BULETINUL INSTITUTULUI POLITEHNIC DIN IAŞI
Publicat de
Universitatea Tehnică „Gheorghe Asachi” din Iaşi
Volumul 63 (67), Numărul 4, 2017
Secția
ELECTROTEHNICĂ. ENERGETICĂ. ELECTRONICĂ

THE DESIGN OF PERFECTLY BALANCED CONSTANT-WEIGHT SUBSTITUTION CODES

BY

LUMINIȚA SCRIPCARIU*

“Gheorghe Asachi” Technical University of Iaşi,
Faculty of Electronics, Telecommunications and Information Technology

Received: October 20, 2017
Accepted for publication: November 27, 2017

Abstract. Constant-weight substitution (CWS) codes (Scripcariu, 2016) can precede the encryption systems, in order to encrypt a constant-weight sequence each time, as a counter measure against side-channel attacks (Black & Urtubia, 2002). These attacks exploit the unbalanced energy or the variable processing time used to deliver different numerical data values, depending on its Hamming weight. CWS codes deliver bit sequences with a constant Hamming weight that ensures identical processing time and power and the attacker cannot extract any useful information by measuring it. In this paper, the characteristics of CWS codes, in general, and of perfectly balanced CWS codes, in particular, are described. Then, the CWS codes design principles are presented and some designed constant weight substitution codes exemplify them. These codes can be implemented as software algorithms or as hardware circuits as well.

Key words: hamming weight; substitution code; cryptographic attack; code design; software algorithm.

1. Introduction

Information security is a critical task of any network security system and it is based on data encryption techniques.

Nowadays, various encryption algorithms exist (Paar & Pelzl, 2010).

*Corresponding author: *e-mail*: lscripca@etti.tuiasi.ro

Numerous *public-key* algorithms – such as Rivest-Shamir-Adleman (RSA), 'Elliptic Curve Digital Signature Algorithm' (ECDSA) and 'Digital Signature Algorithm' (DSA) – or *secret-key* algorithms – such as 'Advanced Encryption Standard' (AES) and 'Secure and Fast ***Encryption*** Routine' (SAFER) - are competing to be the best encryption tool of the moment.

However, all of them have the same weaknesses when a side-channel attack is launched. The input data is not balanced regarding the Hamming weight of the data blocks.

The encryption key and other useful information can be hacked by an attacker measuring the power consumption of the target device (Mangard, 2002). This is the so-called simple power analysis (SPA) attack and it can be used against RSA encryption system. A similar attack method based on energy analysis, Comparative Power Analysis (CPA) attack, can also be used to break RSA (Homma *et al*., 2010).

Another powerful side-channel attack is the Differential Power Analysis (DPA) that can be used successfully against DES and TDES (McEvoy *et al.*, 2007).

DPA can be used in a modified version, called Binary Power Analysis (BPA), to break AES algorithm.

All these cryptographic attacks exploit the weakness of any encryption algorithm, created by the unbalance between the number of bits of '0' and '1'.

The encryption algorithm implementation can be done in various ways (Ferguson *et al*., 2010). It is important to add an initial code step to any encryption algorithm that ensures the energy balance for the data stream.

The energy balance depends on the Hamming weight of the binary form of the numerical value that can be uniformed applying a CWS code.

In this paper, the design of some CWS codes is approached.

The CWS coding properties and principles are presented in the second section of the paper. Some convenient CWS codes are discussed in the third paragraph. Some selected codes are designed and their structures are compared in the fourth section. Finally, the conclusions of the paper are drawn, followed by the references.

2. Description of Constant-Weight Substitution Codes

The Hamming weight of a binary sequence is expressed as the number of '1' bits contained by it. The unbalanced Hamming weight between consecutive sequences affects the robustness of any cryptosystem and it is a tractable aspect (Berlekamp *et al*., 1978).

In order to ensure a balanced delivery of encrypted data, constraints regarding the Hamming weight are imposed and applied by CWS coding techniques.

For a CWS code, all the code words have the same Hamming weight.

Let us denote by N the number of bits in the input block and by M, the length of the code word, in its binary representation, having the Hamming weight equal to k. The resulting code is denoted as CWS (M, k, N).

CWS coding method consists in the substitution of each of *N*-bit input block by a *M*-bit sequence with *k* bits '1'.

The number of all *M*-bit sequences, with the Hamming weight equal to *k*, is given by:

$$m = \binom{M}{k}. \tag{1}$$

In order to design a CWS code, the following inequality must be true:

$$2^N \le \binom{M}{k}. \tag{2}$$

The maximum value of *N* results as:

$$N_{\max} = \left\lfloor \log_2 \binom{M}{k} \right\rfloor \tag{3}$$

The coding rate can be expressed as percentage of *M*:

$$R = \frac{N}{M} \times 100, \ [\%]. \tag{4}$$

The optimal CWS codes are chosen based on the coding rate that must be at least 0.5 and the desired Hamming weight of the code words.

2. Perfectly Balanced Constant-Weight Substitution Codes

CWS codes with k equal to half of M having the same number of '1's and '0's are considered perfectly balanced codes. In this case, *M* has to be an even number.

