Computer Security Symposium 2019
21 - 24 October 2019

Information leakage through passive timing attacks on RSA decryption system

Tomonori HIRATA[1,a)] Yuichi KAJI[1,b)]

Abstract: The running time of an RSA decryption program exposes certain information of the concealed decryption key due to their correlation. However, it is not easy to estimate the amount of information that leaks through the running time because practical decryption programs are complicated and difficult to analyze. This study focuses on two decryption algorithms, and derives well-defined formulas of the mutual information between the running time and the decryption key. The formula can be used to compare the actual amount of information that a passive timing attacker can learn, and contributes to the quantitative discussion of the possible risk of passive timing attacks. Based on the formulas, the amount of information leakage is computed numerically for those two algorithms with practical parameters.

Keywords: RSA, timing attack, Information Theory, quantitative information flow analysis

1. Introduction

1.1 Background of Study

This study investigates information theoretical aspects of *timing attacks* on basic algorithms for RSA decryption. Generally speaking, a computation is an abstract process that determines a specific output from a given input. To automate the computation, the abstract process is given a physical realization such as a computer program or an electronic circuit. Besides explicitly-defined input and output of the computation, these realizations usually emit various *side-information* such as electromagnetic wave, power consumption pattern and running time that are needed to perform the computation. In many cases, the side-information depends on the input, output, and the internal process of computation. This means that an attacker has the chance to learn something about the internal process of the computation by carefully observing the side-information. In a cryptographic computation, this brings some risk; an attacker can mount a *side-channel attack* in which he/she obtains the side-information and tries to determine secret information that is hidden in a realization of a cryptographic computation. When the attack focuses on the running time of the computation, the attempt is called a *timing attack*.

Timing attacks can be categorized to two classes; *active* timing attacks and *passive* timing attacks. Many of existing studies are categorized to the active attacks in which an attacker is able to feed the cryptosystem with inputs of his/her choice. It is known that active timing attacks are quite effective, and become practical threat to various cryptosystems[1], [4]. On the other hand, not so much efforts have been devoted to the passive timing attacks in which attackers cannot control the input to the cryptosystem, because passive attackers are weaker than active attackers and thus less interesting in the context of security study. This does not imply however that passive timing attacks are insignificant in practice. For example, a recent realization of RSA decryption often adopts the technique of *input blinding* as a countermeasure to active timing attacks. The input blinding is a preprocessing operation that converts a ciphertext c to $c' = r^e c \pmod{n}$ where r is a random value and (e, n) is an encryption key. The RSA decryption is then performed as $m' = (c')^d \pmod{n}$, and the final result is obtained by computing $r^{-1}m' \pmod{n}$ where r^{-1} is the inverse of r in the modulus of n. Notice that an attacker is not able to control the value of r, and this is exactly the scenario that is focused by passive timing attacks. Many people believe that the input blinding is an effective measure to active timing attacks, but that belief has not been verified by theoretical evidences. This study changes the situation. We derive formulas that express the amount of information that leaks through passive timing attacks, and show that a passive attacker learns very little information of the decryption key.

1.2 Framework of Discussion

Assume that there is an attacker who is able to access an RSA decoding program and to observe the number of clocks (running time) consumed by the program. The attacker knows the algorithm that is used in the program, but cannot access to the input and output of the program. The objective of the attacker is to guess the decryption key, which is regarded as a secret input or a secret constant that

[1] Graduate School of Informatics, Nagoya University
[a)] hirata@sqlab.jp
[b)] kaji@icts.nagoya-u.ac.jp

is concealed in the program, from the running time of the program. Even though we cannot predict what kind of computation the attacker may employ, the amount of information that the attacker can learn is upper-bounded by the mutual information between the running time and the decryption key.

Köpf introduced an information theoretical framework to discuss the above setting [5], [6]. Let K be a random variable of the secret decryption key, and Z be a random variable of the running time (the number of clocks) of the RSA decryption program. The amount of information of the decryption key that the attacker can obtain is upper-bounded by $I(K;Z)$, which is characterized as

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

where H denotes the (Shannon) entropy of a random variable and $H(\cdot|\cdot)$ denotes the conditional entropy. The problem here is that it is not easy to determine the conditional entropy $H(K|Z)$ or $H(Z|K)$ because of the complicated behavior of the decryption program. Köpf therefore ignored the conditional entropy, and considered to upper-bound: $I(K;Z) \leq H(Z)$ because the conditional entropy $H(Z|K)$ is nonnegative. In practice, it is likely that the attacker can make multiple observations of the running time. Köpf extended the above discussion to the case with multiple observations, and derived a tiny upper-bound of the mutual information between the decryption key and multiple observations of the running time. Kobayashi followed the discussion of Köpf by improving the bound formula of Köpf, and used the improved formula to compare possible risks of some different decryption algorithms[3]. However, the study of Kobayashi also follows the convention of Köpf where the conditional entropy $H(Z|K)$ is ignored for simplicity. The approach misses an important aspects of timing attacks, because the attack is based on the fact that the uncertainty of the key is reduced by the information of running time.

