柔軟かつ安全なべき乗剰余計算

ヴィオム　カミーユ[†]　　桶屋　勝幸[†]

[†]（株）日立製作所 システム開発研究所　〒215-0013 川崎市麻生区王禅寺 1099 番地
E-mail: †{camille,ka-okeya}@sdl.hitachi.co.jp

あらまし　サイドチャネル攻撃に耐性のあるべき乗剰余計算を発表します。他の方式と比較すると処理速度も速く、耐タンパーRSAまたはDSAの実装に適しています。使用メモリをカスタマイズすることで、スマートカード等、様々なプラットフォームにおいて実装が可能になります。

キーワード　RSA、サイドチャネル攻撃、SPA

Secure Exponentiation with High Flexibility

Camille VUILLAUME [†] and Katsuyuki OKEYA [†]

[†] Hitachi, Ltd., Systems Development Laboratory, Kawasaki, Japan.
E-mail: †{camille,ka-okeya}@sdl.hitachi.co.jp

Abstract We present a countermeasure for protecting modular exponentiations against side-channel attacks. Our countermeasure is well-suited for tamper-resistant implementations of RSA or DSA, without significant penalty in terms of speed compared to commonly implemented methods. Thanks to its high efficiency and flexibility, our method can be implemented on various platforms, from smartcards with low-end processors to high-performance servers.

Key words RSA, side channel attacks, simple power analysis

1. Introduction

With the rise of electronic communications, and in particular, electronic commerce, public-key cryptography has become an essential component in our daily life. The de-facto standard for public-key encryption and digital signatures is RSA, and with the development of miniaturization, RSA is now implemented not only on high-performance servers, but also on various mobile devices such as smartcards or mobile phones.

It is believed that with a bitlength of 1024 bits, RSA is secure for middle-term applications, and with 2048 bits, for long-term applications. However, protecting against mathematical attacks is not sufficient in the real world. Indeed, it has been shown that practical implementations of cryptosystems often suffer from critical information leakage through side-channels: timings [4], power consumption [3] for instance. Such side-channel attacks are no theoretical works that researchers secretly run in laboratories with expensive hardware, but practical threats to virtually any application where secrecy matters.

On the one hand, there are numerous countermeasures for defeating side-channel attacks on elliptic curve cryptosystems [2], [5], but on the other hand, there are few of them for RSA. At first sight, it seems that the keylength of RSA is so long that even when side-channel information partially reveals the secret, exhaustive searches often remain ineffective. However, when a sufficiently large part of the secret key is known, RSA can be broken [1]. Thus, despite long secret keys, RSA also needs a decent protection against side-channel attacks.

Unfortunately, many countermeasures that have been developed for elliptic curves cannot be transposed to the case of RSA: on elliptic curves, signed representations of the exponents are heavily used because they yield faster exponentiation algorithms when inversions are cheap. But unlike elliptic curves, RSA does not benefit from cheap inversions, and therefore, unsigned representations are the only option. In other words, countermeasures which are efficient on elliptic curves are generally not practical at all in the case of RSA.

Our contribution is as follows: we show how to transform a countermeasure based on a signed representation [5] into a countermeasure based on an unsigned representation. Then, we improve the flexibility of our countermeasure with a fractional width technique [7]: with the improved method, the size of the pre-computed table can be freely chosen. Our countermeasure is not only highly flexible, and therefore well-suited for a wide range of platforms, from constrained environments such as smartcards to high-performance servers, but in addition, can compete with the commonly implemented exponentiation techniques. In practical situations, our method is only about 5% slower than commonly used methods. And most importantly, our method thwarts several types of attacks, power analysis and cache attacks in particular. Finally, we refine attacks against fractional width techniques and introduce a new tool to evaluate the quality of our countermeasure in the sense of resistance to side channel attacks. We demonstrate that in the SPA model, our countermeasure can prevent information leakage.

2. Side Channel Attacks

We briefly discuss side channel attacks and their countermeasures.

