International Journal of Computer Networks and Communications Security

VOL. 1, NO. 3, AUGUST 2013, 88–94
Available online at: www.ijcncs.org
ISSN 2308-9830

State of the ART in Boolean Functions Cryptographic Assessment

ADHAM M. ELHOSARY[1], NABIL HAMDY[2], ISMAIL ABDEL-GHAFAR FARAG[3], ALAA ELDIN ROHIEM[4]

[1]Departement of Communications, Military Technical College, Egypt

[2]Misr International University, Egypt

[3]Arab Academy for Science and Technology and Maritime Transport (AASTMT)

[4]Technical Research Center of the Armed Forces, Egypt

E-mail: [1]adhosary@gmail.com

ABSTRACT

Boolean functions play important role in cryptography, since in convention a symmetric encryption algorithm can be designed by composing Boolean functions satisfying good cryptographic criteria. In this paper; state of the art in mathematical and practical study of the most important cryptographic criteria of Boolean functions and how to implement algorithms that fulfill these criteria are introduced. Also; the most known constructions for generating Boolean functions that satisfy good cryptographic criteria are summarized.

Keywords: *Boolean Functions, Vectorial Boolean Functions, Cryptographic Criteria.*

1 INTRODUCTION

This paper presents the state of the art in Boolean functions representation and cryptographic assessment that should be considered as a starting point in cryptographic algorithm design and analysis, namely block ciphers, stream ciphers, and also hash functions. It starts by a brief introduction of Boolean functions and their representation methodology. Afterwards; most of the well-known Boolean functions cryptographic criteria are studied giving illustrative examples for the proof of concept.

2 BOOLEAN FUNCTIONS

In Boolean Algebra the set {0,1} is set algebra of interest, since 0 and 1 represent binary elements which are the basic representation of data in computer architecture and therefore F_2 is the finite field of choice. F_2: {0, 1, +, -} which is equivalent to GF(2), F_2^n is the set of all binary vectors of length n (i.e. F_2 vector space) [1], the null vector of F_2^n is 0. F_2^n sometimes is endowed with field structure to form the well-known Galois Field GF (2^n).

A Boolean function is a mapping $f(x)$: $F_2^n \rightarrow F_2^m$ as mentioned before. The total number of Boolean functions of n-vector input is 2^{2^n} which is very large even for values which quite small i.e. for n=8, $2^{2^8} = 2^{256}$ which is approximately equals the number of atoms of the universe! [1].

3 REPRESENTATION

Boolean polynomial is a string which results from a finite number of Boolean operations on a finite number of elements.

The Boolean polynomial $P = x_1 + x_2x_3$ is represented as sum of min terms (monomials), this representation is unique for every Boolean function and is called Disjunctive Normal Form, also the product of max terms of a Boolean function is unique and called Conjunctive Normal Form [2].

3.1 Algebraic Normal form

In cryptography the most used form of representing a Boolean polynomial or function is called Algebraic Normal Form (ANF), it is unique and can be obtained by XOR summation of monomials [2].

$$f(x) = \bigoplus_{J \subset \{1,2,\ldots,n\}} a_J \prod_{j \in J} x_j \qquad (1)$$

e.g. $f(x_1,x_2,.....,x_n) = a_o \oplus a_1x_1 \oplus . . . . . \oplus a_nx_n \oplus a_{12}x_1x_2 \oplus . . . \oplus a_{(n-1)n}x_{n-1}x_n \oplus a_{12\ldots n}x_1x_2\ldots x_n$

A Boolean function f(x): $F_2^n \rightarrow F_2^m$ is always associated with its ANF which can be computed practically using divide and conquer butterfly algorithm called Fast Möbius Transform [1], it can be implemented simply by the C code as provided by [3].

Hence if, in the truth-table of f(x), the binary vectors are ordered in lexicographic order, with the bit of higher weight on the right, the table of the ANF equals the concatenation of the ANFs of the $(n-1)$-variable functions $f(x_1, \cdots, x_{n-1}, 0)$ and $f(x_1, \cdots, x_{n-1}, 0) \oplus f(x_1, \cdots, x_{n-1}, 1)$.

