Chosen-message Electromagnetic Analysis against Cryptographic Software on Embedded OS

Hajime Uno, Sho Endo, Yu-ich Hayashi, Naofumi Homma and Takafumi Aoki
Graduate School of Information Science, Tohoku University
6-6-05, Aramaki Aza Aoba, Aoba-ku, Sendai-shi 980-8579, Japan
uno@aoki.ecei.tohoku.ac.jp

***Abstract*— This paper presents an EMA (electromagnetic analysis) against an RSA software on an embedded operating system (OS) and its countermeasure. First, we describe a pre-processing step to find an appropriate observation point for EMA, where the EM radiation caused by cryptographic operations can be observed in an accurate manner. We employ a specific input pattern, which is suitable for observing EM radiation according to an EM radiation model, and scan a micro magnetic probe on the target device in order to find specific EM radiation pattern for EMA. This pre-processing can be conducted on another programmable device that has the same CPU and OS as the target device. We demonstrate that such EM analysis is effective for an RSA software performed on Android OS. We also confirm the validity of a countermeasure against the presented attack.**

Keywords—side-channel attacks, electromagnetic analysis, RSA, chosen-message SEMA

I. INTRODUCTION

Side-channel attacks on cryptographic devices are attracting extensive attention. It exploits side-channel information such as power dissipation, electromagnetic radiation, or operating times etc. In particular, electromagnetic analysis (EMA) attack [1] that uses electromagnetic (EM) emanations from cryptographic devices is known as a powerful attack since we can pinpoint the cryptographic processing point and measure side-channel information with high accuracy. Many papers about side-channel attacks against cryptographic hardware or cryptographic software on general-purpose processor have been published [2]. However, to the best of our knowledge, there is little previous work about EMA attacks against cryptographic software implemented on practical embedded OSs in literature. Unlike in the case of cryptographic hardware or cryptographic software on general-purpose processor, any device equipped with an embedded OS always runs many background operations in parallel with cryptographic operation. As a result, the signal-to-noise ratio of side-channel information observed from such devices becomes much lower than those from other devices. In addition, information leakage source from such devices is usually unclear. Therefore it is expected to be difficult to apply side-channel attacks to devices equipped embedded O/S.

This paper shows that EMA can be applied to cryptographic software on devices equipped with embedded OSs. In the conventional attack scenarios, the attack point, where targeted EM trace is measured, is given in advance. On the other hand, in embedded OS devices, the cryptographic operation is one of the many operations in parallel, and therefore it is difficult to determine such attack point in advance. This paper first presents how we can search an attack point using a chosen-input method, which makes sure whether the observed EM radiation is accordance with the attack model. Once the attack point determined, conventional EMA is applied at the point. This paper shows that the effectiveness of the proposed method through a simple EM analysis experiment against an RSA software on Android OS. This paper also confirms the validity of countermeasures against this kind of attack.

II. DETERMINING OBSERVATION POINT FOR ELECTROMAGNETIC ANALYSIS

A. Determining observation point based on electromagnetic model

This section describes a pre-processing step to find an appropriate point where we can observe EM radiation caused by cryptographic operations in an accurate manner. This pre-processing is particularly essential for embedded OS devices running many operations in the background. Prior to EMAs on such devices, we search for the observation point using another programmable device equivalent to the target device. In this step, we generate a predictable EM radiation, and then scan the device with micro magnetic probe for observing the expected pattern of EM radiation. The predictable EM radiation is generated as follows. First, we chose two operations that should generate the maximum and minimum amounts of EM radiations, respectively. In the EMA attack, we use the correlation between theoretical values calculated from EM models and observed value. It is well-known that the intensity of EM radiation is correlated with the hamming weight of operands processed by instructions (Hamming-weight (HW) model) [3] in microcontrollers. Therefore, we chose two inputs based on the HW model in order to observe the difference between the EM radiations. More precisely, we perform multiplication operations using the following two inputs:

$$C = (\,\mathrm{ffffffffffffffff}\,)_{16} \times (\,\mathrm{ffffffffffffffff}\,)_{16}, \quad (1)$$

$$C = (\,0000000000000000\,)_{16} \times (\,0000000000000000\,)_{16}. \quad (2)$$

