Middle-East Journal of Scientific Research 23 (3): 459-462, 2015
ISSN 1990-9233
© IDOSI Publications, 2015
DOI: 10.5829/idosi.mejsr.2015.23.03.22093

Comparative Investigation of Methods of Modular Exponentiation

[1]A. Shaikhanova, [2]M. Karpinski, [1]B. Ahmetov, [1]A. Zhauyt and [1]U. Imanbekova

[1]Department of Computer and Software Engineering, Kazakh National Technical University Named after K.I. Satpayev, Almaty 050013, Kazakhstan
[2]University of Bielsko-Biala, Bielsko-Biala 43-309, Poland

Abstract: Main parameters of evaluation of modern information protection systems that implement RSA encryption algorithm are performance, RAM consumption and stability of algorithm of used method of modular exponentiation to the attacks on implementation, particularly in the temporal analysis. Methods of modular exponentiation (binary method, β method and sliding window method) are characterized by the dependence of execution time of their algorithm on the key length and by the maximum number of used memory cells. Comparative investigation of operations of binary method, β method and sliding window method of modular exponentiation with "left-to-right" and "right-to-left"reading bits of exponent was conducted. Algorithm of β method of modularexponentiation has high performance and reasonable consumption of RAM.

Key words: Timecomplexity · Spacecomplexity · binary method · β method · Sliding window method

INTRODUCTION

Traditionally it is accepted to estimate the degree of complexity of algorithm in terms of consumed basic computer resources, such as CPU time and RAM. In this regard, such concepts as time and space complexity of the algorithm are introduced [1, 3].

Time complexity parameter is particularly important for applications with interactive mode of program or for real-time control tasks [2, 7]. It is necessary to spend some time to perform the operations of modular exponentiation algorithms (binary, β, sliding window methods).

Execution time of single operation of the algorithm depends on the speed of processor, so it can be said that in general, every single step of the algorithm is performed during certain time. Basic operations of modular exponentiation algorithms and time spent on each of them can be represented as Table 1:

In general, it can be assumed that ratio between the values of these times is as follows:

On the basis of data in Table 1, we can construct a mathematical model for calculating the time required to perform each of the algorithms for the implementation of methods of modular exponentiation.

$$c \le b \le q \le t \le r \le s \le d \tag{1}$$

Following time is required to perform binary method: "left-to-right" reading:

$$T1(n) = t + c + \sum_{i=k-1}^{0} r_i + \sum_{\{i|n_{i=1}\}} s_i = t + c + \lceil \log n \rceil \cdot r + H(n) \cdot s \tag{2}$$

"right-to-left" reading

$$T2(n) = t + c + b + \sum_{\{i|n_{i=1}\}} s_i + \sum_{i=0}^{k-1} r_i = t + c + b + H(n) \cdot s + \lceil \log n \rceil \cdot r \quad 3)$$

Following time is required to perform β method: "left-to-right" reading:

$$T3(n,w) = t + c + \sum_{i=1}^{\beta-1} s_i + c + \sum_{i=k-1}^{0} (d_i + s_i) = \\ = t + 2c + \left(\frac{\lceil \log n \rceil}{w} + 2^w - 1 \right) \cdot s + \frac{\lceil \log n \rceil}{w} \cdot d \tag{4}$$

"right-to-left" reading:

Corresponding Author: A. Shaikhanova, Department of Computer and Software Engineering, Kazakh National Technical University Named after K.I. Satpayev, Almaty 050013, Kazakhstan.

Table 1: Time required to perform basic operations of exponentiation algorithms

Operation	Time, in ticks	Meaning of the operation
a = b	C	Simple assignment
z = x mod m	B	Modulus assignment
FIND (max$\{n_i \ldots n_j\} \mid i, j+1 \leq w, n_j=1$)	Q	Finding the longest sequence of bits, so that $I-j+1 \leq w$ and $n_j = 1$
n = $(n_{k-1} \ldots n_0)_2$	T	Representation of numbers in binary notation
y = x . x mod m	R	Modulus squared
z = x . y mod m	S	Modular multiplication
z = y$^\beta$ *mod m*	D	Modular exponentiation

$$T4(n,w) = t + b + \sum_{w=1}^{\beta-1} c_w + \sum_{0}^{k-1}\left(d_{\{i|n_i=0\}} + s_{\{i|n_i=1\}} + d_{\{i|n_i=1\}}\right)$$
$$+2c + \sum_{w=\beta-1}^{1} 2s_w = \tag{5}$$
$$= t + \left(2^w + 1\right)c + b + \frac{\lceil \log n \rceil}{w}\cdot d + \left(\frac{\lceil \log n \rceil}{w} - W_0(n) + 2^{w+1} - 2\right)\cdot s$$