- Fast Möbius Transform computation steps:

(i) Write the truth-table of f, in which the binary vectors of length n are in lexicographic order as described above;

(ii) Replace the values of f_1 by those of $f_0 \oplus f_1$; where f_0 and f_1 be the restrictions of f to $F_2^{n-1} \times \{0\}$ and $F_2^{n-1} \times \{1\}$ respectively, in other words; the truth-table of f_0 (resp. f_1) corresponds to the upper (resp. lower) half of the truth-table of f.

(iii) Apply recursively step 2, separately to the functions now obtained in the places of f_0 and f_1.

When the algorithm ends (i.e. when it arrives to functions in one variable each), the global table gives the values of the ANF of f.

Illustrative Example 1: Assuming a Boolean function f of 3-variables defined in it's Disjunctive Form by $f(x_1,x_2,x_3) = \bar{x}_1\ \bar{x}_2\ x_3 + x_1\ \bar{x}_2\ x_3 + x_1\ x_2\ x_3$, it's Truth Table is given by columns 1 to 4 of Table 1 , ANF(f) can be computed through steps on columns 5 to 10.

Table 1: Illustrative Example of ANF of a Boolean function

x_1	x_2	x_3	$f(x)$					ANF	
0	0	0	0	f_0	0	f_0	0	f_0	0
0	0	1	1		1		1	f_1	1
0	1	0	0		0	f_1	0	f_0	0
0	1	1	0		0		1	f_1	1
1	0	0	0	f_1	0	f_0	0	f_0	0
1	0	1	1		0		0	f_1	0
1	1	0	0		0	f_1	0	f_0	0
1	1	1	1		1		1	f_1	1

The results of the table denoted as ANF of

$$f(x_1,x_2,x_3) = x_3 \oplus x_2\ x_3 \oplus x_1x_2x_3$$

3.2 Walsh Transform

The Walsh Transform is an application of another slightly more general transform called Hadamard-Walsh Transform which is a descendant of Discrete Fourier Transform which has been applied to cryptography by Xiao and Massey in [4].

For any Boolean function f its Walsh Transform $\hat{F}$ can be computed as follows:

$$\hat{F}(a) = \sum_{x \in F_2^n} (-1)^{f(x) \oplus a \cdot x}, a \in F_2^n \qquad (2)$$

Walsh Transform is the Discrete Fourier Transform of the sign function f_χ which is used as an indicator instead of Boolean function itself ($f_\chi(x) = -1^{f(x)}$). Walsh transform can be computed easily by a simple divide and conquer butterfly algorithm as mentioned in [1].

Hence, if in the truth-table of the function f (in this case f denotes sign function of f), the input vectors are ordered in lexicographic order with the bit of highest weight on the right, the Walsh transform ($\hat{F}$) table of f equals the concatenation of those of the Walsh transforms of the $(n-1)$-variable functions

$\psi_0(x) = f(x_1, \cdots, x_{n-1}, 0) + f(x_1, \cdots, x_{n-1}, 1)$,and
$\psi_1(x) = f(x_1, \cdots, x_{n-1}, 0) - f(x_1, \cdots, x_{n-1}, 1)$.

- Walsh Transform computation steps:

(i) Write the truth-table of f, in which the binary vectors of length n are in lexicographic order as described above.

(ii) Replace the values of f_0 by those of $f_0 + f_1$ and the values of f_1 by those of f_0 - f_1; where f_0 and f_1 be the restrictions of f to $F_2^{n-1} \times \{0\}$ and $F_2^{n-1} \times \{1\}$ respectively, in other words; the truth-table of f_0 (resp. f_1) corresponds to the upper (resp. lower) half of the truth-table of f.

(iii) Apply recursively step 2, separately to the functions now obtained in the places of f_0 and f_1.

