IEICE TRANS. FUNDAMENTALS, VOL.E106–A, NO.3 MARCH 2023

PAPER Special Section on Information Theory and Its Applications

Information Leakage Through Passive Timing Attacks on RSA Decryption System*,**

Tomonori HIRATA†, ***Nonmember and*** **Yuichi KAJI**†a), ***Member***

SUMMARY A side channel attack is a means of security attacks that tries to restore secret information by analyzing side-information such as electromagnetic wave, heat, electric energy and running time that are unintentionally emitted from a computer system. The side channel attack that focuses on the running time of a cryptosystem is specifically named a "timing attack". Timing attacks are relatively easy to carry out, and particularly threatening for tiny systems that are used in smart cards and IoT devices because the system is so simple that the processing time would be clearly observed from the outside of the card/device. The threat of timing attacks is especially serious when an attacker actively controls the input to a target program. Countermeasures are studied to deter such active attacks, but the attacker still has the chance to learn something about the concealed information by passively watching the running time of the target program. The risk of passive timing attacks can be measured by the mutual information between the concealed information and the running time. However, the computation of the mutual information is hardly possible except for toy examples. This study focuses on three algorithms for RSA decryption, derives formulas of the mutual information under several assumptions and approximations, and calculates the mutual information numerically for practical security parameters.

key words: *timing attack, quantitative information flow analysis, RSA, mutual information, entropy*

1. Introduction

1.1 Background

A timing attack can be a serious issue to practical information systems. In a timing attack, an attacker observes the running time of a target program, and attempts to know about secret information that is concealed in the program. The attack becomes a fatal threat if the attacker can actively choose the input to the target program [1], [6], [11], and countermeasures such as *bucketing* and *blinding* are studied thereafter. The bucketing inserts dummy operations to hide real running time, but it slows down the computation, and furthermore, its effectiveness is questioned recently [12]. The blinding is another countermeasure that is effective to programs for numerical computations. With blinding, the computation is made in three-steps; blinding, the computation of a *core function*, and unblinding (see Fig. 1). In the case of RSA decryption $m = c^d \bmod n$, the blinding operation chooses a random r and sends $c' = r^e c \bmod n$ to the core function. The core function computes $m' = c'^d \bmod n$ as in the usual RSA decryption, and sends m' to the unblind operation. The unblind operation computes $r^{-1}m' \bmod n$ which equals to the correct value of m. With this blinding, an attacker cannot "control" the input to the core function, which contributes to deter an attacker from mounting an *active* (or chosen-input) timing attack. On the other hand, the running time of the decryption steps (blinding, core function and unblinding) still varies according to the decryption key d in the core function. By any chance, an attacker may be able to learn something about d from the running time, but the risk of such *passive* timing attacks has not been understood well.

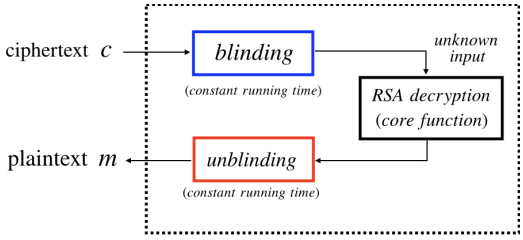

Fig. 1 The blinding structure.

To measure the risk of timing attacks, quantitative information flow analysis is used in general. Quantitative information flow analysis is a method for estimating the amount of confidential information leaked from programs based on the information theory.

1.2 Related Studies

This section briefly reviews preceding works made by Köpf et al. [7], [8] and Kobayashi [5]. They both use mutual information to give an upper bound on the amount of information that leaks through a timing attack.

1.2.1 Köpf's Studies

Köpf, Durmuth and Smith introduced an information theoretic framework to discuss this setting [7], [8]. Köpf et al. give an upper bound on the amount of information leakage of decryption key in the event of a timing attack on the RSA decryption system.

Let K be a random variable of the secret decryption key d, and Z be a random variable of the running time (such as the number of clocks) that was used by a single run of

Manuscript received February 14, 2022.
Manuscript revised June 25, 2022.
Manuscript publicized August 16, 2022.

†The authors are with Graduate School of Informatics, Nagoya University, Nagoya-shi, 464-8601 Japan.

*The preliminary result of this study has been presented in [4]. This manuscript contains the refined proof.

**This study was supported by JSPS Grant-in-Aid for Scientific Research KAKENHI 18K11162 and 21K11886.

a) E-mail: kaji.yuichi.a0@f.mail.nagoya-u.ac.jp

DOI: 10.1587/transfun.2022TAP0006

the RSA decryption program in Fig. 1. We cannot predict what kind of analysis/computation an attacker may perform, but he/she should not be able to learn information that is more than $I(K;Z)$, the mutual information between K and Z. By definition $I(K;Z) = H(Z) - H(Z|K)$, but it is not easy to compute $H(Z|K)$ because the computation needs precise analysis of the relation between the key and the running time of a complicated RSA decryption algorithms. For this reason, [7], [8] ignore the conditional entropy, and focus on $H(Z)$ as an upper-bound of $I(K;Z)$ (and of the information that an attacker may learn), that is,

