Krzysztof BUCHOLC
POZNAN UNIVERSITY OF TECHNOLOGY

Fault Attack Technique Against Software Implementation of a Block Cipher

Dr inż. Krzysztof BUCHOLC

Is a senior lecturer in Poznań University of Technology. He received a Ph.D. from Poznań University of Technology in 1989. His main research areas are: computer architecture, embedded systems, reliability and diagnosis of computer hardware. He is the author or coauthor of more than 50 published papers, 2 patents and 1 textbook.

e-mail: Krzysztof.Bucholc@.put.poznan.pl

Abstract

In this paper a fault attack technique against software implementation of a block cipher is described. Implementation for 8-bit processor (similar to those used in embedded applications e.g. smart cards) was examined. There is shown that 2 specific pairs plaintext – faulty ciphertext suffice to break the cipher. The method was illustrated in PP-1 cipher, but it is applicable to any software implementations of the Substitution-Permutation Networks such that the main key, or all the round keys. It can be deduced from round key for the first round.

Keywords: block cipher, software implementation, fault attack.

Techniki ataku polegające na wprowadzaniu defektów do programowej implementacji szyfru blokowego

Streszczenie

W artykule przedstawiono atak na szyfr blokowy przy użyciu celowo wprowadzanych defektów. Taka metoda ataku (ang. fault attack) należy do najskuteczniejszych metod łamania szyfrów. Najczęściej wykorzystywana jest do atakowania implementacji sprzętowych szyfru. W prezentowanej pracy za przedmiot badań wybrano implementację programową, przy czym procesor wraz z programem w pamięci traktowany był jak układ sprzętowy. Wybrano 8-bitowy procesor (podobny do wykorzystywanego w kartach procesorowych (ang. smart cards)). Eksperymenty przeprowadzono posługując się specjalnie opracowaną maszyną wirtualną rozszerzoną o moduł do wstawiania uszkodzeń. Wykorzystano następujące modele uszkodzeń: sklejenie z zerem, sklejenie z jedynką i odwrócenie wartości bitu. Przedmiotem eksperymentów był szyfr blokowy o nazwie PP-1. W pracy krótko przedstawiono zasadę działania algorytmu szyfrującego. Następnie przeanalizowano wpływ defektów na wyniki szyfrowania. Rozważono defekty pojedyncze i wielokrotne. Szczególną uwagę zwrócono na pojawianie się wyników odpowiadających wersji szyfru zredukowanej do jednej rundy. (Normalnie szyfr wykorzystuje 11 rund). Jednorundowe wersje szyfrów blokowych są bardzo łatwe do złamania. Badania wykazały, że prawdopodobieństwo wystąpienia wyniku odpowiadającego jednorundowej wersji szyfru jest wystarczająco duże, aby można było zastosować tę technikę ataku w praktyce. Stwierdzono także, że pewne obszary pamięci programu są bardziej wrażliwe na defekty niż inne. Koncentrując się na obszarach wrażliwych można znacząco zwiększyć prawdopodobieństwo sukcesu.

Słowa kluczowe: szyfr blokowy, implementacja programowa, atak z wykorzystaniem defektów.

1. Introduction

Errors in an encryption circuit not only disturb communication but also cause hazard for the cipher safety. Encryption circuits are often objects of deliberate error injection, while it is rather rare the case in other sort of digital circuits. Block ciphers can be implemented both in hardware and in software.

The influence of errors on hardware implementation of block ciphers was an object of substantial research effort in last few years [1, 2, 3].

Errors inserted on purpose cause hazard for the cipher security. For example error can reduce the number of rounds or affects the round key schedule procedure in such a way that all the round keys will be the same. In such case the cipher can be easily broken [5, 6].

For example the only successful attacks against AES [7] implementations have been side channel attacks, exploiting information from a physical implementation of the cipher, such as timing information, power consumption, electromagnetic leaks etc. or fault attacks. The algorithm itself is considered as secure.

In our research we consider the software implementation of the PP-1 cipher treated as a piece of hardware. It means that the processor and the program are treated like the encryption circuit. The main objective was to establish to what extend the software implementation of the PP-1 is vulnerable to fault attacks. It is intended as an introductory step to implement tamper resistant implementation of the PP-1.