When the algorithm ends (i.e. when it arrives to functions in one variable each), the global table gives the values of the Walsh transform ($\hat{F}$) of the Boolean function *f*.

Illustrative Example 2: Assuming a 4-variables Boolean function *f* of the Truth Table as given in Table 2 columns 1 to 5, it can be computed by steps on columns 6 through 10.

The Walsh Transform values are called *spectral coefficients* [4], which are up to a constant.

Table 2: Illustrative Example of the Walsh Transform of a Boolean function

x_1	x_2	x_3	x_4	$f(x)$	$(-1)^{f(x)}$				$\hat{F}$
0	0	0	0	1	-1	0	2	0	0
0	0	0	1	1	-1	0	-2	0	0
0	0	1	0	1	-1	-2	-2	4	0
0	0	1	1	0	1	2	2	-4	8
0	1	0	0	0	1	2	-2	-4	0
0	1	0	1	1	-1	-2	2	4	-8
0	1	1	0	0	1	0	-2	0	0
0	1	1	1	0	1	0	2	0	0
1	0	0	0	0	1	-2	-2	0	0
1	0	0	1	0	1	-2	-2	0	0
1	0	1	0	1	-1	0	2	-4	-8
1	0	1	1	0	1	0	2	-4	0
1	1	0	0	0	1	0	-2	-4	-8
1	1	0	1	1	-1	0	-2	-4	0
1	1	1	0	1	-1	2	-2	0	0
1	1	1	1	1	-1	2	-2	0	0

Those values play important role in Boolean functions cryptographic analysis especially the first element and the maximum element of all the spectral coefficients.

4 BOOLEAN FUNCTIONS CRYPTOGRAPHIC CRITERIA

Most of the cryptographic criteria of Boolean functions rely on the finding Algebraic Normal Form and Walsh Transform which are mentioned above, there are a lot of criteria which have been studied broadly, but those criteria that have been mentioned are fundamental and most applicable as follows:

4.1 Rehearsal

- Number of Boolean Functions of n-variables is 2^{2^n}.
- Scalar Boolean Function is a mapping as follows
$$f(x) : F_2^n \rightarrow F_2$$
- Hamming Weight of a Boolean vector *x* is the number of non-zero coordinates of *x*.
$$\mathcal{W}_H(x) = |\{ i \in N \mid x_i \neq 0 \}| \quad (3)$$
- Hamming Weight of a Boolean Function *f* is the number of non-zero outputs of *f*
$$\mathcal{W}_H(f) = |\{ x \in F_2^n \mid f(x)_i \neq 0 \}| \quad (4)$$
- Hamming Distance between two functions *f* and g is denoted by d_H as mentioned in [1] as follows
$$d_H(f, g) = |\{ x \in F_2^n \mid f(x) \neq g(x) \}| \quad (5)$$
$$d_H(f, g) = \mathcal{W}_H(f \oplus g) \quad (6)$$

4.2 Balancedness

It is a necessary property of good cryptographic Boolean functions which guarantees fair (unbiased) output expectancy. A Boolean Function is balanced the its output is equally distributed among 0's and 1's which is mathematically defined as follows, An *n*-variable Boolean Function $f(x)$ is balanced if and only if:

$$|\{ x \in F_2^n \mid f(x)_i \neq 0 \}| = |\{ x \in F_2^n \mid f(x)_i = 0 \}| \quad (7)$$

$\Rightarrow$ If $\mathcal{W}_H(f) = 2^{n-1}$ then f is balanced.

Also, practically Balancedness can be computed if the first element of the Walsh Transform of a function f is 0, therefore the function is balanced.

i.e. $\hat{F}(0) = 0 \Rightarrow f$ is Balanced.

