The Protection of Modular Exponentiation Operands from Their Reconstruction by Simple Power Analysis

Akram A. Moustafa and Saleh Alomar

Abstract: **The goal of this research is to point out the potential vulnerabilities of modular exponentiation operands reconstruction by power dynamic analysis and to elaborate countermeasures. It has been shown that exponent of modular exponentiation which is the secret key of RSA, El-Gamal and DSA can be reconstructed by timing power analysis. As a countermeasure, a special algorithm for modular exponentiation has been worked out. The proposed algorithm does not use conditional operators and includes false operators which inhibit timing power analysis. It has been shown that the implementation of the proposed approach requires about 25 % more time for modular exponentiation.**

Index term: modular exponentiation, SPA-reconstruction exponent, keys.

I. INTRODUCTION

The dynamic intensification of information integration as the basis of computer networks plays a dominant role at the present stage of development in the majority of spheres of human activity. The important condition of information integration expansion is the solution to a problem of effective data protection and the delimitation of access rights to the information in computer systems and networks. The expansion of the use of computer and network technologies in the areas of financial activity and for the management of complex technical systems connected with technology risk requires the guarantee of a high level of reliability for the protection of information in computer control systems and networks. Thus, there is an obvious need for an increase in the effectiveness of information protection from the point of view of the expanding areas which now use integrated systems.

There are a number of factors obviously reducing the reliability of any means of data protection. In recent years, one of the most potentially dangerous threats to information security in the systems of computer management has been unauthorized access to data by means of the measurement and analysis of dynamic changes in a computer's work parameters during programs [1]. The problem of protecting data from reconstruction by means of the measurement and analysis of dynamic changes in power consumption is especially in vital. In modern computer networks, a significant proportion of terminal devices are made with microcontrollers for which the measurement of power consumption and its comparison to executable commands can be carried out fairly simply [1]. At the same time, such terminal devices support network reports on information protection, the use of which requires confidential data: keys and passwords. Modern technologies allow statistical reconstruction of these confidential data on the basis of the analysis of power consumption dynamics [2]. The analysis of such technologies proves that they are most effective for access to constant components in the information protection systems (to keys and passwords). It is especially dangerous to mechanisms of data protection, whose keys are seldom modified. Such mechanisms, in particular, concern algorithms with the open key, the most widespread of which are *RSA, El-Gamal* and *DSA*. Modification of keys at each session of this algorithm essentially defies the concept of an openness of keys. Therefore the keys of the algorithms mentioned are most vulnerable to unauthorized access through the analysis of power consumption dynamics. Similarly, these algorithms underlie the network reports on information security; so that loss of control over their keys threatens with major violations in work of the information protection systems in networks.

Thus, the problem of protection against reconstruction through the analysis of power consumption of algorithm keys which form the basis of modular exponentiation is important and current at the present stage of development of data protection technology in networks.

II. ANALYSIS OF THE PROBLEM OF THE PROTECTION OF DATA FROM POWER ANALYSIS

The basic computing operation on the performance of a program using a great number of algorithms of information protection with the open key, including RSA, is modular exponentiation X^E mod M, and $X, E < M$. The quantity n binary digits of numbers X, E and M essentially exceed digit capacity k of the processor. Each of the numbers X, E and M, could be presented in the form of s words consisting of k bits $(s=n/k)$, each of which lies in an interval from 0 up to 2^k-1: $X=\{x_{s-1},\ldots,x_1,x_0\}$, $E=\{e_{s-1},\ldots,e_1,e_0\}$and $M=\{m_{s-1},\ldots,m_1,m_0\}$, $\forall j\in\{0,\ldots,s-1\}$, $0\le x_j,e_j,m_j\le 2^k-1$. For the instruction of bits of words X, E and *the* M are designations used, with two indices, for example $e_0=\{e_{0,k-1},e_{0,k-2},\ldots,e_{0,1},e_{0,0}\}$; $\forall l\in\{0,\ldots,k-1\}\}$: $x_{j,l}$, $e_{j,l}$, $m_{j,l}\in\{0,1\}$:

Akram A. Moustafa, Computer Science Department, Al al-Bayt University, Mafraq, Jordan, P.O.BOX 922283, 11192, hamarchi@yahoo.com.
Saleh Alomar, Computer Science Department, Al-Ahliyyah Amman University, Amman, Jordan, E-mail: salehalomar@yahoo.com

$$X = \sum_{j=0}^{s-1} x_j \cdot 2^{j \cdot k} = \sum_{j=0}^{s-1} \sum_{l=0}^{k-1} x_{j,l} \cdot 2^{j \cdot (k-1)+l},$$

$$E = \sum_{j=0}^{s-1} e_j \cdot 2^{j \cdot k} = \sum_{j=0}^{s-1} \sum_{l=0}^{k-1} e_{j,l} \cdot 2^{j \cdot (k-1)+l},$$

$$M = \sum_{j=0}^{s-1} m_j \cdot 2^{j \cdot k} = \sum_{j=0}^{s-1} \sum_{l=0}^{k-1} m_{j,l} \cdot 2^{j \cdot (k-1)+l}$$

The process of modular exponentiation is reduced to consecutive performance $\log_2 E=n$ cycles, in each of which the operation of squaring of the result received from a previous cycle and, in addition, depending on current t bit of degree E, is carried out by the operation of multiplication. Proceeding from the order in which digits of degree E are analyzed, two versions of modular exponentiation, from right to left and from left to right, exist.

Algorithm of binary exponentiation from right to left is.

```
1. $A = 1$; S = $X$;
2. for ($j$=0; $j< s$ ; $j$++ )
    2.1. for ($l$=0; $l< k$; $l$++)
    {
        2.1.1. if ( $e_{jl} = 1$) $A = A \cdot S \bmod M$;
        2.1.2. $S = S \cdot S \bmod M$;
    }
Result $A=X^E \bmod M$.
```

For the calculation of weights, squaring is used, and for the formation of the result, multiplication is used. An average of multiplication operations $=1.5 \cdot n$.

In practice, a greater distribution is received by algorithms which assume the analysis of rows of degree E from the high-order binary digits (from left-to right): In this case, the multiplier on performing the operation of multiplication represents a constant number equal to X that creates potential preconditions for the increasing of multiplication speed.

```
1. $A = 1$;
2. for ( $j= s$-1; $j$ >=0; $j$ -- )
    2.1. for ( $l=k$-1; $l$>=0; $l$--)
    {
        2.1.1. $A = A \cdot A$ ;
        2.1.2. if ( $e_{j,l} = 1$) $A = A \cdot X$ ;
    }
Result $A= A=X^E \bmod M$.
```

Using the algorithms based on modular exponentiation a confidential key is the exponent's code E which practically can't be changed. Accordingly, the primary goal of breaking down the protection is to receive a code. As a collateral problem we have the problem of access to the value of code X or the result - $A=X^E \bmod M$ with the unknown X, but constant E [2].

As already noted, a significant number of terminal devices of networks are made with built-in microcontrollers and microprocessors used for data gathering or management by the process equipment. In 2005 the proportion of such terminal devices made up 60 % of the market share [2], with a tendency to grow alongside the development of computer and network technologies, and also with the expansion of areas where they are used. This type of terminal device supports reports from information protection in networks where constant confidentiality parameters (passwords and keys) of cryptographic algorithms are used. For microcontrollers and microprocessors it is fairly to measure the dynamics of power consumption during performance of programs and to compare it with the commands performed. All the processing devices in computer use are constructed on logic gateways whose basic part consists of transistors. The currents proceeding through the transistors depend on their working points and vary in leaps and bounds on switching gates. Thus, the size of a current consumed depends on the value of the data bits that are processed in real time [3]. For the measurement of the current consumed by a computer, a resistor is added to its supply circuit, the pressure on which is measured by the precision digital micro voltmeter [3]. Research on digital micro voltmeters allows the taking of measurements with a frequency of discrimination of 1-2 GHz, and an accuracy of up to 1мкB. Industrial development has opened up for general use, products which provide frequency of digitization of 50 MHz and accuracy 5мW.The majority of microcontrollers, built in microprocessors and smart-cards works with a clock frequency of 20-50 MHz.