$$I(K;Z) \le H(Z).$$

Since the attacker may be able to make multiple observations of the running time, say n times, the amount of leaked information of decryption key can be expressed as $I(K;Z^n)$. Usually, the invocations of the RSA decryption can be regarded as independent with each other, and the discussion is simplified by focusing on a relative frequency of running times rather than the "ordered sequence" of the observed running times. In [7], [8], Köpf et al. define "type" as the relative frequency of realizations $(z_1, z_2, \cdots, z_n)$ of the clock sequence $(Z_1, Z_2, \cdots, Z_n)$ that is obtained by multiple observations of the running time. The type is represented by an m-integer tuple $t[z] = (t_1, t_2, \cdots, t_m)$ where m is the number of possible values of Z. Let assume for example that Z takes either of 5, 10, 15 or 20 as its realized value ($m = 4$ in this case). The type of $z = (15, 10, 15, 20, 15, 5, 15, 10)$ is defined as $t[z] = (1, 2, 4, 1)$ because z contains 1 occurence of the first value 5, 2 occurences of the second value 10, and so on. Since observation times are random variables, the type can be regarded as a random variable which we denote by $T_{m,n}$ hereafter. Since the order of the observed execution times does not affect the mutual information, the equation $I(K;Z^n) = I(K;T_{m,n})$ holds, and

$$I(K;Z^n) \le H(T_{m,n})$$

holds.

Köpf assumes that the types follow a uniform distribution in order to simplify the discussion. Under the assumption, the equation $H(T_{m,n}) = \log_2 |T_{m,n}|$ holds, and

$$I(K;Z^n) \le \log_2 |T_{m,n}|$$

holds. In [7], Köpf introduces a naive upper-bound $|T_{m,n}| \le (n+1)^{|Z|}$. Therefore, the inequation

$$I(K;Z^n) \le |Z| \log_2(n+1)$$

can be derived. In [8], Köpf et al. derive the equation $|T_{m,n}| = \binom{n+|Z|-1}{n}$, and a tighter upper bound

$$I(K;Z^n) \le \log_2 \binom{n+|Z|-1}{n}.$$

1.2.2 Kobayashi's Studies

Kopf et al. assumed that the type $T_{m,n}$ obeys uniform distribution. The assumption makes discussion simple, but it brings a quite loose and useless upper bound of $H(T_{m,n})$. Kobayashi noticed that $T_{m,n}$ indeed obeys a multinomial distribution, and obtained a better upper bound of $H(T_{m,n})$ by using an asymptotic formula of the entropy of multinomial distributions [5]. Because $T_{m,n}$ obeys a multinomial distribution, we can write

$$P_{T_{m,n}}(t_1, \cdots, t_m) = \binom{n}{t_1, \cdots, t_m} p_1^{t_1} \cdots p_m^{t_m}$$

where $\binom{n}{t_1, \cdots, t_m} = \frac{n!}{t_1! \cdots t_m!}$, p_i is the probability of obtaining the i-th execution result, and $t_1 + \cdots + t_m = n$. There is no known closed-form formula of the entropy of multinomial distribution, but Cichoń et al. [2] derived an asymptotic approximation of the entropy of the multinomial distribution as

$$H(T_{m,n}) \approx \frac{1}{2} \log_2 \left((2\pi n e)^{m-1} p_1 \cdots p_m \right) + \frac{1}{12n} \left(3m - 2 - \sum_{j=1}^{m} \frac{1}{p_j} \right)$$

that is good for $n \to \infty$. Kobayashi determined $p_1, \ldots, p_m$ experimentally, and derived a tighter upper bound on the amount of leaked information than in previous studies.

However, Kobayashi also evaluates the amount of information leakage about the decryption key ignoring the conditional entropy term $H(Z^n|K)$. That is, the upper bound of $I(K;Z^n) = I(K;T_{m,n})$ is somewhat loose.

1.3 Contribution of This Study

In the authors' recognition, an essential aspect of timing attacks is perished if $H(Z|K)$ is ignored, because the attack is based on the fact that the uncertainty of the key is reduced by observing the running time of the target. The main goal of this study is to derive both of $H(Z)$ and $H(Z|K)$ for three programs, namely, *Binary method*, *ModBin method* and *CRT-ModBin method* for RSA decryption. The derivation is not very strict in the sense that we use several assumptions and approximations, but our experiments supports that the assumptions and approximations are feasible and reasonable for practical choices of parameters (i.e. a key of several hundred or thousand bits). The formulas can be used to compute $I(K;Z)$, the amount of information that leaks through the running time of the program. Table 1 summarizes the numerical results of the computation of $I(K;Z)$ for the three algorithms with key length l from 128 to 2048. The table will be reviewed in later sections. From this table, we confirmed that the amount of decryption key information leaked by the timing attack is negligible. It is also noted that numerical results of this kind enable us to compare the security of different algorithms and different implementations, which has been made heuristically so far and caused a lot of confusion. The approach and the result of study give theoretical basement to that kind of discussion, and contribute to precise understanding of implementations of cryptosystems.

Table 1 Numerical calculations of $I(K;Z)$.