Where *$w_0(n)$* number of zero bits in the representationof number n to the base βobviously, that in binary image numbers $n \varepsilon \lceil \log n \rceil - H(n)$ of zero bits.

To convert number into β notation, binary image is divided into *n* windows with length of *w* Therefore, the upper bound *$W_0(n)$*:

$$W_0^{\max}(n) = \left\lfloor \frac{\lceil \log n \rceil - H(n)}{w} \right\rfloor \tag{6}$$

On the other hand, the lower bound can easily be determined as:

$$W_0^{\min}(n) = \left\lfloor \frac{\left(\lceil \log n \rceil - H(n)\right)\cdot w}{(w-1)\cdot\lceil \log n \rceil} \right\rfloor \tag{7}$$

Following time is required to perform sliding window method:

"left-to-right" reading:

$$T5(n,|w_i|) = b + s + \sum_{j=1}^{2^{|w_i|}-1} s_j + t + 2c +$$
$$+\sum_{i=0}^{k-1}\left((r+c)_{\{i|n_i=0\}} + (q+s+c+r)_{\{i|n_i\neq 0\}}\right) =$$
$$= b + s + \left(2^{|w_i|} - 1\right)s + t + 2c + \tag{8}$$
$$+(k - H(n))(r+c) + p(q+s+c) + r(|w_o| + \ldots + |w_i|) =$$
$$= t + b + 2c + kr + 2^{|w_i|}s + p(q+s+c) + (k - H(n))c =$$
$$= t + b + (2 + p + \lceil \log n \rceil - H(n))c + \lceil \log n \rceil r + \left(2^{|w_i|} + p\right)s + pq$$

"right-to-left" reading:

$$T6(n,|w_i|) = t + b + \sum_{\{j=1,3,\ldots,2^{|w_i|}-1\}} c_j + c +$$
$$+\sum_{i=k-1}^{0}\left((r+c)_{\{i|n_i=0\}} + (q+s+c+d)_{\{i|n_i\neq 0\}}\right) +$$
$$+\sum_{\{v=2^{|w_i|}-1,\ldots,5,3\}} (2s_v) + c = t + b + \left(2^{2|w_i|-2} + 1\right)c +$$
$$(k - H(n))(r+c) + p(q+s+c+d) + 2^{2|w_i|-1}s + c =$$
$$= t + b + \left(2^{2|w_i|-2} + 2 + \lceil \log n \rceil - H(n) + p\right)c +$$
$$+(\lceil \log n \rceil - H(n))r + \left(2^{2|w_i|-1} + p\right)s + pq + pd \tag{9}$$

Where p is number of windows,

$(|w_0| + \ldots + |w_i|)$ Sum of all odd windows that equal to Hamming weight, since these windows consist of only single bits.

Obviously that $p_{\max} = \left\lceil \frac{\log n}{2} \right\rceil$, *a* $p_{\min} = \left\lceil \frac{H(n)}{w_i} \right\rceil$ (10)

Thus, in general, for the investigation of the execution time of this algorithm following average value can be used

$$p = \frac{\left\lceil \frac{H(n)}{w_i} \right\rceil + \left\lceil \frac{\log n}{2} \right\rceil}{2} \tag{11}$$

Determination of the most productive algorithm of modular exponentiation:It is obvious that to improve performance of asymmetric encryption devices it is necessary to determine the most productive of known algorithms of modular exponentiation that are used in such devices. We consider solution of this problemon the example of described above binary, β and sliding window methods.

As mentioned above, total execution time of the algorithm of binary method is dependent only on the length of binary image of number n. Execution time of the algorithm of β method depends not only on the length of the binary image of number n, but also on the value of β (i.e. on the number w). Execution time of the algorithm of sliding window method depends on the length of the binary image of number n and on the width of odd window [6-9]. Taking this into account, it is possible to investigate dependence of the execution time of the algorithm on the length of binary image of number n.