There are many works describing how to insert faults into cryptographic circuit [8]. In this paper we simply assume that this is possible and focus on the results.

To break the cipher means to find the secret key. In this paper this term is used for finding the secret key or finding all the round keys.

The paper is organized as follows: Section 2 contains short description of the PP-1 cipher. In section 3 we describe experiment with fault insertion to software implementation of the PP-1 block cipher. Section 4 contains brief analysis of the influence of errors on the PP-1 safety. Section 5 contains some concluding remarks.

2. Overview of the PP-1 structure

The PP-1 [4] is an n-bit (n = 64, 128, 192, ...) scalable block cipher. The key length is n or $2n$. The PP-1 is an involutional Substitution-Permutation Network. It uses one S-box, which is an involution and a bit permutation, which also is an involution. As a result the same algorithm can be used both for encryption and decryption. The PP-1 structure is shown in Fig.1 and Fig.2.

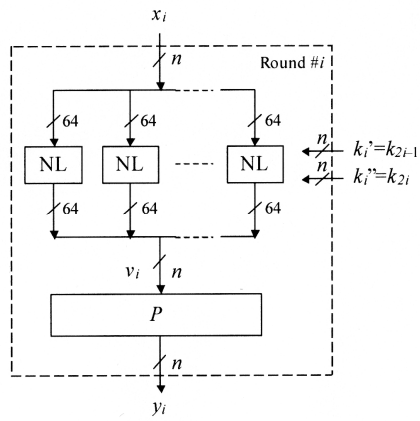

Fig. 1. The PP-1 cipher
Rys. 1. Szyfr PP-1

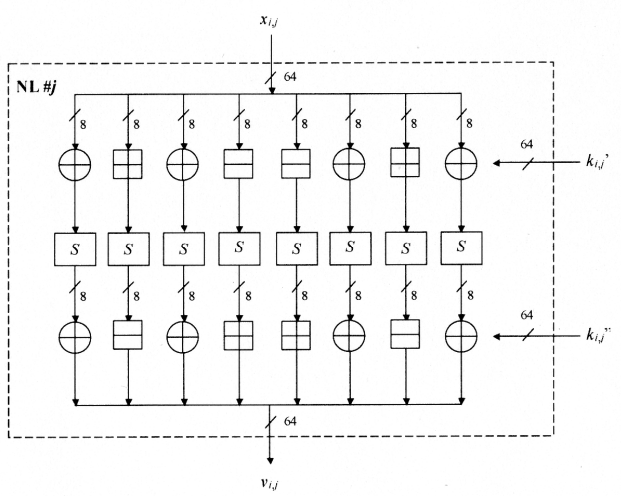

Fig. 2. The structure of NL element
Rys. 2. Struktura elementu NL

Symbols ⊕, +,– stand for xor, addition, and subtraction, on 8-bit arguments. The S element is an 8×8 S-box. The *P* block is an *n*-bit permutation. The number of rounds depends on *n*. There are 11, 22, 32 and 43 rounds for *n*=64, 128, 192, 256 respectively. In each round two *n*-bit round keys are used.

Results presented in this paper are obtained for the 64-bit data block 128-bit key version of PP-1.

3. Analysis of faults influence on encryption process

3.1. Investigated implementation

For simulation of the faulty encryption device an assembler implementation of PP-1 for 8-bit processor (8080) installed on virtual machine was used. The virtual machine was specially written for this purpose and augmented with program for fault insertion, and results collection.

The 8080 architecture was chosen as a model of 8-bit processor. It is close to architectures used in 8-bit embedded systems, but it is simpler. Only main data processing path of the cipher was implemented. Round keys were computed separately. The program encrypts one 64-bit data block using 128-bit key. Its size is 814 bytes.

3.2. The fault model

Fault can be either a hardware defect or a software mistake. Fault may or may not cause an error. It means that an error is a manifestation of a fault. In our experiment we focus on faults in memory containing the encryption/decryption program.