	Binary	ModBin		CRT-ModBin
l	$I(K;Z)$	$H(Z)$	$I(K;Z)$	$I(K;Z_p,Z_q)$
128	4.6311	8.4684	3.2113	5.5556
256	5.0907	9.8066	3.7143	6.4226
512	5.5693	11.2205	4.2479	7.4286
1024	6.0583	12.6755	4.7879	8.4959
2048	6.5527	-	-	9.5758

Algorithm 1 Binary method algorithm

Input: $c, d = (d_{t-1} \cdots d_0)_2, n$
Output: $m = c^d \bmod n$
1: $m = c$
2: **for** $i = t-2$ to 0 **do**
3: $\quad m = m^2 \bmod n$
4: $\quad$ **if** $d_i = 1$ **then**
5: $\quad\quad m = mc \bmod n$
6: $\quad$ **end if**
7: **end for**
8: return m

2. Preliminary

2.1 RSA Decryption Algorithm

The RSA decryption is realized by a computation

$$m = c^d \bmod n \tag{1}$$

where n is a product of two primes p and q, d is a decryption key, and c and m are a ciphertext and a plaintext, respectively. Throughout this paper, we let l be the bit length of the modulus n. In practice, n is very large and several algorithms have been investigated for efficient computation of (1). Among such algorithms, we focus Binary method (Algorithm 1), ModBin method (Algorithm 3) and CRT-ModBin method (Algorithm 4).

In these algorithms, the computation is carried out based on the binary representation $d_{t-1} \cdots d_0$ of the decryption key d, where t is the most significant nonzero bit of the l-bit binary representation of d (i.e. $d_{l-1} = \cdots = d_t = 0$ and $d_{t-1} = 1$), and called the *actual key length* of d. The ModBin method utilizes a Montgomery reduction (Algorithm 2) and a predetermined constant r that satisfies $n < r$ and $\gcd(r,n) = 1$ (the choice of r is arbitrary but it is common to take r as a power of 2). The algorithm also uses two additional constants $r_2 = r^2 \bmod n$ and $n' = (-1) \cdot n^{-1} \bmod r$. In the CRT-ModBin method, we are using $d_p = d \bmod (p-1)$, $d_q = d \bmod (q-1)$, and q^{-1} which is the integer satisfying $qq^{-1} \equiv 1 \bmod p$.

2.2 Notations and Assumptions

We would like to estimate how much information of d leaks through a running time of the RSA decryption program. It is assumed that the decryption is implemented with the blinding technique, and that the blinding and unblinding always take constant running time. This means that the variation of the running time of the program all comes from the core function. An attacker knows the algorithm that is used in the core function, but neither control nor know the input that is given to the core function.

Algorithm 2 Montgomery reduction MR(x)

Input: x, n, n', r
Output: The result of Montgomery reduction of x
1: $m_1 = (\, x \bmod r \,)\, n' \bmod r$
2: $m_2 = (\, x + m_1 n \,) \,/\, r$
3: **if** $m_2 < n$ **then**
4: $\quad$ return m_2
5: **else**
6: $\quad$ return $m_2 \bmod n$
7: **end if**

Algorithm 3 ModBin method

Input: $c, d = (d_{t-1} \cdots d_0)_2, n, r_2$
Output: $m = c^d \bmod n$
1: $c' = \mathrm{MR}(cr_2)$
2: $m' = c'$
3: **for** $i = t-2$ to 0 **do**
4: $\quad m' = \mathrm{MR}(m'm')$
5: $\quad$ **if** $d_i = 1$ **then**
6: $\quad\quad m' = \mathrm{MR}(m'c')$
7: $\quad$ **end if**
8: **end for**
9: $m = \mathrm{MR}(m')$
10: return m

Algorithm 4 CRT-ModBin method

Input: $c, p, q, d_p, d_q, q^{-1}, n$
Output: $m = c^d \bmod n$
1: $c_p = c \bmod p$
2: $c_q = c \bmod q$
3: $m_p = \mathrm{ModBin}(c_p, d_p, p)$
4: $m_q = \mathrm{ModBin}(c_q, d_q, q)$
5: return $m_q + q(q^{-1}(m_p - m_q) \bmod p)$

For information theoretical discussion, we use K as the random variable of the decryption key d, and T and W as the random variables of the actual key length and the Hamming weight of d, respectively. And we assume that ciphertexts are generated randomly. At the beginning, an attacker has no information of d, which we model as the following Assumption 1.

Assumption 1: The decryption key K distributes uniformly in the space of l-bit binary integers.

We note that Assumption 1 is not very precise, as an attacker should know that d is odd and $d_0 = 1$. It is not difficult to accommodate this condition in the discussion, but we assume for simplicity that d_0 is not known to an attacker. With this assumption, we have, for example,

$$P_{T,W}(t,w) = \binom{t-1}{w-1} \cdot 2^{-l}. \tag{2}$$

Consider that either of Algorithms 1, 3 or 4 is im-

plemented in the core function of an RSA decryption program. Usually, the word size of a processor is much smaller than the key length l, and the multiplication of l-bit integers requires considerably longer time than simple atomic operations. Similar attention is needed for the modulo computation a mod b for two l-bit integers a and b; if $a \geq b$, then we have to perform a *reduction* (the computation of the remainder) and the computation of a mod b takes long time; if $a < b$, on the other hand, then the reduction is not necessary and almost no computation is needed to perform a mod b. These observation suggests that the running time of the program depends on the number of multiplications and the number of reductions that are performed in the program. Let Y_1 and Y_2 be random variables of the numbers of multiplications and reductions that are performed in the program, respectively, and Z be a random variable of the number of clocks consumed by the program (for simplicity, we may call Z a *running time*).

