Scientific Research Publishing

Journal of Computer and Communications, 2019, 7, 169-178
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Correlated Extra Reductions Defeat Fixed Window Exponentiation

Xiaohan Meng

College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
Email: xiaohanmeng@nuaa.edu.com

How to cite this paper: Meng, X.H. (2019) Correlated Extra Reductions Defeat Fixed Window Exponentiation. *Journal of Computer and Communications*, **7**, 169-178.
https://doi.org/10.4236/jcc.2019.710016

Received: October 12, 2019
Accepted: October 23, 2019
Published: October 30, 2019

Abstract

The security of modular power algorithm is a very important research topic, which is the core operation of public key cryptography algorithm. Since the first timing attack was public in 1996, the attacker can exploit time differences between specific events to recover a secret key. In 2016, Dugardin took advantage of extra reductions to attack a regular exponentiation algorithm, which did not entirely adapt the fixed window method with Montgomery's algorithm. The central thesis of this paper is that there exists a positive correlation between extra reductions of pre-computation and post-computation when the calculation has the same multiplier factor. In this article, basing on this dependency we present an attack method, and confirm the feasibility and effectiveness of it by conducting simulation experiments. Experimental results verify that the method can effectively attack modular power algorithm.

Keywords

Side-Channel Attack, Montgomery Modular Multiplication, Extra Reduction Leakage

1. Introduction

Side-channel attack is an increasingly important area in applied cryptography. In a side-channel attack, the attacker is able to detect relevant physical information in the computation of a cryptographic algorithm and thus to get knowledge of the secret key. Meanwhile, due to the limitation of the computing resources and processing power, the master key of the cryptographic algorithm is mostly cut into several sub-key blocks and participates in operations in a certain order. The attacker can recover the sub-key block value by using subtle correlation, after obtaining enough sub-key block values and combing with the algorithm design, and the master key value can be restored. This behavior makes it possible to re-

cover the long key within a limited complexity. Hence, there is an urgent need to notice the safety problems caused by the side-channel attack. text styles are provided.

RSA algorithms [1] are commonly used in a wide range of public key cryptosystem in the embedded world from the smartcard technology to the IoT service. From a mathematical viewpoint, the security of the RSA algorithm relies on the difficulty in factoring large integer, but the practical implementation of the algorithm is not absolutely safe. The modular power algorithm is the core operation of the widely used public key cryptographic algorithm, and RSA is no exception. Or, more specifically, an RSA decryption or encryption computation is based on modular exponentiation consisting of many squares and multiplications. With the introduction of side-channel attack, many of the previously proposed module modular power algorithms may have potential security problems, and there is still much room for research on evaluating their security issues.

Since Kocher first implemented the timing attack of the RSA algorithm [1], various new attack techniques have been presented by the researchers for different cryptographic algorithms, like simple power attack [2], differential power attack [3], electromagnetic attack [4], correlation power attack [5] and so on. At the same time, there are a lot of studies of recovering the secret key with less time. But Kocher's timing attack will not work if the algorithm [6] one or more invalid modular multiplication operation to make the time computation constant in the Montgomery implementation. In the literature on [7], a new timing attack is introduced on the RSA with Chinese Remainder Theorem. The papers [8] [9] [10] performed timing attacks on RSA implementations in OpenSSL or mbedTLS, not only RSA with Chinese Remainder Theorem, but also extend to exponentiation algorithm, and optimized exponentiation algorithm. Surveys such as that conducted by Schindler [11] have showed that exclusive exponent blinding (without additional countermeasures) does not always prevent timing attacks on RSA.

Recently in CHES 2016, Dugardin *et al.* [12] pointed out binary exponentiation algorithms is vulnerable to side-channel attack even with message blinding and regular exponentiation. They presented a new dependency based on extra reductions in a sequence of multiplies and squares, which is a negative correlation between the extra reduction of two consecutive calculations. They also explained it from a mathematical viewpoint and exploit this correlation to successfully attack the RSA with regular exponentiation method in a real environment.