RESULTS AND DISCUSSION

Shows this dependence for the averaged values of the Hamming weight (*H(n)*) and number of zeros in the β image of number n ($W_0(n)$), as well as for different values of w, the width of odd window and values of *c=1 b=1.5, q=1.6, t=1.6, r=15, s=16, d=19* (the ratio between the variables correspond to the number of ticks that the processor spends to perform the corresponding operations) [11].

In this case, $T1(n)$ and $T2(n)$ -$T3(n2)$ and $T3(n4)$ execution time of "left-to-right" β algorithm of modular exponentiation at $w = 2$ and $w = 4$, correspondingly. $T4(n2)$ and $T4(n4)$ executiontimeof "right-to-left" β algorithm of modular exponentiation at $w = 2$ and $w =4$, correspondingly. $T5(n3)$ and $T6(n3)$ executiontime of "left-to-right" and "right-to-left" method of sliding window at the length of the window $w_i = 3$.

Analysis of Fig. 1 shows that the execution time of algorithms of modular exponentiation is linear. In addition, the fastest algorithms are that of "left-to-right" and "right-to-left" β method andthe most time consuming is the algorithm of binary method.

Fig. 2 and Fig. 3 shows, respectively, the dependence of speed of algorithms of "left-to-right" and "right-to-left" β method on the value of power of w base at different key length and at averaged value of the Hamming weight.

Using data from Fig.2 and Fig.3 we can determine the optimum base at which there is the smallest delay in work of algorithm, i.e. the minimum *T*3 and *T*4 , respectively and thus provide maximum productivity for given values of the exponent.For algorithms of β method values of *w* presented in Table 2 will be the best.

Space complexity of the algorithm, i.e. consumption of computer memory for its execution, becomes critical when the volume of data to be processed is almost equal to amount of RAM.In modern computers acuteness of this problem is reduced due to increase in amount of random access memory (RAM) and to usage of multilevel storage system. For programs that implement the algorithm very large, almost unlimited, memory space (virtual memory) is available. Lack of main memory only leads to a slowdown through the exchange of data with

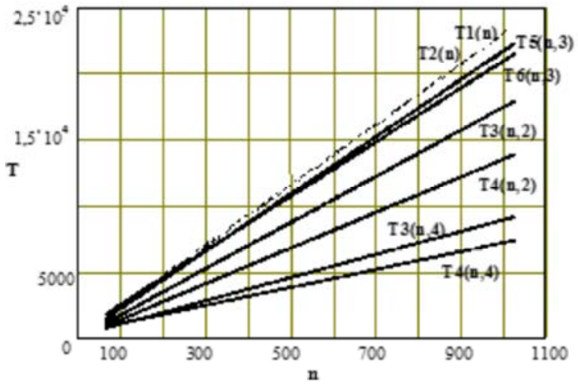

Fig. 1: Evaluation of the performance characteristics of investigated algorithms

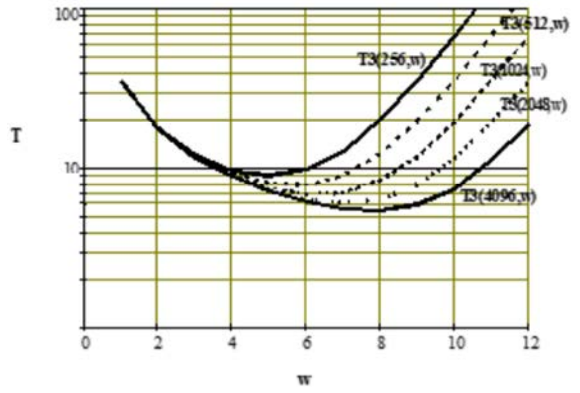

Fig. 2: Dependence of the speed of algorithm of "left-to-right" β method on the value of the power of w base