Assumption 2: We have $Z = c_0 + c_m Y_1 + c_r Y_2$, where c_0 is a constant that corresponds to the number of clocks for the blinding, unblinding, and other atomic operations in the core function, c_m and c_r are the numbers of clocks for a multiplication and a reduction, respectively.

The constant c_0 is irrelevant to information theoretical discussion, and we let $c_0 = 0$ for the simplicity and assume $Z = c_m Y_1 + c_r Y_2$ hereafter.

It is understood from the pseudo-programs that Y_1 and Y_2 are determined from T and W, and the following lemma holds.

Lemma 2.1: $I(K;Z) = I(T,W;Z)$

Proof: For the Markov chain $K \Rightarrow (T,W) \Rightarrow (Y_1,Y_2) \Rightarrow Z$ (Fig. 2), the data processing inequality (Lemma 2.8.1 of [3]) guarantees that $I(K;Z) \leq \min\{I(K;T,W), I(T,W;Z)\}$. The values of T and W are uniquely determined for a given value of K, and therefore $H(T,W|K) = 0$. It follows that

$$\begin{aligned} I(T,W;Z) &= H(T,W) - H(T,W|Z) \\ &\leq H(T,W) \\ &= H(T,W) - H(T,W|K) \\ &= I(K;T,W) \end{aligned}$$

and we have $I(K;Z) \leq I(T,W;Z)$. The equality holds if and only if $I(T,W;Z|K) = 0$, which is true in our case because

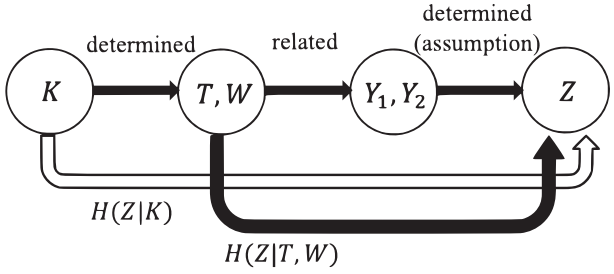

Fig. 2 Bayesian network for some random variables.

$I(T,W;Z|K) = H(T,W|K) - H(T,W|K,Z) = H(T,W|K) - H(T,W|K) = 0$ (note: our discussion corresponds to the case of $I(Y;Z) \geq I(X;Z)$ in the proof of Lemma 2.8.1 of [3] whose equality holds if and only if $I(Y;Z|X) = 0$). □

3. Binary Method

The probability distributions of Y_1 and Y_2 in the Binary method (Algorithm 1) are investigated first. Let t and w be realized values of T and W, respectively. It is easily understood from the pseudo-program that the number of multiplications Y_1 always equal to $(t-1)+(w-1)$.

The number of reductions Y_2 is not obvious because modulo computations do not always invoke reductions. Instead of tracing the values of m^2 (at line 3) and mc (at line 5) precisely, we put the following simple assumption and regard the modulo computations as Bernoulli trials.

Assumption 3: At lines 3 and 5 of the Binary method (Algorithm 1), the value of m distributes uniformly in $\{0,\ldots,n-1\}$.

Write $Y_2 = Y_{2,1} + Y_{2,2}$ where $Y_{2,1}$ and $Y_{2,2}$ denote the number of reductions that are performed at lines 3 and 5 of the program, respectively.

Lemma 3.1: If $n \to \infty$, then $P_{Y_{2,1}|T,W}(t-1|t,w) = 1$ and $P_{Y_{2,2}|T,W}(w-1|t,w) = 1$ (and hence Y_2 is always $t+w-2$).

Proof: A reduction is performed at line 3 if $m^2 \geq n$, which is an event with probability $1 - 1/\sqrt{n} \to 1$ by Assumption 3. Since line 3 is visited $t-1$ times, $Y_{2,1} = t-1$ if $n \to \infty$.

The discussion is little bit complicated for line 5 because of the uniformly distributing c. In the fifth line, a reduction is not needed if $mc < n$, which holds with probability 1 if $c = 0$, and with probability $1/c$ if $c > 0$. This means that the distribution of $Y_{2,2}$ is defined as a superposition of n different binomial distributions that correspond to n different values of c that are equiprobable;

$$P_{Y_{2,2}|T,W}(y|t,w) = \frac{1}{n}\cdot 0 + \frac{1}{n}\sum_{c=1}^{n-1} Bin[w-1, y, 1-(1/c)]$$

where $Bin[a,b,p] = \binom{a}{b} p^b (1-p)^{a-b}$.

Next consider $P_{Y_{2,2}|T,W}(w-1|t,w)$ which is

$$\begin{aligned} &\frac{1}{n}\sum_{c=1}^{n-1}\left(1-\frac{1}{c}\right)^{w-1} = 0 + \frac{1}{n}\sum_{c=2}^{n-1}\left(1-\frac{1}{c}\right)^{w-1} \\ &\geq \frac{1}{n}\sum_{c=2}^{n-1}\int_{c-1}^{c}\left(1-\frac{1}{x}\right)^{w-1} dx \\ &= \frac{1}{n}\int_{1}^{n-1}\left(1-\frac{1}{c}\right)^{w-1} dc \\ &= \frac{1}{n}\int_{1}^{n-1}\sum_{i=0}^{w-1}\binom{w-1}{i}\left(-\frac{1}{c}\right)^{w-1-i} dc \end{aligned}$$

$$= \frac{1}{n}\sum_{i=0}^{w-1}\left(\binom{w-1}{i}\int_1^{n-1}\left(-\frac{1}{c}\right)^{w-1-i}dc\right)$$
$$= \frac{1}{n}\sum_{i=0}^{w-3}\left(\binom{w-1}{i}\left[-\frac{1}{i+2-w}\left(-\frac{1}{c}\right)^{w-2-i}\right]_1^{n-1}\right)$$
$$+ \frac{1}{n}\binom{w-1}{w-2}[-\log c]_1^{n-1}$$
$$+ \frac{1}{n}\binom{w-1}{w-1}\left[-\left(-\frac{1}{c}\right)^{-1}\right]_1^{n-1}.$$