For that, is there any other correlations of extra reductions exist in the implementation of the modular power algorithm to be utilized? In order to improve efficiency, there is a lot of improved modular power algorithm, which mainly focuses on classical exponentiation. The research to date has tended to focus on a regular exponentiation algorithm rather than the fixed window exponentiation. The aim of this essay is to explore the relationship between pre-computation and post-computation in the fixed window exponentiation.

In this paper, we propose a strong positive correlation between the extra reductions during the Montgomery Modular Multiplication of pre-computation and post-computation.

This new dependency can be used to recover the secret key because the iteration in the post-computation could share common operand with pre-computation. In addition, we show it by conducting simulation experiments.

This paper demonstrates that our attack can successfully attack modular power algorithm based on fixed window exponentiation. Our attack does not require explicit knowledge of the message, neither does require cryptographic parameters. This work will generate fresh insight into the security of modular power algorithm.

The rest of paper is organized as follows. In Section 2, we show that some correlation between extra-reductions of pre-computation and post-computation, and explain our attack in detail. In Section 3, experiment and experimental results are presented. In Section 4, we conclude our paper.

2. Our Attack

This section points out that there exists a new correlation in the fixed window exponentiation algorithm and how to apply it using our attack.

2.1. Vulnerability of the Fixed Window Exponentiation Algorithm

Each modular multiplication operation has two operands. It is clear that two operations would be absolutely independent when they do not share one operand. Instead, if two operations have common operands there is a correlation between them. More precisely, when the extra reduction appear in both two modular multiplication operations, there exists strong positive correlation. It is because of when the operand is sufficiently large, both operations are likely to have an extra reduction at the same time.

We studied each step of the algorithm in detail. Obviously, the fixed window algorithm always executes pre-computation and post-computation. It can be seen that there is a common multiplier factor between the pre-computation modular multiplication ($m[i]=m[i-1]*m \bmod n$) in the fixed window exponentiation Algorithm and the post-computation ($c=c*m[k_j] \bmod n$) in the fixed window exponentiation Algorithm . That is when the index $i-1$ is equal to k_j, the same multiplier factor exists for a certain step. Therefore, there is a significant relation between modular power operation in pre-computation and post-computation. In this paper, we attempt to recover the private key using side-channel attack by relating the extra reductions to the key.

2.2. Knowledge of Recovering the Key

We recover secret key bit by bit using the Pearson correlation coefficient [13]. It has a value between +1 and −1, where 1 is a total positive linear correlation, 0 is no linear correlation, and −1 is total negative linear correlation. When the correlation coefficient value is high, the random variables are related, and it means

the hypothesis on the sub-key is correct. On the contrary, the correlation coefficient value less than 0 means that the initial guess was wrong. In practice, a good hypothesis is usually determined to give the highest correlation of all possible hypotheses.

For fixed window exponentiation algorithm, it functions by scanning the bits of an exponent from left to right. When Pearson's correlation coefficient of extra reduction information in post-computation and pre-computation is large, each window key value in post-computation is most likely the corresponding pre-computation index value. Thus, our attack ensures the recovery of bits of the key at a time, from most significant to least significant.

2.3. Method of the Attack

To verify our correlation predictions, we use l length static key k to perform n times the cryptographic operation with fixed window exponentiation algorithm and capture the corresponding side-channel information.

In the modular exponentiation, the secret exponent k is split into windows of fixed size w at each iteration where the most significant bit is 1. For each encryption $1 \le t \le n$, $1 \le i \le 2^w - 1$, $0 \le j \le l/w - 1$, we can get extrareduction $\left(X^t pre_i, X^t post_j\right)$. For all $1 \le i \le 2^w - 1$ and $1 \le t \le n$, if the extrareduction is existing, $X^t pre_i$ is 1. Otherwise, $X^t pre_i$ is 0. Likewise if the extrareduction is existing (resp. missing) in post-computation, the value of $X^t post_j$ is 1 (resp. 0) for all $0 \le j \le l/w - 1$ and $1 \le t \le n$.