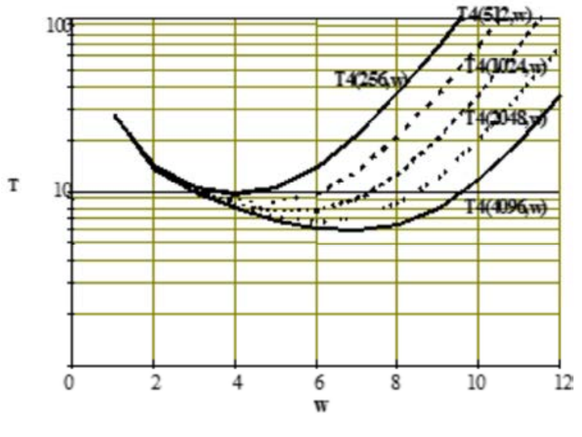

Fig. 3: Dependence of the speed of algorithm of "right-to-left" β method on the value of the power of w base

Table 2: The optimum values of the power of bases of β method at different length of *n* key.

	w	
Length of *n* key	"left-to-right" β method	"right-to-left" β method
4096	8	7
2048	7	6
1024	6	5
512	6	5
256	5	4

Table 3: The maximum number of memory cells involved in the execution of algorithms of modul are xponentiation

Modular exponentiation algorithm	The number of memory cells
Binary	2
β	2^w
Sliding window	$2^w i$

the disk. Special techniques are used to minimize the loss of time in this exchange.It is the usage of cache memory and hardware preview of program commands on the required number of steps ahead that allows totransferrequired values from disk to main memory in advance [5, 11-14]. When performing considered modular exponentiation algorithms maximum number of registers according to Table 3 are busy in computer memory.

Analysis of Table 3 shows that the largest consumption of memory is during the execution of sliding window algorithm, since length of the largest window can be equal to the half length of the key. In the case of β modular exponentiation algorithm consumption of memory depends on the chosen notation, i.e.on the value.

CONCLUSION

Parameters of time complexity and space complexity were investigated. It was found that best methods for application are β method and sliding window method of modular exponentiation with "left-to-right" reading bits of exponent.

REFERENCES

1. Bellezza, A., 2001. Countermeasures against side-channel attacks for elliptic curve cryptosystems. In: Cryptology ePrint Archive, pp: 96-103.
2. Biham, E. and A. Shamir, 1997. Differential fault analysis of secret key cryptosystems. 17th annual international cryptology conference on advances in cryptology, Lecture notes in computer science, 1294: 513-525.
3. Kshetri, N. and S. Murugesan, 2013. EU and US Cybersecurity Strategies and Their Impact on Businesses and Consumers. Computer, 10(46): 84-88.
4. Kurose, J.F. and K.W. Ross, 2011. Computer networking: A top-down approach, 6th edn. Addison-Wesley, 10: 215-225.
5. Mangard, S., E. Oswald and T. Popp, 2007. Power analysis attacks: Revealing the secrets of smart cards. Springer, 3: 39-46.
6. Stallings, W., 2013. Cryptography and network security. Principles and practice, 6th edn. Prentice Hall, 5(1): 62-78.
7. Vasyltsov I.V., 2009. Ataky special nohovydunakry ptoprystroyitametodyborot, byznymy (The attacks of specialty peoncry ptodevices and methods of dealing with them), 5(36): 1244-1256.
8. Ross, T.J., 1995. Fuzzy Logic with Engineering Applications, McGraw-Hill In, pp: 589-600.
9. Ozyer, T., 2007. Intrusion detection by integrating boosting genetic fuzzy classifier and data mining criteria for rule pre-screening. Journal of Network and Computer Applications, 30: 99-113.
10. Ros, F.J., J.A. Martinez and P.M. Ruiz, 2014. A survey on modeling and simulation of vehicular networks.Communications, mobility and tools, Computer Communications, 43: 1-15.
11. Hong, S.M., 1996. New Modular Multiplication Algorithms for Fast Modular Exponentiation. Theory and Àpplication of Ñryptographic Òechniques, 15th annual international conference, pp: 166-177.
12. Messerges, T.S., 1999. Power Analysis Attacks of Modular Exponentiation in Smartcards. Cryptographic Hardware and Embedded Systems, pp: 144-157.
13. Vasyltsov, I., 2005. Power and Fault Analysis in ECC. Problems and Solutions, e-Smart 2005 Conference, pp: 142-155.
14. Hanley, N., 2007. Correlation Power Analysis of Large Word Sizes. Irish Signals and Systems Conference (ISSC 2007), pp: 13-14.