1.3 Contribution of This Study

The contribution of this study is the derivation of well-defined formulas of $H(Z)$ and $H(Z|K)$ for basic algorithms of the RSA decryption. We focus on a simple *Binary method* that implements a modulo exponentiation of large numbers by iterative calculations of rather small numbers, and a variance of the Binary method where calculations are made in the *Montgomery forms* (unfortunately, a practical decoding algorithm that utilizes the *Chinese-Remainder Theorem* is not discussed in this study because of the complication of the program). In these algorithms, the number of multiplications and the number of reductions (a reduction is the computation to determine a remainder in a modulo computation) give significant contribution to the running time of the program. We derive formulas that characterize the relation among the running time of the program, actual key length, and the Hamming weight of the decryption key, and use the formulas to derive $H(Z)$ and $H(Z|K)$. This enables the estimation of $I(K;Z)$, which works as the upper-bound of the amount of information that a passive attacker can learn from the timing information, and thus helps understanding how serious passive timing attacks are for these decryption algorithms.

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:     m = m^2 mod n
4:     if d_i = 1 then
5:         m = mc mod n
6:     end if
7: end for
8: return m
```

2. Preliminary

2.1 RSA Decryption Algorithm

The decoding of the RSA is realized by a computation

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

where n is a product of two primes, d is a decryption key, c is a ciphertext that is encoded as an integer with $0 \leq c \leq n-1$, and m is the result of the decoding. For the sake of security, n is taken as a very large integer with several thousand bits, and a naive computation of (1) requires large computational complexity. To avoid this issue, acceleration algorithms are investigated. Among such algorithms, this study focuses on *Binary method* and *ModBin method* which are introduced in this section. We do not discuss the decoding algorithm that makes use of the Chinese Reminder Theorem in this study, though it is widely used in practical implementations.

In the following discussion, we write l for the bit-length of the modulus n, and $d_{l-1} \cdots d_0$ for the l-bit binary representation of the decryption key d, that is, $d_i \in \{0,1\}$ for $0 \leq i \leq l-1$, and

$$d = \sum_{i=0}^{l-1} d_i 2^i.$$

If $d_{l-1} = \cdots = d_t = 0$ and $d_{t-1} = 1$, then t is called as the *actual key length*.

In the Binary method, which is also known as the Square-and-Multiply method, we consider a binary representation of a decryption key and make iterative computation according to the binary bits of the key[7]. The algorithm is described as the pseudo-code as in Algorithm 1.

The ModBin method is an improvement of the Binary method where computation is made through Montgomery operations. In Montgomery operations, we need to select in advance a constant r that is greater than n and relatively prime to n. An arbitrary value can be taken as r as far as it satisfies the conditions, and we usually select r as a power of two, because divisions and reductions with respect to r can be implemented by simple shift operations if r is a power of

Algorithm 2 Montgomery reduction MR(x)

Input: x
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** $m_2 \leq n$ **then**
4: return m_2
5: **else**
6: return $m_2 \bmod n$
7: **end if**

Algorithm 3 ModBin method

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

two. Let n' be $nn' \equiv -1 \pmod r$. Algorithm 2 reduces an integer to its Montgomery form and vice versa[5], and commonly called a *Montgomery reduction*. Using Montgomery reduction, ModBin algorithm can be described as in Algorithm 3, where $r_2 = r^2 \pmod n$ [5].

2.2 Notations and Assumptions