Let us define a matrix PRE of n queries of length $2^w - 1$, which can be represented extra reduction occurrence in the pre-computation and define another matrix $POST$ of n queries of length l/w to express whether extra reduction present in the post-computation. Two vectors $Xpre_i$ and $Xpost_j$ respectively are columns in the pre-computation matrix and post-computation matrix representing a modular exponentiation, for $\left(Xpre_i, Xpost_j\right) \in \{0,1\}^2$ and $1 \le i \le 2^w - 1$, $0 \le j \le l/w - 1$. The matrix PRE and matrix $POST$ are represented as:

$$PRE = \begin{bmatrix} X^1 pre_1 & X^1 pre_2 & \cdots & X^1 pre_{2^w-1} \\ X^2 pre_1 & X^2 pre_2 & \cdots & X^2 pre_{2^w-1} \\ \vdots & \vdots & \ddots & \vdots \\ X^n pre_1 & X^n pre_2 & \cdots & X^n pre_{2^w-1} \end{bmatrix}$$

$$POST = \begin{bmatrix} X^1 post_0 & X^1 post_1 & \cdots & X^1 post_{l/w-1} \\ X^2 post_0 & X^2 post_1 & \cdots & X^2 post_{l/w-1} \\ \vdots & \vdots & \ddots & \vdots \\ X^n post_0 & X^n post_1 & \cdots & X^n post_{l/w-1} \end{bmatrix}$$

Next, we need compute the estimated probability the attacker respectively computes the Pearson correlation $r\left(Xpre_i, Xpost_j\right)$ as:

$$r\left(Xpre_i, Xpost_j\right) = \frac{n\sum X^t pre_i X^t post_j - \sum X^t pre_i \sum X^t post_j}{\sqrt{n\sum X^t pre_i^2 - \left(\sum X^t pre_i\right)^2}\sqrt{n\sum X^t post_j^2 - \left(\sum X^t post_j\right)^2}} \quad (1)$$

For each j, the attacker can observe $2^w - 1$ correlation coefficient in total. $r\left(Xpre_i, Xpost_j\right)$ is a measure of the linear correlation between two variables $Xpre_i$ and $Xpost_j$. In $r\left(Xpre_i, Xpost_j\right)$ we can determine whether there is a subtle correlation between two modular operations. For all multiplication share common operand, they all show significant correlation. So, if two modular operations do not share the same operand, we expect to get a small correlation value using sufficiently large vectors. Therefore, the maximal correlation value implies the sharing of one operand. Hence, the secret key can be directly recovered $\max\left(r\left(Xpre_i, Xpost_j\right)\right)$.

We illustrate $r\left(Xpre_i, Xpost_j\right)$ by **Figure 1** using 1,000,000 queries. It shows that the color depth of each position is different, which represents the strength of correlation.

As can be seen from the **Figure 1**, the correlation values are perfect on the diagonal, other positions are fairly shallow. This result may be explained by the fact that diagonal values represent correlations between identical values, while others represent correlations with different values.

And we also can clearly see the white diagonal area, which means a strong negative correlation between pre-computation and post-computation. When the output of the previous operation is equal to the input of the following operation, and if an extra reduction has been occurred in the previous operation, the result will be smaller, thus there is less likelihood for an extra reduction to exist in the following operation.

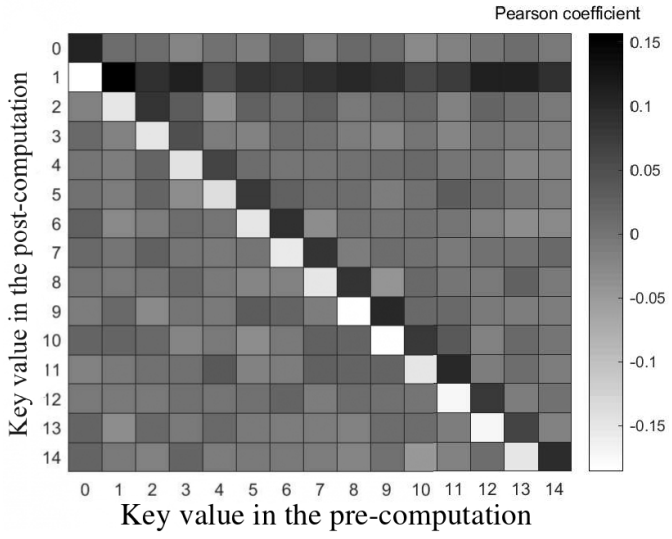

Figure 1. Pearson's correlation between $Xpre_i$ and $Xpost_j$ for simulation of RSA with window size = 4.