Three types of faults were considered: stack bit at zero (s-a-0), stack bit at one (s-a-1), and invert bit (bit-flop). Single and multiple faults were investigated. For single and double faults all possibilities were checked. For 3 faults and more (up to 30), faults were chosen in random. We focused on the case where one or several faults were inserted. In real systems it is often difficult to insert precisely one fault. One to several is more probable.

3.3. Results of the experiments

Each experiment consists of four stages: loading the program into virtual machine, fault insertion, performing of encryption and recording results. Five types of results were considered: proper, improper corresponding to 1 round version of PP-1, other improper, no result, and infinite loop. The most interesting is improper corresponding to 1 round version result, because it is most useful for breaking the cipher. Details are described in the next section.

For each fault model and for each number of faults the experiment was repeated 10 000 times. Summarized results for single fault and multiple faults are shown in Table 1 and Table 2.

Tab. 1. Single faults
Tab. 1. Pojedyncze defekty

Fault model	Proper	No result	Round 1	Different results	Infinite loop
s-a-0	7791	305	48	727	114
s-a-1	6528	588	125	1137	216
Bit flop	4519	893	173	1759	330

Tab. 2. Multiple faults (bit flop model)
Tab. 2. Defekty wielokrotne (model: odwrócenie bitu)

Number of faults	Proper	No result	Round 1	Different faulty results	Infinite loop
1	4519	893	173	1759	330
2	2016	1738	212	3111	591
3	891	2430	203	4012	799
4	447	3050	184	4333	967
5	183	3604	157	4293	1107
6	93	4152	126	4069	1205
7	46	4599	99	3819	1336
8	22	4927	67	3590	1374
9	6	5259	48	3342	1404

As we can se in Tables 1 and 2, for single fault and for 2 to 8 faults, significant percentage of the results correspond to those for 1 round version of PP-1 (with the same key). In section 4 we will exploit this for attack on the cipher.

Some parts of the cipher implementation may be more vulnerable to faults than other. To check this, the whole memory space occupied by the application was divided into 10 intervals. For each interval round 1 results to different faulty results ratio was calculated. Results for single fault bit flop model are shown in Fig. 3.

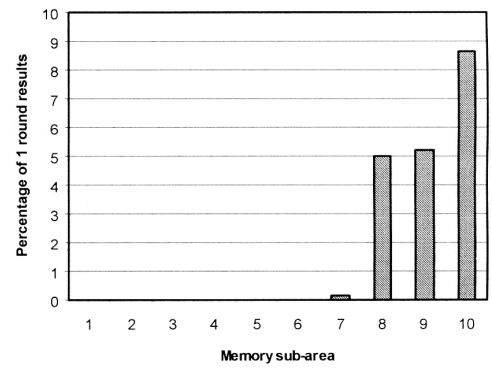

Fig. 3. Distribution of round 1 results to the number of all erroneous results ratio
Rys. 3. Udział wyników dla pierwszej rundy w zależności od obszaru pamięci

As was expected, some areas are more vulnerable. It would be profitable to locate them when breaking real circuit. But the average ratio – about 2% is also very interesting.

4. The attack

4.1. General concept

Let us assume that there is an encryption device with a processor. We can use it for the encryption any data but the key is secret. We have access to the device and we can insert randomly faults into its memory. Our goal is to find the secret key.

Let us assume that we can perform a series of experiments with our encryption device inserting faults during the encryption process. We get a series of results. Some of these results are proper, and some are erroneous. Among erroneous results some are the same as results for fault-free one-round version of the cipher with the same key.

It is well known that one round block cipher can be easily broken (e.g. only 2 pairs plaintext-ciphertext are required to brake one round AES [5]).

We do not know which result corresponds to those produced by the one round version of the cipher. But we can check every obtained result. In this way we get round key (in PP-1 two round keys) for round 1. Knowing the first round key (keys), we can find the main key or find all the next rounds keys we will break the whole cipher. For PP-1 this is quite simple as the first round keys (denoted as #0 round in algorithm description [4]) consist of two halves of the main key.

Two interesting problems arise:

- What is the complexity of restoring the key?
- How many experiments with fault insertion are required to obtain "good" (useful for cipher breaking) faulty ciphertexts with given probability?

We will address these problems below.

