AUSTRALASIAN JOURNAL OF COMBINATORICS
Volume **29** (2004), Pages 231–247

Simpler methods for generating better Boolean functions with good cryptographic properties

L. Burnett, W. Millan, E. Dawson and A. Clark

Information Security Research Centre,
Queensland University of Technology,
GPO Box 2434, Brisbane
Queensland 4001
Australia
{ld.burnett, b.millan, e.dawson, a.clark}@qut.edu.au

Abstract

We present two heuristic optimisation methods for generating N-variable boolean functions which exhibit particular cryptographic properties. Each method targets one or more properties and consistently produces a large number of N-variable boolean functions with those properties. The first method presented is shown to outperform any other heuristic technique previously reported, in terms of generating highly nonlinear, low autocorrelation balanced boolean functions. The second method discussed in this paper again outperforms any other existing heuristic technique used to generate resilient functions with high nonlinearity and algebraic degree maximising Siegenthaler's inequality.

1 Introduction

The extensive research on the analysis of boolean functions which has taken place in recent years has been important in both the design and cryptanalysis of cipher systems. Boolean functions exhibiting strong cryptographic properties have become an essential part of secure symmetric cipher systems. Not only is it common for block and stream ciphers to incorporate boolean functions into their design, but they often also form the foundation of cryptographic hash functions.

Modern symmetric block ciphers such as the Advanced Encryption Standard, Rijndael [13], depend on their component s-boxes for complexity largely in the form of diffusion and nonlinearity. In turn, the strength of these s-boxes is derived from the boolean functions comprising them. The stronger the properties exhibited by the combined boolean functions, the stronger the formed s-box will be. A number of stream ciphers (for example [4], [16]) use boolean functions to combine the output of multiple linear feedback shift registers. These combining boolean functions must simultaneously exhibit high nonlinearity and a degree of correlation immunity which reflects an acceptable tradeoff of these properties in order for the stream cipher to be considered resistant to correlation attacks such as the divide-and-conquer attack [20]. Further, boolean functions with low autocorrelation can reduce the likelihood of a stream cipher exhibiting predictable periodic sequences in the output. This is of particular significance to self-synchronising stream ciphers and stream ciphers with memory [6] to resist resynchronisation attacks. In some cryptographic hash functions, boolean functions are utilised in the round function. The use of highly nonlinear boolean functions can prevent the hashed output exhibiting excessive degrees of bit bias. This minimises the occurrence of collisions in the output of the hash function.

For many years, heuristic techniques have been shown to be an effective and flexible way of generating strong boolean functions and s-boxes. Heuristic techniques involve directed search methods and have been successful in not only finding functions with "good" properties but also are able to produce a large number of such functions. Some of the best known techniques for heuristic optimisation include Hill Climbing, Genetic Algorithms, and Simulated Annealing. Examples of significant results produced by these methods can be found in [11], [12], [3] respectively.

Among the advantages of heuristic techniques over algebraic constructions are the ability of heuristics to generate a large number of boolean functions with the desired properties. Further, the non-deterministic nature of heuristic techniques results in an inherent randomness that tends to generate predominantly strong functions with a complex structure. Whilst algebraic constructions are typically able to produce optimal functions, the basis of some algebraic constructions result in weaker functions with a structure which is more easily cryptanalysed.

In this paper we present two simple heuristic methods which have proven to be powerful tools for generating a large number of balanced boolean functions with target cryptographic properties for enhancing security. These methods provide researchers with alternative means for generating strong boolean functions for applications which require the existence of the cryptographic properties they are able to produce. The first heuristic is a very simple method that outperforms other currently known heuristics in that it is capable of discovering balanced boolean functions with better combinations of the properties of high nonlinearity and low autocorrelation that are currently achievable by other heuristics. As an example, the first method presented easily generates many 8-variable balanced boolean functions with a nonlinearity of 116 and a maximum autocorrelation value of 16, which have not previously been found directly by heuristic techniques. This first method generates a large number of boolean functions with good properties in a short period of time.

