-bit chip without using the crypto-coprocessor [4].

2. The 8-Bit IC Cards

The IC card device we adopted for implementing and evaluating digital signature algorithms is the Hitachi H8/3113 [5] IC card device. It contains an 8-bit H8/300 microprocessor, 32-kbyte ROM, 16-kbyte EEPROM, 2.5-kbyte RAM, an arithmetic coprocessor, and a random number generator. 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 device 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. 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.

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.

We wrote assembly codes to evaluate the performance of ESIGN and RSA algorithms in the IC card. Hardware emulator of IC card is used to real-time emulate performance of the assembly codes.

3. The ESIGN and RSA Digital Signature Algorithms

The RSA [18] digital signature scheme was first published in 1978. In this scheme, each entity first chooses two large prime numbers. Let P and Q, denote 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 $(P-1) \times (Q-1)$ equals to one. Now, we apply the extended Euclidean algorithm [10] to find a positive integer D satisfies

$$D = E^{-1} \, mod \, (P-1) \times (Q-1) \tag{1}$$

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 IC cards and is needed for generating digital signature while public key is needed for verifying digital signature. In practice, a message digest algorithm, such as the NIST's SHA-2 [16], is used with the RSA algorithm for signature generation and signature verification. Therefore, instead of directly applying RSA digital signature scheme on a long message, we could efficiently sign on the message's hash value, signature

ESIGN (abbreviation for Efficient digital SIGNature) digital signature scheme was first proposed in 1990 and is now a part of the ISO 14888-3 international standard [9]. ESIGN is also one of the crypto algorithms selected for the IEEE P1363a [6] standard. There are variants of ESIGN scheme with different security strength, from the classical digital signature algorithm to the provably secure against single occurrence chosen message attack (SO-CMA) and to the provably secure against chosen message attack (CMA). In this paper, we will use ESIGN to denote the core component of all ESIGN's variants where we are mainly concerned about the implementation issues. In the basic ESIGN scheme, each entity chooses two large prime numbers, let P and Q also denote private key but the *public key N* of entity is equal to P^2Q while the private key is P and Q. A public security parameter E (exponent) also is needed, $4 \leq E$. To generate a signature on a message M; first we need a (encoding) 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:

$$W_0 = \left\lceil \frac{\left(f \| 0^{2k}\right) \quad - R^E \, mod \, N}{P \, Q} \right\rceil \tag{2}$$

$$T = W_0 \times \left(E \times R^{E-1}\right)^{-1} \quad mod \, P \tag{3}$$

$$S = R + T \times PQ \tag{4}$$

where $\lceil x \rceil$ denotes the ceiling function for the largest integer less than x, f is a k-bit (hashed) message representative, and $\left(f \| 0^{2k}\right)$ is a *3k*-bit value obtained by putting 2k-bit zeroes as the least significant bits. Signature verification process involves the computation of S^E *mod N* and check for the equality of the most significant k bits of S^E mod N with the message representative f.

4. The Implementation of Digital Signature Algorithms on 8-Bit IC Cards

Multiple-precision modular arithmetic [10] is required at both ESIGN and RSA digital signature algorithms. However, most general-purpose IC cards feature limited RAM/ROM and slow 8-bit CPU that makes 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 IC card environment where RAM area is usually only

```
Given:    $M, E, N \underline{<} M < N < R = 256^n$
Find:     $C = M^{E}\ mod\ N$
Solution:
Let PROD(A, B) indicate the operation of modular
  multiplication, PROD(A, B)= $A * B$ mod $N$
Assume that binary presentation of exponent $E$ is
  $\{e_{2t-1}e_{2(t-1)}..e_{2j+1}e_{2j}e_{2j-1}...e_1e_0\}$ ,

Step 1.   Pre-compute
            M3 ←PROD(PROD(M, M), M)
          Initialize flag ← 0
Step 2.   for i = $t$ -1 to 0 do
          if flag = 1, then $C$ ←PROD($C$, $C$);
          case ($e_{2i+1}e_{2i}$)
          '00':  if flag = 1 do
                   $C$←PROD ($C$, $C$)
          '01':  if flag = 1 do
                   $C$ ←PROD ($C$, $C$);
                   $C$ ←PROD ($C$, M)
                 else
                   $C$ ←PROD ($C$, M); flag ←1
          '10':  if flag = 1 do
                   $C$ ←PROD ($C$, M);
                   $C$←PROD ($C$, $C$)
                 else
                   $C$←PROD ($M$, $M$); flag←1
          '11':  if flag = 1 do
                   $C$ ←PROD ($C$, $C$);
                   $C$ ←PROD ($C$, M3)
                 else
                   $C$←PROD ($C$, M3); flag←1
```

Fig. 1 Modular exponentiation algorithm for 8-bit IC cards.

2 kbytes 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 ESIGN and RSA digital signature scheme on the H8/300 general-purpose IC cards. Figure 1 shows the modular exponentiation algorithm, it is based on the well-known left-to-right 2-ary exponentiation. We scan 2 bits of exponent at once 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.

