A NEW APPROACH INTO CONSTRUCTING S-BOXES FOR LIGHTWEIGHT BLOCK CIPHERS

Hristina Mihajloska
Faculty of Natural Sciences and Mathematics
Institute of Informatics
Skopje, Macedonia

Danilo Gligoroski
Norwegian University of Sciences and Technology
Department of Telematics
Trondheim, Norway

ABSTRACT

In this paper we can give an overview of some of the popular block ciphers, rather a strong focus is put on the construction of their S-boxes. While we were studding S-boxes we saw that there is a room for improvement. The focus is on putting quasigroups of order 4 with good cryptographic properties and their e and $d-$ transformations, in order to achieve also small bit S-boxes.

I. INTRODUCTION

According to the father of ubiquitous computing Mark Weiser just now we are living in the third wave in computing. This new wave also known as a calm technology is when technology retreats into the background of our lives and affects all of computer disciplines. The huge deployment of pervasive devices promises on one hand many benefits (e.g. make our lives better), but on the other hand, many of these devices are security sensitive (make our lives more complicated). In this manner cryptography takes starring role.

Modern cryptography intersects disciplines of mathematics and computer science, so algorithms in it has become progressively more complex and its applications more worldwide. Two main concepts for encrypting plaintext are used in cryptography, according to the secret keys: symmetric i.e. secret key cryptography and asymmetric i.e. public key cryptography. In the first one the secret key is unique and the same key is used in the phase of encryption and decryption, but in the other concept, there are two types of key, secret and public key which are used for encrypting (is used public key) and decrypting (is used secret key) the message.

In the area of symmetric cryptography there is a new trend known as lightweight cryptography for cryptographic components that can be efficiently implemented into pervasive devices, as well as for ciphers that are particularly suitable for this purpose. So this trend enforces small and fast secure algorithms that their implementation can require as lightweight hardware area as possible.

II. ABOUT BLOCK CIPHERS

One of the several cornerstones of the modern secure communications is the cryptographic primitives called “Block Ciphers”. They are symmetric cryptographic primitives that try to closely resemble the concept of secret and ideally random permutation.

Every block cipher typically consists of a short sequence of simple operations called round function. This function is repeated r times, so these loops are called rounds. The first round takes an n-bit plaintext block as input, transformed that and leaves it to the next round. The last round outputs the ciphertext. As well, each round depends on the round key which is derived from the k-bit secret key. Because block cipher is symmetric cryptographic primitive, receiver must be able to uniquely decrypt the ciphertext beside he/she has a secret key. This can only be achieved if the round function is bijective for any given value of the secret key.

Round function can be constructing in one of the following ways:

1) *Feistel Network*

This round function is named after H. Feistel one of the IBM researchers who designed Lucifer and DES. Feistel round function is designed on that way to split the input block into two equal pieces (L_0, R_0). The right part is leaving unchanged and forms the left part of output L_{i+1}. The right part of the output is getting by adding to the left part of the input a transformed copy of R_i by the round function f. For each round $i = 0, 1, 2, \dots, n-1$

$$L_{i+1} = R_i,$$
$$R_{i+1} = L_i + f(R_i, K_i)$$

Then the ciphertext is (L_n, R_n)

2) *SP-Network*

SP network comes from substitution and permutation network where round function is built with combining layers of simple invertible functions:

substitution, called S-boxes and permutation, called P-boxes. S-boxes play on a small unit of data so they have to be distinguished with highly nonlinear properties if they want to confuse the input data into the cipher. The P-boxes, on the other hand, are simple linear transformations which operate on the complete input block and their goal is to diffuse the effect from the substitution layer [4]. Most popular block cipher based on an SP network is AES (Rijndael). We have also notice that SP network functions take a part in the functions of many Feistel ciphers.

III. S-BOXES

As we mentioned before, the S-boxes act a fundamental role for the security of almost all modern block ciphers. So they have to be selected very carefully to make the cipher resistant against all kinds of attacks. There is no formal definition for S-boxes, but in general they are defined as a lookup tables or Boolean functions.

A vectorial Boolean function $f: \mathcal{F}_2^n \rightarrow \mathcal{F}_2^m$ also called (n, m) S-box can be represented by the vector $(f_0, f_1, \dots, f_{m-1})$ where f_i are Boolean functions from $\mathcal{F}_2^n$ into $\mathcal{F}_2$ for $0 \leq i \leq m-1$, which we also call the component or output functions of the S-box [3].