2.1 Methodology of Side Channel Attacks

Side channel attacks take advantage of the correlation between secret values and physical emanations such as timings [4] or power consumption [3]. On smartcards, which do not have any embedded power supply, the most powerful approach is probably to measure the power consumption of the device supplied from the outside [3]. One can classify power analysis attacks into two main classes: simple power analysis (SPA) and differential power analysis (DPA). The approach of SPA is to identify regions of a power trace which directly depend on the secret key. For example, in the frame of an RSA exponentiation computed with the binary method, the binary representation of the (secret) exponent is scanned; for the bit-value zero, a square is computed, whereas a square and a multiplication are calculated when the bit-value is one. Thus, it is easy to see that the knowledge of the operation sequence (square or square-multiplication) is equivalent to the knowledge of the secret exponent. A similar problem also exists in the case of elliptic curves, where the ability of distinguishing the two types of elliptic operations, namely point doublings and point additions, also leads to the secret scalar. DPA is more sophisticated: the idea is to guess the value of the secret bit-by-bit, and try to confirm or infirm the guess for each bit thanks to statistical analysis of several power traces.

2.2 Countermeasures

It is not that difficult to protect cryptosystems against DPA. By definition, DPA requires that the same secret is used to perform several cryptographic operations with each time a different input value. Signature schemes such as DSA and EC-DSA use a new random ephemeral as exponent for each new signature, and as a consequence, are naturally immune to DPA. In the case of RSA, a well-known technique to defeat DPA is to blind the secret exponent, that is, to add a random multiple of the group order to the secret. Because $c^{d+r\phi(n)} = c^d \bmod n$, blinding does not change the result of the exponentiation, but in the same time, changes the exponent itself and the side-channel information that arises from multiple power traces is not correlated. On the contrary, despite the relative simplicity of the idea behind SPA, it is not easy to design secure and efficient SPA countermeasures. However, SPA-resistance is always necessary, and is a prerequisite to DPA resistance. For instance, if only one random ephemeral exponent of DSA or EC-DSA is revealed, the secret key of the signature scheme can be easily inferred. Similarly, from the point of view of the attacker, a blinded RSA exponent $d + r\phi(n)$ is as good as the secret itself.

On elliptic curve cryptosystems, it is common to use representations of the scalar with a large digit set and to pre-compute some small multiples of the base point in order to speed up the scalar multiplication. In particular, thanks to window methods with digit set $\{0, 1, 2, \ldots, 2^w - 1\}$, one can reduce the computational cost of the scalar multiplication given that small multiples of the base point P are pre-computed: $2P, 3P, \ldots, (2^w - 1)P$. Although standard window methods aim at greater efficiency only, they can also be enhanced to SPA-resistant scalar multiplication schemes, where the secret is recoded with a fixed pattern, using a signed representation with digit set $\{-2^w, 1, 2, \ldots 2^w - 1\}$, where 0 is absent [5]. More precisely, an additional conversion step is applied in each window in order to remove zero digits: (1) replace the digit 0 with -2^w, and add a carry of $+1$ to the next window when scanning the scalar from right to left, (2) replace the digit 2^w with -2^w, and add a carry of $+2$ to the next window, (3) replace the digit 2^w+1 with 1 and add a carry of $+1$ to the next window, (4) otherwise leave the digit as it is. Thanks to this conversion technique, the scalar can be recoded with a fixed pattern: nonzero digits are always followed by exactly $w - 1$ zero digits. The advantage of this approach is that the operation pattern becomes regular as well: point additions are always followed by exactly w point doublings, which makes SPA impractical. Because the point $-2^w P$ must be readily available, the scheme requires that computing inversions in the group (in that case, elliptic point negative) is easy. Unfortunately, this is not the case for RSA.

3. SPA-Resistant Unsigned Techniques

We show how to construct an SPA countermeasure for RSA, where the computation of inverses is too costly.

3.1 SPA-Resistant Unsigned Integral Width

Our approach is to extend Möller's recoding to the unsigned case. To obtain the unsigned digit set $\{1, 2, \ldots, 2^w\}$, the key idea of our method is to use *negative* carries rather than positive carries: (1) replace the digit 0 with 2^w, and add a carry of -1 to the next window when scanning the scalar from right to left, (2) replace the digit -1 with $2^w - 1$, and add a carry of -1 to the next window, (3) otherwise leave the digit as it is.