The second method presented in this paper is easily able to generate known optimal m-resilient boolean functions with high nonlinearity, some of which other heuristic techniques have been incapable of generating. This method was able to successfully produce a large number of examples of boolean functions with almost all of the optimal cryptographic property combinations: high nonlinearity, varying degrees of resilience and maximum algebraic degree, that have been discovered. The results obtained are particularly significant given that the algorithm used is substantially simpler than existing heuristic techniques. In addition, this method is able to efficiently generate a wide range of functions with these optimal combinations of properties directly within a single method and without requiring linear transformations to achieve any of its varying degrees of resilience.

This paper is organised in the following manner. Necessary definitions and a description of the notation used can be found in Section 2. Section 3 describes a method for generating highly nonlinear balanced even-dimensional boolean functions with low autocorrelation and presents experimental results which demonstrate that this method is at least equal to, and in some instances superior to any other reported heuristic technique. In Section 4, a second method for generating optimised resilient boolean functions for $5 \leq N \leq 9$ is proposed, with experimental results supporting the value of this method in generating highly nonlinear resilient boolean functions. In Section 5 we draw conclusions about the methods we have developed and make some suggestions for future work which could be done in this area.

2 Preliminaries

We now introduce the notation used throughout this paper and present some necessary and well established definitions.

2.1 Notation and Well Known Definitions

A single output N-variable boolean function is a mapping from N bits to a single bit. An NxM s-box is a mapping from N input bits to M output bits. Each of the M-bit binary output vectors is a boolean function containing 2^N elements. The truth table representation of an N-variable boolean function $f(x)$ is a vector containing 2^N elements, each element $\in \{0, 1\}$. A more convenient form of a boolean function is its polarity truth table representation, denoted by $\hat{f}(x)$, containing 2^N elements $\in$ $\{1, -1\}$. $\hat{f}(x)$ may be derived from $f(x)$ by the equation, $\hat{f}(x) = 1 - 2f(x)$.

The Algebraic Normal Form (ANF) is another representation of a boolean function. The coefficients $a_i \in \{0, 1\}$ $(i = 0,..,N)$ form the elements of the ANF of $f(x)$, an N-variable boolean function. Every ANF representation corresponds to a unique boolean function truth table. The Algebraic Normal Form of a boolean function of N variables is written in the form:

$$\begin{aligned} f(x) &= a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus ... \oplus a_Nx_N \oplus a_{12}x_1x_2 \oplus a_{13}x_1x_3 \oplus ... \oplus \\ & \quad a_{(N-1)N}x_{N-1}x_N \oplus ... \oplus a_{123...N}x_1x_2x_3...x_N. \end{aligned}$$

The algebraic degree, or simply degree, of a boolean function is the order of the largest product term in the function's ANF, which we shall denote by *deg*. The well known inequality due to Siegenthaler [20] gives an upper bound on the degree of a boolean function as it relates to the number of input variables, N, and the degree of correlation immunity, m. This has been defined as

$$N \geq m + d + \epsilon \text{ where } \epsilon = \begin{cases} 0 & \text{if function is balanced} \\ 1 & \text{if function is unbalanced} \end{cases}$$

The Hamming weight of a boolean function $f(x)$ of N variables denoted $hw(f)$ is defined as $\Sigma_{x=0}^{2^N-1} f(x)$, the number of ones in its truth table. The Hamming distance between two boolean functions $f(x)$ and $g(x)$, denoted by $hd(f, g)$, is the number of differing elements in corresponding positions between the two truth tables.