4.3 Algebraic Degree of a Boolean Function Balancedness

It is the degree of the ANF of the function; it is also called Non-Linear order of the function [1], but not to be confused with Non-Linearity as will be studied afterwards. Simply, the Algebraic Degree $d^\circ f$ is the number of variables in the highest order monomial with nonzero coefficient of the ANF of the function f.

e.g.: let $f(x_1,x_2,x_3) = x_3 \oplus x_2 x_3 \oplus x_1x_2x_3$ be the ANF of function $f \implies d^\circ f = 3$

4.4 Non Linearity of Boolean Function

It is the minimum Hamming distance between a Boolean Function f and all affine functions in F_2^n. A Boolean Function must be highly non-linear to provide confusion and resistance against Fast Algebraic Attack [1].

$$N_f = \min_{\phi \in \mathcal{A}_n} d_H(f, \phi), \qquad (8)$$

Where $\mathcal{A}_n$ is the set of all affine functions on F_2^n.

It can be found practically by the maximum absolute value of the Walsh transform of the function f as follows:

$$N_f = 2^{n-1} - \frac{1}{2} \max |\hat{F}| \qquad (9)$$

But the issue is whether this non-linearity is high enough or not, the answer could be found by upper bound of non-linearity, which is unknown for Boolean functions number of variables more than 8 [1].

For Boolean Function f of number of variables n:

$$\text{If n is even} \Rightarrow N_f \le 2^{n-1} - 2^{\frac{n}{2}-1} \qquad (10)$$

$$\text{If n is odd} \Rightarrow N_f \le 2^{n-1} - 2^{\frac{n-1}{2}} \qquad (11)$$

4.5 Algebraic Degree of a Boolean Function Balancedness

Algebraic Immunity is a new term in assessment of cryptographic Boolean functions that has been introduced as factor proof resistance against FastAlgebraic Attack. For a Boolean function f a Boolean function g can be found where $f \times g = 0$ such that $g \neq 0$, and multiplication is Hadamard product, the function g is called annihilator of function f. The minimum algebraic degree of the set of annihilators of the function f is called Algebraic Immunity and denoted by $AI(f)$. The upper bound of Algebraic Immunity of Boolean functions of n input variables is $\lceil n/2 \rceil$ [1].

4.6 Resiliency

A balanced n-variable function f is called m-resilient (n and m are positive integers such that $m<n$) if any of its output sub-functions obtained by fixing at most m of its input coordinates is balanced. Practically, any n variable Boolean Function f is m-resilient if and only if f is balanced, and $\hat{F}(u) = 0$ for all $u \in F_2^n$ such that $\mathcal{W}_H(u) \le m$. In Illustrative Example 2 since all vectors of Hamming weight 1 have $\hat{F} = 0$, therefore f is 1-resilient, rather than it is not 2-resilient i.e. there does not exist a correlation between the function output and a single bit of its input, but if exist for at least 2 bits of its inputs which makes it subjected to Correlation attack as stated by Siegenthaler. Another example is the function that is used in the design of the stream cipher GRAIN which is

$$\begin{aligned} g(x) = 1 + x^{32} + x^{37} + x^{72} + x^{102} + x^{128} \\ + x^{44}x^{60} + x^{61}x^{125} + x^{63}x^{67} \\ + x^{69}x^{101} + x^{80}x^{88} + x^{110}x^{111} \\ + x^{115}x^{117} \end{aligned}$$

The function $g(x)$ is balanced highly non-linear function with non-linearity 260096 and resiliency 4.

4.7 Correlation Immunity

A Boolean Function Correlation Immunity is the same as resiliency, but the condition of balancedness for the function itself is not necessary.

i.e. Boolean Function f is m-correlation immune if and only if $\hat{F}(u) = 0$ for all $u \in F_2^n$ such that $\mathcal{W}_H(u) \le m$. The conclusion is also the same but with different nomenclature.