We use the following random variables throughout the paper.

- K : Decryption key (the integer d in (1)).
- T : The actual key length of the decryption key.
- W : The Hamming weight of the decryption key.
- Y_1 : The number of multiplications that are performed in a decoding calculation.
- Y_2 : The number of reductions that are performed in a decoding calculation. We define that a reduction is performed in a modulo computation $a \bmod b$ only if $a \geq b$ (if $a < b$, then no reduction is performed).
- Z : The total number of clocks consumed for the multiplications and reductions that are performed in a decoding calculation. For simplicity, we may call Z as a *running time* instead of the number of clocks.

A constant l is used to denote the bit length of the modulus n of the decryption key.

In practice, it is difficult for an attacker to learn some information of the decryption key from public information. In this study, we formalize this condition as the following Assumption 1 .

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

From this assumption, the probabilities $P_T(t)$ and $P_W(w)$ are determined as

$$P_T(t) = \frac{1}{2^{l-t+1}},$$
$$P_W(w) = \binom{l}{w} \cdot \left(\frac{1}{2}\right)^w \cdot \left(\frac{1}{2}\right)^{l-w} = \frac{1}{2^l} \cdot \binom{l}{w}.$$

It is also understood that

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

As for the running time of the algorithms, we assume the following.

Assumption 2 The variance of the running time of the algorithm is brought only by the number of multiplications and reductions that are performed in the algorithm.

We note that this assumption does not hold in a strict sense because iterations and conditional branches in the above mentioned algorithms involve some additional operations. However, the number of clocks for these additional operations are much smaller than those for multiplications and reductions, and thus we ignore the effect of these rather minor operations for simplicity of the discussion.

Assumption 2 implies that the running time Z depends only on the number of multiplications Y_1 and the number of reductions Y_2. On the other hand, from the pseudo-programs of the Binary method and ModBin method, it is understood that the number of multiplications Y_1 and the number of reductions Y_2 are affected only by the actual key length T and the Hamming weight W of the decryption key K. Consequently, the relation among random variables are modeled by a Markov chain

$$K \Rightarrow (T, W) \Rightarrow (Y_1, Y_2) \Rightarrow Z.$$

The data processing lemma[2] that is widely known in Information Theory guarantees that

$$I(K;Z) \leq \min\{I(K;T,W), I(T,W;Z)\},$$

but a stronger result can be obtained in the above scenario.

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

Proof: Note that the actual key length T and the Hamming weight W are uniquely determined from the decryption key K. Therefore, there is a function $t(k)$ and $w(k)$ that denote the actual key length and the Hamming weight of a key k, respectively, and $P_{Z|K}(z|k) = P_{Z|T,W}(z|t(k), w(k))$ holds for arbitrary values of z and k. With this relation and basic transformations of equations, we can show that $H(Z|K) = H(Z|T,W)$, from which

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

3. Binary method

3.1 Probability distributions

Let assume that $T = t$ and $W = w$ in the Binary method (Algorithm 1). The multiplication in the third line is executed $(t-1)$ times, and the multiplication in the fifth line is executed $(w-1)$ times. Therefore, the number of multiplications in the decoding calculation is exactly $y_1 = (t-1)+(w-1)$.

The modulo computations in the third line and the fifth line invoke reductions only if $m^2 \geq n$ and $mc \geq n$, respectively. However, it is difficult to spot the value of m in each occasion because m is updated iteratively. To get around this issue, we put the following assumption with which modulo computations are regarded as Bernoulli trials.