A boolean function of the form $f(x) = a_0 \oplus a_1x_1 \oplus a_2x_2 \oplus ... \oplus a_Nx_N$ where a_i (i = 0,...,N) $\in \{0, 1\}$ is called an affine function. If $a_0 = 0$, then the function is a linear function. A linear transformation on an N-variable boolean function, $f(x)$, is defined as the resultant function, $g(x)$, being produced by replacing the input vector x with the product of x with the transformation matrix, A, where A is an NxN non-singular binary matrix, expressed as $g(x) = f(Ax)$. The Walsh Hadamard Transform (WHT) of the polarity truth table of a boolean function $\hat{f}(x)$, denoted by $\hat{F}(\omega)$, is a measure of the correlation between a function and the set of all affine functions. It is defined by the equation $\hat{F}(\omega) = \Sigma_x \hat{f}(x)\hat{L}_\omega(x)$ where $\hat{L}_\omega(x)$ is the linear function defined by $\omega \in \{1,\text{-}1\}$.

If a function is boolean, then the values in its WHT vector must satisfy Parseval's Equation:

$$\Sigma_\omega \left(\hat{F}(\omega)\right)^2 = 2^{2N}$$

The converse is not always true. That is, if a function satisfies Parseval's Equation, it may not necessarily be boolean.

We denote the concatenation of two (N-1)-variable boolean functions, $f(x)$ and $g(x)$, to form an N-variable boolean function, $h(x)$, by $h(x) = f(x) \parallel g(x)$ (or conventionally $f \parallel g$). In terms of the Algebraic Normal Form, the concatenation, $h(x) = f(x) \parallel g(x)$, can be expressed as:

$$\begin{aligned} h(x_1, x_2, .., x_N) &= (1 \oplus x_N)f(x_1, .., x_{N-1}) \oplus x_N g(x_1, .., x_{N-1}) \\ &= f(x) \oplus x_N(f(x) \oplus g(x)) \end{aligned}$$

The Walsh Hadamard Transform of $h(x)$ can be achieved computationally by the following process:

$$\begin{aligned} \hat{H}(\omega) &= \hat{F}(\omega) + \hat{G}(\omega) \ for \ \omega \in \{0, .., 2^{N-1} - 1\} \\ \hat{H}(\omega + 2^{N-1}) &= \hat{F}(\omega) - \hat{G}(\omega) \ for \ \omega \in \{0, .., 2^{N-1} - 1\} \end{aligned}$$

A bent function is an unbalanced N-variable boolean function (N even) which achieves the maximum possible nonlinearity of 2^{N-1} - $2^{\frac{N}{2}-1}$. The hamming weight of a bent function is $2^{N-1} \pm 2^{\frac{N}{2}-1}$ and all bent functions have degree $\leq \frac{N}{2}$ for $N >$ 2.

2.2 Some Properties of Boolean Functions

We now define the cryptographic properties of boolean functions that will be of interest to this paper.

Autocorrelation: Denote the autocorrelation function of $\hat{f}(x)$ (the polarity truth table of $f(x)$) by $\hat{r}(\alpha)$. This function may be written as

$\hat{r}(\alpha) = \Sigma_x \hat{f}(x)\hat{f}(x \oplus \alpha)$ with α and $x \in \{0,..,2^N - 1\}$ and $\hat{r}(0) = 2^N$.

From the autocorrelation function (AC), we define the measure commonly referred to as the absolute indicator or maximum absolute autocorrelation value, as:

$|AC_{max}| = \max |\hat{r}(\alpha)| \; \alpha \in \{1,..,2^N - 1\}$

Hereinafter in this paper we shall simply use AC_{max} to refer to the maximum absolute autocorrelation value.

The sum-of-square indicator, also derived from the autocorrelation function, may be calculated as follows:

$\sigma = \Sigma_\alpha \hat{r}^2(\alpha) \quad \alpha \in \{0,..,2^N - 1\}$

Balance: An N-variable boolean function $f(x)$ is said to be balanced if $hw(f) = 2^{N-1}$, or, $\#\{\text{x} \mid f(x) = 0\} = \#\{\text{x} \mid f(x) = 1\}$.

