Power Attack of MARS and Rijndael*

WU Wen-ling, HE Ye-ping, FENG Deng-guo, QING Si-han

(*Engineering Research Center for Information Security Technology, The Chinese Academy of Sciences, Beijing* 100080, *China*);
(*State key laboratory of Information Security, Institute of Software, The Chinese Academy of Sciences, Beijing* 100080, *China*)
E-mail: wwl@ercist.iscas.ac.cn
http://www.ios.ac.cn
Received July 17, 2000; accepted September 27, 2001

Abstract: MARS and Rijndael are analyzed by using power attack. It is shown that the complexities on MARS are 2^{208}, 2^{168} and 2^{116} for 256, 192 and 128 bits key respectively, the complexities on Rijndael are 2^{131}, 2^{99} and 2^{67} for 256, 192 and 128 bits key respectively. Although the complexities are too big to implement, the approach presented in this paper reduces greatly the key size of MARS and Rijndeal.
Key words: key; block cipher; power attack; key schedule; complexity

P.Kocher, J.Jaffe,and B.Jun[1] presented power attack on smart card implementations of cryptographic algorithms，which is easy to carry out and very effective in practice. The basic idea behind power attack is that the power consumed by the smart card at any particular time during the cryptographic operation is related to the instruction being executed and to the data being processed. For instance, multiplication consumes more power than addition, and writing 1's consumes more power than writing 0's. The limitation of the above approach is that the attacker must know all ciphertexts or all plaintexts, which is almost impossible in reality. Biham and Shamir[2] introduced a variant of power attack, in which the attacker need not know either the inputs or the outputs of the encryption algorithm, and need not know the software implementation details in the smart card. For the sake of simplicity, we assume that the protocol used in the smart card always performs its sub-protocols in the same order, and always requires same number of clock cycles for executing each sub-protocol. As a result, we can align the power consumption graphs of different executions of the protocol, then compare consumed power of each instruction.

The attack can be performed in two steps:

Step 1. The attacker finds the tiny sections of the power consumption graph which are related to the key scheduling parts of the encryption operations. The attacker can accomplish this task in two sub-steps.

1) The attacker executes a large number of trials on a single smart card for different data, then compares the power consumption graphs at each clock cycle, and discards those clock cycles in which the graphs show a significant variability of the power consumption, due to the differences of the processed data. The remaining clock

* Supported by the National Natural Science Foundation of China under Grant No.60103023 (国家自然科学基金); the National Grand Fundamental Research 973 Program of China under Grant No.G1999035802 (国家重点基础研究发展规划 973 项目)

WU Wen-ling was born in 1966. She is an associate professor of the Institute of Software, CAS. Her current research interests are design and analysis of block cipher. **HE Ye-ping** was born in 1962. He is an associate professor of the Institute of Software, CAS. His current research interests are design and analysis of block cipher. **FENG Deng-guo** was born in 1965. He is a professor and director of the State key laboratory of Information Security, the Institute of Software, CAS. His current research areas are theory and technology of information security. **QING Si-han** was born in 1939. He is a professor and director of the Engineering Research Center for Information Security Technology, CAS. His current research areas are theory and technology of information security.

cycles represent operations which are data independent.

2) The attacker repeats 1) for several smart cards with different keys, and finds their common data independent regions. Among these regions, the attacker discards the clock cycles which have small variability among the different smart cards. The remaining clock cycles which are related to the key schedule are needed by the attacker.

Step 2. The attacker collects the Hamming weights of each byte of sub-keys. In software implementation on an 8-bit smart card the iterated block cipher usually computes each sub-key just before it is used, since the small RAM capability makes it difficult to calculate all sub-keys in advance. The sub-keys are computed and stored in RAM in chunks of 8 bits, and consumed power during the write operation is related to the number of 1's in the 8 written bits. So it is possible to predict the Hamming weights of each byte of sub-keys[2].

Step 3. The attacker studies the key scheduling algorithm carefully, extracts the master key or information about the master key by using the Hamming weights of each byte of sub-keys.

It has been pointed out[2] that step 1 and step 2 are easy to implement practically. Therefore, it is assumed in this paper that the attacker knows the Hamming weights of each byte of sub-keys.

NIST started to collect AES (Advanced Encryption Standard) candidates in 1997, and publicized five finalist candidates in 1999，which include MARS and Rijndael. It has been shown that the data requirements for successful differential or linear attacks on MARS and Rijndael exceed 2^{128}, the total amount of all possible plaintexts. The results in this paper show that the complexities of power attack on MARS are 2^{208}, 2^{168} and 2^{116} for 256, 192 and 128 bit key respectively, The complexities of power attack on Rijndael are 2^{131}, 2^{99} and 2^{67} for 256, 192 and 128 bit key respectively. Although the complexities are too big to implement, the approach presented in this paper reduces greatly the key size of MARS and Rijndael.