On the one hand, it is easy to see that Möller's algorithm terminates, and that if the original bitlength of the scalar was $\mathcal{L}$, the recoded scalar has at most $\mathcal{L}+1$ digits. On the other hand, in the case of the above rules for generating an unsigned representation with a fixed pattern, the situation is different: there is no guarantee that the algorithm will terminate because a carry can propagate indefinitely. To ensure a correct termination, we treat the case of the most significant bit separately: if a carry remains at the end of the recoding, we use the most significant bit to neutralize it, and reduce the length of the exponent. If instead of d, a blinded exponent $d + r\phi(n)$ with random r is used as DPA countermeasure, this approach is safe[注1]. But if not, there is a direct information leakage, because the length of the recoded exponent depends on the value of some secret bits. To remove this leakage, we extend the bitlength of the exponent by 2, and fix the value of the 2 most significant bits $d_{\mathcal{L}+1} = 1$ and $d_{\mathcal{L}} = 0$. This is always possible because $c^{d+\phi(n)} = c^d \bmod n$: in other words, adding $\phi(n)$ to the exponent does not change the value of the exponentiation modulo n. By repeatedly adding $\phi(n)$, one can always set $d_{\mathcal{L}+1}$ to 1 and $d_{\mathcal{L}}$ to 0 (because $d < \phi(n) < n < 2^{\mathcal{L}}$). If the value of d and n are fixed (which is typically the case when d is a secret key), this calculation can be performed once for all at the key generation stage. Now, since $d_{\mathcal{L}} = 0$, independently from the value of the previous bits, the corresponding recoded digit is $u_{\mathcal{L}} \neq 0$ and a carry γ is generated. Finally, $u_{\mathcal{L}+1} = d_{\mathcal{L}+1} - 1 = 0$, therefore the length of the recoded expansion is *always* reduced by one.

3.2 SPA-Resistant Unsigned Fractional Width

A disadvantage of the previous method, and more generally, of table-based exponentiations, is that there are only limited choices for the table size. Since the table size and the cost of pre-computations grow exponentially with the width w, number of bits which are scanned simultaneously, large values of w become quickly impractical. However, it would be useful to be able to select *any* table size. Fractional width recodings make this possible, thanks to a degenerated width-w pre-computed table where some values are missing [6]. In addition, SPA-resistant fractional width methods exist in the case of elliptic curves, where it is advantageous to use signed representations [7]. We show that our unsigned (integral width) SPA-resistant recoding technique (Algorithm 1) can also be enhanced to an unsigned fractional width recoding.

Algorithm 1: Conversion to unsigned representation

```
INPUT: $\mathcal{L}+2$-bit exponent $d = (10d_{\mathcal{L}-1} \ldots d_0)_2$, width $w$;
OUTPUT: Recoded exponent $(u_{\mathcal{L}} \ldots u_0)$;

$i \leftarrow 0; \gamma \leftarrow 0;$
while $i \leqq \mathcal{L} - w$ do
    $u_i \leftarrow (d_{i+w-1} \ldots d_i)_2 - \gamma;$
    if $u_i \leqq 0$ then $u_i \leftarrow u_i + 2^w; \gamma \leftarrow 1;$ else $\gamma \leftarrow 0;$
    $u_{i+1} \leftarrow 0, \ldots, u_{i+w-1} \leftarrow 0; i \leftarrow i + w;$
if $i < \mathcal{L}$ then
    $u_i \leftarrow (d_{\mathcal{L}-1} \ldots d_i)_2 - \gamma;$
    if $u_i \leqq 0$ then $u_i \leftarrow u_i + 2^{\mathcal{L}-i}; \gamma \leftarrow 1;$ else $\gamma \leftarrow 0;$
    $u_{i+1} \leftarrow 0, \ldots, u_{\mathcal{L}-1} \leftarrow 0;$
$u_{\mathcal{L}} \leftarrow 2 - \gamma;$ return $(u_{\mathcal{L}} \ldots u_0);$
```