Note that $\left(1-\frac{1}{c}\right)^{w-1} \geq \int_{c-1}^{c}\left(1-\frac{1}{x}\right)^{w-1}dx$ holds because $f(x) = \left(1-\frac{1}{x}\right)^{w-1}$ is a monotonically increasing function for $x > 0$. If $n \to \infty$, then the first term (the summation) and second term in the above formula diminish to zero because $1/n \to 0$. The third term remains undiminished, and we can write

$$P_{Y_{2,2}|T,W}(w-1|t,w) \geq \frac{n-2}{n} \to 1.$$

This lower bound and a trivial upper bound $P_{Y_{2,2}|T,W}(w-1|t,w) \leq 1$ implies that $P_{Y_{2,2}|T,W}(w-1|t,w) = 1$ if $n \to \infty$, and therefore $Y_{2,2} = w-1$. □

The discussion of Y_2 involves some assumptions and approximations. To verify the feasibility of the discussion, an experiment is conducted. We selected decryption keys with $t = 1024$ and $w = 512$, run the Binary method with these keys for randomly generated 10,000 ciphertexts, and counted the number of reductions actually performed in each run of the program. With this experiment, we confirmed that the number of reductions is $1534 = t + w - 2$ in all of the 10,000 trials for all keys (see Fig. 3). The verification is also made for keys with different parameters, and all the results supports that the assumptions and approximations are feasible. Based on this verification, we let $P_{Y_2}(t+w-2) = 1$ in the following discussion.

If n is sufficiently large, then T and W decide the values of Y_1 and Y_2 uniquely, and hence decide the value of Z uniquely also. This implies that $H(Z|T,W) = 0$ and $I(T,W;Z) = H(Z)$. To compute $H(Z)$, we clarify the probability distribution of Z. With Assumption 2, we have $Z = c_m Y_1 + c_r Y_2 = (t+w-2)(c_m+c_r)$. If $Z = z$, then t and w must satisfy $t + w = 2 + z/(c_m+c_r)$, from which $t \geq 1 + z/(2(c_m+c_r))$ and $w = 2 - t + z/(c_m+c_r)$. Consequently

$$P_Z(z) = \sum_{t=1+z/(2(c_m+c_r))}^{l} P_{T,W}(t, 2-t+z/(c_m+c_r))$$
$$= 2^{-l}\sum_{t=1+z/(2(c_m+c_r))}^{l}\binom{t-1}{1-t+z/(c_m+c_r)}$$

by (2). With this formula, we can compute $H(Z) = I(K;Z)$ in the Binary method. Take $c_m = c_r = (l/32)^2$ as in [7], and the mutual information $I(K;Z) = H(Z)$ is computed as in Table 1.

We note that $H(Z)$ has been recognized as an upper-bound of $I(K;Z)$ in the discussion of Köpf and Durmuth for [7] and Kobayashi for [5], but it turned out that $H(Z)$ coincides with $I(K;Z)$ in the Binary method. It is also noted that, in the Binary method, the realized value of Z does not change as far as the same decryption key is used. This implies that an attacker is not able to obtain additional information even if he/she makes multiple observations of the running time, and $I(K;Z^s) = H(Z)$ for an arbitrary s.

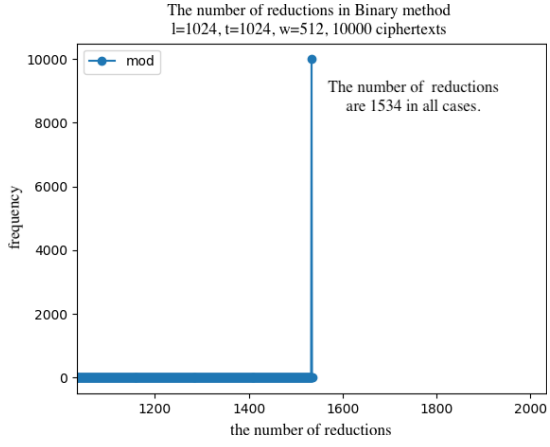

Fig. 3 The number of Y_2 in Binary method.

4. ModBin Method

In the ModBin algorithm (Algorithm 3), the function MR is called $t + w$ times in total. The function call of MR at lines 1, 4 and 6 invokes three multiplications each (one to compute the argument and two inside of MR), and the function call of MR at line 9 (which is visited only once) invokes two multiplications only. Therefore, the total number of multiplications is always $y_1 = 3(t+w) - 1$.

