Secure Evaluation of Modular Functions

[Published in R.J. Hwang and C.K. Wu, Eds., *2001 International Workshop on Cryptology and Network Security*, pp. 227–229, Taipei, Taiwan, September 26–28, 2001.]

Marc Joye[1], Pascal Paillier, and Sung-Ming Yen[2]

[1] Gemplus Card International, France
{marc.joye, pascal.paillier}@gemplus.com
[2] National Central University, Taiwan
yensm@csie.ncu.edu.tw

Abstract. This paper presents a simple and efficient method of protection against fault analysis when the underpinning cryptosystem uses modular arithmetic. The proposed method applies whatever the modular function to be evaluated and the used algorithms. Moreover, it only requires a very little overhead of extra computations, especially when the modulus is represented in diminished-radix form or when at least one factor of the modulus is known.

Keywords. Fault analysis, modular reduction, diminished-radix representation.

1 Introduction

In September 1996, newspaper publications cited a Bellcore press release *New Threat Model Breaks Crypto Codes*: a new 'Potential Serious Problem' was reported. This launched a new type of cryptanalysis, the so-called *Fault Analysis* [1–3, 5, 6, 10–13, 15, 19, 25]: the presence of faults may leak some secret information.

This paper presents a simple and efficient method of protection against fault analysis when the underpinning cryptosystem uses modular arithmetic. This method thus applies to almost all public-key systems (RSA [22], El Gamal [9], etc. . .) and also to some private-key systems (e.g., XMX [17]).

For some moduli M having a special form, computations modulo M can be speeded up (e.g., when $M = 2^s \pm \mu$) [14, §4.3.2]. Furthermore, if a given modulus M can be transformed into $M' = 2^{s'} \pm \mu' = rM$ for some (small) integer r, then $f(x) \bmod M$ can be derived from $f(x) \bmod M'$ by Chinese remaindering. This was exploited by Quisquater [20, 21] and Walter [24] (see also [18]). If such algorithms are used, checking modular functions for faults is almost free. The same conclusion holds when the (partial) factorization of the modulus is known.

Previous solutions suggested to do calculations twice. Such solutions are not satisfactory because they do not detect permanent faults. Other solutions proposed to verify the correctness by comparing the "inverse" result with the input (e.g., the RSA signature S on a message M can be verified by comparing whether $S^v \equiv M \pmod{n}$, where v is the public verification exponent and n is the RSA-modulus). This is also not satisfactory because this is time-consuming and the inverse map (when it exists) is not always known. The main advantage of the proposed method is its very general nature, it remains applicable whatever the modular function to be evaluated and the used modular algorithms. Moreover, this method is much less time-consuming than previous ones, and allows to parameterize the probability that an error goes undetected.

The next section reviews a previous method by Shamir to check RSA computations for faults. Section 3 explains how to securely evaluate an *arbitrary* modular function. Then, in Section 4, some short-cuts are pointed out when modular reductions are performed with diminished-radix moduli, and when some factors of the modulus are known. Finally, Section 5 concludes the paper.

2 Shamir's Trick

An elegant method was pointed out by Shamir [23] to prevent the leakage of the RSA secret factors from faulty Chinese remaindering based RSA signatures [6, 11].

Let $n = pq$ be an RSA modulus and let (e, d) be a matching pair of verification/signature exponents. To sign a message representative m, the signer first chooses a (small) random number r relatively prime to n. Then s/he computes $s_{rp} = m^d \bmod rp$ and $s_{rq} = m^d \bmod rq$. If $s_{rp} \equiv s_{rq} \pmod{r}$ then the computations are assumed correct, and s is computed by applying Chinese remaindering on $(s_{rp} \bmod p)$ and $(s_{rq} \bmod q)$.

The problem in Shamir's method is that, within CRT mode of concrete applications, the value of d is unknown: only $d_p = d \bmod p - 1$ and $d_q = d \bmod q - 1$ are available. This means that d must be re-computed; note the same conclusion holds when s_{rp} (resp. s_{rq}) is computed as $s_{rp} = m^{d \bmod \phi(rp)} \bmod rp$ (resp. $s_{rq} = m^{d \bmod \phi(rq)} \bmod rq$) where ϕ denotes the Euler totient function.

The next section presents a solution that alleviates the aforementioned drawback. It also generalizes the methodology behind Shamir's trick.

3 Our Method

Suppose that $y = f(x) \bmod M$ has to be evaluated. Instead of directly computing y, a (small) random number r is first chosen and $y_r = f(x) \bmod r$ is evaluated. Also, the value of $z = f(x) \bmod rM$ is evaluated. Then, if $z \not\equiv y_r \pmod{r}$, an error has occurred; otherwise the computations are supposed correct and $y = z \bmod M$. Schematically, we have:

$$\begin{tikzcd}
z = f(x) \bmod rM \arrow[dr] & & y_r = f(x) \bmod r \arrow[dl] & \\
& z \bmod r \overset{?}{=} y_r \arrow[rr, "\text{no}"] \arrow[d, "\text{yes}"'] & & \text{ERROR} \\
& y = z \bmod M & &
\end{tikzcd}$$

Fig. 1. Evaluation of $\boldsymbol{y = f(x) \bmod M}$.