The modular multiplication algorithm is shown in Fig. 2. Instead of using conventional method of multiplication following by division approach, a *lookahead determination* technique [13] was developed so as to reduce the RAM usage while it still provides excellent performance.

Our modular multiplication algorithm requires only n+2 blocks of RAM, where n is the length of N in blocks for the calculation of $C \leftarrow A * B\ mod\ N$. The computation time is about twice the computation time needed for the multiplication of $A * B$. Montgomery algorithm is [12] the most common technique used in hardware implementation of many smart-card arithmetic coprocessors [4] and the speed of our algorithm is similar to that of the Montgomery algorithm [12] while our algorithm requires much small amount of RAM making it very suitable for firmware implementation on smart cards [13].

```
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 = $A * B$ mod N
Solution:
Step 1.   C ← 0 (Byte length of C is $n$+2)
          $i \leftarrow n$ -1
Step 2.   C ← $2^8$ 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 ← 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 ← C - $2^8 * N$
          else if $q$ = 2, then C← C - $2^9 * N$
Step 7.   If $2^8 C_n + C_{n-1} \geq (N_{n-1} + 1)\left(2^8 - 1\right)$,
          then C ← C - ($2^8$-1) $* N$
Step 8.   $i \leftarrow i$ - 1
          If $i \geq 0$, 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 ← C - $q * N$
Step 12.  If C < N, then return C = $\sum_{j=0}^{n-1} C_j 2^{8j}$
Step 13.  C ← C - N, Goto Step 12
```

Fig. 2 Modular multiplication algorithm [13] for 8-bit IC cards.

```
Given:    $A, N$
Find:     $T$ such that $T * A \equiv 1\ mod\ N$
Solution:
Step 1.   C ← $N$
          D ← $A$
          X' ← 0
          X ← 1
          counter ← 0
Step 2.   If $D$ is more than one 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 $Q$ ←1
          $C$ (new) ← $D$ (old)
          $D$ (new) ← $C$ (old) - $Q * D$ (old)
Step 3.   $D$ = 0, then goto Step 5
Step 4.   $X'$ (new) ← $X$ (old)
          $X$ (new) ← $X'$ (old) + $Q * X$ (old)
          If $C \leq D$,
          then swap $C$ with $D$, swap $X$ with $X'$
          else counter ← counter +1
          Goto Step 2
Step 5.   If (counter = 1 mod 2),
          then return $N$ - $X$
          else return $X$
```

Fig. 3 Modular inverse algorithm [20] for 8-bit IC cards.

The proposed modular inverse algorithm, see Fig. 3, is based on the classical extended Euclidean algorithm and the Lehmer's algorithm [10], [11], but we adopt *approximation division* [13] instead of direct long division (see Step 2). We also reduce the number of variables and keep values of all variables positive so that we won't need to check the sign bits of operands during arithmetic operations.

Although there are binary extended gcd algorithms available, but the bit-shift operations involved in these binary algorithms take more time than addition operations for multiple-prevision integers on general-purpose IC cards. Therefore, we carefully examine the available algorithms and came out with a fast algorithm suitable for the H8/300 IC cards. Our experiences indicate that such an approach provides a 10 to 20 times performance improvement over the original extended Euclidean algorithm.

5. Implementation of the Digital Signature Algorithms

The ESIGN and RSA algorithms were implemented on Hitachi's H8/3113 IC cards running at internal 5 MHz. Assembly codes for modular arithmetic algorithms were written and emulated using Hitachi's hardware development system. Performance of our modular multiplication and modular inverse algorithms are summarized in Table 1 and Table 2.

Assembly codes for ESIGN and RSA are then written. Total program size is about 3 kbytes for each digital signature algorithm. To improve the performance of ESIGN, we revise Eq. (3) into

$$T = W_0 \times E \times \left(E \times R^E\right)^{-1} modP, \tag{5}$$

This way instead of performing $R^{E-1} modP$, we perform $\left(R^E modN\right) modP$, where $R^E modN$ is already computed at Eq. (2).

Besides, the implementation complexity (in terms of code size) of an algorithm is as important as the algorithm performance on the IC 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 5 MHz. The total RAM usage in the implementation is 466 bytes.

In Fig. 4, we gave the performance analysis of ESIGN on H8/300 IC card with different values of security parameter E, where *R^E mod N* denotes the operation of $R^E mod\, N$ in Eq. (2), *R^E mod P* denotes the $\left(R^E modN\right) modP$ in Eq. (5), INV is the modular inverse operation needed in Eq. (5), and *Misc* denotes all other operations in ESIGN signature generation.

By using fast modular inverse algorithm, the critical operation involved in the ESIGN algorithm is the operation of $R^E mod\, N$. Our results are somewhat different from the NESSIE's report on the 8051-based IC card Implementation of ESIGN [15]. 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.

$$T_1 = R^E mod\ N \tag{6}$$

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

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 IC cards.