The principle of our technique is the same as in the original signed SPA-resistant fractional width recoding [7]: compute simultaneously the digits x and y, where x correspond to the width w and y to the width $w-1$. The core idea in Algorithm 2 is that the choice of recoding a sequence of bits with the width w or $w-1$ looks random to the attacker. Recall that B is the set of exponents of the pre-computed values, define the width $w = \lceil \log_2(k) \rceil$ and the probability $p = k/2^{w-1} - 1$. Then, for x computed with width w and y with width $w-1$, we apply the following rules: (1) if $x \leqq 2^{w-1}$ then choose x with probability p or y with probability $1-p$, (2) else if $x \in B$ (in other words, g^x is pre-computed), choose x (this occurs with probability p), (3) else choose y (in that case, g^x is not pre-computed, this happens with probability $1-p$). Therefore, for randomly chosen exponents, the width w is chosen with probability p and $w-1$ with probability $1-p$. Additionally, since the set B is randomized for each new recoding, the two patterns can actually appear for the same sequence of bits.

3.3 Efficiency and Comparisons

Next, we describe the advantages of our fractional technique in terms of speed and memory, and compare its performances with that of commonly used exponentiation methods. For a table size k and a bitlength n, k elements of n bits (in-

[注1]: Blinding alone is not sufficient, because it protects only against DPA, not SPA.

Algorithm 2: Conversion to unsigned representation

```
INPUT: $\mathcal{L}+2$-bit exponent $d=(10d_{\mathcal{L}-1}\dots d_0)_2$, table size $k$, index set $B$;
OUTPUT: Recoded exponent $(u_{\mathcal{L}}\dots u_0)$;

(1) $i \leftarrow 0$; $\gamma \leftarrow 0$; $w \leftarrow \lceil \log_2(k) \rceil$;
(2) while $i \leq \mathcal{L}-w$ do
    (a) $x \leftarrow (d_{i+w-1}\dots d_i)_2 - \gamma$; $y \leftarrow (d_{i+w-2}\dots d_i)_2 - \gamma$;
    (b) if $x \leq 0$ then $x \leftarrow x + 2^w$; $\gamma_x \leftarrow 1$; else $\gamma_x \leftarrow 0$;
    (c) if $y \leq 0$ then $y \leftarrow y + 2^{w-1}$; $\gamma_y \leftarrow 1$; else $\gamma_y \leftarrow 0$;
    (d) if $x \leq 2^{w-1}$ then
        i. $rnd \leftarrow$ generate $w-1$ random bits;
        ii. if $rnd < k - 2^{w-1}$ then $u_i \leftarrow x$; $\gamma \leftarrow \gamma_x$; $r \leftarrow w$;
        iii. else $u_i \leftarrow y$; $\gamma \leftarrow \gamma_y$; $r \leftarrow w-1$;
    (e) else if $x \in B$ then $u_i \leftarrow x$; $\gamma \leftarrow \gamma_x$; $r \leftarrow w$;
    (f) else $u_i \leftarrow y$; $\gamma \leftarrow \gamma_y$; $r \leftarrow w-1$;
    (g) $u_{i+1} \leftarrow 0, \dots, u_{i+r-1} \leftarrow 0$; $i \leftarrow i + r$;
(3) if $i < \mathcal{L}$ then
    (a) $u_i \leftarrow (d_{\mathcal{L}-1}\dots d_i)_2 - \gamma$;
    (b) if $u_i \leq 0$ then $u_i \leftarrow u_i + 2^{\mathcal{L}-i}$; $\gamma \leftarrow 1$; else $\gamma \leftarrow 0$;
    (c) $u_{i+1} \leftarrow 0, \dots, u_{\mathcal{L}-1} \leftarrow 0$;
(4) $u_{\mathcal{L}} \leftarrow 2 - \gamma$; return $(u_{\mathcal{L}}\dots u_0)$;
```

cluding the basis of the exponentiation g are pre-computed and stored in RAM. In contrary to other methods, which only allows 1, 2, 4, $\dots, 2^w, \dots$ as table size, the fractional window method is much more flexible: any table size can be chosen. This is not only an advantage to fully occupy the available memory on constrained environments, but also means that the optimal table size k can be chosen on large-memory profiles: in that case, the fractional width method yields an exponentiation method which is faster than integral width techniques.