The probability that an error is undetected is equal to $1/r$. If r is a 20-bit integer, this probability is already smaller than 10^{-6}; r can thus be considered as a security parameter.

4 Special Cases

The evaluation of modular functions can be simplified if the modulus has a special form $M' = rM$. In that case, the computation of $z = f(x) \bmod rM$ is implicitly done, resulting in a gain of speed in the verification process (see Fig. 1). We will illustrate this topic for algorithms based on diminished-radix transformation, but it can also be generalized to augmented-radix moduli algorithms.

Furthermore, when the (partial) factorization of the modulus is known, the computation of $y_r = f(x) \bmod r$ (see Fig. 1) can be optimized.

4.1 Diminished-radix transformation

Let $M = \sum_{i=0}^{s-1} m_i\, 2^i$ and $M' = \sum_{i=0}^{s'-1} m'_i\, 2^i$ be the binary expansions of M and M', respectively. The modulus M' is called a diminished-radix (DR) modulus if it has the special form

$$M' = rM = 2^{s'} - \mu$$

where $r, \mu < 2^s$. Moreover, given an arbitrary modulus M, it is always possible to transform M into a DR modulus M'. A valid choice for the normalization factor r is $\lfloor 2^{s'}/M \rfloor$, and $\mu = 2^{s'} \bmod M$.

Proof. Obviously, $M' = 2^{s'} - \mu = 2^{s'} - (2^{s'} \bmod M) = \lfloor 2^{s'}/M \rfloor M = rM$. $\square$

Note that the full division by M is not required to evaluate r [8]. Once M has been transformed into its DR form $M' = rM$, $z = f(x) \bmod M'$ can be computed with only shifts, additions and single-precision multiplications [14, pp. 268–275] (see also [20, 21, 24, 18]).

In conclusion, to compute $y = f(x) \bmod M$, the modulus M is first transformed to a DR modulus $M' = rM = 2^{s'} - \mu$. Then, $z = f(x) \bmod M'$ and $y_r = f(x) \bmod r$ are evaluated. If $z \equiv y_r \pmod{r}$, then $y = z \bmod M$. Note that since computations modulo M' are faster, $z \bmod M$ can efficiently be computed as $(rz \bmod M')/r$.

Proof. Straightforward from $rz \bmod rM = rz - \lfloor \frac{rz}{rM} \rfloor rM = r\,(z \bmod M)$. $\square$

4.2 Chinese remaindering

Let $M = \prod_{i=1}^{t} p_i^{e_i}$ be the prime factorization of M; $y = f(x) \bmod M$ can then be computed from $f(x) \bmod p_i^{e_i}$ $(i = 1, \dots, t)$ via Chinese remaindering. More generally, we will assume that the modulus M is given by $M = \prod_{i=1}^{g} f_i$ with $\gcd(f_i, f_j) = 1$ for $i \neq j$. In that case, the verification process can also be speeded up. We will give the method for $M = f_1 f_2$ with $\gcd(f_1, f_2) = 1$, but it readily extend to 3 or more co-prime factors f_i. Since $M = f_1 f_2$, $y = f(x) \bmod M$ can be computed from $y_1 = f(x) \bmod f_1$ and $y_2 = f(x) \bmod f_2$. As depicted in Fig. 1, y_i $(i = 1, 2)$ is computed by independently evaluating $z_i = f(x) \bmod r_i f_i$ and $y_{r,i} = f(x) \bmod r_i$ for some (small) random number r_i; then if $z_i \equiv y_{r,i}$ $\pmod{r}$, the algorithm outputs $y_i = z_i \bmod f_i$. If the same value r is chosen for r_1 and r_2 (i.e., $r_1 = r_2 = r$), then the computation of $y_{r,1}$ and $y_{r,2}$ can be avoided; the algorithm now checking whether $z_1 \equiv z_2 \pmod{r}$. If this equivalence holds, then $y_1 = z_1 \bmod f_1$ and $y_2 = z_2 \bmod f_2$.

5 Conclusions

A general and inexpensive method for checking modular functions was proposed. When the modular algorithm uses diminished-radix moduli (or augmented-radix moduli) or when at least one factor of the modulus is known, this verification is almost free. Note, however, that in the general case the extra overhead is negligible: only 20 additional bits are needed to offer an error-free result with probability at least $(1 - 10^{-6})$.

Acknowledgements. Part of this work was supported by the National Science Council of the Republic of China under contracts NSC87-2213-E-032-012 and NSC87-2811-E-032-0001.

