Multiple Error Detection in Block Ciphers

K. Bucholc, E.Idzikowska

Abstract **-We present two methods of multiple error detection in substitution blocks in block ciphers. Permanent errors and transient errors are considered. Proposed methods require relatively low circuit overhead. These methods ere demonstrated on an 8-input, 8-output substitution block.**

1. INTRODUCTION

Errors in an encryption circuit not only disturbs communication but also causes hazard for the cipher safety. Encryption circuits are often objects of deliberate error injection which rarely is the case in other sort of digital circuits. Therefore error detection in cryptographic circuits is of great importance.

A substitution box (S-box) is a basic component of block ciphers. Substitution boxes are used to obscure the relationship between the plaintext and the ciphertext. There are m inputs and n outputs in such a box.

Error detection in cryptographic hardware is the subject of many research efforts [1,2].

There are many techniques which can be used for error detection in digital circuits. All are based on adding redundancy to the circuit. The required extra hardware varies greatly. It can be relatively small, adding a few percent to the circuit area, but it may also lead to doubling of the hardware.

Parity code based solutions require relatively small increase of the hardware complexity. Examples of solutions can be found in [1]. The drawback of the parity code methods is that they can only detect single errors and any odd number of errors.

In this paper we will focus on error detection in s-boxes. We will show that for permanent errors parity based method can detect error with high probability. For transient errors bigger redundancy is required. But we will show that limiting possible errors, to those most probable when induced on purpose, leads to a simplified method of error detection. The paper is organized as follows: In section 2 we describe basic properties of s-boxes – important for error detection.

K. Bucholc, E.Idzikowska, *Department of Electric Engineering, Poznan University of Technology* pl. M.Sklodowskiej-Curie 5, 60-965 Poznan, Poland # Krzysztof.Bucholc@put.poznan.pl, Ewa.Idzikowska@put.poznan.pl

Section 3 presents a method of permanent error detection, while section 4 describes a method of transient error detection in an S-box.

2. SUBSTITUTION BLOCKS PROPERTIES

Let us consider a substitution block shown in Fig.1. There are m inputs and n outputs. The number of inputs and outputs may be equal – for example in AES cipher m=n=8, or m>n (e.g. DES 6x4) or m<n (e.g. MARS 9x32). Each m-bit input vector is substituted with n-bit output vector. S-boxes are carefully designed to make differential cryptanalysis and linear cryptanalysis as difficult as possible.

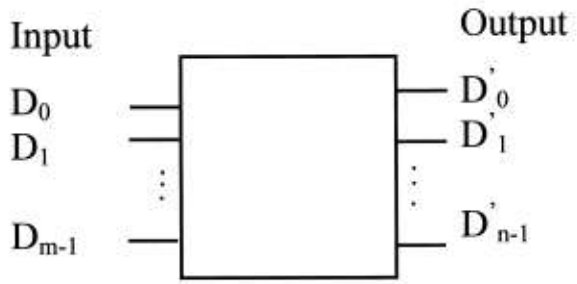

Fig.1. The m x n S-box

Let us consider a n x n S-box. For testing purposes it can be treated as a black-box with n-inputs and n outputs. Such a s-box can be implemented as a ROM memory.

Set of Boolean functions used to compose the substitution block should poses such properties as: high nonlinearity, high algebraic order and resilience to provide protection against cryptanalysis [5]. In result the transformation in the s-box does not preserve the parity of data. To illustrate this property we will check how the parity is preserved in the AES [3] cipher s-box. The substitution table of the s-box is shown in Fig. 2.

For each input value and its corresponding output value the parity was computed. Results are shown in Table 1.

It can be seen that the s-box transformation does not preserve parity. Furthermore if the input is even the probability that the output is also even is nearly the same that the output is odd.

```
     0    1    2    3    4    5    6    7
00 0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x68, 0x6F, 0xC5,
   0x30, 0x01, 0x67, 0x2B, 0xFE, 0xD7, 0xAB, 0x76,
10 0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0,
   0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0,
20 0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC,
   0x34, 0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31, 0x15,
30 0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A,
   0x07, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75,
40 0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0,
   0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84,
50 0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B,
   0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58, 0xCF,
60 0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85,
   0x45, 0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F, 0xA8,
70 0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5,
   0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2,
80 0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17,
   0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73,
90 0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88,
   0x46, 0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB,
A0 0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C,
   0xC2, 0xD3, 0xAC, 0x62, 0x91, 0x95, 0xE4, 0x79,
B0 0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9,
   0x6C, 0x56, 0xF4, 0xEA, 0x65, 0x7A, 0xAE, 0x08,
C0 0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6,
   0xE8, 0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B, 0x8A,
D0 0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E,
   0x61, 0x35, 0x57, 0xB9, 0x86, 0xC1, 0x1D, 0x9E,
E0 0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94,
   0x98, 0x1E, 0x87, 0xE9, 0xCE, 0x55, 0x28, 0xDF,
F0 0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68,
   0x41, 0x99, 0x2D, 0x0F, 0xB0, 0x54, 0xBB, 0x16
```