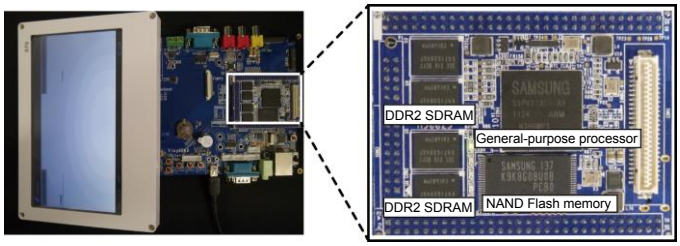


Figure 1 Experimental board.

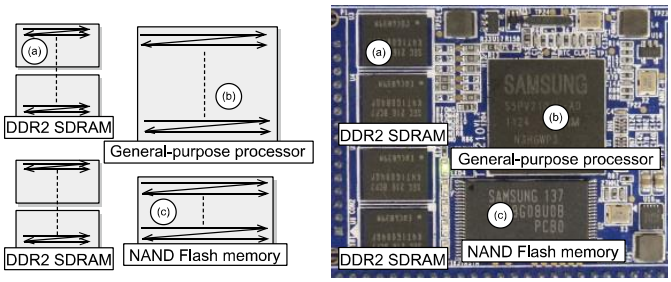


Figure 2 Probing areas.

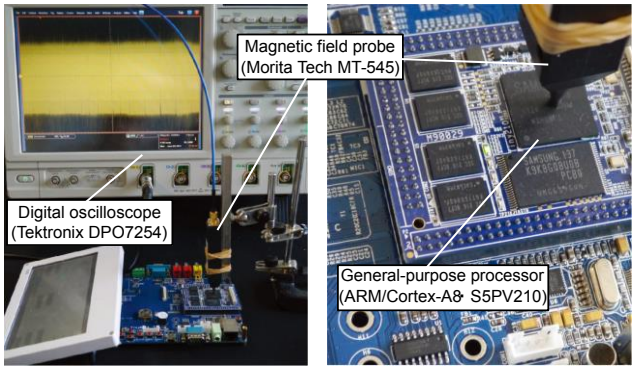


Figure 3 Overview of experimental setup.

Here, we expect that the intensity of EM radiation during the computation of (1) would be larger than that of (2) because the HW of operands in (1) is maximum while that in (2) is minimum. As a result, the observed EM trace would be a observation point.

B. Preliminary experiment

Figure 1 shows an experimental device equipped with an embedded general-purpose processor with Android OS. We measure EM waveforms from the board using a micro magnetic probe in running the repeating operations described above. We scan all the possible point on CPU, NAND Flash memory, and DDR2 SDRAM as shown in Figure 2. Table 1 and Figure 3 summarize the experimental condition. The probe touched the surface of objects during the observation. The coil face of the probe was placed vertically to the board, and the horizontal component of the EM radiation was measured by the probe.

Figure 4 shows the EM waveforms obtained from the device. Figure 4 (a) ~ (c) show the traces obtained at the points (a) ~ (c) shown in Figure 2. From Figure 4, we can confirm that EM radiation from the point (b) on the CPU correlates with the HW model because of the rectangular shape. This result suggests that we can chose the point (b) as the observation point.

Table 1 EXPERIMENTAL CONDITION

Experimental board	
CPU	ARM/Cortex-A8・S5PV210
Operating frequency	1GHz
Experimental setting	
Sampling frequency	500MSa/s
Magnetic probe	Morita Tech MT-545
Amplifier	YKC-1000AMP
Oscilloscope	Tektronix DPO7254

ALGORITHM 1
Left-to-right binary method

Input : $X, N, E = (e_{k-1}, \ldots, e_1, e_0)_2$
Output : $Z = X^E \bmod N$

1: $Z = 1$;
2: **for** $i = k-1$ **downto** 0
3: $Z = Z \times Z \bmod N$; -squaring
4: **if** ($e_i = 1$) **then**
5: $Z = Z \times X \bmod N$; -multiplication
6: **end if**
7: **end for**

III. SEMA AGAINST RSA CRYPTOSYSTEM

This section shows the effectiveness of our EMA through a simple EMA experiment with the above device. We briefly describe the trick of EMA on RSA cryptosystem. We then show the experimental results.