The other definition is an $m \times n$ S-Box can be represented as a lookup table with 2^m words of n bits each. This tables might be fixed as normally used, like in Data Encryption Standard (DES), but in some ciphers they are generated dynamically from the key; e.g. the Blowfish and the Twofish encryption algorithms.

In particular there are many research papers that applied criterions for making good S-boxes which make the cipher resistant against differential and linear cryptanalysis. Some of the formal design criterions of S-boxes for cryptographic purposes are [9]:

1) *Nonlinearity*

Basic criterion for every cryptosystem that is widely accepted is that it must be nonlinear. There are many aspects for defining nonlinearity. According to Rueppel [13] the nonlinearity of a Boolean function (when we present our S-box by the vectorial Boolean function) can be measured by the Hamming distance to the set of affine functions and is calculated by

$$\delta(f) = 2^{n-1} - \frac{1}{2} max|\chi_f|,$$

where χ_f is called the Walsh spectrum of f.

The algebraic degree is a first measure of nonlinearity of a Boolean function. Every Boolean map $f: \mathcal{F}_2^n \rightarrow \mathcal{F}_2^m$ has a unique expression as

$$f(x_1, x_2, \dots, x_n) = \sum_{I \subseteq \{1,2,\dots,n\}} x^I a_I$$

with coefficients $a_I \in \mathcal{F}_2^m$. So the degree of this Boolean map f as a polynomial is calculated with $Deg\, f = \max\{\, \#I \mid a_I \neq 0\}$ and is called algebraic degree of f.

2) *Strict Avalanche Effect*

A function $f: \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2^m$ exhibits the avalanche effect if and only if

$$\sum_{x \in \mathbb{Z}_2^n} wt\left(f(x) \oplus f\left(x \oplus c_i^{(n)}\right)\right) = m\, 2^{n-1}$$

for all $i\ (1 \leq i \leq n)$.

$c_i^{(n)}$ refers to an n dimensional vector with Hamming weight 1 at the i −th position.

In other words, this means that, whenever an input is change (a single bit is flipped), an average of one half of the output bits changes.

Also Webster and Tavares [14] introduced Strict Avalanche Criterion (SAC) in order to get strong S-boxes.

A function $f: \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2^m$ satisfies the SAC, or f is a strong S-box, if for all $i\ (1 \leq i \leq n)$ there hold the following equations:

$$\sum_{x \in \mathbb{Z}_2^n} wt\left(f(x) \oplus f\left(x \oplus c_i^{(n)}\right)\right) = (2^{n-1}, 2^{n-1}, \dots, 2^{n-1})$$

In particular, if $f: \mathbb{Z}_2^n \rightarrow \mathbb{Z}_2$ satisfies the SAC, f is called a Boolean strong S-box.

3) *Cross correlation of Avalanche Variables*

Avalanche variables are binary components of the avalanche vectors, defined as the exclusive-or sums

$$V_i = S(x) \oplus S(x \oplus c_i^{(n)})$$

where $S(x)$ refers to an S-box.

By the [14] for the given set of avalanche vectors generated with complementing the single input bit, all the avalanche variables should be independent pairwise. The degree of independences is measured by the correlation coefficient $\rho(A, B)$ which is calculated with

$$\rho(A, B) = \frac{cov(A, B)}{\sigma(A) * \sigma(B)}$$

where A and B are two random avalanche variables.

4) *Bijection*

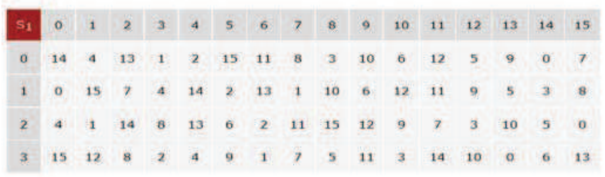
For an n-bit input function f to be a bijection it must satisfy a necessary and sufficient condition that any linear combination of a Boolean function has Hamming weight 2^{n-1}. In other words this means that every possible input vector is mapped into a unique output vector.

IV. S-BOXES FROM SOME OF THE POPULAR BLOCK CIPHERS

S-boxes from DES

DES is a symmetric block cipher which uses 56-bit secret key for encrypting 64-bit plaintext. It is Feistel Network cipher with 16 rounds in each of it is using 48-bit keys obtained by permutations and shifts on the original secret key. The 48-bit input is separated into eight 6-bit blocks. Each block is subjected to a substitution made from S-boxes, from which the output is 4-bit block.