Correlation Immunity: An N-variable boolean function $f(x)$ is m^{th}-order correlation immune, denoted by $CI(m)$, if, for every ω such that $1 \leq hw(\omega) \leq m$, $\hat{F}(\omega) = 0$. The effect of a linear transformation on the Walsh Hadamard Transform of a boolean function is that it permutes the values in the Walsh Hadamard Transform but does not change the magnitude of those values. Thus, where a boolean function is m^{th}-order correlation immune ($m \leq N - 2$), it may be possible to increase the degree of correlation immunity by performing a linear transformation on the boolean function.

Global Avalanche Criteria (GAC): The concept of GAC was first proposed by Zhang and Zheng [21] whereby the overall avalanche characteristic of a boolean function is evaluated with the use of the absolute indicator (or maximum absolute autocorrelation value), as well as the sum-of-square indicator.

Nonlinearity: The nonlinearity of a boolean function $f(x)$, denoted by NL_f, is the minimum hamming distance to the set of all affine functions. Mathematically, the relationship between the nonlinearity of a boolean function $f(x)$ and the Walsh Hadamard Transform of the function is given by the following equation:

$NL_f = \frac{1}{2} \text{ x } (2^N - WHT_{max})$

where WHT_{max} represents the maximum absolute value of the Walsh Hadamard Transform.

Resilience: An N-variable boolean function which is both balanced and m^{th}-order correlation immune is known as an m-resilient boolean function.

Strict Avalanche Criteria (SAC): An N-variable boolean function $f(x)$ is said to satisfy strict avalanche criteria if, for every s such that $hw(s) = 1$, $\Sigma_x f(x) \oplus f(x \oplus s) = 2^{N-1}$. Alternatively, the strict avalanche criteria is said to be satisfied for a boolean function, $f(x)$, iff the autocorrelation function of $\hat{f}(x)$ contains zero values in all positions s where $hw(s) = 1$.

3 Generation of Highly Nonlinear Balanced Boolean Functions with Low Autocorrelation, N even

In this section, we describe the method which we use for generating even-dimensional highly nonlinear balanced boolean functions with low autocorrelation. This is referred to as “Method 1”. The importance of each of these properties relates to a quantifiable measure of the resistence of boolean functions to various forms of cryptanalysis. A high measure of nonlinearity is required for a higher probability of resisting linear approximation attacks. The balance property ensures that no bitwise bias in the truth table of a boolean function exists. Low autocorrelation enhances the difficulty of approximating a boolean function's first order derivatives. In particular, the existence of these combined properties provides a higher degree of resistance to linear and differential cryptanalytic attacks [1, 9].

The basic principle of Method 1 is to concentrate the heuristic search process to regions of the N-dimensional input space (N even) that are expected to exhibit high nonlinearity. By definition, these regions comprise functions that are at greater distances from the affine set of boolean functions. The significance of concentrating the search in this manner becomes important for efficiency as we consider larger and larger dimensions of N input variables.

As with all heuristic methods, particularly for the purposes of efficiency, the first important step is to always begin with a good starting function. This should be one which has reasonably good fitness in the hope that subsequent steps of the method will “discover” better functions with an improved fitness measure. In the case of Method 1, the targeted fitness measure is nonlinearity. Many heuristic techniques simply initiate the process by choosing a starting function at random, or by iteration to a randomly chosen function with a nonlinearity measure above a reasonable value, specified by the code. The approach which we have taken is to indeed begin with a good starting function, although rather than hope to get, or iterate to, a reasonably good starting function, we randomly choose a starting function known to have the maximum nonlinearity value. In the case of an even dimensional input space, this function will be a bent or perfect nonlinear function. Bent functions only exist for even-dimensional N. Thus, in Method 1, for even N, we select our starting function randomly from a constructed subset of N-variable bent functions. In addition to achieving maximal nonlinearity, bent functions have the property of zero autocorrelation. This makes them an obvious choice of starting function when trying to achieve high nonlinearity and low autocorrelation. As Method 1 makes small incremental changes to the starting bent function, the resultant boolean function from Method 1 possesses a similar structure and like characteristics to the chosen starting bent function.