Assumption 3 At the third line and the fifth line of the Binary method, 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 modulo computations that are performed at the third and the fifth lines of the program, respectively. At the third line of the program, a reduction is not needed if $m^2 < n$ which holds with probability $\alpha = \sqrt{n}/n = 1/\sqrt{n}$ under Assumption 3. The third line is visted $(t-1)$ times during the computation, and each visit invokes a reduction with probability $1-\alpha$. Therefore, $Y_{2,1}$ obeys a binomial distribution

$$P_{Y_{2,1}|T,W}(y|t,w) = \binom{t-1}{y} \alpha^{t-1-y}(1-\alpha)^y$$

for $0 \leq y \leq t-1$. The discussion for $Y_{2,2}$ is little bit complicated. 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;

$$P_{Y_{2,2}|T,W}(y|t,w) = \frac{1}{n}\binom{w-1}{y} 1^{w-1-y}0^y + \frac{1}{n}\sum_{c=1}^{n-1}\binom{w-1}{y}\left(\frac{1}{c}\right)^{w-1-y}\left(1-\frac{1}{c}\right)^y.$$

Lemma 3.1 If n is large, then

$$P_{Y_{2,1}|T,W}(y|t,w) \approx \begin{cases} 1 & \text{if } y = t-1, \\ 0 & \text{otherwise,} \end{cases}$$

$$P_{Y_{2,2}|T,W}(y|t,w) \approx \begin{cases} 1 & \text{if } y = w-1, \\ 0 & \text{otherwise.} \end{cases}$$

See Appendix A.1 for the proof of Lemma 3.1.

For practical n with several thousand bits, we can safely replace $\approx$ in Lemma 3.1 with equalities, and regard $P_{Y_{2,1}|T,W}$ and $P_{Y_{2,2}|T,W}$ as single-point distributions that are 1 for $y = t-1$ and $y = w-1$, respectively. With this replacement, we have

$$P_{Y_2}(y_2) = \begin{cases} 1 & \text{if } y_2 = (t-1)+(w-1), \\ 0 & \text{otherwise} \end{cases}$$

because the third and the fifth lines are visited $(t-1)$ and $(w-1)$ times, respectively.

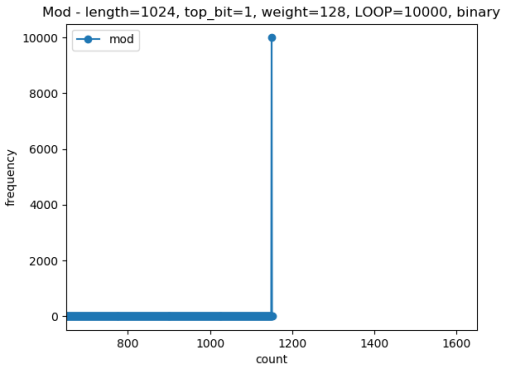

Fig. 1 The number of reductions in Binary method

To confirm if the above replacement is appropriate or not, we conducted an experiment and evaluated the relative frequency of the number of reductions by running the Binary method for randomly generated 10,000 ciphertexts. Fig. 1 shows the result of the experiment for $l = 1024$ and a decryption key that is randomly selected to satisfy $t = 1024$ and $w = 128$. The horizontal axis represents the number of reductions, and the vertical axis represents the frequency. From Fig. 1, we can see that the number of reductions is 1150 for all 10,000 ciphertexts. Note that 1150 is exactly equal to $(t-1)+(w-1)$ with $t = 1024$ and $w = 128$ that are used in this experiment.

3.2 Derivation of $I(K;Z)$ (Binary method)

We follow the same convention as Köpf and assume that a multiplication and a reduction over l-bit integers need $(l/32)^2$ clocks each(this should be $(l/64)^2$ if we consider modern 64-bit processors, but the choice is not essential and we follow the convention in literature). With this convention, we need

$$\begin{aligned} z &= (y_1+y_2)\left(\frac{l}{32}\right)^2 \\ &= 2((t-1)+(w-1))\cdot\left(\frac{l}{32}\right)^2 \end{aligned} \tag{2}$$

clocks in total to perform all multiplications and all reductions in the Binary method. Notice that $H(Z|T,W) = 0$ because z is uniquely determined once t and w are specified. Therefore,

$$I(T,W;Z) = H(Z) - H(Z|T,W) = H(Z). \tag{3}$$

Remind that $I(K;Z) = I(T,W;Z)$ by Lemma 2.1, and therefore $I(K;Z) = H(Z)$.

To compute $H(Z)$, we rewrite (2) as $w+t = f(z)$ with

$$f(z) = \frac{z}{2}\cdot\left(\frac{32}{l}\right)^2 + 2.$$

Tab. 1 Numerical calculations for binary method

l, the bit length of n(bits)	$I(K;Z)$ (bits)
128	4.6311
256	5.0907
512	5.5693
1024	6.0583
2048	6.5527

Using this function f, $P_Z(z)$ is calculated as

$$\begin{aligned} P_Z(z) &= P_{W+T}(f(z)) \\ &= \sum_{t=f(z)/2}^{l} P_T(t) \cdot P_{W|T}(f(z)-t|t) \\ &= \frac{1}{2^l} \sum_{t=f(z)/2}^{l} \binom{t-1}{f(z)-t-1}. \end{aligned}$$