The graph also displays each cell in the first row is darker in color. A possible explanation for this is that there is always a common multiplier factor between the pre-computation and the post-computation, when the sub-key value is 1. In summary, these results confirmed that there exists a strong positive correlation between extra reductions of the pre-computation and the post-computation, provided that they have common operands. In other words, when Pearson correlation is the largest, the key in the post-computation would probably be the same as the key in the pre-computation.

Based on the value of this correlation coefficient is maximum, $\hat{k}_j = k_{i-1}$. $\hat{k}_j$ is estimated private key value in an attack and k_{i-1} is pre-computing the corresponding value.

But when the real key $k_j = 2^w - 1$, there is no corresponding pre-computation of the key value in our proposed method. The cause of the state is that key value is calculated to the maximum window in the pre-computation. For $m[i] = m[i-1] * m \bmod n$, the maximum value of i is $2^w - 2$, thereby the maximum value of $i-1$ is $2^w - 2$. That is to say, for this case, the attacker always guessed the incorrect key. It can render the algorithm less efficient. So we propose the threshold for the successful attack to fix it.

When the guess is correct, let us denote num as the total number of successful recovery of k_j. We also can compute the mean of the minimum correlation coefficient $\hat{E}r_{min}\left(k_j\right)$ for each key bit by:

$$\tau = \hat{E}r_{min}\left(k_j\right) = \frac{\sum_{q=1}^{num} \min\left(r\left(Xpre_{k_j}, Xpost_{k_j}\right)\right)}{num} \approx 0.0408. \tag{2}$$

The minimal values would depend on the key length in this step.

To estimate the key $\hat{k}_j$, we define decision function F_{FWA}:

$$\hat{k}_j = F_{FWA} = \begin{cases} k_{i-1} & \text{if } \max\left(r\left(Xpre_i, Xpost_j\right)\right) \geq \tau, \\ 2^w - 1 & \text{otherwise.} \end{cases} \tag{3}$$

2.4. Summary of the Attack

The entire attack process is divided into three parts. First, the attacker needs to separately collect the information about the extrareduction in the pre-computation and post-computation. So we get matrix *PRE* and matrix *POST*. The attacker then calculates the Pearson correlation value between each column of the matrix *PRE* and matrix *POST*. For each column of the matrix *POST*, the attacker can get a corresponding set of Pearson correlation coefficient. Finally, the attacker estimates the key through the decision function.

Algorithm 1 describes our attack to recover a secret key. Line 4 computes the Pearson correlation $r\left(Xpre_i, Xpost_j\right)$ for each bit using corresponding extra-reduction information. Line 9 recovers the entire estimated private key using the calculation of the threshold.

Algorithm 1. Our attack.

Require: $(Xpre_i, Xpost_j), w$,a set of n queries of l bits
Ensure: An estimation $k \in \{0,1\}^{l-1}$ of the secret exponent
1: **for** $i = 1 \to 2^w - 1$ **do**
2: **for** $j = l/w - 1 \to 0$ **do**
3: $r(Xpre_i, Xpost_j) \in \{0,1\}^2$
4: **end for**
5: **end for**
6: compute $\hat{k_j}$ using $r(Xpre_i, Xpost_j)$
7: **for** $j = l/w - 1 \to 0$ **do**
8: **for** $i = 1 \to 2^w - 1$ **do**
9: $\hat{k_j} = F_{FWA}(\max r(Xpre_i, Xpost_j), \tau)$
10: **end for**
11: **end for**
12: **return** $\hat{k}$

3. Experiment

In this section, we introduce our experiment and discuss the efficiency of the attack method. We put the correlation technique on a simulation of a fixed window exponentiation algorithm to testify our theoretical correlation predictions

3.1. Experiment Setup

We simulated our attack against the latest version 2.6.0 of mbed TLS with the private primes parameters defined by RSA-1024-p and RSA-1024-q. All of the experiments presented were run using an Intel Core i7-6700 CPU running at 3.410 GHz with 8 GB of RAM on windows. The secret keys length is 1024-bit. The experiment are as follows:

For different window size, we need to do repeated experiments.