4.8 Boolean Function Derivative

Derivative of a Boolean function is necessary for computing Propagation Criteria and Strict Avalanche Criteria that will be mentioned afterwards. For any n-variable Boolean function f and any vector b in F_2^n. Derivative of f in the

direction of b the Boolean function is denoted by $D_b f$

$$D_b f(x) = f(x) \oplus f(x+b) \qquad (12)$$

4.9 Propagation Criteria

The Propagation Criteria (*PC*) was introduced in cryptography by Preneel et al. in [5]. This property describes the behavior of a Boolean Function when one/some of its inputs is/are changed. So satisfying high levels, diffusion principal is achieved by a Boolean Function [1].

A Boolean Function f is said to satisfy the propagation criterion PC(k), if the Derivative of a Boolean function $D_b f(x) = f(x) \oplus f(x+b)$ is balanced for all $\{d \in F_2^n \mid 1 \leq \mathcal{W}_H(d) \leq k\}$ [5]. Practically PC can be studied by computing the Distributed Difference Table, which can be computed for a given Boolean Function f by squaring the Walsh Transform of the Function and Applying Inverse Walsh Transform of the result, the output is studied then starting with d (input vectors) of Hamming Weight 1 (Ignoring the first element in the table).

Illustrative Example 3: From function mentioned in Illustrative Example 2 the Walsh Transform of the function f is obtained, so the result can be directly squared and then the inverse Walsh transform of this squared result is computed. After that the output (assuming k=1) can be studied with respect to the Hamming Weight of the input vectors (d) respectively, i.e. those vectors of Hamming Weight 1 come first, if they succeed (all corresponding values are 0), it means that k=1 succeeds, so f satisfies PC(1), then proceed to those vectors of Hamming Weight 2, …etc. If the output of one vector of a certain weight is not zero therefore PC fails for this value of k, and the previous value of k is retained.

In this Illustrative Example the results of the Difference Table for all input vectors of Hamming Weight 1 equals to 0, therefore this function satisfies PC(1).

4.10 Strict Avalanche Criteria

Strict Avalanche Criteria (SAC) was introduced by Webster and Tavares in [6]; it corresponds to PC (1), i.e. if a Boolean Function satisfies Propagation Criteria of 1, therefore it satisfies Strict Avalanche Criteria, e.g. the function used in Illustrative Example 3 satisfies SAC.

4.11 Autocorrelation of a Boolean function

Differential cryptanalysis is one of the bases of a proven secure cipher algorithm. It relies on constructing a Table that will be discussed later on which is named Difference Distribution Table for certain Boolean functions.

Autocorrelation of Boolean function corresponds to constructing Distributed Difference Table. Autocorrelation of a Boolean function f is defined in [3] as follows:

$$r_f(a) = \sum_{x \in F_2^n} f(x).f(x \oplus a) \qquad (13)$$

Also,

$$r_f = \hat{F}^{-1}(\hat{F}^2(f)). \qquad (14)$$

5 VECTORIAL BOOLEAN FUNCTIONS

A Vectorial Boolean function $\boldsymbol{F}(x) : F_2^n \rightarrow F_2^m$ is a function that maps from vector space F_2^n (finite field) of all binary vectors of length n to the vector space to the vector space F_2^m of all binary vector of length m, such that m > 1. Vectorial Boolean functions are denoted by (n, m)-functions are used mostly in building Substitution Boxes (S-Boxes) which provide non linearity and confusion to cipher algorithms, especially block ciphers. The Scalar Boolean functions $f_1, \ldots, f_m$ are called co-ordinate functions of Vectorial Boolean function $\boldsymbol{F}(x)$, and defined by, for every input $x \in F_2^n$, $\boldsymbol{F}(x)=(f_1(x), \ldots, f_m(x))$. Any linear combination of the co-ordinate functions is called component function of $\boldsymbol{F}$ and denoted by $\boldsymbol{v.F}$, where $v \in F_2^n$ (implicitly co-ordinate functions are included in the set of components functions). Component functions are mainly used to describe the cryptographic criteria of Vectorial Boolean Functions.