1 Key Schedule of MARS

For the sake of simplicity, we only describe the 128-bit key schedule of MARS, where the word size is 32 bits. Let $K=(K_0,K_1,K_2,K_3)$ be 128 bit master key，the 40 word sub-keys can be obtained as follows:

Step 1. Expand the master key into 15 words:

$T[0]=K_0$ $T[1]=K_1$ $T[2]=K_2$ $T[3]=K_3$ $T[4]=4$ $T[5]=$ … $T[14]=0$

Step 2. Repeat the following process four times, where each iteration outputs the next ten words of sub-keys:

(a) Do a linear transformation for $T[0],\ldots,T[14]$:

for $i=0,...,14$

$$T[i]=((T[i-7 \bmod 15]\oplus T[i-2 \bmod 15])<<<3)\oplus(4i+j)$$

where j is the iteration number(j=0 for the first iteration, 1 for the second, etc.)

(b) Repeat the following operation four times:

$$T[i]=(T[i]+S[\text{low 9bits of } T[i-1 \bmod 15]])<<<9$$

$$i=0,1,...,14$$

(c) Take ten words in $T[0],...,T[14]$ and reorder them as the next ten word sub-keys. This is done as follows:

$$K[10j+i]=T[4i \bmod 15] \quad i=0,1,...,9,$$

where j is the iteration number.

Step 3. Finally, we go over the sixteen words which are used in the cipher for multiplication, and modify them to have some properties. Because this step is independent of our analysis, it is omitted here.

2 Power Attack on the Key Schedule of MAR

Now we describe (b) of Step 2 in detail:

Let $T_0[0],\ldots,T_0[14]$ and $T_1[0],\ldots,T_1[14]$ be input and output of the first iteration respectively.

$$\begin{aligned}
T_1[0] &= (T_0[0] + S(\text{low9bits of } T_0[14])) <<< 9\\
T_1[1] &= (T_0[1] + S(\text{low9bits of } T_1[0])) <<< 9\\
T_1[2] &= (T_0[2] + S(\text{low9bits of } T_1[1])) <<< 9\\
T_1[3] &= (T_0[3] + S(\text{low9bits of } T_1[2])) <<< 9\\
T_1[4] &= (T_0[4] + S(\text{low9bits of } T_1[3])) <<< 9\\
T_1[5] &= (T_0[5] + S(\text{low9bits of } T_1[4])) <<< 9
\end{aligned} \tag{1}$$

Let $T_2[0],\ldots,T_2[14]$ be output of the second iteration.

$$\begin{aligned}
T_2[1] &= (T_1[1] + S(\text{low9bits of } T_2[0])) <<< 9\\
T_2[2] &= (T_1[2] + S(\text{low9bits of } T_2[1])) <<< 9\\
T_2[3] &= (T_1[3] + S(\text{low9bits of } T_2[2])) <<< 9\\
T_2[4] &= (T_1[4] + S(\text{low9bits of } T_2[3])) <<< 9\\
T_2[5] &= (T_1[5] + S(\text{low9bits of } T_2[4])) <<< 9
\end{aligned} \tag{2}$$

Let $T_3[0],\ldots,T_3[14]$ be output of the third iteration.

$$\begin{aligned}
T_3[2] &= (T_2[2] + S(\text{low9bits of } T_3[1])) <<< 9\\
T_3[3] &= (T_2[3] + S(\text{low9bits of } T_3[2])) <<< 9\\
T_3[4] &= (T_2[4] + S(\text{low9bits of } T_3[3])) <<< 9\\
T_3[5] &= (T_2[5] + S(\text{low9bits of } T_3[4])) <<< 9
\end{aligned} \tag{3}$$

Let $T_4[0],\ldots,T_4[14]$ be output of the fourth iteration.

$$\begin{aligned}
T_4[2] &= (T_3[2] + S(\text{low9bits of } T_4[1])) <<< 9\\
T_4[3] &= (T_3[3] + S(\text{low9bits of } T_4[2])) <<< 9\\
T_4[4] &= (T_3[4] + S(\text{low9bits of } T_4[3])) <<< 9\\
T_4[5] &= (T_3[5] + S(\text{low9bits of } T_4[4])) <<< 9
\end{aligned} \tag{4}$$

From (c) we get

$$K[0] = T_4[0], K[4] = T_4[1], K[8] = T_4[2], K[1] = T_4[4], K[5] = T_4[5]$$
$$K[9] = T_4[6], K[2] = T_4[8], K[6] = T_4[9], K[3] = T_4[12], K[7] = T_4[13]$$

It is noted that the attacker knows the Hamming weights of each byte of sub-keys in the power attack, so the attacker knows the Hamming weights of each byte of $T_4[1],T_4[2],T_4[4],T_4[5]$. The attack is as follows:

Step 1. Deduce $(T_3[2],T_3[3],T_3[4],T_3[5])$ by using equation (4) and the Hamming weights of each byte of $(T_4[1],T_4[2],T_4[4],T_4[5])$. The average number of all possible values of $(T_3[2],T_3[3],T_3[4],T_3[5])$ is 2^{98}.