S-boxes in DES are designed as a lookup tables with 4 rows and 16 columns, where every row is a permutation of all hexadecimal numbers. When 6-bit input is entered in this S-box, the first and last bit from the sequence is taking to represent a 2-digit binary number in the range of 0 to 3. This number is presented the row in the lookup table. The middle four bits of the sequence result in 4-digit binary number in the range of 0-15. So the unique substitution number which is obtain in intersect of that row and column, which is in the range of 0 to 15 is represented 4-bit output from DES S-box [7]. The first S-box from DES eight S-boxes is given on Figure 1.



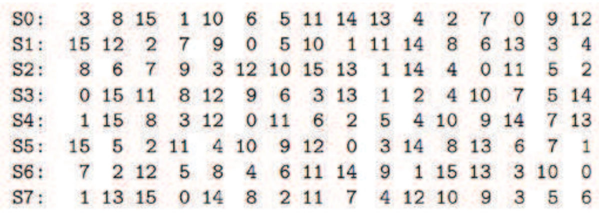
S1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Figure 1: The first DES S-box

S-boxes from Serpent

Serpent is a SP-Network with 32-rounds operating on 32-bits words, giving a 128-bits block. In other words, Serpent encrypts 128-bit plaintext into 128-bit ciphertext in 32 rounds, with the help of 33, 128-bit subkeys. For external computation, every block is written as a plain 128-bits hexadecimal number. On the other side, internally, all values are represented in little-endian mode, where the last word is the most significant, and the first word (word 0) is the least significant word. Therefore the 32 rounds use 8 different S-boxes each one map four input bits to the same number output bits. Because, Serpent has 32 rounds, 8 S-boxes and 128-bit blocks, each of these S-boxes is used in exactly four rounds, where it is used 32 times in parallel.

The S-boxes were generated in the following manner, which was inspired by RC4. They used a matrix with 32 arrays each with 16 entries. The matrix was initialized with the 32 rows of the DES S-boxes and transformed by swapping the entries in the r-th array depending on the value of the entries in the $(r-1)$-st array and on an initial string representing a key. If the resulting array has the desired (differential and linear) properties, save the array as a Serpent S-box. Repeat the procedure until 8 S-boxes have been generated [1]. One representative from the set of good Serpent's S-boxes is given on the Figure 2.



S0:	3	8	15	1	10	6	5	11	14	13	4	2	7	0	9	12
S1:	15	12	2	7	9	0	5	10	1	11	14	8	6	13	3	4
S2:	8	6	7	9	3	12	10	15	13	1	14	4	0	11	5	2
S3:	0	15	11	8	12	9	6	3	13	1	2	4	10	7	5	14
S4:	1	15	8	3	12	0	11	6	2	5	4	10	9	14	7	13
S5:	15	5	2	11	4	10	9	12	0	3	14	8	13	6	7	1
S6:	7	2	12	5	8	4	6	11	14	9	1	15	13	3	10	0
S7:	1	13	15	0	14	8	2	11	7	4	12	10	9	3	5	6

Figure 2: The S-boxes used in the block cipher Serpent

V. CONSTRUCTING S-BOXES WITH QUASIGROUPS

There are many research papers on the field of quasigroups. In our investigation we are concerned for small order quasigroups with good properties for generating S-boxes, unlike the large order quasigroups which are much more inefficient to implement, especially in hardware.

In the lightweight cryptography the choice of 4x4 rather than 8x8 bits S-boxes is hardware driven; 4-bit S-boxes require less than a quarter of the area of 8-bit S-boxes. However, if we work with 4-bit S-boxes we must be very careful with their choice, because they are cryptographically weaker than S-boxes with 8-bits input and 8-bits output.

In this paper talking about binary quasigroups we refer to small order quasigroups.

From classification of binary quasigroups according to their Boolean representation, there are two classes, linear and non-linear quasigroups [8, 5]. When we consider them and minimal features for good S-boxes, we decide to work only with non-linear quasigroups.