A. SEMA against RSA cryptosystem

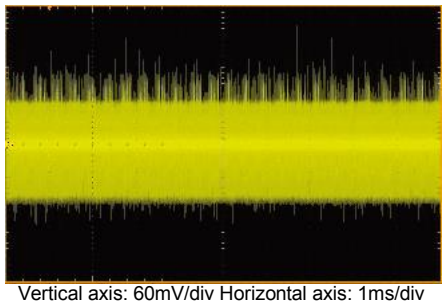
The RSA cryptosystem employs modular exponentiation for encryption and decryption as follows:

$$C = P^E \bmod N, \quad (3)$$

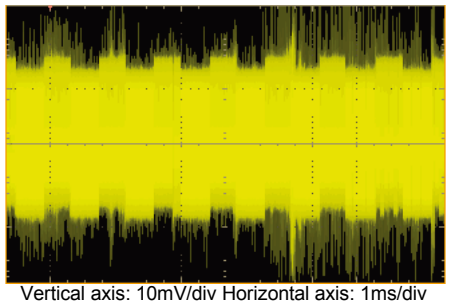
$$P = C^D \bmod N, \quad (4)$$

where P is the plaintexts, C is the ciphertext, E and N are the public key, and D is the secret key. P, C, N and D are at least 1,024 bits in length for security reasons.

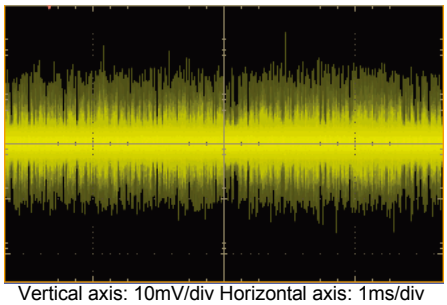
Binary methods are commonly used for the modular exponentiations, which perform multiplication and squaring sequentially according to the bit pattern of the exponent E or D. ALGORITHM 1 shows a left-to-right binary method for scanning the bits of the exponent from MSB to LSB. This algorithm always performs a squaring at Line 3 independently of the scanned bit value, but the multiply operation at Line 5 is only executed if the scanned bit is 1.


Vertical axis: 60mV/div Horizontal axis: 1ms/div

(a)


Vertical axis: 10mV/div Horizontal axis: 1ms/div

(b)


Vertical axis: 10mV/div Horizontal axis: 1ms/div

(c)

Figure 4 Electromagnetic waveform measured at the points (a) ~ (c) shown in Figure 2.

ALGORITHM 2
Montgomery multiplication (*MontMult*)

Input :	$X=(x_{m-1},\ldots x_1,x_0)_{2^r}$, $Y=(y_{m-1},\ldots y_1,y_0)_{2^r}$ $N=(n_{m-1},\ldots n_1,n_0)_{2^r}$, $w=-N^{-1} \bmod 2^r$
Output :	$Z=XY2^{-r\cdot m} \bmod N$

```
1  : Z = 0;
2  : for i = 0 to m-1 do
3  :   c = 0;
4  :   t_i = (z_0 + x_i y_0) · w mod 2^r;
5  :   for j = 0 to m-1 do
6  :     q = z_j + x_i y_j + t_i n_j + c;
7  :     if (j≠0) then
8  :       z_{j-1} = q mod 2^r;
9  :     c = q >> r;
10 :   end for
11 :   z_{m-1} = c;
12 : end for
13 : if (Z > N) then
14 :   return Z-N else return Z;
```

ALGORITHM 3
Modular exponentiation

Input :	$X, N, R=2^k, E=(e_{k-1},\ldots,e_1,e_0)_2$
Output :	$Z=X^E \bmod N$

```
1  : W = -N^{-1} mod R;
2  : Y = MontMult(X, 1, N, W) ;
3  : Z = MontMult(1, 1, N, W) ;
4  : for i = k-1 to 0 do
5  :   Z = MontMult(Z, Z, N, W) ;      -squaring
6  :   if (e_i = 1) then
7  :     Z = MontMult(Z, Y, N, W) ;    -multiplication
8  :   end if
9  : end for
10 : Z = MontMult(Z, 1, N, W) ;
```