The following is our brief proof for the above claim. Under the condition of knowing the Hamming weights of each byte of $(T_4[1],T_4[2],T_4[4],T_4[5])$, the average number of all possible $T_4[2]$ is 2^{20}. The average number of all possible values of the least 9-bits of $T_4[2]$ is 2^6. Hence by using the first equation in (4), we can deduce $T_3[2]$ which has 2^{26} possible values on average. Therefore, the average number of all possible $(T_3[2],T_3[3])$ is 2^{58}. For each $(T_3[2],T_3[3])$, we deduce $T_3[4]$ and $T_3[5]$. Under the condition of knowing the Hamming weights of each byte of $T_4[4],T_4[5]$, the average number of all possible $T_3[4]$ and $T_3[5]$ is 2^{20}. Finally we have proven that the average number of all possible values of $(T_3[2],T_3[3],T_3[4],T_3[5])$ is 2^{98}.

Step 2. For any $(T_3[2],T_3[3],T_3[4],T_3[5])$, we deduce $(T_2[2],T_2[3],T_2[4],T_2[5])$ by using equation (3). The average number of all possible values of $(T_2[2],T_2[3],T_2[4],T_2[5])$ is 2^{107}.

Because $S(\text{lowest9bits of } T_3[1])$ has 2^9 possibilities at most, the average number of all possible $T_2[2]$ is 2^9

for fixed ($T_3[2],T_3[3],T_3[4],T_3[5]$). Once $T_2[2]$ is fixed, $T_2[3],T_2[4]$ and $T_2[5]$ are also fixed respectively.

Step 3. For any $T_2[2],T_2[3],T_2[4],T_2[5]$, we deduce ($T_1[2],T_1[3],T_1[4],T_1[5]$) by using equation (2). The average number of ($T_1[2],T_1[3],T_1[4],T_1[5]$) is 2^{116}.

Step 4. For any ($T_1[2],T_1[3],T_1[4],T_1[5]$), we deduce ($T_0[2],T_0[3],T_0[4],T_0[5]$) by using equation (1). The average number of all possible ($T_0[2],T_0[3],T_0[4],T_0[5]$) is 2^{116}.

From (a) of step 2 in key schedule, $T_0[0],T_0[1]$ and $T_0[14]$ are constants, which are independent of the master key. So from the first and second equation in (1), we know $T_1[1]$ is constant. Furthermore, ($T_0[2],T_0[3],T_0[4],T_0[5]$) is determined only by ($T_1[2],T_1[3],T_1[4],T_1[5]$).

Step 5. Using （a） of step 2 in the key schedule, we get the master key $K=(K_0,K_1,K_2,K_3)$ by ($T_0[2],T_0[3],T_0[4],T_0[5]$).

The deduction procedure is the following:

$$T_0[2]=(K[0]<<<3)\oplus 8 \qquad T_0[3]=(K[1]<<<3)\oplus 12$$
$$T_0[4]=(K[2]<<<3)\oplus 16 \qquad T_0[5]=(K[3]<<<3)\oplus 20$$

Step 6. Test the deduced key.

The complexity of the above attack is 2^{116} on average. During the attack we only use a part of known information, the complexity can be further reduced if using more information. Similarly, the above attack on MARS needs 2^{208} and 2^{168} trials for 256 and 192 bit key respectively.

3 Key Schedule of Rijndael

For the sake of simplicity, we only give the 128-bit key schedule of Rijndael. Rijndael with 128-bit key needs 11 128-bit sub-keys, each of which consists of four words $W[i]\,(0\le i\le 43)$, where $(W[0],W[1],W[2],W[3])$ is the master key. The other $W[i]$ are obtained as follows:

$$W[4]=W[0]\oplus S(Rot1(W[3]))\oplus Rcon[1]$$
$$W[5]=W[1]\oplus W[4]$$
$$W[6]=W[2]\oplus W[5]$$
$$W[7]=W[3]\oplus W[6]$$
$$W[8]=W[4]\oplus S(Rot1(W[7]))\oplus Ron[2]$$
$$\vdots$$