By means of the tools described it is possible to execute a binding of commands of the program being performed (computer devices of a particular type are practically invariable) to the diagram of power consumption change. Currently, two technologies of data reconstruction from the results of power consumption dynamics measurement are widespread. The first of these is known as SPA (Simple Power Analysis) - the simple analysis of capacity [4]. The essence of this technology consists in the direct interpretation of measurements of power consumption during the performance of a recognized program. With accurate use of SPA it is possible to establish the sequence of the carrying out of commands. As each of commands takes a certain time, SPA technology is linked to the time analysis of the performance of program commands. [1].

Other technology using power consumption dynamics as measurements for the reconstruction of process able data is the differential analysis of capacity (DPA-Differential Power Analysis). This technology was also described for the first time in 1999 [5] and focused on data reconstruction constantly used in programs by statistical processing a large number of power consumption diagrams during the performance of the same program. Specifically, DPA technology can be effective in the analysis of a large number of power consumption diagrams during performance of the same program, through the use of a cryptographic algorithm which processes different data, but uses keys which remain constant. This situation is typical of the use of microcontrollers and smart-cards as the terminal devices of computer networks.

Through the results of the an analysis of publications [2], concerning technologies of data reconstruction using SPA and DPA, there are currently no mathematical models of bit influence on process able information on power consumption. However, this influence does take place and can be revealed by the statistical method on which the concept of DPA is constructed.

As a first approximation it is possible to say that SPA is effective for data reconstruction depending on the order of commands during performance while DPA allows the reconstructing of the data directly used in calculations. From this, for the reconstruction of exponent *E* the greatest danger is represented by SPA as follows from the results of the above algorithms; it does not directly participate in calculations, but defines the order of commands, so we should analyze the potential danger of using an application SPA to obtain an exponent's digit values.

One of the central questions of such an analysis is the estimate of operations of squaring and multiplication. The results obtained in work [2] allow a positive estimate of such opportunities even provided that for both operations the same algorithm of multiplication is used. As a result, the values of all digits *E* could be restored easily enough by tracing the diagram of the sections, corresponding to figure 2.1.1 (for exponentiation from right-to left) or figure 2.1.2. (for exponentiation from left-to right).The second danger of using application SPA consists in the opportunity of tracing the diagram of change in the command sequence depending on the conditional transition which is performed as a result of the analysis of the current exponent bit. The danger discussed does not depend on whether it is distinguishable through the diagram of operations of squaring and multiplication. Using such technology, the reconstruction *E* could be performed simply enough.

DPA technology represents a danger in the solution of a problem of code *X* reconstruction through the analysis of the diagram of the unknown value of exponent *E.* It is evident that such an opportunity provides an opportunity to obtain modular exponentiation algorithm from left-to right, as in figure 2.1.2, where the constant code X is taking part. This code can potentially be restored by the statistical analysis of power diagrams on the performance of the corresponding operations. Hence, from the point of view of operands protection of modular exponentiation from reconstruction using DPA, the algorithm from right-to the left in which there are no constant operands is preferable.
Thus, the analysis conducted has shown that base algorithms of modular exponentiation do not provide protection against the reconstruction of an exponent's code *E* using SPA.

III. REVIEW OF METHODS OF POWER ANALYSIS COUNTERACTION

Currently, a number of solutions for counteracting of data reconstruction by means of the analysis of power consumption dynamics are suggested [1-5] all of which remain within the limits of the following conditions:

- The masking of all operands which depend on constant keys, through the varying of the casual codes on each performance of the algorithm;
- The casual rearrangement of operations which do not affect the result of the working of the algorithm;
- An introduction of casual commands not influencing the result and complicating the "binding" of the power consumption diagram to the operations of the algorithm.