Because of $w \leq t$, $w+t=f(z)$, and $t \leq l$, the range of t is $f(z)/2 \leq t \leq l$. The entropy $H(Z)$ is computed by using this $P_Z(z)$.

We note that $H(Z)$ was an upper-bound of $I(K;Z)$ in the discussion of Kopf and Kobayashi. In this study, we showed that that upper-bound is tight and $H(Z)$ coincides with $I(K;Z)$. It is also noted that the distribution of Z was not discussed so far in existing studies, but we could write it down as a well-defined formula in the maximum key length l. The formula enables arithmetic computation of $H(Z)$ instead of time-consuming and costly experiments.

Tab. 1 shows the values of $I(K;Z)$, the amount of information that leaks through the running-time of Binary method, for several choices of l (the bit length of n). We can see that the running-time leaks small number of bits of the decryption key, and a passive timing attacker learns very little about the decryption key.

4. ModBin method

4.1 Probability distributions

First, we determine the number of multiplications that are preformed in a decoding calculation of the ModBin algorithm(Algorithm 3). Notice that multiplications are explicitly performed at lines 1, 4, 6 of the ModBin algorithm. Also notice that two multiplications are always performed (at lines 1 and 2) in each call of the MR algorithm (Algorithm 2, Montgomery reduction). Therefore, lines 1, 4 and 6 of the ModBin algorithm induce three multiplications each, and the line 9 induces two multiplications. The line 4 is executed $t-1$ times and the line 6 is executed $w-1$ times. Consequently, the number of multiplications that are preformed in the decoding calculation is derived as follows.

$$\begin{aligned} y_1 &= 3 + 3(t-1) + 3(w-1) + 2 \\ &= 3(t+w) - 1 \end{aligned} \tag{4}$$

Next, we focus on the number of reductions that are preformed in the decoding calculation. To simplify the discussion, we ignore two reductions in the first and the ninth lines of ModBin method 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, the number of reductions will be characterized by Bernoulli trials.

Assumption 4 At the fourth and the sixth line of the ModBin algorithm, m' distributes uniformly in $\{0,\ldots,n-1\}$.

In the main loop of the ModBin algorithm, MR($m'm'$) is called from the fourth line and MR($m'c'$) is called from the sixth line. 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, we separate the random variable Y_2 as $Y_2 = Y_{2,1}+Y_{2,2}+\cdots+Y_{2,6}$, where $Y_{2,1}$ and $Y_{2,2}$ are random variables of the number of reductions that are performed by modulo computations ($m'm' \bmod r$) and (($\cdot$)$n' \bmod r$) at the first line of MR($m'm'$), respectively, $Y_{2,3}$ denotes the number of reductions that are performed by ($m_2 \bmod n$) at the sixth line of MR($m'm'$), and $Y_{2,4}, Y_{2,5}$ and $Y_{2,6}$ denote corresponding numbers in MR($m'c'$). Similar to the dicussion for the Binary method, it is understood that $Y_{2,1}$, $Y_{2,2}$ and $Y_{2,3}$ obey binomial distributions

$$\begin{aligned} P_{Y_{2,1}|T,W}(y|t,w) &= \binom{t-1}{y} \alpha_{m'm'}^{t-1-y}(1-\alpha_{m'm'})^y, \\ P_{Y_{2,2}|T,W}(y|t,w) &= \binom{t-1}{y} \beta_{m'm'}^{t-1-y}(1-\beta_{m'm'})^y, \\ P_{Y_{2,3}|T,W}(y|t,w) &= \binom{t-1}{y} (1-\gamma_{m'm'})^{t-1-y}\gamma_{m'm'}^y, \end{aligned}$$

where $\alpha_{m'm'}$, $\beta_{m'm'}$ and $\gamma_{m'm'}$ are probabilities of $m'm' < r$, $sn' < r$ and $m_2 > n$, respectively, with $s = m'm' \bmod r$.

Lemma 4.1 We have $\alpha_{m'm'} = \sqrt{r}/n$, $\beta_{m'm'} = 1/n'$ and $\gamma_{m'm'} = n/(3r)$.

Proof: The inequalities $m'm' < r$ and $t'n' < r$ hold if $m' < \sqrt{r}$ and $t < r/n'$, respectively, and thus $\alpha_{m'm'} = \sqrt{r}/n$ and $\beta_{m'm'} = (r/n')/r = 1/n'$. As for $\gamma_{m'm'}$, note that $m_2 \leq n$ if and only if $(m')^2 + m_1 n \leq nr$ in MR($m'm'$), which is written $m_1 \leq r - (m')^2/n$. The expected value of $(m')^2$ is derived as