Recall that the upper width w is defined as $w=\lceil \log_2(k) \rceil$, and that probability of choosing the width w rather than $w-1$ is $p=\frac{k}{2^{w-1}}-1$. Multiplications occur only when there is a nonzero digit in the representation, therefore, for an n-bit exponent, there are on average $n/(w-1+p)$ multiplications. Then, the memory and average computational cost of exponentiations based on the unsigned fractional representations are as follows:

$$\begin{cases} \mathcal{M}_F = k \cdot \mathcal{L} \text{ bits} \\ \mathcal{C}_F = \left(2^{w-2}+\mathcal{L}\right) S + \left(k - 2^{w-2} - 1 + \frac{\mathcal{L}}{w-1+p}\right) M \end{cases} \quad (1)$$

where S and M stand for the cost of squares and multiplications, respectively, $w=\lceil \log_2(k) \rceil$ and $p=\frac{k}{2^{w-1}}-1$.

The sliding window method is often utilized for practical implementations of exponentiations; for instance, OpenSSL uses the sliding window with 16 pre-computed values $\{g, g^3, g^5, \dots, g^{31}\}$ [9]. The idea of the sliding window is to consider odd pre-computed values only, reducing the size of the table and the number of multiplications. For the same memory, the sliding window is faster than the SPA-resistant fractional window. However, our fractional window method benefits from a higher flexibility. We implemented both techniques with the NTL library [10] and compared the algorithms running with their optimal parameters on our platform; Table 1 summarizes our implementation results. It comes out that, although the fractional width is slightly slower than the sliding window, the performance drop is very small in practice: only 5% for 512-bit and 1024-bit exponentiations and when the algorithms run in their optimal settings. For the same memory consumption, our method is 7% slower than the sliding window 512-bit exponentiation, and 5% than the 1024-bit exponentiation.

表 1 Memory and cost of 1024-bit exponentiation methods

Binary method	0 bytes	15.05 ms
Sliding window, $w=6$	4,096 bytes	11.49 ms
Our technique, $k=32$	4,096 bytes	12.09 ms
Our technique, $k=53$	6,784 bytes	12.05 ms

4. Security Analysis

We now analyze the security of the unsigned SPA-resistant fractional width technique in the sense of SPA.

4.1 Non-uniform Digit Distribution

It has been shown that the signed fractional width recoding leaks some information about the secret [8]. The reason for this (relatively small) information leakage is the *degenerated* width w pre-computed table: the fact that some pre-computed values are missing in the table can be used to speed up attacks. Obviously, since it uses exactly the same principle for its recoding (but with a different digit set), the unsigned fractional width algorithm (Algorithm 2) can be targeted by this kind of attack as well. In the following, we refine the technique described in [8] by considering the influence of the degenerated table not only on blocks of bits recoded with the width w but also $w-1$.

4.1.1 Non-uniformity when w is selected.

The first point is that when the width w is selected rather than $w-1$, some digits do not appear in the recoding [8]. Indeed, to construct a length-k pre-computed table, one takes a length-2^w table and remove some values at random from its upper half part. Therefore, the exponents of the removed pre-computed values are absent from the representation. In other words, each time the width w is selected, there are only k possible digits instead of 2^w. For instance, consider $k=3$; from the length-4 table $\{g^1, g^2, g^3, g^4\}$, one value is chosen among g^3 and g^4, and removed from the table. In our example, we remove g^3. Then, our representation admits the following digit set: $B=\{1,2,4\}$.