Masking of operands is the most effective way of counteracting reconstruction of constant keys through the analysis of power consumption dynamics [4].

For the masking of key RSA-exponents *E* the best known approach [1]is the additive mask *E* product of a random number *r* on $\phi(M)=(p-1)\cdot(q-1)$ in which the connection $p\cdot q=M$. is used. The masked exponent E can be presented, thus, in the form of $E' = E + r\cdot\phi(M)$. Any *X* is performed $X^{r\cdot\phi(M)} \bmod M = 1$, then то $X^{E'} \bmod M = X^{E} \bmod M$. It means that for the removal of a mask there is no need for special operations. The drawback of the stated masking method is that the imposing of an additive mask $r\cdot\phi(M)$ essentially increases word length *n'* of exponent *E'*. For the exponent's known value of the module *M*, a code $\phi(M)$ is simple enough to define. If not used in a combination with masking as a special means of protection from SPA, then values *n'* and *E'*, as shown in the previous section, can be reconstructed by the simple analysis of the power consumption diagram. Considering, that $r\cdot\phi(M) >> E$, the approximate value *r'* can be defined, as quotient $r' \approx E'/\phi(M)$. Owing to the fact that sizes *E* and $\phi(M)$ are values of one order (*n*-digit binary numbers), then true value $r \le r'$ also differs from *r*' on units (at close values *E* and ϕ(M*)*, it is most probable, that $r = r'-1$). It means, that it is possible to restore, basically, value *r* by small selection, and so to reconstruct its value.
Hence, without using special measures for counteracting SPA, masking exponent's *E* does not exclude the opportunity of reconstruction of key RES. For the counteraction of SPA, the random rearrangement of operations using a combination of two base algorithms modular exponentiation [2] is most widely applied. For this purpose the casual, lying in an interval from 0 up to *s-1*starting point $d\in\{0,\dots,s-1\}$is selected. Modular exponentiation from right-to the left from 0 up to point *d*, with fixing result in a variable *A* and degrees in variable *S* is also performed. Then, having as initial received values *A* and *S* , modular exponentiation from left-to right from *s-1* up to *d+1*is performed. The use of such a method, in the opinion of its authors [2] means that on performing SPA it is difficult to establish the order of following exponent's bits. However, it is necessary to bear in mind, that the number of variants of a starting point *d* is very insignificant (no more than 10^3) and all these variants have to be looked through to be aware of *X* and result X^E mod *M*. In practice, with a constant exponent *E*, having at our disposal the power consumption diagram of the realization of several operations in modular exponentiation, it is fairly simple to reconstruct *E*.

Another method of counteracting SPA is the rearrangement of squaring and multiplication operations. This way assumes as an algorithmic basis the exponentiation from right-to left. The essence of the method is that all squaring, retaining those results (values *S*) which participate in the operations of multiplication in the memory,. (Figure 2.1.1. corresponding algorithm) are performed at the start. Then from the memory the remaining squares participating in multiplication are all taken and the operations of multiplication [1] are all performed one after another. Below is the corresponding modification of the algorithm from right to left.

```
1. $A = 1$;  $S = X$; $h = 0$;
2. for ($j=0$; $j< s$ ; $j$++ )
    2.1. for ($l=0$; $l< k$; $l$++)
    {  2.1.1.  if ( $e_{jl} = 1$)
               {
                    2.1.1.1.  $Z[h] = S$;
                    2.1.1.2.  $h$ ++ ;
```

```
            }
        2.1.2.  $S = S·S$ mod $M$;
    }
  3. for ($q$ =0; $q<h$; $q$++)  $A = A·Z[q]$ mod $M$;
  Result $A=X^E$ mod $M$.
```

The drawback of such a method is that with the use of SPA it is possible to check the condition which is performed in figure.2.1.1. Fragments of the power consumption diagram on performance of commands in figure 2.1.1.1. and 2.1.1.2. can be detected . The existing ways of counteraction SPA, has shown that they do not provide reliable protection against reconstruction of exponent E with the analysis of power consumption dynamics.