In the first approach we present S-box as a binary quasigroup. In this case our S-box will be 4x2 bits because 2x2 bits are input and 2 bits are output. For example, if we get 45-th binary quasigroup (from lexicographical ordering [6]) which belong to the class of non-linear quasigroups and we have the 4-bits sequence $B = b_1 b_2 b_3 b_4 = 1011$ as input in this S-box the output will be computed in the following way. Bits corresponding to $b_1 b_2$ select a row in the quasigroup, while the bits corresponding to $b_3 b_4$ select a column. The value $S(B) = 11$ is then a 2-bits output in that row and column.

*	00	01	10	11
00	00	01	11	10
01	11	10	00	01
10	10	00	01	11
11	01	11	10	00

Q-S-boxes of size 6x4 (DES like)

The second approach is to present S-box with using already known and investigate quasigroup string transformations, $e_{l,*}$ and $d_{l,*}$ by Markovski, Gligoroski at al. [10, 11, 12]. In this case our S-box will be 6x4 bits and also will be present as a lookup table, like DES S-box. In this manner we named this S-box, Q-S-box (DES like).

Graphical representation of $e-$transformation is given on the Figure 3, where output bits are received as follows:

$$c_1 = l * a_1,\ c_2 = c_1 * a_2, \dots, c_n = c_{n-1} * a_n$$

where $*$ is a quasigroup operation on the set Q i.e. consider a quasigroup $(Q,*)$.

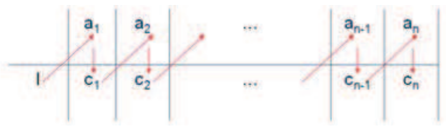


Figure 3: Quasigroup $e-$transformation

For example, if we get the same quasigroup from above example and we have the input sequence $B = b_1 b_2 b_3 b_4 b_5 b_6 = 101011$ then the output will be computed with e-transformation. So the first two bits will present the leader, the second collection of two bits will be a_1, and the third collection of two bits will be a_2 (from Figure 3), so we have:

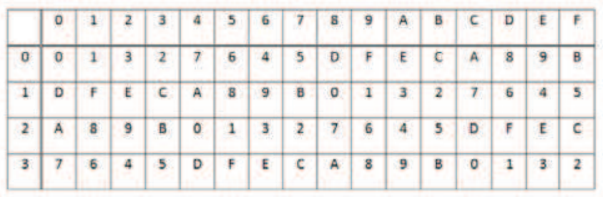
$$b_1 b_2 * b_3 b_4 = s_1 s_2$$

$$s_1 s_2 * b_5 b_6 = s_3 s_4$$

*	00	01	10	11
00	00	01	11	10
01	11	10	00	01
10	10	00	01	11
11	01	11	10	00

The value $S(B) = s_1 s_2 s_3 s_4 = 0111$ is then the 4-bits output from this kind of S-box.

According to this example we can calculate all 4-bits outputs from all possible 6-bits inputs, and we obtain Q-S-box for the 45-th binary quasigroup (given on Figure 4).



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	3	2	7	6	4	5	D	F	E	C	A	8	9	B
1	D	F	E	C	A	8	9	B	0	1	3	2	7	6	4	5
2	A	8	9	B	0	1	3	2	7	6	4	5	D	F	E	C
3	7	6	4	5	D	F	E	C	A	8	9	B	0	1	3	2

Figure 4. Q-S-box (DES like)

Q-S-boxes of size 4x4 (Serpent like)

Going step by step into implementing quasigroups as S-boxes we saw that we have an opportunity to make 4x4 bits S-box. To achieve our goal we bring into play the orthogonal property of two quasigroups. The concept of orthogonality can be described in algebraic language as: Two quasigroups $(Q_1,*), (Q_2,*)$ are orthogonal if the system of equations $x * y = a$ and $x * y = b$ have a unique solution for every pair of elements $a, b \in Q$ [2].

If $B = b_1 b_2 b_3 b_4$ are the input bits, then according to the orthogonal property where

$$b_1 b_2 *_{Q_1} b_3 b_4 = s_1 s_2$$

$$b_1 b_2 *_{Q_2} b_3 b_4 = s_3 s_4$$

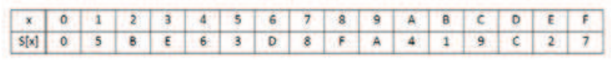
The output is $S(B) = s_1 s_2 s_3 s_4$; 4-bits output that we are searching for.