Some combinations (M, $M/2$, $N_{\max}$) that can be used to design CWS codes are presented in Table 1.

Large values of the coding rate, close to 1, are preferable.

High coding rate ensures a small reduction of the transmission speed.

A coding rate equal to 0.5 leads to halving the data rate and doubling the processing time.

A code with a coding rate value greater than 75 % diminishes the data rate only with 25%, while a coding rate value higher than 90% leads to a data speed loss less than 10 %.

High coding-rates are recommended for high-speed data communication systems.

However, it is difficult to design 'large' CWS codes and to implement the corresponding software coding algorithms or the hardware code structures.

A trade-off between data rate degradation and the design complexity should be done.

Table 1
Parameters of Some Perfectly Balanced CWS Codes

Code word length (*M*) [bits]	Hamming weight (*k*)	No. of code sequences	Maximum input sequence length (*N*) [bits]	Coding rate (*R*) [%]
4	2	6	2	50
6	3	20	4	66
8	4	70	6	75
10	5	252	7	70
12	6	924	9	75
14	7	3,432	11	78.57
16	8	12,870	13	81.28
18	9	48,620	15	83.33
20	10	184,756	17	85
22	11	705,432	19	86.36
24	12	2,704,156	21	87.5
26	13	10,400,600	23	88.46
32	16	601,080,390	29	90.625
64	32	1.8E19	60	93.75

3. Design Principles of Perfectly Balanced CWS Codes

For relative small values of *N*, the CWS code can be simply implemented, based on the code table, as a substitution box, shortly called 'S-box', by writing the code table into a memory.

For example, CWS (8, 4, 4) code has the code table given in Table 2. The code output values are expressed in hexadecimal format.

The code table maps the input sequences to those combinations of two hexadecimal digits that have the Hamming weight equal to 2: {3, 5, 6, 9, A, C}. We choose the most balanced four digits from the set in order to build the code: {5, 6, 9, A}. Digit 3 corresponds to the bit sequence 0011, while digit C is the equivalent of 1100. Both of them are unbalanced regarding the Hamming weight.

Table 2
CWS (8, 4, 4) Code Table

Input	00	01	10	11
00	55	56	59	5A
01	65	66	69	6A
10	95	96	99	9A
11	A5	A6	A9	AA

Table 2 is read as follows: the 4-bit input sequence is read as a combination of two 2-bit blocks and the output value is written with two hexadecimal digits. For example, if the input decimal value is 14, the corresponding input bit sequence is 1110.The output hexadecimal value is read

on the fourth row and the third column of the table and it is equal to 0x.A9 that corresponds to the decimal value 169 and the binary sequence 10101001.

In general, the number of cells of the code table is equal to 2^N. The values of N that can be considered small enough to implement the CWS code with S-boxes depend on the memory technology.

Nowadays, the size of the memory can be very large (up to TB) while the access time, consisting in the memory latency and the transfer time, is less than 100 nanoseconds. This time value corresponds to 10 Mega operations per second.

The memory size is not a limitation to the design of a CWS code with S-boxes. But the encoding speed is limited by the memory access time and by the size of the data block because a whole data block is read from the memory in one stage of the encoding algorithm.

For example, if the data block consists of 24 bits, the data rate is up to 240 Mbps. But for a CWS code having a coding rate of 75% (3:4), the throughput diminishes to 180 Mbps.

Faster ways to run a CWS code are given by its implementation as a software algorithm that processes input data block-by-block, or by designing a hardware encoding structure according to the CWS code table, using HDL (Hardware Description Language) and a high-speed logic circuit technology (Hadia *et al.*, 2011).

For large values of *M*, *N* and *k*, a different approach of CWS code design is used to manage data.

The particular case of the so-called 'large codes' with sizes multiple of 4 is considered first. The code size can be expressed as a number of bytes (B) in order to write the code table more easily.

For example, CWS (64, 32, 60) code can be seen as it is applied on 16 hexadecimal digits and the code generates 15 hexadecimal digits, resulting in a coding rate equal to 15:16.

This way of structuring the output sequences facilitates the code table design for large code sizes.

Let us design the CWS (16, 8, 8) code that has a coding rate equal to 1:2. It is difficult to write the code table having 256 rows. But it is easier to establish rules for the encoding algorithm by writing each output value as a sequence of two hexadecimal digits.

The code word must have a Hamming weight equal to 8 while it is composed of 4 hexadecimal digits or two 8-bit sequences of weight 4.

The following sets can be used:

a) the set of digits of weight equal to 1: s1 = {1, 2, 4, 8};

b) the set of digits of weight equal to 2: s2 = {3, 5, 6, 9, A, C};

c) the set of digits of weight equal to 3: s3 = {7, B, D, E};

Only the digits of weight 2 are used and from this set, only the most balanced digits are selected. The resulting bytes (written as two-hexadecimal

digit combinations), having the weight equal to 4, are grouped into the following set:

S4 = {55, 56, 59, 5A, 65, 66, 69, 6A, 95, 96, 99, 9A, A5, A6, A9, AA}

Combining these 16 bytes, 256 sequences of 16 bits are obtained.