We move our attention to the number of reductions. For simplicity, we ignore two reductions in the lines 1 and 9 of Algorithm 3 because their contribution to the running time is little. In the ModBin algorithm, all modulo computations are performed in the MR algorithm; two modulo computations at line 1 and one modulo computation at line 6. Similarly to the previous section, we assume the following.

Assumption 4: At lines 4 and 6 of the ModBin algorithm (Algorithm 3), m' distributes uniformly in $\{0,\ldots,n-1\}$.

Unfortunately, the assumption does not simplify the discussion drastically as in the previous section. The number of reductions that are performed in MR($m'm'$) and MR($m'c'$) are different in general because $m'm'$ and $m'c'$ obey different distributions. For detailed discussion, let $Y_2 = Y_{2,1} + Y_{2,2} + \cdots + Y_{2,6}$, where $Y_{2,1}$ and $Y_{2,2}$ are random variables of the number of reductions that are performed by modulo computations (x mod r) and (($\cdot$)n' mod r) at line 1 of MR($m'm'$), respectively, $Y_{2,3}$ denotes the number of reductions that are performed by (m_2 mod n) at line 6 of

MR($m'm'$), and $Y_{2,4}, Y_{2,5}$ and $Y_{2,6}$ denote corresponding numbers in MR($m'c'$).

Lemma 4.1: If $n \to \infty$, then $P_{Y_{2,1}|T,W}(t-1|t,w)$, $P_{Y_{2,2}|T,W}(t-1|t,w)$, $P_{Y_{2,4}|T,W}(w-1|t,w)$, and $P_{Y_{2,5}|T,W}(w-1|t,w)$ all approximate to 1, and

$$P_{Y_{2,3}|T,W}(y|t,w) = Bin[t-1, y, n/(3r)],$$
$$P_{Y_{2,6}|T,W}(y|t,w) = \frac{1}{n} \cdot \sum_{c'=0}^{n-1} Bin[w-1, y, c'/(2r)],$$

where $Bin[a,b,p] = \binom{a}{b} p^b (1-p)^{a-b}$.

Proof: The approximations for $Y_{2,1}$, $Y_{2,2}$, $Y_{2,4}$ and $Y_{2,5}$ can be shown similarly to Lemma 3.1. As for $Y_{2,3}$ and $Y_{2,6}$, the probability of $m_2 \geq n$ in MR($m'm'$) is $n/(3r)$, and that of MR($m'c'$) is $c'/(2r)$ [11]. Therefore, $P_{Y_{2,3}|T,W}(y|t,w)$ is the binomial distribution with probability $n/(3r)$, and $P_{Y_{2,6}|T,W}(y|t,w)$ is the superposition of binomial distributions where c' distributes uniformly. □

The computation of $P_{Y_{2,6}|T,W}$ can be done by replacing the sum by integration, and then by making some recursive computation based on a partial integration. First,

$$\begin{aligned} & P_{Y_{2,6}|T,W}(y|t,w) \\ &= \frac{1}{n} \sum_{c'=0}^{n-1} \binom{w-1}{y} \left(1 - \frac{c'}{2r}\right)^{w-1-y} \left(\frac{c'}{2r}\right)^{y} \\ &\approx \frac{1}{n} \int_0^{n-1} \binom{w-1}{y} \left(1 - \frac{x}{2r}\right)^{w-1-y} \left(\frac{x}{2r}\right)^{y} dx \\ &= \frac{2r}{n} \binom{w-1}{y} \int_0^{(n-1)/(2r)} (1-h)^{w-1-y} h^y dh. \end{aligned}$$

To proceed the computation further, we define $I(u,s) = \int_0^{\alpha} (1-h)^s h^u dh$ where α is a constant and s, u are variables. Using partial integration, $I(u,s)$ is expanded as follows:

$$I(u,s) = \alpha^{u+1}(1-\alpha)^s \cdot \frac{1}{u+1} + \frac{s}{u+1} I(u+1, s-1).$$

From the above result, we can compute $I(u,s)$ with only s recursion. With these investigations, we can calculate $P_{Y_{2,6}|T,W}(y|t,w)$ with manageable complexity (see Fig. 4).

Similarly to the previous section, we let $P_{Y_{2,1}|T,W}(t-1|t,w) = P_{Y_{2,2}|T,W}(t-1|t,w) = 1$ and $P_{Y_{2,4}|T,W}(w-1|t,w) = P_{Y_{2,5}|T,W}(w-1|t,w) = 1$ by assuming $n \to \infty$. With this simplification, we have

$$P_{Y_2|T,W}(y|t,w) = \sum_{y_3+y_6=y-2(t+w-2)} P_{Y_{2,3}|T,W}(y_3|t,w) P_{Y_{2,6}|T,W}(y_6|t,w). \quad (3)$$

The derivation of (3) is based on several assumptions and approximations. To verify if the derivation is reasonable or not, numerical experiments are conducted. Fig. 5 shows the distribution of $P_{Y_2|T,W}$ computed from (3) with

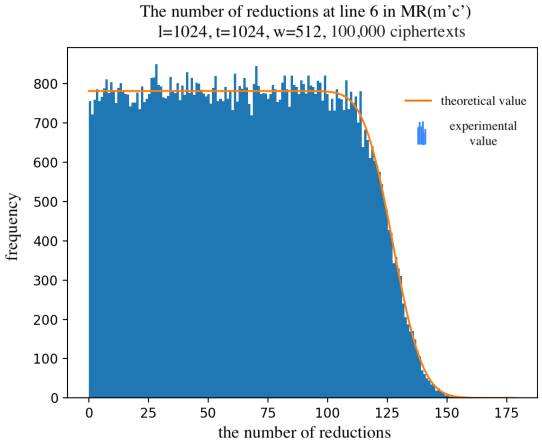

Fig. 4 Comparing $P_{Y_{2,6}|T,W}(y_6|t,w) \times 100,000$ and experiment value in ModBin method.