Bent functions are not weight balanced and have a hamming weight of 2^{N-1} $\pm$ $2^{\frac{N}{2}-1}$. As balance is an important property for cryptographic applications, we seek to achieve balance while retaining high nonlinearity and low autocorrelation. In Method 1, this is done by progressively setting or clearing k-bits at a time, and retaining resultant functions which remain above a certain nonlinearity threshold. It is a well known fact that a one bit change in the truth table of a boolean function results in a change to the Walsh Hadamard Transform of the function $\in \{2,\text{-}2\}$. This in turn produces the effect of a small increase or decrease to the nonlinearity of the function. The key factor of Method 1 is the generation of boolean functions with a specified hamming distance from the original starting function and subsequently from retained functions. By doing so, we are maintaining much of the distance needed to provide us with boolean functions with high nonlinearity and low autocorrelation. The degree of bit changing, which depends on the imposed hamming distance/s, allows a branching effect to take place.

Define *MAXCHANGES* to be the maximum number of distinct k-bit changes to be considered for each function. Note that our experiments using Method 1 were performed by choosing $k = 2$. *MAXCHANGES* is fixed for each run of the program and it's most optimal value may be determined by experimentation. However, typically *MAXCHANGES* took a random value for each program run. NL_{min} is defined as the minimum nonlinearity value which is acceptable for the resulting boolean function. The basic algorithm for Method 1 is provided below:

1. Let Δ be a constructed subset of N-variable bent functions.
2. Select a random function $r = (r_0, r_1, ..., r_{2^N-1})$, where $r \in \Delta$.
3. Define $T0$ as the set of positions i, where $r_i = 0$, $T1$ as the set of positions j, where $r_j = 1$, and hw(r) as the hamming weight of r.
4. Call the REPLACE function.
5. Return to Step 1 to obtain more functions, if needed.

REPLACE function:

1. Check hw(r):
   (a) If hw$(r) < 2^{N-1}$, select $a_1 \in T0$, $a_2 \in T0$, where $a_1 \neq a_2$.
   (b) If hw$(r) > 2^{N-1}$, select $a_1 \in T1$, $a_2 \in T1$, where $a_1 \neq a_2$.
   (c) If hw$(r) = 2^{N-1}$ then selection criteria met; output r, NL, AC and exit REPLACE function.
2. Derive candidate function c from r, with $a_1 = 1 \oplus a_1$; $a_2 = 1 \oplus a_2$.
3. Check NL of c.
4. If $NL \geq NL_{min}$, call the REPLACE function recursively.
5. If $NL < NL_{min}$, discard c, return to Step 1 (to a limit of *MAXCHANGES* times).

In our experiments we have used bent functions based on the construction by Maiorana-McFarland [10]. Note, however, that any bent function construction technique may be used to generate the random function, r.

3.1 Results for Method 1

Trials of Method 1 successfully generate many examples of balanced N-variable (N even) highly nonlinear functions with low autocorrelation. Some of the best examples of boolean functions which we have generated using Method 1 are shown below:

N	NL	deg	AC_{max}	σ
6	26	5	16	6,784
8	116	7	16	89,728
10	488	8	40	1,268,608
10	488	9	40	1,272,448
12	2000	11	64	18,757,120
12	2002	11	64	18,776,704
14	8100	13	104	284,931,328
14	8102	13	104	284,891,392
14	8104	13	112	284,919,808

Table 1: Even-dimensional balanced boolean functions with optimal combination of above properties generated by Method 1. An example of boolean functions with selected property combinations from Table 1 can be found in the Appendix.

Table 2 provides the highest nonlinearity and lowest sum-of-square indicator combination of properties for even N, $6 \leq N \leq 14$.

N	NL	σ
6	26	6,784
8	116	86,656
10	488	1,262,464
12	2000	18,743,680
12	2002	18,754,048
14	8100	284,866,432
14	8102	284,891,392
14	8104	284,919,808

Table 2: Even-dimensional balanced boolean functions with optimal combination of nonlinearity and sum-of-square indicator generated by Method 1

In the table below we list the maximal nonlinearity values which we have achieved by Method 1 for balanced boolean functions in the range N = 6,..,14 inclusive (N even).