The code table can be written as a 16-by-16 array, in the ascending order, from top to bottom and from left to right.

The rows and the columns are numbered using hexadecimal digits (from 0 to F), and the input data is read as a combination of two hexadecimal digits. The most significant digit is used to read the row and the least significant one identifies the column in the code table. From each table cell, a 4-digit sequence is read.

The rows and the columns of the table can be indexed, from 0 to 15.

The value set S4 can also be indexed from 0 to 15.

The input value generates two indexes (i and j) that are used by the encoding algorithm to provide the code word by reading the corresponding values from S4 set.

For example, if the input decimal value is 124, it corresponds to a hexadecimal number 7C, so the indexes are $i = 7$ and $j = 12$. The corresponding numbers in S4 are 6A and A5 so the output code hexadecimal sequence is 6AA5 that corresponds to the bit sequence 0110101010100101. The Hamming weight of the obtained code word is 8, while the input sequence has the Hamming weight equal to 5. Another case can be studied. Let us consider the input decimal value 0 that is written in hexadecimal form as 0x.00, having indexes, i and j, equal to 0. The output code word is 0x.5555 with the weight equal to 8 while the input sequence weight is 0.

The Hamming weight of the codeword is always the same, no matter what the input sequence weight is, according to the purpose of a CWS code.

5. Conclusions

The Constant-Weight Substitution (CWS) codes can be used to encode data before an encryption code is applied in order to produce constant weight sequences. Using the perfectly balanced CWS codes, the processing time and the power consumption of the encryption module are the same regardless of the input value and the side-channel attacks are counter fought. Perfectly Balanced CWS codes are presented, up to 64-bit code sequences. The principles of designing these codes are described and applied on two different codes: a small size code and a large size code. Both can be implemented with S-boxes or as software algorithms or in hardware, with high-speed logical circuits. Different numerical bases can be used to compact the code words, to index sequences and to simplify the design of the codes.

REFERENCES

Berlekamp E., McEliece R., Van Tilborg H., *On the Inherent Intractability of Certain Coding Problems*, IEEE Trans. Inform. Theory, **24**, 384–386 (1978).

Black, J., Urtubia, H., *Side-Channel Attacks on Symmetric Encryption Schemes: the Case for Authenticated Encryption*, Proc. of the 11th USENIX Security Symposium, San Francisco, USA, 327-338, 2002.

Ferguson N., Schneier B., Kohno T., *Cryptography Engineering – Design Principles and Practical Applications*, Wiley Publishing, Inc., 2010.

Hadia S.K., Patel R.R., Kosta Y.P., *FinFET Architecture Analysis and Fabrication Mechanism*, Proc. of IJCSI Intern. J. of Computer Science Issues, **8**(**5**), *1*, 235-240 (2011).

Homma N., Miyamoto A., Aoki T., Satoh A., Shamir A., *Comparative Power Analysis of Modular Exponentiation Algorithms*, IEEE Trans. Comput., **59**, *6*, 795-807 (2010).

Mangard S., *A Simple Power-Analysis (SPA) Attack on Implementation of the AES Key Expansion*, Proc. of Inform. Security and Cryptology, ICISC 2002, 343-358 (2002).

McEvoy R., Tunstall M., Murphy C.C., Marnane W.P., *Differential Power Analysis of HMAC Based on SHA-2, and Countermeasures*, 8th Workshop on Inform. Security Applications – WISA 2007, Lecture Notes in Computer Science, **4867**, 317-332 (2007).

Paar C., Pelzl J., *Understanding Cryptography*, Springer-Verlag Berlin, 2010.

Scripcariu L., *Constant-Weight Substitution Codes Against Side-Channel Attacks*, Bul. Inst. Politehnic, Iaşi, s. Electrot., Energ., Electron., **62(66)**, *3*, 51-60 (2016).

PROIECTAREA CODURILOR DE SUBSTITUȚIE CU PONDERE CONSTANTĂ PERFECT ECHILIBRATE

(Rezumat)

Codurile de substituție cu pondere constantă (CWS) pot fi aplicate înaintea codurilor de criptografiere, astfel încât să se cripteze de fiecare dată secvențe cu aceeași pondere, ca o contramăsură împotriva atacurilor pe canale laterale. Aceste atacuri exploatează variabilitatea timpului şi a energiei folosite pentru criptarea blocurilor de date, care depind de ponderea lor Hamming. Codurile CWS produc secvențe binare cu aceeaşi constantă Hamming care asigură că timpul şi energia de procesare sunt aceleaşi şi atacatorul nu poate extrage informații utile prin măsurarea lor. În lucrare, sunt discutate mai întâi caracteristicile codurilor CWS, în general, şi ale celor perfect echilibrate, în particular. În continuare, sunt descrise principiile de proiectare a acestor coduri şi se prezintă proiectarea câtorva astfel de coduri ca exemple. Aceste coduri pot fi implementate fie ca algorimi software, fie folosind circuite de criptare.