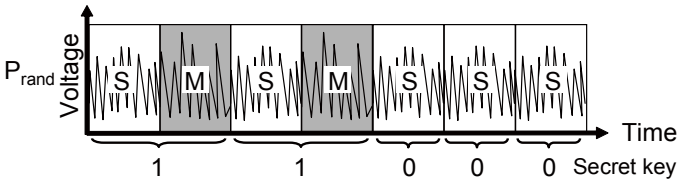


Figure 5 SEMA attack against RSA cryptosystem.

One major method to speed up the exponentiation is to use Montgomery's modular multiplication algorithm (*Mont-Mult*) [4].For integers X and Y, where $0 \le X, Y < N < 2^k = R$, define *MontMult* to be $XYR^{-1} \bmod N$. ALGOTITHM 2 shows a high-radix Montgomery multiplication algorithm for the fast and compact implementation. The k-bit operands are divided into $m \times r$-bit blocks and processed in a nested loop. ALGORITHM 3 shows a modular exponentiation algorithm combining ALGORITHM 1 and ALGORITHM 2.

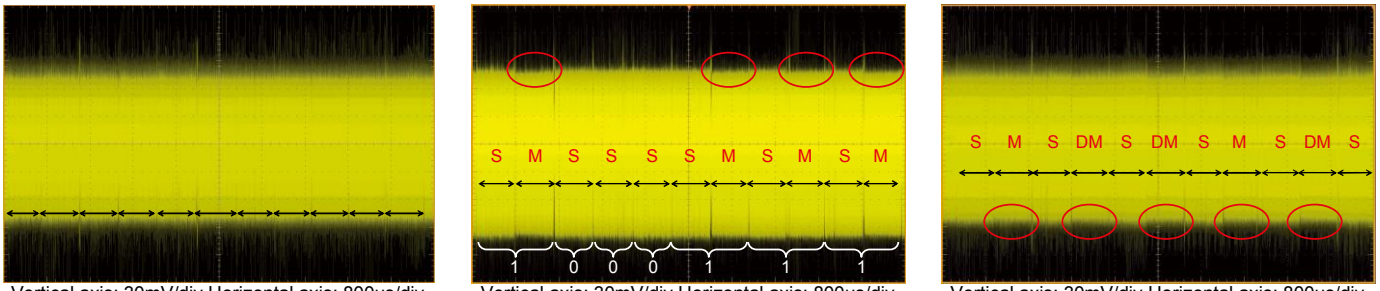
Simple EM analysis (SEMA) is known as basic and powerful side-channel attack for RSA. The concept of the SEMA against RSA cryptosystem is to distinguish between multiplication and squaring in the EM waveform. Figure 5 shows an image of the SEMA against an RSA module using the left-to-right binary method, where the key bit pattern '11000' can be detected.

B. SEMA experimentation and counteremeasure

The EM waveform was observed during RSA operations showed in ALGORITHM 3 from the point determined in Section Ⅱ. The experimental condition is the same as Table 1. We performed a chosen-message SEMA [5], which enhances the difference between squaring and multiplication using particular input. The basic idea of this attack is that one input of multiplication depends on the message. In this experiment, the input value X is equal to $C \times R^{-1} \bmod N$, where $C = (100\ldots000)_2$ and $R = 2^k$. This is because the input $C \times R^{-1}$ is converted to the input in the Montgomery domain with Equation (5) at Line 2 in ALGORITHM 3 as follows, prior to the Montgomery multiplication.

$$Y = C \times R^{-1} \times R \bmod N = C = (100\ldots000)_2. \quad (5)$$

Therefore, $Y = (100\ldots000)$ is always multiplied in ALGOLITHM 2, and the output of the multiplication is identical to the other input.



Vertical axis: 30mV/div Horizontal axis: 800μs/div (a)
Vertical axis: 30mV/div Horizontal axis: 800μs/div (b)
Vertical axis: 30mV/div Horizontal axis: 800μs/div (c)

Figure 6 SEMA attack against RSA software.