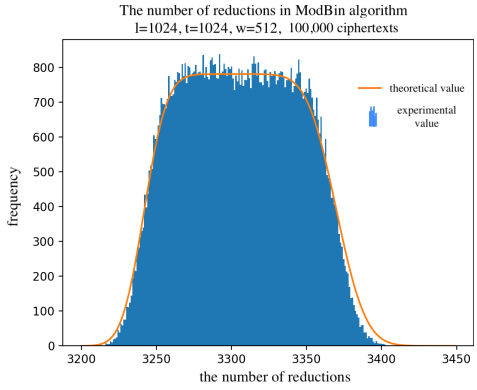

Fig. 5 Comparing $P_{Y_2|T,W}(y|t,w) \times 100,000$ and experiment value in ModBin method.

$l = 1024, t = 1024$ and $w = 512$, and the relative frequency of the number of reductions that are performed in an actual run of Algorithm 3. We can confirm that the formula finely explains the result of the experiment, and we conjecture that the derivation of (3) is sufficiently reasonable. We also conducted the above experiments with many different keys and confirmed the reasonability in (3).

The running time Z is characterized by Y_1 and Y_2. Let c_m and c_r be the numbers of clocks that are needed to perform a multiplication and a reduction, respectively. We have $Z = (3(t+w)-1)c_m + c_r Y_2$ because $Z = c_m Y_1 + c_r Y_2$ by Assumption 2 and $Y_1 = 3(t+w)-1$. If $Z = z$, then $Y_2 = (z - (3(t+w)-1)c_m)/c_r$, and therefore

$$P_{Z|T,W}(z|t,w) = P_{Y_2|T,W}\left(\frac{z-(3(t+w)-1)c_m}{c_r}|t,w\right).$$

Notice that $P_{Y_2|T,W}$ has been derived in (3), and $P_{Z|T,W}(z|t,w)$ can be numerically computable for given z, t and w. We also have $P_Z(z) = \sum_{(t,w)} P_{Z|T,W}(z|t,w) \cdot P_{T,W}(t,w)$, where $P_{T,W}(t,w)$ is given as (2). We can compute $H(Z)$ by using P_Z, $H(Z|T,W)$ by using $P_{Z|T,W}$ and $P_{T,W}$, and $I(K;Z) = I(T,W;Z) = H(Z) - H(Z|T,W)$. We remark that the values of $I(K;Z)$, $H(Z)$ and $H(Z|T,W)$ de-

pend on n/r because $P_{Y_{2,3}|T,W}$ and $P_{Y_{2,6}|T,W}$, and consequently $P_{Z|T,W}$ and $P_{Y_2|T,W}$, involve n/r in their expressions. Numerical computation shows that $I(K;Z)$ tend to decrease as n/r increases, even though the phenomenon is not justified mathematically because the expression is too complicated to analyze. If this phenomenon holds in general, then it suggests that using n that is slightly smaller than a power of 2, namely r, is advantageous in decreasing the information leakage of the decryption key.

With setting $0.80 < n/r < 0.81$, the result of numerical computation is shown in Table 1, where $c_m = (l/32)^2$ but $c_r = (l/32)$ since a reduction can be replaced by shift operation because r is usually a power of 2, and the reduction at line 6 of MR can be replaced subtraction operation because $0 \le m_2 < 2n$ [5].

5. CRT-ModBin Method

For the CRT-ModBin method (Algorithm 4), it is difficult to take the same approach as previous sections because $d_p = d \bmod (p-1)$ and $d_q = d \bmod (q-1)$ are used instead of d. Therefore, we take an approach that is different from previous sections at two points.

The first difference is the target of an attacker; we consider a scenario that the attacker tries to find d_p and d_q instead of d. Let $K = (K_p, K_q)$ where K_p and K_q are random variables of d_p and d_q. Similarly, we introduce random variables T_p, T_q, W_p and W_q for the actual key lengths and the Hamming weights of d_p and d_q. We assume for simplicity that T_p and T_q are independent, and so are W_p and W_q. The above assumption is conjectured to be reasonable because p and q are unknown prime numbers chosen independently and there is no mathematical relationship between d_p and d_q. The second difference we introduce is the ability of an attacker; we assume that an attacker is able to know the running time Z_p for the computation of $\mathrm{ModBin}(c_p, d_p, p)$ and the running time Z_q for the computation of $\mathrm{ModBin}(c_q, d_q, q)$. This assumption seems unrealistic at a glance, but not impossible if an attacker has some control over a target system, as considered in Flush+Reload attack [13]. In the Flush+Reload attack, an attacker is able to run a program in the same system as the target decryption program, and the attacker's program share the same cache memory as the target program. The attacker flushes out the cache memory, let the target program run, and reloads the memory line which might be accessed by the target program. If the reload operation takes a short time, then the memory line must be in the cache, which is the evidence that the target program has accessed the memory line. If the reload takes long time, then the target program did not access that memory line. In the case of CRT-ModBin method, a Flush+Reload attacker who watches an access to d_q is able to detect when the program starts the computation of $\mathrm{ModBin}(c_q, d_q, q)$, and thus separate Z to Z_p and Z_q.