$$\frac{1}{n} \sum_{1 \leq m' \leq n} (m')^2 \approx \frac{n^2}{3},$$

and the expected value of $r-(m')^2/n$ equals to $r-n/3$. The probability of $m_1 \leq r-(m')^2/n$ is $(r-n/3)/r = 1-n/(3r)$, and $\gamma_{m'm'} = n/(3r)$. □

With this lemma, the probability distribution $P_{Y_{2,3}|T,W}$ is written as

$$P_{Y_{2,3}|T,W}(y|t,w) = \binom{t-1}{y} \left(1-\frac{n}{3r}\right)^{t-1-y} \left(\frac{n}{3r}\right)^y.$$

Lemma 4.2 If n is large, then $P_{Y_{2,1}|T,W}(t-1|t,w) \approx 1$ and $P_{Y_{2,2}|T,W}(t-1|t,w) \approx 1$, that is, the realized values of $Y_{2,1}$ and $Y_{2,2}$ are almost always $t-1$.

Proof: If n is large, then $\alpha_{m'm'}$ and $\beta_{m'm'}$ converge to 0 and the lemma holds due to the property of binomial distributiuons. □

In the discussion for $Y_{2,4}$, we need to consider a superposition of binomial distributions each corresponds to a possible value of c'. Remark that $m'c' < r$ holds with probability 1 if $c' = 0$ and $r/(c'n)$ if $c' > 0$, and we have

$$P_{Y_{2,4}|T,W}(y|t,w) = \frac{1}{n}\binom{w-1}{y}1^{w-1-y}0^{y} + \frac{1}{n}\sum_{c=1}^{n-1}\binom{w-1}{y}\left(\frac{r}{c'n}\right)^{w-1-y}\left(1-\frac{r}{c'n}\right)^{y}.$$

Fortunately, $P_{Y_{2,5}|T,W}$ is a simple binomial distribution because sn' is irrelevant with c':

$$P_{Y_{2,5}|T,W}(y|t,w) = \binom{w-1}{y}\beta_{m'c'}^{w-1-y}(1-\beta_{m'c'})^{y}$$

where $\beta_{m'c'} = \beta_{m'm'} = 1/n'$. The discussion for $Y_{2,6}$ is the most complicated. The inequality $m_2 \le n$ in MR$(m'c')$ is transformed as $m_1 \le r - m'c'/n$, which holds with probability $1 - m'c'/(nr)$. The expected value of m' is $n/2$, and the expected value of this probability is $1 - c'/(2r)$. We need to consider a superposition of binomial distributions which are characterized by the probabilities $1-c'/(2r)$ with $c' = 0, \ldots, n-1$:

$$P_{Y_{2,6}|T,W}(y|t,w) = \frac{1}{n}\sum_{c'=0}^{n-1}\binom{w-1}{y}\left(1-\frac{c'}{2r}\right)^{w-1-y}\left(\frac{c'}{2r}\right)^{y}.$$

It is difficult to compute $P_{Y_{2,6}|T,W}(y_6|t,w)$ in a straightforward way because n is too large, and we approximate $P_{Y_{2,6}|T,W}(y_6|t,w)$ using the classification quadrature method.

Lemma 4.3

$$P_{Y_{2,6}|T,W}(y_6|t,w) \approx \frac{2r}{n}\binom{w-1}{y}\int_0^{(n-1)/(2r)}(1-t)^{w-1-y}t^{y}dt$$

See Appendix A.2 for the proof of Lemma 4.3.

To proceed the computation further, we define

$$I(r,s) = \int_0^{\alpha}(1-t)^{s}t^{r}dt$$

where α is a constant and s, r are variables. Using partial integration, $I(r,s)$ is expanded as follows.

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

From the above result, we can compute $I(r,s)$ with only s recursion. With these investigations, we can calculate $P_{Y_{2,6}|T,W}(y_6|t,w)$ with manageable complexity.

To verify the approximation, we conducted an experiment and compared the result of the experiment with the above derived distribution $P_{Y_{2,6}|T,W}$. Fig. 2 shows the theoretical and experimental distributions of $P_{Y_{2,6}|T,W}$. The bars in the graph show the results of experiment, and the curve shows the formula that we have derived. We can confirm that our

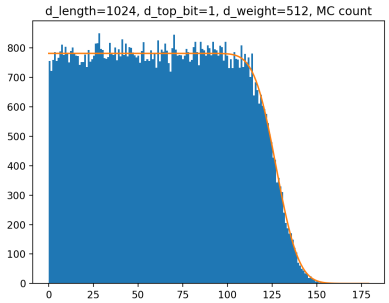

Fig. 2 Comparing $P_{Y_{2,6}|T,W}(y_6|t,w)$ and experiment value in ModBin method

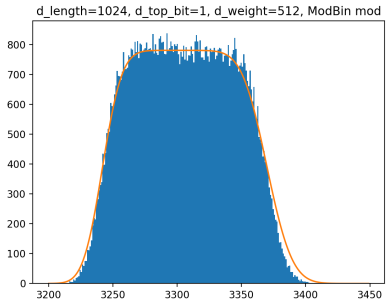

Fig. 3 Comparing $P_{Y_2|T,W}(y|t,w)$ and experiment value in ModBin method

derivations well explain the results of the experiment, and assumptions that are posed in the discussion are reasonable and acceptable.

Lemma 4.4 If n is large, then $P_{Y_{2,4}|T,W}(w-1|t,w) \approx 1$ and $P_{Y_{2,5}|T,W}(w-1|t,w) \approx 1$, that is, the realized values of $Y_{2,4}$ and $Y_{2,5}$ are almost always $w-1$.

Proof: The proof for $P_{Y_{2,4}|T,W}$ is obtained in a similar manner to the discussion for the Binary method case, and $P_{Y_{2,5}|T,W}(w-1|t,w) = 1$ is obtained because $\beta_{m'c'} = 1/n'$ converges to 0. □

In general, $P_{Y_2|T,W}$ is defined as