4.1.2 Non-uniformity when $w-1$ is selected.

The second point is that the distribution of digits recoded with the width $w-1$ is not uniform. This problems occurs because when a missing digit is scanned in the secret (case where $x > 2^{w-1}$ and $x \notin B$), the width $w-1$ is selected. But this event occurs only for some digits x, and the corresponding value of y is uniquely determined by the relationship $x = y + 2^{w-1}d_{i+w-1}$. Since there are only $2^w - k$ for $x \notin B$, there are also $2^w - k$ possible values for y (instead of 2^{w-1}). Therefore the case where $x > 2^{w-1}$ and $x \notin B$ introduces non-uniformity in the distribution of digits recoded with the width $w-1$. Assume for instance a table size $k = 5$ and digit set $B = \{1, 2, 3, 4, 5\}$. Suppose that during the recoding, $x = 6$ is computed. Now, there are two possibilities: either $(d_{i+2}d_{i+1}d_i)_2 = 6$ and the carry is zero ($\gamma = 0$), or $(d_{i+2}d_{i+1}d_i)_2 = 7$ and the carry is not zero ($\gamma = 1$). In both case, $(d_{i+1}d_i)_2 - \gamma = y = 2$. But when $x \leqq 4$ and $w - 1 = 2$ is (randomly) selected, the distribution of the four digits $\{1, 2, 3, 4\}$ is uniform. Consequently, the digit 2 has a higher probability than 1, 3 and 4 when the width $w - 1 = 2$ is selected. Note that although we concentrate on the case of the unsigned SPA-resistant fractional window in this paper, the signed technique shares similar properties.

4.2 Side-Channel Information and Entropy

Although the non-uniformity of the digit distribution is intuitively a problem, even when the bias is known, it is difficult to evaluate how serious the threat is. To explicitly evaluate SPA-resistance of the fractional width techniques, the calculation of the entropy of such representations is helpful.

[Definition 1] (Side-Channel Entropy) Let p be a probability function $p : \{0, 1, \ldots, 2^{\mathcal{L}} - 1\} \rightarrow 0..1$ with $\sum_{\delta \in \{0,1,\ldots,2^{\mathcal{L}}-1\}} p(\delta) = 1$, representing a bias in the sea r ch space $\{0, 1, \ldots, 2^{\mathcal{L}} - 1\}$ obtained by side-channel analysis. We call side-channel information entropy the term:

$$S = -\sum_{\delta \in \{0,1,\ldots,2^{\mathcal{L}}-1\}, p(\delta) \neq 0} p(\delta) \log_2(p(\delta)). \tag{2}$$

Assume for example that an attacker tries to find a secret $d \in \{0, 1, 2, 3\}$. Without the help of side-channel analysis, all candidates in the search space are equiprobable: $p(0) = p(1) = p(2) = p(3) = 1/4$ and the entropy is $S = 2$ bits. But imagine now that the attacker identified a feature in side-channel information which is more likely to occur for $\delta = 3$ than for the other candidates. For instance, $p(3) = 1/2$, and $p(0) = p(1) = p(2) = 1/6$. In that case, the entropy is reduced to $S = 1.79$ bits. More generally, when all values for d are equiprobable even when SPA information is available, the entropy reaches its maxima, namely $S = \mathcal{L}$ bits. In that case, the representation is a perfect SPA countermeasure. When the values are not uniformly distributed (some values are forbidden or simply less probable), $S < \mathcal{L}$ bits. One interpretation of the entropy is the number of bits that remain secure (unknown to the attacker) despite side-channel information leakage.

Note that fractional window methods have two distinct groups of events. First, the selection of the pre-computed table; in this case, we assume that the Ω_B possible choices for the pre-computed table are *equiprobable* and *indistinguishable* by SPA. And second, the recoding itself, where some choices between the widths w and $w-1$ are random. We call S_w and S_{w-1} the entropy of w-blocks (that is, a block of bits recoded with width w) and $w-1$-blocks. Then, we abuse notations and call "entropy of the fractional width recoding" the term $S = \log_2 |\Omega_B| + \mathcal{L}_w S_w + \mathcal{L}_{w-} S_{w-1}$, where $\mathcal{L}_w$ and $\mathcal{L}_{w-1}$ are the number of w-blocks and $w-1$-blocks. In other words, we define the entropy of the fractional width recoding as the sum of the entropy of the pre-computed table and the entropy of each block. For an $\mathcal{L}$-bit secret key recoded with our fractional width method, the average entropy $\overline{S_F}$ and the worst entropy $\tilde{S_F}$ are:

$$\begin{cases} \overline{S_F} = \log_2 |\Omega_B| + \frac{\mathcal{L}}{w-1+p} \left(pS_w + (1-p)\, S_{w-1}\right) \\ \tilde{S_F} = \log_2 |\Omega_B| + \min\left(\frac{\mathcal{L}}{w} S_w, \frac{\mathcal{L}}{w-1} S_{w-1}\right) \end{cases} \tag{3}$$

We first evaluate the contribution of the randomized pre-computed table to the entropy of the fractional width method. There are $k - 2^{w-1}$ exponents randomly chosen in the set of 2^{w-1} elements $\{2^{w-1} + 1, 2^{w-1} + 2, \ldots, 2^w\}$. Therefore, the entropy of the pre-computed table is:

$$\log_2 |\Omega_B| = \sum_{i=k-2^{w-1}}^{2^{w-1}} \log_2 i - \sum_{i=1}^{2^w - k} \log_2 i. \tag{4}$$

Then, we study the entropy of digits recoded with the width w. We consider the digits from the upper half table $x > 2^{w-1}$ first. If $x > 2^{w-1}$ (probability 1/2), since w has been chosen, $x \in B$: there are only $k - 2^{w-1}$ digits in the upper half table. Thus, when w is selected, the probability of a digit from the upper half table is $\frac{1}{2(k-2^{w-1})}$. Next, we consider digits from the lower half table ($x \leqq 2^{w-1}$). When $x \leqq 2^{w-1}$ (probability 1/2), since the lower half of the table is full, there are 2^{w-1} possible digits. Therefore, in the lower half table, the digits have a probability of $\frac{1}{2 \cdot 2^{w-1}}$. Clearly, digits in the upper half table have a greater probability than digits in the lower half table. This difference is the origin of entropy loss when w is selected. More precisely, the entropy of a block of width w is:

$$S_w = \frac{w+1}{2} + \frac{1}{2} \log_2 \left(k - 2^{w-1}\right) \tag{5}$$

Note that in the case of a perfect countermeasure, we have $S_w = w$ bits.

The case of the width $w-1$ is slightly different. When $x \leq 2^{w-1}$ (probability 1/2), the choice among the 2^{w-1} digits is uniform. But when $x > 2^{w-1}$ (probability 1/2), among the 2^{w-1} possible digits, only $2^w - k$ can be selected (because $x \notin B$). Therefore, $k - 2^{w-1}$ digits have a probability of appearance of $\frac{1}{2}\frac{1}{2^{w-1}}$, whereas $2^w - k$ digits have a probability of appearance of $\frac{1}{2}(\frac{1}{2^{w-1}} + \frac{1}{2^w - k})$. As a consequence, the entropy of a block of width $w-1$ is:

$$S_{w-1} = \frac{k-2^{w-1}}{2^w} w - \frac{2^w+2^{w-1}-k}{2^w} \log_2\left(\frac{1}{2^w} + \frac{1}{2^{w+1}-2k}\right) \quad (6)$$