5.1 Characteristic Function

The characteristic function $\boldsymbol{\chi_F}$ of Vectorial Boolean (n, m)-function $\boldsymbol{F}$ is a scalar Boolean function of (n+m) number of variables and denoted by

$$\boldsymbol{\chi_F}(y \parallel x) = \begin{cases} 1, & \text{if } y = \boldsymbol{F}(x) \text{ and} \\ 0, & \text{otherwise.} \end{cases} \qquad (15)$$

where, $y \parallel x$ is a decomposition of the input to $\boldsymbol{\chi_F}$ into a lower part x on n-bits and an upper part y on m-bits.

5.2 Algebraic Degree

The Algebraic Degree of a Vectorial Boolean (n, m)-function $\boldsymbol{F}(x)$ is equal to the maximal Algebraic Degree of its component functions.

5.3 Balancedness

Vectorial Boolean Functions Balancedness plays important role in cryptography. An (n, m)-function $\boldsymbol{F}$ is called balanced if and only if its component functions are balanced.

5.4 Nonlinearity

The nonlinearity $nl(\boldsymbol{F})$ of an (n, m)-function $\boldsymbol{F}$ is the minimum nonlinearity of all the component functions of $\boldsymbol{F}$.

5.5 Algebraic Immunity

The Algebraic Immunity of a Vectorial Boolean (n, m)-function is equal to the minimum Algebraic Immunity of its component functions [7].

Table 3: Calculating Distributed Difference Table of a Boolean Function

x_1	x_2	x_3	x_4	$f(x)$	$\hat{F}$	$\hat{F}^2$				$\hat{F}^{-1}(\hat{F}^2)$
0	0	0	0	1	0	0	0	16	16	16
0	0	0	1	1	0	0	0	16	16	0
0	0	1	0	1	0	0	32	16	0	0
0	0	1	1	0	8	64	32	16	0	0
0	1	0	0	0	0	0	32	-16	0	0
0	1	0	1	1	-8	64	32	-16	0	0
0	1	1	0	0	0	0	0	16	-16	-16
0	1	1	1	0	0	0	0	16	-16	0
1	0	0	0	0	0	0	0	-16	-16	0
1	0	0	1	0	0	0	0	16	16	-16
1	0	1	0	1	-8	64	-32	-16	0	0
1	0	1	1	0	0	0	32	16	0	0
1	1	0	0	0	-8	64	-32	16	0	0
1	1	0	1	1	0	0	32	-16	0	0
1	1	1	0	1	0	0	0	-16	16	0
1	1	1	1	1	0	0	0	16	-16	16

5.6 Walsh Transform

Walsh Transform of Vectorial Boolean Function $\boldsymbol{F}$ can be performed theoretically by mean of the characteristic function of $\boldsymbol{F}$ which is a scalar Boolean function of (n+m) number of variables, or by mean of its component functions which is practically mostly used.

5.7 Vectorial Boolean Functions Applications

S-Boxes are parts of iterative Block Ciphers and they play a central role in their robustness. Iterative Block Ciphers are the iterations of a transformation depending on a key over each block of plaintext. The iterations are called rounds and the key used in a single iteration is called Round Key. The Round Keys are computed from the secret key (called the Master Key) by a key scheduling algorithm.

The rounds consist of Vectorial Boolean Functions combined in different ways involving the Round Key. The most important function used in constructing a round function is the Substitution Box (denoted by S-Box). The S-Box is the only source of nonlinearity in any cipher. This nonlinearity property strictly adheres to propagation criteria property, so together the diffusion property is fulfilled.

Whenever S-Boxes are to be used, nonlinearity should be the most important criteria to be kept in consideration. Two families of nonlinear Vectorial Boolean Functions are of great interest in S-Boxes design due to their nonlinearity characteristics. Namely The Perfect Nonlinear functions (called Bent Functions) and Almost Perfect Nonlinear functions.