Fig. 2. The AES S-box

Let us denote the parity of D by P(D) . S(D) denotes substitution – its value is equal to the S-box output for D placed on the input.
Let D'=S(D). For well designed S-box there is:

$$probability(P(D')eqP(D)) = probability(P(D')neP(D)) = 0.5 \quad (1)$$

or

$$probability(P(D')eqP(D)) \approx probability(P(D')neP(D)) \approx 0.5 \quad (2)$$

Table 1. Parity distribution in AES S-box

Input		Output	
		Even	Odd
Even	128	65	63
Odd	128	63	65

For example the values of probability(P(D') eq P(D)) and probability(P(D') ne P(D)) are 0.5078125 and 0.4921875 respectively for even input and 0.4921875 and 0.5078125 for odd inputs [4].

3. PERMANENT ERROR DETECTION

A. Data processing in the block cipher encryption circuit

In typical encryption circuit most operations are repeated many times to obtain the required level of security. In each round an s-box is used. The number of rounds in selected ciphers is shown in Table 2. Block size is modern block ciphers is 16 bytes or more. It means that for 10-round AES with 128-bit block size encryption of one block of data requires 160 substitutions in the s-box.

Table 2. Rounds in selected block ciphers

Cipher	Number of rounds
AES (Rijndael)	10 or 12 or 14
Twofish	16
Serpent	32

B. Error detection

Let us suppose that there is a method of error detection in s-box, such that while processing one vector of bits (e.g. 8 bits in AES) detects error with probability p, and that consecutive vectors are independent. In such case, using n vectors we detect the error with probability

$$Pd = 1-(1-p)^n \quad (3)$$

where Pd denotes the probability of error detection.

For practical purposes it should suffice to detect an error, with high enough probability, before sending out the faulty block.

For p=0.5 the numbers of vectors required to detect error with probability 0.999, 0.9999 and 0.99999 are 10, 14 and 17 respectively.

The method is based on the fact that consecutive vectors are independent. In reality consecutive vectors are obtained by processing previous vectors. But because of the nature of the cipher circuits even correlated data quickly become statistically independent. It means that they pass most of the test used for random numbers testing.

```
PT=000102030405060708090A0B0C0D0E0F
KEY=000102030405060708090A0B0C0D0E0F

Round
1   00000000000000000000000000000000
2   B5C9179EB1CC1199B9C51B92B5C8159D
3   2B65F6374C427C5B2FE3A9256896755B
4   D1015FCBB4EF65679688462076B9D6AD
5   8E17064A2A35A183729FE59FF3A591F1
6   D7557DD55999DB3259E2183D558DCDD2
7   73A96A5D7799A5F3111D2B63684B1F7F
8   1B6B853069EEFC749AFEFD7B57A04CD1
9   107EEADFB6F77933B5457A6F08F046B2
10  8EC166481A677AA96A14FF6ECE88C010
```

Fig.3. AES S-box input data in rounds 1-10 for plain text the same as the key

There may be problem however with initial round. In AES

cipher there is AddRoundKey operation in initial round which sums modulo 2 the plaintext and the key. If the data block is the same as the key, we get 16 identical bytes - all zeros. (See Fig. 3).

The proposed method can still be used. Only longer sequence of data is required – 25, 29 and 31 instead of 10, 14 and 17. It must be emphasized that in practice for well chosen key such situation happens very rarely. It means that in most cases the required level of data detection is achieved for 10, 14 and 17 operations of data substitution using S-box.

The position of the error detection circuit is shown in Fig. 4. The contents of the s-box can be protected using error detection codes. But errors on the input and the output remain undetected. Therefore we focus on errors on inputs and outputs of the s-box. Places where the faults may appear are marked by circles in Fig. 4.