The goal of this work is the creation of methods of protection against the restoration of an exponent's code E resulting from the analysis of power consumed by a computer during the performance of modular exponentiation.

IV. THE ORGANIZATION OF MODULAR EXPONIENTATION FOR COUNTERACTING SIMPLE POWER ANALYSIS

The analysis of known methods of counteraction against reconstruction of the exponents which are performed using SPA has shown that a key problem is the lack of opportunity of tracking through SPA the conditional operators which are performing consecutive testing of digits of E. For protection from SPA, it is necessary to modify the algorithm of modular exponentiation so as to exclude conditional operators from it. As a basis for the modified algorithm, modular exponentiation from the right to left is chosen as it does not use constant elements and accordingly it is steadier for is DPA. Below is the suggested modification of modular exponentiation algorithm:

```
  1. $A[0] = 1$; $A[1]=1$;  $S = X$; $y$ =1;
  2. for ($j$=0; $j< s$ ; $j$++ )
        2.1. for ($l$=0; $l< k$; $l$++)
        {
           2.1.1. $q = e_j \& y$;
           2.1.2. $e_j = e_j >>1$;
           2.1.3. $A[q] = S·A[1]$ mod $M$;
           2.1.4. $S = S·S$ mod $M$;
        }
  Result $A[1] =X^E$ mod $M$.
```

In the suggested modification there are no conditional operators: in each cycle, one operation of squaring and multiplication is performed that does not allow definition by means of SPA value of the category process able in this cycle's exponent digit. The exclusion of conditional operators is carried out thanks to the introduction of a parasitic operation of multiplication whose result is fixed in A [0], and the sum of the correct result is carried out in A [1]. It is fairly clear that due to the introduction of parasitic operation of multiplication, the computing complexity of modular exponentiation increases by 25 %.

As the second most effective method of counteracting SPA-reconstruction exponent E, the performance of modular exponentiation using additive chains is suggested. Additive chain V in length L for positive integer E is named the sequence $u_0,u_1,…,u_s$ of positive numbers and the sequence connected with them, $w_1,w_2,…,w_s$ pairs $w_i=<i_1,i_2>$, $0≤i_1,i_2<i$, which possess the following properties:

1. $u_0 = 1$ и $u_L = E$;
2. $\forall\ u_i$, $1≤ i≤ L$, $u_i = u_{i_1} + u_{i_2}$

The algorithm of modular exponentiation X^E mod M on the basis of additive chains has the following appearance:

```
  1. $G[0] = X$;
  2. for( $j$=1; $j$<=$L$; $j$++)  $G[j] = G[j_1]·G[j_2]$ mod $M$;
  Result: $G[L] = X^E$ mod $M$.
```

For example, for $E=12$, it is possible to generate an additive chain in the form of a set of numbers {1,2,3,6,12} to which correspond the pairs: w_1=<0,0>, w_2=<0,1>, w_3=<2,2>, w_4=<3,3>. For the same exponent $E=12$, it is possible to generate another additive chain which is set up with a set of numbers {1,2,4,5,10,12} to which correspond the pairs: w_1=<0,0>, w_2=<1,1>, w_3=<0,2>, w_4=<3,3>, w_5=<1,4>. According to the algorithm described above, the calculation X^{12} using the last chain is performed in the following sequence: G [0] =X;

$G[1]=G[0]·G[0]$ mod $M = X^2$ mod M;
$G[2]=G[1]·G[1]$ mod $M = X^4$mod M;
$G[3]=G[0]·G[2]$ mod $M = X^5$ mod M;
$G[4]=G[3]·G[3]$ mod $M = X^{10}$ mod M;
$G[5]=G[1]·G[4]$ mod $M = X^{12}$ mod M.