For example, if we get 4-th and 40-th binary quasigroups (from lexicographical ordering) which belong to the class of pairs of orthogonal quasigroups and we have the 4-bits sequence $B = b_1 b_2 b_3 b_4 = 1011$ as input, the output is the value $S(B) = s_1 s_2 s_3 s_4 = 0001$.

$*_4$	00	01	10	11
00	00	01	10	11
01	01	00	11	10
10	11	10	01	00
11	10	11	00	01

$*_{40}$	00	01	10	11
00	00	01	11	10
01	10	11	01	00
10	11	10	00	01
11	01	00	10	11

According to this example we can calculate all 4-bits outputs from all possible 4-bits inputs, and we obtain Q-S-box (Serpent like) for the 4-th and 40-th binary quasigroups (given on Figure 5).



x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S[x]	0	5	B	E	6	3	D	8	F	A	4	1	9	C	2	7

Figure 5. Q-S-box (Serpent like)

However, we check which of the pairs of 576 x 576 binary quasigroups are orthogonal. We found that 144 pairs fulfill our requirements but unfortunately none of them belong to the class of non-linear quasigroups. All 142 pairs are part of the class of linear quasigroups which are not adequate for constructing S-boxes.

VI. Conclusion

Studding and going into details in the area of S-boxes of the block ciphers, open a new horizon of possibilities that can be investigated in order to improve and prove some of the features for constructing optimal S-boxes.

One such approach of building S-boxes is by means of the quasigroups, which we presented. But in order our newly created S-boxes to reach the stage of Serpent/Present, we have to conduct many trials and tests on the properties of our S-boxes, not only to see their linearity. However, linearity is the first and the initial condition for any structure to be viewed as an S-box.

References

[1] R. Anderson, E. Biham, and L. Knudsen, “Serpent: A Proposal for the Advanced Encryption Standard,” NIST AES proposal, 1998

[2] V. Belousov, “Parastrophic-orthogonal quasigroups”, Quasigroups and Related Systems, vol 13, pages 25 – 72, 2005

[3] A. Braeken, "Cryptographic Properties of Boolean Functions and S-Boxes," PhD thesis, Katholieke Universiteit Leuven, B. Preneel (promotor), 209+16 pages, 2006

[4] C. de Canniere, A. Biryukov, B. Preneel, “An Introduction to Block Cipher Cryptanalysis”, invited paper, Proceedings of the IEEE, vol.94, issue 2, page 346-356, Fev. 2006

[5] V. Dimitrova, “Quasigroup Processing String, their Boolean Representations and Application in Cryptography and Coding Theory”, PhD thesis, Ss. Cyril and Methodius University, Skopje, S. Markovski (promotor), 2010

[6] V. Dimitrova, “Quasigroup Transformations and their Applications”, MsC thesis, Ss. Cyril and Methodius University, Skopje, S. Markovski (promotor), 2005

[7] J. Gargiulo, “S-Box Modifications and Their Effect in DES-like Encryption Systems”, SANS Institute Reading Room site

[8] D. Gligoroski, V. Dimitrova, and S. Markovski, “Quasigroups as Boolean functions, their equation system and Groebner bases,” Groebner Bases, Coding and Cryptography, Springer Berlin Heilderberg, 2009, pages 415-420

[9] K. Kim, “A Study on the Construction and Analysis of Substitution Boxes for Symmetric Cryptosystems”, Ph.D Thesis, Yokohama National University, Japan, 1991

[10] S. Markovski, D. Gligoroski, S. Andova, “Using quasigroups for one-one secure encoding”, in Proceedings of VIII-th Conference for Logic and Computing - LIRA'97, September 1997, Novi Sad, FR Yugoslavia

[11] S. Markovski, D. Gligoroski, V. Bakeva, “Quasigroup String Processing: Part 1”, Maced. Acad. of Sci. and Arts, Sc. Math. Tech. Scien. XX 1-2(1999) 13-28

[12] S. Markovski, “Quasigroup String Processing and its Applications in Cryptography”, Proceeding of the 1-st MII 2003 Conference, Thessaloniki, pages 278-290, 2003

[13] R. A. Rueppel, “Linear Complexity and Random Sequences”, Proc. of EUROCRYPT ’85, Springer-Verlag, 1986

[14] A. F. Webster and S. E. Tavares, “On The Design Of S-Boxes”, Proceedings of CRYPTO’85, Springer-Verlag, vol. 218, pages 523-534, 1986