4.2. Attack with one faulty ciphertext

In this section we will focus on the PP-1 cipher shown in Fig. 1 and Fig. 2. Only one NL element is used. Let us assume that we obtained one faulty ciphertext, which corresponds to round 1. Let us consider the right-most 8-bit section in Fig.2.

We denote the 8-bit right-most parts of k_0' and k_0'' by $k1$, $k2$, the 8-bit right-most part of the plaintext by m, and the 8-bit right-most part of ciphertext by c.

In this case

$$c = S(m \oplus k1) \oplus k2 \tag{1}$$

$$k2 = S(m \oplus k1) \oplus c \tag{2}$$

Similarly for 8-bit section with operations plus and minus (Fig.2) we have

$$c = S(m + k1) - k2 \tag{3}$$

$$k2 = S(m + k1) - c \tag{4}$$

There are 2^8 (256) pairs of $k1$, $k2$ that satisfy (2). The same is true for (4). It means that 128-bit key can be broken using 2^{64} pairs plaintext ciphertext instead of 2^{128} required in brute-force method.

4.3. Attack with two faulty ciphertexts

Let us assume that we have two different pairs plaintext ciphertext *(m*1,*c*1*) (m*2,*c*2*)* for round 1. For *xor xor* section (Fig. 2) we have the system of equations:

$$\begin{aligned} c1 &= S(m1 \oplus k1) \oplus k2 \\ c2 &= S(m2 \oplus k1) \oplus k2 \end{aligned} \tag{5}$$

Similarly, for *plus minus* section we have:

$$\begin{aligned} c1 &= S(m1 + k1) - k2 \\ c2 &= S(m2 + k1) - k2 \end{aligned} \tag{6}$$

We know $m1$, $c1$, $m2$, $c2$ and we want to find $k1$, $k2$. The problem is: how many solutions there exist for systems of equations (5), and (6)?

The answer is not obvious because of the nonlinear element S. Solutions were found by checking all possible $m1$, $m2$, $c1$, $c2$ such that $m1 \neq m2$ and $c1 \neq c2$. Results are shown in Tables 3 and 4.

Tab. 3. Solutions distribution for *xor xor*
Tab. 3. Rozkład rozwiązań dla *xor xor*

Solutions count	Found times	%
0	1076920320	50,54
1	0	0
2	1038090240	48,72
3	0	0
4	15728640	0,74
Total	2130739200	100

Tab. 4. Solutions distribution for *plus minus*
Tab. 4. Rozkład rozwiązań dla *plus minus*

Solutions count	Found times	%
0	779812864	36,5982
1	784203776	36,8043
2	392986624	18,4437
3	135462912	6,3576
4	30998528	1,4548
5	5505024	0,2584
6	1376256	0,0646
7	393216	0,0185
Total	2130739200	100

As we can see in Table 3, if a solution for *xor xor* exists, there are 2 unequally distributed possibilities: in about 98.5% we get two pairs of $k1$, $k2$ whereas in about 1.5% cases we have 4 different pairs of $k1$, $k2$. For *plus minus* section there exist from 1 to 7 solutions (Table 4).

We will use these results to estimate the number of pairs plaintext ciphertext required to break the cipher. As we can see in Fig.2 there are four *xor xor* sections and for *plus minus* sections (the order of operations + – is of no importance). The smallest possible numbers of solutions are 2 for *xor xor* and 1 for *plus minus*. The total number of pairs plaintext ciphertext required to break the cipher is:

$$Lmin = 2^4 \times 1^4 = 16 \tag{7}$$

The biggest possible numbers of solutions are 4 for *xor xor* and 7 for *plus minus*. In this case the total number of pairs plaintext ciphertext required to break the cipher is:

$$Lmax = 4^4 \times 7^4 = 614656 \tag{8}$$

Weighted average is:

$$Laverage = 466.00215 \tag{9}$$

Generally for PP-1 with n-bit data block and $2n$-bit key we have

$$Lmax\_n = 4^{n/16} \times 7^{n/16} \tag{10}$$

Results (7), (8), (9) and (10) show that attack with two faulty ciphertexts is much more practical than the attack with one faulty ciphertext. The main problem is to find two different pairs plaintext faulty ciphertext such that $m1 \neq m2$ and $c1 \neq c2$ are proper results for round 1. We will address this problem in the next section.