1) Generated a random plaintext;

2) Simulate the RSA encryption process using k in the code blocks;

3) Save whether an extra-reduction is performed ($X^t pre_i = 1$) or not ($X^t pre_i = 0$) in pre-computation and is presented ($X^t post_j = 1$) or not ($X^t post_j = 0$) in post-computation;

4) Repeat steps 1-3 n times;

5) Pearson correlation analysis of collected data $\left(Xpre_i, Xpost_j\right)$ using Matlab;

6) Recover key according to the result of the correlation;

7) Record the queries n and the bits of recovered keys until the number of recovered keys does not change.

For different window size, we need to do repeated experiments.

3.2. Experimental Results

We at most run 10,000 queries to key recovery attempts for 1024-bit RSA with random input messages.

Figure 2 plots the correlation between pre-computation and post-computation at different window sizes when the lock key values are equal in pre-computation and post-computation. As the window size increases, the relationship becomes more obvious between pre-computation and post-computation. Additionally, the smaller the window size, the wider the range of Pearson coefficients.

It can be seen from **Figure 3**, when the requires is less than 3500, the smaller window size, the better the attack effect. In other words, as the window size is larger, the attack requires more queries to reach a given success rate. This is caused by **Figure 2**. We also can notice that as the number of queries increases, the success rate of guessed key value also increases. And when the number of queries is enough, we successfully recovered the key using side-channel information on the total number of extra reduction. From a statistical point of view, the reason for this is that the sample size is large, estimating the precision of unknown parameters will increase. as the window size decreases, the percentage of each key bit that is correctly guessed gradually increases.

Table 1 illustrates the number of that queries are approximately needed to recover all secret key in different window size using our attack method. The number of queries also mean time spent on the attack. Without regard to noise, for smaller window size, the key recovery method needs approximately 6000 total queries. There is a little difference between different window size on the queries of the attack.

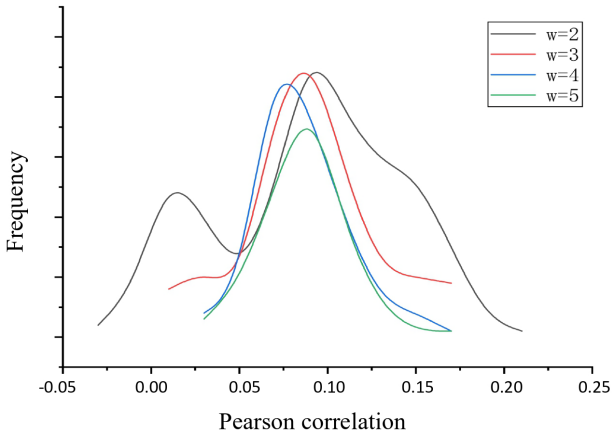

Figure 2. Pearson correlation of window key between pre-computation and post-computation.

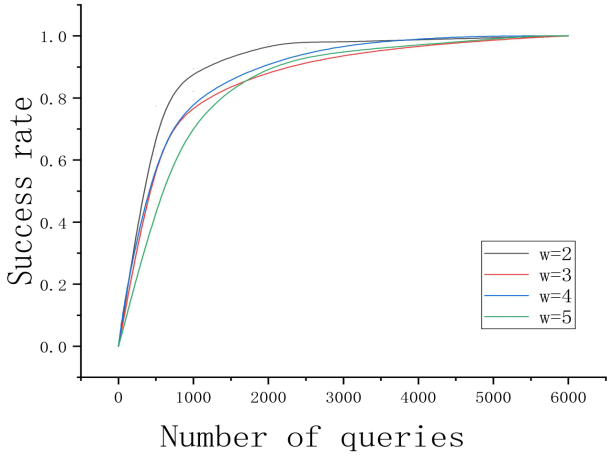

Figure 3. Success rate for our attack as number of queries.