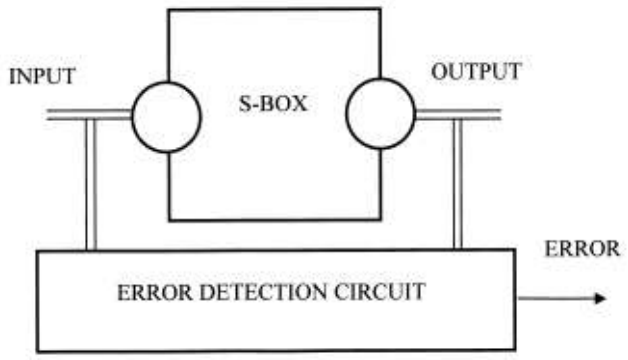

Fig. 4. The error detection circuit

The error detection is based on the parity code usage. The parity for each s-box input and for its corresponding output is computed. Then the cross parity is computed as modulo 2 sum of parities of input and output (4).

$$CP = P(INPUT) \oplus P(OTPUT) \quad (4)$$

Where CP and P – denote cross parity and parity respectively
Proper values of CP are stored in the error detection circuit.

For each input and output the value of CP is derived and compared with the stored pattern. If error is detected the ERROR signal is activated. Both single and multiple errors, including even number errors, are detected with probability close to ½.

Let us consider the AES s-box. To check the multiple error detection, even number of stack-at –zero and stack-at-one errors were simulated. Then random input data was delivered to the INPUT. The number of vectors needed to detect error was counted and recorded. This procedure was repeated for all possible input vectors. Results are shown in Fig. 5. As can be seen it takes one vector to detect nearly half of the faults. Results in Fig.5. tally with those computed from (3).

The main advantage of the method is that it adds small overhead. For 8x8 s-box 256 bits of stored pattern and two 8-input parity generators are required.

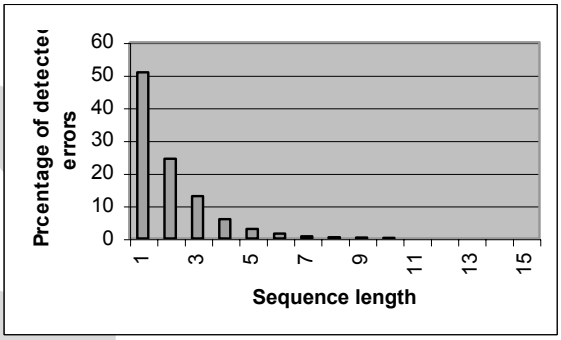

Fig. 5. Percentage of faults detected with 1, 2, 3… vectors

4. TRANSIENT ERROR DETECTION

Transient error detection requires another approach as there is not possible to use cumulative detection power of the sequence of vectors. The content of the memory used in s-box can be protected using error detecting or error correcting code. The overhead depends on the expected level of protection and does not exceed 100 per cent. If errors on input and output, as shown in Fig. 4, are also taken in to account bigger overhead is required. Adding one parity bit leads to overhead 100 per cent [1]. Every next extra bit doubles the overhead.

Table 3. The overhead added by error detection method in n x n S-box

Method	Overhead	
	Circuit	Memory
Parity	1 n-input parity checker	2^n
Cross-Parity	2 n-input parity checkers	2^n
Sum of 1's	1 n-input adder of 1's	$\log_2 n 2^n$
Cumulated Sum of 1's	2 n-input adders of 1's	$(\log_2 n+1)2^n$

We analyzed four solutions: parity bit, cross-parity, sum of 1's and cumulated sum of 1's for s-box input and output. Two

faults models were considered: general error model and unidirectional error model. The former does not put any restrictions – the data vector can be transformed into any vector. The latter model assumes that all changes in the data vector due to the error are of the same direction i.e. from 0 to 1 or from 1 to 0. This kind of errors are more probable when errors are inserted on purpose.

The overhead required by considered methods is shown in Table 3.

Results are shown in figures 6-9 and summarized in Table 4.