ALGORITHM 4

Square-and-multiply always method

Input :	$X, N, E = (e_{k-1}, \ldots, e_1, e_0)_2$
Output :	$Z = X^E \bmod N$

1: $Z = 1$;
2: **for** $i = k-1$ **downto** 0
3: $Z_0 = Z \times Z \bmod N$; --squaring
4: $Z_1 = Z_0 \times X \bmod N$; --multiplication
5: $Z = Z_{ei}$
6: **end for**
7: **return** Z

Figure 6 shows measured EM traces, where (a) and (b) indicate the waveforms for input data with random values and $X = C \times R^1$, respectively. Note that all the EM traces shown in Figure 6 are obtained through a low-pass filter (LPF). The cut-off frequency is set to 20MHz, which was determined by the frequency characteristics of the waveform. As shown in Figure 6 (a), no difference between squaring and multiplication operations was observed. In contrast, multiplication and squaring operations can be distinguished in Figure 6 (b), where the peak envelopes of multiplication operations (i.e., red circle) are different from those of other parts. Using the difference, we correctly estimated the key bit pattern '100111' .

There are some countermeasures against this SEMA attacks. ALGORITHM 4 shows the square-and-multiply always method [6], which is known as the most typical countermeasure technique. The method inserts dummy multiplications if the scanned bit is 0 in the left-to-right binary method shown ALGORITHM 1. Figure 6 (c) shows the observed EM trace of the square-and-multiply always technique. From the figure, the multiplication and squaring operations are observed alternately, and the key bit pattern cannot be detected. Note here that, this countermeasure has vulnerability to other attacks which determine dummy multiplications by fault injection. Montgomery powering ladder method [7] and Square-multiply exponentiation method [8] are proposed for countermeasures against such fault injection attacks.

IV. CONCLUSION

This paper presented a chosen-input EMA against an RSA software on Android OS. First, we described a pre-processing step with specific inputs to scan a micro magnetic probe on the target device and find the observation point for EMA. The two selected inputs could enhance the difference in EM radiation intensity under the EM radiation model called HW model. From preprocessing step, we pinpointed the observation point from EM waveforms measuring on the CPU, the Flash memory, the DDR2 SDRAMs during the multiplication operation. As a result, we found that the specific pattern according to the HW model was observed only on the CPU. We demonstrated the key recovery from an EM waveform observed during RSA operations with a chosen-message SEMA technique at the observation point. In addition, we presented that the square-and-multiply always method was effective as a countermeasure against the presented attack.

References

[1] D. Agrawal, B. Archambeault, J. R. Rao and P. Rohatgi " The EM Side-Channel(s)," in CHES 2002, Lecture Notes in Computer Science, Vol.2523, pp.29—45, 2002

[2] P. Kocher, J. Jaffe , and B. Jun, "Differential power analysis, " CRYPTO 1999, Lect, Notes Comput. Sci., vol.1666, pp.388-397, Aug, 1999

[3] S. Mangard, E. Oswald, T. Popp, "Power Analysis Attacks : Revealing the secret of smart cards, " Springer, 2007.

[4] P. L. Montgomery, "Modular multiplication without trial division," Math, Comp, Vol.44, No.170, pp.519—521, 1985

[5] A. Miyamoto, N. Homma, T. Aoki and A. Satoh, "Chosen-message SPA attack against FPGA-based RSA hardware implementation, " In Proc. of the 2008Int. Conf. on Field Programmable Logic and Applications. pp.35—40, Sep. 2008.

[6] J. S. Coron, "Resistance against differential power analysis for elliptic curve cryptosystems," CHES 1999:Proceedings of the First International Workshop on Cryptographic Hardware and Embedded Systems, pp.292—302, Spring-Verlag, Aug, 1999.

[7] M. Joye and S. M. Yen, "The Montgomery power-ing ladder, "CHES 2002, Lecture Notes in Computer Science, Vol.2523, pp.291—302, Aug. 2002.

[8] M. Joye, "Highly regular right-to-left algorithms for scalar multiplication, " CHES 2007, Lecture Notes in Computer Science, Vol.4727, pp.135—147, Sept 2007.