5.7.1 Bent Functions

A Boolean (n, m)-function $\boldsymbol{F}$ is called Bent Function if and only if the Walsh transform coefficients of $\widehat{\boldsymbol{F}}$ are all $\pm 2^{n/2}$, that is, $W(\hat{f})^2$ is constant. Bent Functions exist only for even dimensions (n-even), and only if m $\leq$ n/2. Obviously Bent functions satisfy the nonlinearity upper bound, due to their Walsh transform coefficients, but unfortunately Bent functions are surjective unbalanced functions, so they should be carefully used cipher design. But fortunately any *n*-variable Bent function satisfies PC(*n*).

Maiorana-McFarland Bent functions are the most famous construction of Bent functions used in S-Box design. It can be constructed as follows.

$$F(x,y) = L\big(x.\pi(y)\big) + H(y) \qquad (16)$$

Where $x.\pi(y)$ is calculated in $\mathrm{F}_{2^{n/2}}$, where L is any linear or affine mapping from $\mathrm{F}_{2^{n/2}}$ onto F_2^m, π is any permutation of $\mathrm{F}_{2^{n/2}}$, and H is any (n/2, m)-function.

5.7.2 Almost Perfect Nonlinear Functions

An (*n, m*)-function *F* is called Plateaued if and only if, every component function *v.F* is Plateaued, that is the values of its Walsh transform belong to the set $\{0, \pm\lambda_v\}$.

An (*n, m*)-function *F* is called Almost Perfect Nonlinear (APN) if and only if, every component function *v.F* is Plateaued, where

$$\boldsymbol{\lambda_v} = \begin{cases} 2^{(n+1)/2}, & \text{if } n \text{ is odd and} \\ 2^{\frac{n}{2}+1}, & \text{if } n \text{ is even.} \end{cases} \qquad (17)$$

An (*n, m*)-functions of the form $F(x)=x^d$, is called Power Function. If $d=2^n-2$, where *n* is odd, then $F(x)$ is called Inverse Function, which is used in the S-Box of the AES Block Cipher ($d=254$). Inverse Function is an APN function which is Balanced and Bijective, so it satisfies very good cryptographic criteria.

6 CONCLUSION

The state of the art in Boolean functions representation and cryptographic assessment have been presented, they should be considered as a starting point in cryptographic algorithm design and analysis, namely block ciphers, stream ciphers, and also hash functions. Moreover; well-known algorithms that were provided to represent functions such as Moebius Transform which is used in finding Algebraic Normal Form of a given truth table of any function, and also calculating Walsh transform of any given Boolean function, which have been shown to be a principal pillar in establishing a successful cryptographic function assessment.

7 REFERENCES

[1] Claude Carlet, "Boolean Functions for Cryptography and Error Correcting Codes," in Boolean Models and Methods in Mathematics, Computer Science, and Engineering.: Cambridge University Press, 2010, pp. pp. 257-397.

[2] Thomas W. Cusick and Pantelimon Stănică, Cryptographic Boolean Functions and Applications, Fisrt ed.: Academic Press is an imprint of Elsevier, 2009.

[3] Antoine Joux, Algorithmic Cryptanalysis.: Chapman & Hall/CRC Cryptography and Network Security Series, 2012.

[4] Guo-Zhen Xiao and James L. Massey, "A Spectral Charaterization of Correlation Immune Combinig Functions," in IEEE Transactions on Information Theory, 3rd ed., 1988, vol. 34, pp. pp. 569-571.

[5] W. Van Leekwijck, L. Van Linden, R. Govaerts, and J. Vandewalle B. Preneel, "Propagation Characteristic of Boolean Functions," in EuroCrypt 90 Proceedings, Advances in Cryptology.: Springer Verlag, 1991, pp. pp. 161-173.

[6] Melek M. Yucel, "Introduction to Cryptography - Course IAM-501 material, Inernet," in Institute of Applied Mathematics, Middle East Technical University, Ankara, Turkey.

[7] Claude Carlet, "Vectorial Boolean Functions for Cryptography," in Boolean Models and Methods in Mathematics, Computer Science, and Engineering.: Cambridge University Press, 2010, pp. pp. 398-469.