References

1. R. Anderson and M. Kuhn. Tamper resistance — a cautionary note. In *Proceedings of the Second USENIX Workshop on Electronic Commerce*, pages 1–11. USENIX Association, 1996.
2. R. Anderson and M. Kuhn. Low cost attacks on tamper resistant devices. In B. Christianson, B. Crispo, M. Lomas, and M. Roe, editors, *Security Protocols*, volume 1361 of *Lecture Notes in Computer Science*, pages 125–136. Springer-Verlag, 1998.

3. F. Bao, R.H. Deng, Y. Han, A. Jeng, A.D. Narasimbalu, and T. Ngair. In B. Christianson, B. Crispo, M. Lomas, and M. Roe, editors, *Security Protocols*, volume 1361 of *Lecture Notes in Computer Science*, pages 115–124. Springer-Verlag, 1998.
4. P. Barrett. Implementing the Rivest Shamir and Adleman public key encryption algorithm on a standard digital signal processor. In A.M. Odlyzko, editor, *Advances in Cryptology – CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 311–323. Springer-Verlag, 1987.
5. E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In B.S. Kaliski Jr, editor, *Advances in Cryptology – CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer-Verlag, 1997.
6. D. Boneh, R.A. DeMillo, and R.J. Lipton. On the importance of checking cryptographic protocols for faults. In W. Fumy, editor, *Advances in Cryptology – EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 37–51. Springer-Verlag, 1997.
7. A. Bosselaers, R. Govaerts, and J. Vandewalle. Comparison of three modular reduction functions. In D.R. Stinson, editor, *Advances in Cryptology – CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 175–186. Springer-Verlag, 1994.
8. J.-F. Dhem, M. Joye, and J.-J. Quisquater. Normalisation in diminished-radix modulus transformation. *Electronics Letters*, 33(23):1931, November 1997.
9. T. El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, IT-31(4):469–472, 1985.
10. P. Gutman. Secure deletion of data from magnetic and solid-state memory. In *6th USENIX Security Symposium Proceedings*, pages 77–89. USENIX Association, 1996.
11. M. Joye, A.K. Lenstra, and J.-J. Quisquater. Chinese remaindering cryptosystems in the presence of faults. *Journal of Cryptology*, 12(4):241–245, 1999.
12. M. Joye, J.-J. Quisquater, F. Bao, and R.H. Deng. RSA-type signatures in the presence of transient faults. In M. Darnell, editor, *Cryptography and Coding*, volume 1355 of *Lecture Notes in Computer Science*, pages 155–160. Springer-Verlag, 1997.
13. B.S. Kaliski Jr. Comments on a new attack on cryptographic devices. Unpublished manuscript.
14. D.E. Knuth. *The art of computer programming*, volume 2. Addison-Wesley, 2nd edition, 1981.
15. O. Kocar. Hardwaresicherheit von mikrochips in chipkarten. *Datenschutz und Datensicherheit*, 20(7):421–424, July 1996.
16. A.J. Menezes, P.C. van Oorschot, and S.A. Vanstone. *Handbook of applied cryptography.* CRC Press, 1997.
17. D. M'Raïhi, D. Naccache, J. Stern, and S. Vaudenay. XMX: a firware-oriented block cipher based on modular multiplications. In E. Biham, editor, *Fast Software Encryption*, volume 1267 of *Lecture Notes in Computer Science*, pages 166–171. Springer-Verlag, 1997.
18. G. Orton, L. Peppard, and S. Tavares. A design of a fast pipelined modular multiplier based on a diminished-radix algorithm. *Journal of Cryptology*, 6:183–208, 1993.
19. I. Peterson. Chinks in digital armor – exploiting faults to break smart-card cryptosystems. *Science News*, 151(5):78–79, February 1997.
20. J.-J. Quisquater. Fast modular exponentiation without division. Presented at the rump session of EUROCRYPT '90, Århus, Denmark.

21. J.-J. Quisquater. Procédé de codage selon la méthode dite RSA par un microcontrôleur et dispositifs utilisant ce procédé. Demande de brevet français, No de dépôt 9002274 (U.S. Patent #5,166,978), 25 February 1990.
22. R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
23. A. Shamir. How to check modular exponentiation. Presented at the rump session of EUROCRYPT '97, 11–15th May 1997, Konstanz, Germany.
24. C.D. Walter. Faster modular multiplication by operand scaling. In J. Feigenbaum, editor, *Advances in Cryptology – CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 313–323. Springer-Verlag, 1992.
25. Y. Zheng and T. Matsumoto. Breaking real-world implementations of cryptosystems by manipulating their random number generation. In *Pre-proceedings of the 1997 Symposium on Cryptography and Information Security*, 29th January–1st February 1997, Fukuoka, Japan.