4 Power Attack on the Key Schedule of Rijndael

Before launching the attack, let's discuss the following problem first. Let $Y,Z\in F_2^8$, given $Z, y=W_H(Y)$ and integer $x\,(0\le x\le 8)$. How many Y satisfy $W_H(Y\oplus Z)=x$ on average? The number of Y is not larger than C_8^y apparently. Let $z=W_H(Z)$, if $y+z-x<0$, then the number of Y satisfying $W_H(Y\oplus Z)=x$ is zero. If $y+z-x=0$, then the number of Y satisfying $W_H(Y\oplus Z)=x$ is C_{8-z}^{y}. If $y+z-x$ is odd，then the number of Y satisfying $W_H(Y\oplus Z)=x$ is zero. If $y+z-x=2i$， then the number of Y satisfying $W_H(Y\oplus Z)=x$ is $C_z^iC_{8-z}^{y-i}$. Hence, we have the following result:

Let $Y,Z\in F_2^8$, given $Z, y=W_H(Y)$ and integer $x(0\le x\le 8)$, then the average number of Y satisfying $W_H(Y\oplus Z)=x$ is about

$$\sum_A C_z^iC_{8-z}^{y-i}\Big/|A|\approx 15,$$

where A is the set of (z,y,i) satisfying $W_H(Y\oplus Z)=x$.

For 128-bit key schedule of Rijndael，suppose the Hamming weights of each byte of $W[i]\,(0\le i\le 43)$ are

given, which are marked $x_i^1, x_i^2, x_i^3, x_i^4$ respectively. For any fixed $W[3]$ whose number is $C_8^{x_3^1} \times C_8^{x_3^2} \times C_8^{x_3^3} \times C_8^{x_3^4}$, we perform the following steps.

Step 1. Search $W[0]$ satisfying $W[4] = W[0] \oplus S(Rot1(W[3])) \oplus Rcon[1]$, and compute relevant $W[4]$.

There are about 15^4 $W[0]$ under the condition of knowing $(x_0^1, x_0^2, x_0^3, x_0^4)$ and $(x_4^1, x_4^2, x_4^3, x_4^4)$.

Step 2. For every $W[4]$ obtained in Step 1, search $W[1]$ satisfying $W[5] = W[1] \oplus W[4]$ and compute relevant $W[5]$.

There are about 15^4 $W[1]$.

Step 3. For every $W[5]$ obtained in Step 2, search $W[2]$ satisfying $W[6] = W[2] \oplus W[5]$ and compute relevant $W[6]$. There are about 15^4 $W[2]$.

Step 4. Compute $W[7] = W[3] \oplus W[6]$ and test the Hamming weights of each byte of $W[7]$. If the above two values are not equal, then discard the relevant $W[6]$; if they are equal, then test other equations.

Suppose for a wrong key, the probability with all equations holding is very small, the possible number of trials of the above attack is about

$$15^{12} \times (C_8^{x_3^1} \times C_8^{x_3^2} \times C_8^{x_3^3} \times C_8^{x_3^4}) \approx 2^{67}.$$

Similarly, the above attack on Rijndael needs 2^{131} and 2^{99} trials for 256 and 192 bit key respectively.

5 Conclusions

The five finalist AES candidates have been shown that they are resistant to differential, linear and related-key cryptanalysis. In this paper we have proven that if they are used in smart cards, the power attack on MARS needs 2^{208}, 2^{168} and 2^{116} trials for 256, 192 and 128 bits key respectively. We also show that the power attack on Rijndael needs 2^{131}, 2^{99} and 2^{67} trials for 256, 192 and 128 bits key respectively. Although the number of trials is too big to implement, the approach presented in this paper reduces greatly the key size of MARS and Rijndael.

References:

[1] Kocher, P., Jaffe, J., Jun, B. Differential power analysis. 1998. http://www.cryptography.com/dpa.

[2] Biham, E., Shamir, A. Power analysis of the key scheduling of the AES candidates. 1999. http://www.cs.technion.ac.il/~biham.

[3] MARS, Rijndael. 1999. http://www.nist.gov/aes.

MARS 和 Rijndael 的能量攻击

吴文玲, 贺也平, 冯登国, 卿斯汉

(中国科学院 信息安全技术工程研究中心,北京 100080);

(中国科学院 软件研究所 信息安全国家重点实验室,北京 100080)

摘要: 使用能量攻击对 MARS 和 Rijndael 进行了深入分析.结果表明:对于 256,192 和 128 比特密钥的 MARS 算法,能量攻击的复杂度平均分别为 2^{208},2^{168} 和 2^{116}.对于 256,192 和 128 比特密钥的 Rijndael 算法,能量攻击的复杂度平均分别为 2^{131},2^{99} 和 2^{67}.虽然攻击的复杂度实际上无法达到,但是此攻击方法大大降低了 MARS 和 Rijndae 的密钥规模.

关键词: 密钥;分组密码;能量攻击;密钥编排;复杂度

中图法分类号: TP309 文献标识码: A