As the resulting algorithm does not contain conditional operators, time analysis of power consumption dynamics on its performance will not allow the restoration of value E, even if it were possible to distinguish the operations of multiplication and squaring. As for the exponent's digit capacity length n used in practice, (the order 1024) there are a great number of variants in its decomposition in additive chains, so that to restore any particular chain is obviously not possible through the results of the analysis of power consumption dynamics.

The effectiveness of using of additive chains at modular exponentiation as means of counteracting SPA is due to the fact that in the practice of exponent E, having a closed key RSA of the subscriber which does not vary. It allows generating, using random numbers, an additive chain according to which can be compiled the program which will be performed directly.

A failing of additive chains utilization is that an additional memory size for array G and storages of pairs w which set communications of chains components is requires.
Thus the required memory size is defined by length L of a chain. If we consider that the average quantity of units in *n-digit* exhibitor E is equal $n/2$, the lower limit of value of length L of a chain makes $n + \log_2 n - 2$. For the carrying out of the resulting algorithm, storage of relatively small number of elements of array G is required: basically, it is possible to execute the algorithm on the storage of only 2 numbers of the specified array. For storage of pairs w, size V_w of memory defined by the formula is required:

$$V_w = 2 \cdot (n + \log_2 n - 2) \cdot \lfloor \log_2 (n + \log_2 n - 2) \rfloor \qquad (1)$$

For example, the RSA with *n=1024*, V_w = 2838 bytes does not exceed the size of the built- in memory of the majority of modern built in microcontrollers.

V. CONCLUSIONS

The research carried out directed towards the development of methods of counteracting the restoration of exponents, performed by measurement and analysis of dynamics of a consumed computing platform of capacity at modular exponentiation allow us to make the following conclusions:

- The operation of modular exponentiation underlies some cryptographic algorithms, such as the exponent of the specified operation as a part of the closed key of these algorithms, which rarely varies
- The greatest danger to the reconstruction of exponents is the time analysis of power consumption by a computing platform on performing an operation of modular exponentiation.
- An effective method of counteracting the exponent's reconstruction through the time analysis of power consumption is the exclusion of conditional operators. The algorithm satisfying this requirement including parasitic operations which complicate the effective analysis of power consumption dynamics is suggested. The carrying out of the suggested algorithm requires 25 % more time in comparison with normal modular exponentiation.
- An approach to a solution to the problem by counteracting the time analysis of consumption of capacity for the reconstruction of a confidential exponent's code is suggested. The approach is based on the use of technology of modular exponentiation on the basis of additive chains.

The suggested means of counteracting access to the closed keys of information protection algorithms, in which the basic operation of modular exponentiation lies, can be used effectively to increase of information security in computer networks.

REFERENCES

[1] Cohher P. Timing Attack on Implementations of Diffie-Hellman, RSA, DSS and other systems, Proceeding of Advances in Cryptology-"Crypto-96." LNCS-1882. Springer-Verlag.-1996, pp.104-113.

[2] Messerges T.S., Dabbish E.A., Sloan R.H. Power Analysis Attacks of Modular Exponentiation in Smartcards, Proceeding of 1-th International Workshop "Cryptographic Hardware and Embedded Systems " (CHES-1999), LNCS-1717. Springer-Verlag.-1999. pp. 145-157.

[3] Akkar M.L., Bevan C., Dischamp P., Moyart D. *Power* analysis, what is now possible, Proceeding of International Workshop "Asiacrypt-2000." LNCS-1976. Springer-Verlag.-2000, pp. 489-502.

[4] Mayer-Sommer R., Smartly analyzing the simplicity and power of simple power analysis on smartcards, Proceeding of 2-th International Workshop "Cryptographic Hardware and Embedded Systems" (CHES-2000), LNCS-1965. Springer-Verlag, 2000, pp. 78-92.

[5] Kocher P., Jaffe J., Jun B. Differential Power Analysis, Proceeding of Crypto' 1999, pp.388-404.