$$P_{Y_2|T,W}(y|t,w) = \sum_{y_1+\cdots+y_6=y} P_{Y_{2,1}|T,W}(y_1|t,w)\cdots P_{Y_{2,6}|T,W}(y_6|t,w).$$

If n is large, then we can replace the approximations in Lemmas 4.2 and 4.4 as equalities. In this case the product of the above equation goes to zero unless $y_1 = y_2 = t-1$ and $y_4 = y_5 = w-1$. Therefore,

$$P_{Y_2|T,W}(y|t,w) = \sum_{y_3+y_6=y-2(t-1)-2(w-1)} P_{Y_{2,3}|T,W}(y_3|t,w)P_{Y_{2,6}|T,W}(y_6|t,w). \quad (5)$$

Different from the case of Binary method, Y_2 varies and has multiple possible values even if the values of T and W are fixed.

Fig. 3 shows the theoretical and experimental distributions of $P_{Y_2|T,W}$. The bars in the graph show the results of experiment, and the curve shows the formula that we have

derived. Like $P_{Y_{2,6}|T,W}(y_6|t,w)$, we can confirm that our derivations well explain the results of the experiment.

4.2 Derivation of $I(K;Z)$ (ModBin algorithm)

We follow the same convention as Köpf and assume that a multiplication over l-bit integers needs $(l/32)^2$ clocks and a reduction over l-bit integers needs $l/32$ clocks because a reduction is implemented by shift operations by taking r as a power of two. With this convention, we need

$$\begin{aligned} z &= y_1 \cdot \left(\frac{l}{32}\right)^2 + y_2 \cdot \left(\frac{l}{32}\right) \\ &= (3\,(t+w)-1) \cdot \left(\frac{l}{32}\right)^2 + y_2 \cdot \left(\frac{l}{32}\right) \qquad (6) \end{aligned}$$

clocks in total to perform all multiplications and all reductions in the ModBin method algorithm, where y_2 is stochastically determined from t and w according to (5). Because of the variation of Y_2, the value of Z is not determined uniquely from T and W. This implies that $H(Z|T,W) > 0$, and we need both values of $H(Z)$ and $H(Z|T,W)$ to compute $I(T,W;Z) = H(Z) - H(Z|T,W)$. For the computation of $H(Z)$, we need to derive $P_Z(z)$ which is written as

$$P_Z(z) = \sum_{(y_1,y_2)\in S(z)} P_{Y_1,Y_2}(y_1,y_2)$$

where $S(z) = \{(y_1,y_2) \mid (l/32)^2 y_1 + (l/32) y_2 = z\}$ gives the combinations of Y_1 and Y_2 values that make $Z = z$. The joint probability $P_{Y_1,Y_2}(y_1,y_2)$ is extended as

$$\begin{aligned} P_{Y_1,Y_2}(y_1,y_2) &= \sum_{w+t=g(y_1)} P_{Y_2,T,W}(y_2,t,w) \\ &= \sum_{w+t=g(y_1)} P_{Y_2|T,W}(y_2|t,w) \cdot P_{T,W}(t,w). \end{aligned}$$

To discuss the conditional probability $P_{Z|T,W}(z|t,w)$, remind the equation (6) which can be written as

$$y_2 = \frac{32z}{l} - (3(t+w)-1)\left(\frac{l}{32}\right) = h(z,t,w).$$

Notice that $P_{Z|T,W}(z|t,w) = P_{Y_2|T,W}(h(z,t,w)|t,w)$, and that $P_{Y_2|T,W}$ has been already derived in (5). Consequently

$$\begin{aligned} &H(Z|T,W) \\ &= \sum_{t,w} P_{T,W}(t,w) \sum_z -P_{Z|T,W}(z|t,w) \log P_{Z|T,W}(z|t,w) \\ &= \sum_{t,w} P_{T,W}(t,w) \sum_z -P_{Y_2|T,W}(h(z,t,w)|t,w) \\ &\qquad \cdot \log P_{Y_2|T,W}(h(z,t,w)|t,w). \end{aligned}$$

Use $H(Z)$ and $H(Z|T,W)$ and we can estimate $I(K;Z)$ numerically, with the effect of $H(Z|T,W)$ taken into consideration. Tab. 2 shows the values of $I(K;Z)$, the amount of information that leaks through the running-time of ModBin method. Unfortunately, the computation for $l = 2048$ is not yet completed at the time of submission of this paper. We can see that the running-time leaks small number of bits of the decryption key, and a passive timing attacker learns very little about the decryption key.

Tab. 2 Numerical calculations for ModBin method

l, the bit length of n(bits)	$I(K;Z)$ (bits)
128	3.4616
256	3.9979
512	4.7949
1024	5.3642
2048	-

We can see that the amount of leaked information in the ModBin method is smaller than that in the Binary method. This is because the running time varies even if the parameters T and W are determined.

5. Conclusion

It has been difficult to evaluate the risk of passive timing attacks because we did not know the amount of secret information that is leaked through the running time of a program. 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 basic algorithms for RSA decryption.