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

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

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

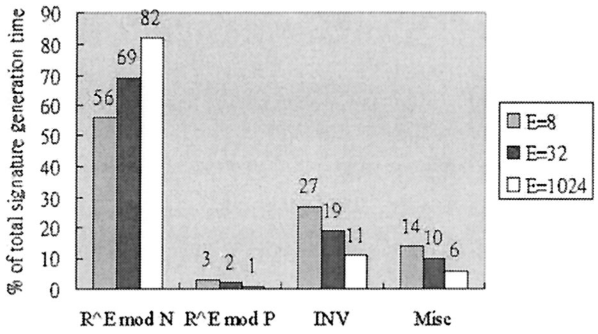

Fig. 4 Performance analysis of ESIGN signature generation on H8/300 IC cards.

$$T_2 = \left(E \times R^E\right)^{-1} \times R \quad mod\ P \tag{7}$$

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

$$W_0 = \left\lceil \frac{\left(f \| 0^{2k}\right) \; - \; T_1 \quad mod\ N}{PQ} \right\rceil \tag{8}$$

$$T = W_0 \times T_2 \quad mod\, P \tag{9}$$

$$S = R + T \times PQ \tag{10}$$

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

6. Conclusions

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 ESIGN and RSA digital signature algorithms on general-purpose 8-bit IC cards. The results show that it takes less than 0.5 second to generate 1152-bit ESIGN signature on H8/3113 IC cards without a coprocessor while 1152-bit RSA signature takes more than 150 seconds.

We are currently in the process of improving our results and implementing ESIGN on the Infineon 8051-compatible IC cards. For ESIGN with large value of public exponent, say $1024 \leq E$, we could have a fast modular squaring algorithm and the use of the Chinese Remainder Theorem could give us further improvement. As the factoring of 1024-bit integer is within our reach [19] and we need to use larger size of public key in ESIGN and RSA, the performance advantage of ESIGN over RSA become remarkable.

Acknowledgement

The authors wish to thank anonymous reviewers for their valuable comments of the earlier manuscript of this paper. Their suggestions have greatly enhance the quality of this paper.

References

[1] 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," Proc. First National Conference on Digital Government, pp.48–55, May 2001.

[2] W. Ford and M.S. Baum, Secure Electronic Commerce: Building the Infrastructure for Digital Signatures and Encryption, Prentice Hall, 1997.

[3] E. Fujisaki and T. Okamoto, "How to enhance the security of public-key encryption at minimum cost," Public Key Cryptography, PKC'99, eds. H. Imai and Y. Zheng, Lecture Notes in Computer Science, vol.1560, pp.53–68, Springer, 1999.

[4] H. Handschuh and P. Paillier, "Smart card crypto-coprocessors for public-key cryptography," Smart Card Research and Applications, Lecture Notes in Computer Science, vol.1820, Springer, pp.386–394, 2000. http://www.gemplus.com/smart/r_d/publications/pdf/HP00copr.pdf

[5] Hitachi Single-Chip Microcomputer H8/3113 Hardware Manual, Hitachi Ltd., 1998.

[6] IEEE P1363a: Standard Specifications For Public Key Cryptography: Additional Techniques, http://grouper.ieee.org/groups/1363/P1363a/

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

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

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

[10] D.E. Knuth, The Art of Computer Programming—Seminumerical Algorithms, vol.2, Sect. 4.3, 3rd ed., Addison-Wesley, 1998.

[11] A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone, Handbook of Applied Cryptography, CRC Press Series on Discrete Mathematics and Its Applications, CRC Press, 1996.

[12] P.L. Montgomery, "Modular multiplication without trial division," Math. Comput., vol.44, no.170, pp.519–521, 1985.

[13] H. Morita and C.H. Yang, "Lookahead determination for modular multiplication," IEICE Trans. Fundamentals, vol.E76-A, no.1, pp.70–77, Jan. 1993.

[14] T. Okamoto, "A fast signature scheme based on congruential polynomial operations," IEEE Trans. Inf. Theory, vol.36, no.1, pp.47–53, Jan. 1990. See also http://info.isl.ntt.co.jp/esign/

[15] NESSIE, Performance of Optimized Implementations of the NESSIE Primitives, version 1.0, pp.41–42, Oct. 2002. https://www.cosic.esat.kuleuven.ac.be/nessie/deliverables/D21-v1.pdf

[16] National Institute of Standards and Technology, "Secure hash standard," FIPS PUB 180-2, Aug. 2002. http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf

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

[18] R.L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," Commun. ACM, vol.21, no.2, pp.120–126, Feb. 1978.

[19] A. Shamir and E. Tromer, "On the cost of factoring RSA-1024," RSA CryptoBytes, vol.6, no.2, pp.10–19, 2003. http://www.wisdom.weizmann.ac.il/~tromer/papers/cbtwirl.pdf

[20] C.H. Yang, "Modular arithmetic algorithms for smart cards," IEICE Technical Report, ISEC92-16, Aug. 1992.