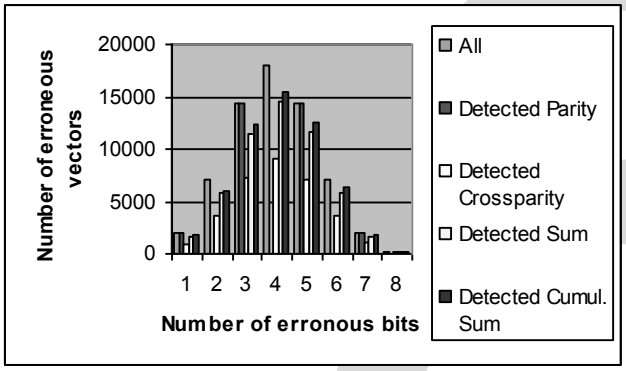

Fig. 6. Error detection. Erroneous input. All errors

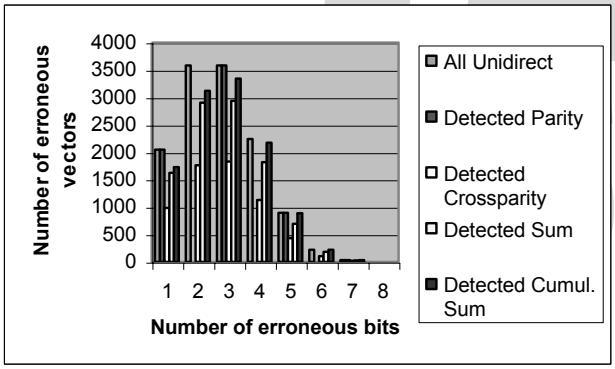

Fig. 7. Error detection. Erroneous input. Unidirectional errors

Table 4. Percentage of detected errors

	All Errors		Unidirectional Errors	
	Input	Output	Input	Output
Parity	50.20	50.20	52.02	52.02
Cross-Parity	50.20	50.20	49.72	52.02
Sum of 1's	80.68	80.68	80.86	100
Cumulated Sum of 1's	86.24	80.68	91.36	100

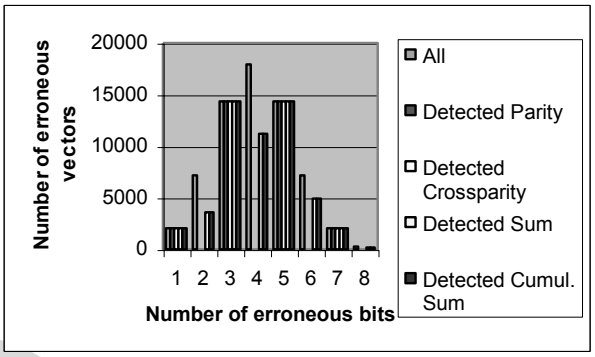

Fig. 8. Error detection. Erroneous output. All errors

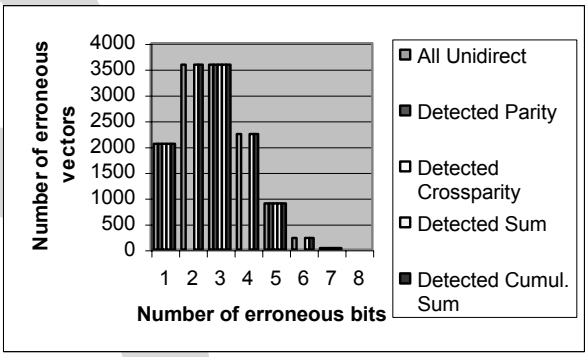

Fig. 9. Error detection. Erroneous output. Unidirectional errors

As can be seen in table 4 method based on counting 1's in both the input vector and the output vector detect all multiple unidirectional errors on outputs and 91.36% of errors placed on inputs.

5. CONCLUDING REMARKS

Detecting permanent errors requires smaller hardware overhead. Two parity checkers and 2n bits of stored pattern suffice to detect error with accuracy high enough for practical application. For example for AES 8 x 8 s-box the probability of input error detection, during encryption of one 128-bit block, can be as high as 1-2-160 . Detecting transient errors requires more redundancy. In this paper we only considered methods which require relatively small overhead not exceeding (log2n+1)2n . We considered multiple errors – up to n erroneous bits for n-bit vector. In most digital circuits probability of such errors is small. Usually much smaller then errors with single erroneous bit. But for cryptographic circuits multiple errors deserve careful investigation because they are quite probable when errors are inserted on purpose to break the cipher.

ACKNOWLEDGMENT

This research was partly supported by the Polish Ministry of Education and Science as a 2005−2008 research project.

REFERENCES

[1] Bertoni G., Bregeveglieri L., Koren I., Maistri P., Piuri V., Error Analysis and Detection Procedures for a Hardware Implementation of the Advanced Encryption Statandard, IEEE Trans. On Computers Vol. 52 No 4, April 2003, pp.492-505.

[2] Karri R., Kuznetsov G., Goessel M., Parity Based Concurrent Error Detection in Symmetric Block Cyphers, proc. of International Test Conference 2003, pp.919-926.

[3] National Inst. Of Standards and Technology, Federal Information Processing Standard 197, The advanced Encryption Standard(AES), November 2001,http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[4] Bucholc K., Idzikowska E., A metod of multiple error detection in substitution blocks for block ciphers, Proceedings of the 13th International Multi-Conference on Advanced Computer Systems, Szczecin 2006, vol. 1, 259-264

[5] Millan W., New Cryptographic Applications of Boolean Function Equivalence Classes. ACISP 2005, Springer, LNCS vol 3574, pages 572-583.

[6] Joshi N., Karri R.,,Invariance based concurrent error detection for the advanced encryption standard, Patent application 01/18/07, http://www.freshpatents.com/Invariance-based-concurrent-error-detection-for-the-advanced-encryption-standard-dt20070118ptan20070014395.