For the Binary method, we showed that the mutual information $I(K;Z)$ equals to the entropy $H(Z)$ of the running time of the program, which essentially improves the upper-bound discussion of Köpf[5] and Kobayashi[3]. This study also derived a formula of the entropy $H(Z)$. Since $H(Z)$ equals to $I(K;Z)$, the formula gives the upper-bound of the secret information that a passive timing attacker is able to learn from the running time. For the ModBin method, the conditional entropy $H(Z|T,W)$ gives an essential contribution to $I(K;Z)$. We have derived formulas of $H(Z)$ and $H(Z|T,W)$ in l, and $I(K;Z)$ is now determined by an arithmetic calculation. The results contribute to the quantitative understanding of the risk of passive timing attacks for the decryption algorithms.

The authors however consider that there is a problem that is still unsolved. This study did not discuss about the most practical RSA decryption method that makes use of the Chinese Remainder Theorem (CRT method). In the CRT method, the decryption computation is highly related to the factors of the modulus n rather than d. With some modification of the framework of the discussion, we might be able to clarify the relation between the running time of the CRT method and factors of the modulus n.

References

[1] D. Brumley and D. Boneh, Remote timing attacks are practical, 12th Conference on USENIX Security Symposium, 2003.

[2] T. M. Cover and J. A. Thomas, Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing), pp. 34-43, 2006.

[3] Y. Kobayashi, Information Theoretical Security Evaluation of Timing Attack for RSA Cryptosystem, Nara Institute of Science and Technology, Masters Thesis, 2015. in Japanese

[4] P.C. Kocher, Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems, CRYPTO '96, pp. 104-113, 1996.

[5] B. Köpf and M. Durmuth, A provably secure and efficient countermeasure against timing attacks, Proc. of the 22nd IEEE Computer Security Foundations Symposium, New York, NY, pp.324–335, July 2009.

[6] B. Köpf and G. Smith, Vulnerability bounds and leakage resilience of blinded cryptography under timing attacks, Proc. of the 23rd IEEE Computer Security Foundations Symposium, Edinburgh, UK, pp44–56, July 2010.
[7] C. Paar and J. Pelzl, Understanding Cryptography, Springer, pp.180–181, 2010.

Appendix

A.1 Proof of Lemma 3.1

Proof: Remind that $P_{Y_{2,1}|T,W}$ is a binomial distribution that is defined for a constant probability $\alpha = 1/\sqrt{n}$. If n is large, then $\alpha \to 0$ which makes $P_{Y_{2,1}|T,W}(t-1|t,w) = 1$ and $P_{Y_{2,1}|T,W}(y|t,w) = 0$ for $y \neq t-1$. Next consider $P_{Y_{2,2}|T,W}(w-1|t,w)$ which is

$$\begin{aligned}
&\frac{1}{n}\sum_{c=1}^{n-1}\left(1-\frac{1}{c}\right)^{w-1} \\
&\approx \frac{1}{n}\int_1^{n-1}\left(1-\frac{1}{c}\right)^{w-1} dc \\
&= \frac{1}{n}\int_1^{n-1}\sum_{i=0}^{w-1}\binom{w-1}{i}\left(-\frac{1}{c}\right)^{w-1-i} dc \\
&= \frac{1}{n}\sum_{i=0}^{w-1}\left(\binom{w-1}{i}\int_1^{n-1}\left(-\frac{1}{c}\right)^{w-1-i} dc\right) \\
&= \frac{1}{n}\sum_{i=0}^{w-2}\left(\binom{w-1}{i}\left[-\frac{1}{i+2-w}\left(-\frac{1}{c}\right)^{w-2-i}\right]_1^{n-1}\right) \\
&\quad + \frac{1}{n}\binom{w-1}{w-1}\left[-\left(-\frac{1}{c}\right)^{-1}\right]_1^{n-1}
\end{aligned}$$

If n is large, then the first term (the summation) in the above formula diminish to zero because $1/n \to 0$. The second term remains undiminished, and we can write

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

A.2 Proof of Lemma 4.3

$$\begin{aligned}
&P_{Y_{2,6}|T,W}(y_6|t,w) \\
&= \frac{1}{n}\sum_{c'=0}^{n-1}\binom{w-1}{y}\left(1-\frac{c'}{2r}\right)^{w-1-y}\left(\frac{c'}{2r}\right)^{y} \\
&\approx \frac{1}{n}\int_0^{n-1}\binom{w-1}{y}\left(1-\frac{x}{2r}\right)^{w-1-y}\left(\frac{x}{2r}\right)^{y} dx \\
&\approx \frac{2r}{n}\binom{w-1}{y}\int_0^{(n-1)/(2r)}(1-t)^{w-1-y}\,t^{y} dt
\end{aligned}$$