4.4. Obtaining faulty ciphertexts

In section 3 we presented results of the experiments with faults insertion. Now we use these results to estimate the number of experiments needed to obtain the required two faulty ciphertexts.

We need to perform two series of experiments with fault insertion – one series for each plaintext. The probabilities of finding the required faulty ciphertext are shown in Table 5. We considered 2 cases:
- faults equally distributed,
- faults focused on sensitive area (see Fig. 3).

As it can be seen in Table 5, to find faulty ciphertext with probability 90% requires 145 experiments, when faults are equally distributed, whereas 26 experiments suffice when faults are focused on sensitive area. Two series of experiments are required to obtain 2 faulty ciphertexts.

Tab. 5. Probability of finding the faulty ciphertext useful for breaking the cipher
Tab. 5. Prawdopodobieństwo znalezienia błędnego kryptogramu przydatnego do złamania szyfru

Number of experiments	Probability of finding round 1 ciphertext	
	Equally distributed	Sensitive area
10	0,160135	0,596201
20	0,294627	0,836946
26	0,364750	0,905370
50	0,582124	0,989264
60	0,649040	0,995665
70	0,705241	0,998250
80	0,752442	0,999293
90	0,792085	0,999715
100	0,825379	0,999885
145	0,901828	0,999994
150	0,927030	0,999999
200	0,969508	0,99999999

5. Conclusion

The basic idea of the proposed attack is to get erroneous result of the encryption by inserting fault (faults). We considered two cases. Attack with one faulty ciphertext and attack with two faulty ciphertext. We assume that erroneous result is those produced by 1 round and verify this hypothesis.

The attack with two faulty ciphertexts is more practical, despite the fact that more experiments with fault insertion are required to obtain 2 faulty ciphertext, because it requires less processing to verify found solutions. For example expected time of all computations needed to find the key with probability 0.99 is less than half second on notebook with 2 GHz Celeron M processor.

We focused on PP-1, but this approach is applicable to any software implementations of Substitution-Permutation Networks such that the main key or all the round keys can be deduced from round key for the first round.

This research also proved that considered implementation of PP-1 is very vulnerable to fault attack. Further work is required to obtain more tamper resistant implementation of the PP-1.

6. References

[1] Bertoni G., Bregeveglieri L., Koren I., Maistri P., An Operation-Centered Approach to Fault Detection in Symmetric Cryptography Ciphers, IEEE Trans. On Computers Vol. 56 No 5, May 2007, pp. 635-649.
[2] Bertoni G., Bregeveglieri L., Koren I., Maistri P., Piuri V., Error Analysis and Detection Procedures for a Hardware Implementation of the Advanced Encryption Standard, IEEE Trans. On Computers Vol. 52 No 4, April 2003, pp.492-505.
[3] Bucholc K., Idzikowska E , Multiple Error Detection in Substitution Blocks for Block Ciphers. Advances in Information Processing and Protection, ed. J.Pejas and K.Saeed, Springer US, 2007, pp. 181-190.
[4] Chmiel K., Grocholewska-Czuryło A., P. Socha, Stokłosa J., Scalable cipher for limited resources. Polish Journal of Environmental Studies, Vol. 17/4C/2008 pp. 371-377.
[5] Joye M., Mannet P., Rgaud J.-B., Strengthening hardware AES implementations against fault attacks, IET Inf. Secur., vol. 1, 2007, pp.106-110.
[6] National Inst. Of Standards and Technology, Federal Information Processing Standard 197, The advanced Encryption Standard (AES), Nov. 2001,http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.
[7] Piret G. and Quisquater J.-J., A differential fault attack technique against SPN structures, with application to the AES and Khazad, Proc. of CHES 2003, pp. 77–88.
[8] Skorobogatov S. and Anderson R.. Optical fault induction attacks. Cryptographic Hardware and Embedded Systems Workshop (CHES-2002), pages 2-12, 2002. Lecture Notes in Computer Science No. 2523.

Artykuł recenzowany