Lemma 5.1: $I(K; Z_p, Z_q) = I(Z_p; T_p, W_p) + I(Z_q; T_q, W_q)$

Sketch of Proof: The random variable K is replaced by K_p and K_q, and then by T_p, T_q, W_p and W_q as in the previous sections. Use the fact that Z_p (resp. Z_q) depends only on T_p and W_p (resp. T_q and W_q). □

Notice that $I(Z_p; T_p, W_p)$ and $I(Z_q; T_q, W_q)$ are the mutual information of ModBin method, but the length of "keys" d_p and d_q is $l/2$ in the CRT-ModBin method. Consequently, $I(K; Z_p, Z_q)$ is twice as that of $l/2$-bit ModBin method (see Table 1).

We emphasize that the attacker model is different for CRT-ModBin method, and the mutual computation in the Table 1 cannot be compared naively. Just regard the mutual information in Table 1 for the CRT-ModBin method as an upperbound of the information leakage for the attacker who cannot separate the running time to Z_p and Z_q.

6. Conclusion

This study derived well-defined formulas that give approximations of the mutual information $I(K;Z)$ between the running time Z and the decryption key K that is concealed in RSA decryption algorithms. The mutual information $I(K;Z)$ is an important measure of the security of a program against passive timing attacks, but the computation of $I(K;Z)$ is quite difficult in general. Existing studies such as [7] and [5] focused an obvious upper-bound $H(Z)$ instead of $I(K;Z)$, but it can make fatal misleads. For example, one may consider that the ModBin method leaks more information than the Binary method because $H(Z)$ of the ModBin method is greater than $H(Z)$ of the Binary method (see Table 1, and pay attention that $H(Z) = I(K;Z)$ for the Binary method). This study revealed that such an implication is completely wrong; $I(K;Z)$ of the ModBin method is less than $I(K;Z)$ of the Binary method, and the ModBin method leaks less information than the Binary method. The study allows precise security comparison of different algorithms, and helps understanding the essential risk of passive timing attacks.

References

[1] D. Brumley and D. Boneh, "Remote timing attacks are practical," 12th Conference on USENIX Security Symposium, pp.1–13, 2003.
[2] J. Cichoń and Z. Gołębiewski, "On Bernoulli sums and Bernstein polynomials," 23rd International Meeting on Probabilistic, Combinatorial and Asymptotic Methods in the Analysis of Algorithms, pp.179–190, 2012.
[3] T.M. Cover and J.A. Thomas, Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing), pp.34–43, 2006.
[4] T. Hirata and Y. Kaji, "Information leakage through passive timing attacks on RSA decryption system," International Symposium on Information Theory and Its Applications, pp.392–396, 2020.
[5] Y. Kobayashi, "Information theoretical security evaluation of timing attack for RSA cryptosystem," Nara Institute of Science and Technology, Masters Thesis, 2015 (in Japanese).
[6] P.C. Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems," CRYPTO'96, pp.104–113, 1996.
[7] B. Köpf and M. Durmuth, "A provably secure and efficient countermeasure against timing attacks," Proc. 22nd IEEE Computer Security Foundations Symposium, pp.324–335, 2009.
[8] B. Köpf and G. Smith, "Vulnerability bounds and leakage resilience

of blinded cryptography under timing attacks," Proc. 23rd IEEE Computer Security Foundations Symposium, pp.44–56, 2010.
[9] N. Kunihiro and J. Honda, "RSA meets DPA: Recovering RSA secret keys from noisy analog data," Intl. Conf. on Cryptographic Hardware and Embedded Systems, pp.261–278, 2014.
[10] C. Paar and J. Pelzl, Understanding Cryptography, pp.180–181, Springer, 2010.
[11] W. Schindler, "A timing attack against RSA with the Chinese remainder theorem," Intl. Conf. on Cryptographic Hardware and Embedded Systems, pp.109–124, 2000.
[12] T. Terauchi and T. Antonopoulos, "A formal analysis of timing channel security via bucketing," Intl. Conf. on Principles of Security and Trust, pp.29–50, 2019.
[13] Y. Yarom and K. Falkner, "FLUSH+RELOAD: A high resolution, low noise, L3 cache side-channel attack," USENIX Security Symposium, pp.719–732, 2014.

Tomonori Hirata received the B.E.degree in information and computer sciences from Nagoya University in Aichi, in 2019, and Master's degree in Graduate School of Informatics, Nagoya University, in 2021.

Yuichi Kaji was born in Osaka, Japan, on December 23, 1968. He received the B.E., M.E., and Ph.D. degrees in information and computer sciences from Osaka University, Osaka, Japan, in 1991, 1992 and 1994, respectively. In 1994, he joined Nara Institute of Science and Technology, Nara, Japan. In 2003 and 2004, he visited the University of California Davis and the University of Hawaii at Manoa as a visiting researcher. He transferred to Nagoya University in 2016. His current research interests include the theory of error correcting codes, fundamental techniques for information security, and the theory of automata and rewriting systems. He is a member of IPSJ and IEEE.