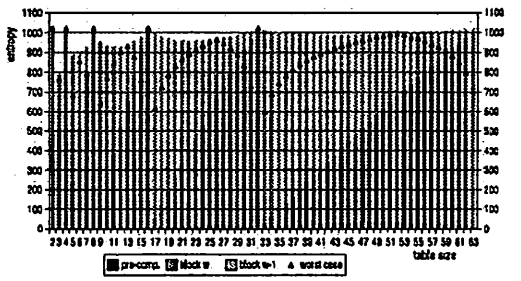

図 1 Average and worst entropy, 1024 bits

4.3 Consequences on Security

Despite the entropy loss of the unsigned fractional window method, we claim that the representation is still a good SPA countermeasure. Indeed, one can interpret entropy as the equivalent bitlength of the secret when the SPA information can be fully utilized. Therefore, in practice, an SPA countermeasure is at least as strong as its SPA information entropy. As a consequence, if the worst-case entropy is large enough, we know for sure that the secret data is safe. With our technique, if the table length k is carefully chosen, the entropy granted by the unsigned fractional width method is sufficient to thwart all known attacks against partially-compromised RSA secret keys. In particular, we remark that in Fig. 1, for from $k = 37$ to $k = 61$, the worst-case entropy is greater than 800 bits. Additionally, there is currently no attack which can take advantage of the bias of the fractional width technique: typically, to be effective, such attacks require the knowledge of the upper or lower bits, or consecutive bits. Thus, we argue that in practice, the security level of the unsigned fractional width technique is even higher than its SPA entropy.

Unfortunately, this is not the case for other exponentiation methods. The binary exponentiation has an entropy of 0 bits, because distinguishing squares from multiplications allows to fully revealing the secret key. In the case of the sliding window method, it is possible to distinguish blocks of consecutive zeros, where no multiplication occurs. Thus, the average entropy of the sliding window representation is $S_{SW} = \mathcal{L}/2$ bits. And in the worst case, the entropy can go as low as 0 bits.

5. Conclusion

We presented a new countermeasure for protecting modular exponentiations against side-channel attacks. Our countermeasure is inspired by the signed fractional width technique, but has an unsigned digit set, which is necessary for achieving high efficiency with RSA or DSA. Our countermeasure is not only highly secure, but is also efficient; thanks to its higher flexibility, our method is about as fast as commonly used exponentiation techniques such as the sliding window method. In fact, in optimal settings and in a practical situation, our method is only 5% slower than the sliding window exponentiation for 512-bit and 1024-bit moduli.

Acknowledgement

This work was partly supported by National Institute of Information and Communications Technology (NICT).

文 献

[1] Coppersmith, D.: Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known. In U. Maurer (Ed.): Advances in Cryptology–Eurocrypt'96. LNCS **1070**, pp.178-189. Springer-Verlag (1996).

[2] Coron, J.-S.: Resistance Against Differential Power Analysis for Elliptic Curve Cryptosystems. In Ç. Koç and C. Paar (Eds.): Cryptographic Hardware and Embedded Systems–CHES'99. LNCS **1717**, pp.292-302. Springer-Verlag (1999).

[3] Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. In M. Wiener (Ed.): Advances in Cryptology–Crypto'99. LNCS **1666**, pp.388-397. Springer-Verlag (1999).

[4] Kocher, P.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In N. Koblitz (Ed.): Advances in Cryptology–Crypto'96. LNCS **1109**, pp.104-113. Springer-Verlag (1996).

[5] Möller, B.: Securing Elliptic Curve Point Multiplication against Side-Channel Attacks. In G. Davida, Y. Frankel (Eds.): Information Security–ISC'01. LNCS **2200**, pp.324-334. Springer-Verlag (2001).

[6] Möller, B.: Improved Techniques for Fast Exponentiation. In P.J. Lee, C.H. Lim (Eds.): Information Security and Cryptology–ICISC'02. LNCS **2587**, pp.298–312. Springer-Verlag (2003).

[7] Okeya, K., Takagi, T.: A More Flexible Countermeasure against Side Channel Attacks Using Window Method. In C. Walter, Ç. Koç, C. Paar (Eds.): Cryptographic Hardware and Embedded Systems–CHES'03. LNCS **2779**, pp.397-410. Springer-Verlag (2003).

[8] Okeya, K., Takagi, T., Vuillaume, C.: On the Exact Flexibility of the Flexible Countermeasure Against Side Channel Attacks. In H. Wang, J. Pieprzyk, V. Varadharajan (Eds): Information Security and Privacy–ACISP'04. LNCS **3108**, pp.466-477. Springer-Verlag (2004).

[9] OpenSSL: The Open Source Toolkit for SSL/TLS. `www.openssl.org`

[10] Shoup, V.: NTL: A Library for Doing Number Theory. `www.shoup.net/ntl/`