Table 1. Result of our attack with different window size.

Window size	2	3	4	5
Number of queries	≈6000	≈6000	≈5000	≈5000
Success rate	≈100%	≈100%	≈100%	≈100%

4. Conclusion

In this paper, we analyze the vulnerability of the fixed window exponentiation algorithm with respect to side-channel attacks in detail. We find a new dependency relationship, namely a strong positive correlation between the extra reduction of pre-computation and post-computation at the end of Montgomery modular multiplications. Further, we exploit it to attack an RSA exponentiation with unknown the plaintext, modulus, and secret exponent.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Kocher, P.C. (1996) Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Koblitz, N., Ed., *Advances in Cryptology-CRYPTO*'96, Springer Berlin, Heidelberg, Berlin, Heidelberg, 104-113. https://doi.org/10.1007/3-540-68697-5_9

[2] Kocher, P., Jaffe, J. and Jun, B. (1999) Differential Power Analysis. *Proc Crypto*, **1666**, 388-397. https://doi.org/10.1007/3-540-48405-1_25

[3] Kocher, P., Jaffe, J., Jun, B. and Rohatgi, P. (2011) Introduction to Differential Power Analysis. *Journal of Cryptographic Engineering*, **1**, 5-27. https://doi.org/10.1007/s13389-011-0006-y

[4] Quisquater, J.-J. and Samyde, D. (2000) A New Tool for Nonintrusive Analysis of Smart Cards Based on Electro-Magnetic Emissions, the SEMA and DEMA Methods. *Eurocrypt* 2000 *Rumpsession*.

[5] Brier, E., Clavier, C. and Olivier, F. (2004) Correlation Power Analysis with a Leakage Model. *International Workshop on Cryptographic Hardware and Embedded Systems*, Springer, 16-29. https://doi.org/10.1007/978-3-540-28632-5_2

[6] Joye, M. and Yen, S.-M. (2002) The Montgomery Powering Ladder. *International Workshop on Cryptographic Hardware and Embedded Systems*, Springer, 291-302. https://doi.org/10.1007/3-540-36400-5_22

[7] Schindler, W. (2000) A Timing Attack against RSA with the Chinese Remainder Theorem. *Ches*. https://doi.org/10.1007/3-540-44499-8_8

[8] Arnaud, C. and Fouque, P.A. (2013) Timing Attack against Protected RSA-CRT Implementation Used in PolarSSL. *International Conference on Topics in Cryptology*, *Proc Crypto*. https://doi.org/10.1007/978-3-642-36095-4_2

[9] Brumley, B.B. and Tuveri, N. (2011) Remote Timing Attacks Are Still Practical. *European Symposium on Research in Computer Security*, Springer, 355-371. https://doi.org/10.1007/978-3-642-23822-2_20

[10] Koc, K.K. and Hung, C.Y. (1992) Adaptive m-Ary Segmentation and Canonical

Recoding Algorithms for Multiplication of Largebinary Numbers. *Computers & Mathematics with Applications*, **24**, 3-12.
https://doi.org/10.1016/0898-1221(92)90209-Z

[11] Schindler, W. (2015) Exclusive Exponent Blinding May Not Suffice to Prevent Timing Attacks on RSA. *International Workshop on Cryptographic Hardware and Embedded Systems*, Springer, 229-247.
https://doi.org/10.1007/978-3-662-48324-4_12

[12] Dugardin, M., Guilley, S., Danger, J.L., Najm, Z. and Rioul, O. (2016) Correlated Extra-Reductions Defeat Blinded Regular Exponentiation. *Cryptographic Hardware and Embedded Systems CHES*, **9813**, 3.
https://doi.org/10.1007/978-3-662-53140-2_1

[13] Benesty, J., Chen, J.D., Huang, Y.T. and Cohen, I. (2009) Pearson Correlation Coefficient. Noise Reduction in Speechprocessing. Springer, 1-4.
https://doi.org/10.1109/TASL.2008.919072