N	6	8	10	12	14
NL	26	116	488	2002	8104

Table 3: Highest nonlinearity found by Method 1 for balanced N-variable boolean functions, $6 \leq N \leq 14$, N even

We now discuss our results in comparison with related work in this area. In 2000, the authors of [2] used the heuristic techniques of simulated annealing and simulated annealing combined with hill climbing to produce the then best known results of balanced boolean functions with high nonlinearity and low autocorrelation. These properties can be directly compared to our results in Table 1. For $8 \leq N \leq 12$, N even, we have found a great number of N-variable balanced boolean functions with consistently lower AC_{max} values for identical values of nonlinearity. In addition, Method 1 is capable of finding functions which achieve lower AC_{max} values with higher nonlinearity than in [2]. A further comparison can be made, using Table 3, between the maximum nonlinearity they were able to achieve for balanced functions for $6 \leq N \leq 12$, N even, and the balanced boolean functions we were able to generate with this property using Method 1. For each number of even inputs N in this range, Method 1 either achieved an equal nonlinearity value or a higher nonlinearity value.

In 2002, Maitra [7] used a combination of computer search and algebraic construction to demonstrate a lower bound for the sum-of-square indicator than previously known. The paper gave specific results for N=6 and N=15 for the combined properties of balance, nonlinearity, absolute indicator, sum-of-square indicator, SAC and degree. We provide below an example truth table (in hex notation) of a 6-variable boolean function generated by Method 1 with the same combined properties but more optimal absolute indicator and sum-of-square indicator values:

Example 1: 6-variable balanced boolean function with deg = 5, NL = 26, AC_{max} = 16, σ = 6784 and satisfying SAC can be expressed as

04511b5e37e23e6d

In [7], a linear transformation was applied to their function in order to achieve the SAC property. This was not necessary for Method 1 as it was quickly able to generate examples of functions with the properties above satisfying SAC.

Further optimal results were obtained in 2002 [3] by combining simulated annealing, hill climbing and linear transformation. From Table 3 in Section 3 of their paper we are able to compare their combined balance, nonlinearity, degree, and AC_{max} properties for $6 \leq N \leq 12$ (N even) with the results obtained from our Method 1 in Table 1. For all even N in this range, Method 1 was able to at least equal their results for these properties, and for most values of $N \geq 8$ could improve on both the nonlinearity and AC_{max} values. In addition, the sum-of-square indicator values in our Table 2 above are as low or lower than those found in Table 4 of [3].

Method 1 has been successful in obtaining the best known combinations of even-dimensional balanced boolean functions with high nonlinearity and low autocorrelation, as well as successfully making improvements to a number of these best known combinations of boolean function properties. Furthermore, Method 1 is reasonably efficient and capable of generating a large number of these N-variable (N even) boolean functions with optimal properties. Method 1 initiates its search from functions with maximal nonlinearity and minimal autocorrelation. As a result, for these values of N, although the focus of the search is on nonlinearity, boolean functions with good autocorrelation are achieved with little effort.

4 Generation of Optimised Resilient Boolean Functions for $5 \leq N \leq 9$

In this section, we describe a method which we have designed for generating optimised resilient boolean functions. For this method, optimality is defined as the best known combination of balance, high nonlinearity and degree of correlation immunity, together with an algebraic degree which maximises Siegenthaler's inequality. In this paper this method is referred to as "Method 2". Resilient functions provide a high degree of tolerance against correlation attacks.

For Method 2, we operate in the Walsh Hadamard spectrum. This enables us to force the generation of functions that satisfy correlation immunity goals. Operating in the Walsh Hadamard spectrum also enables direct limiting of maximal values within the Walsh Hadamard vector, which has a direct relation to nonlinearity.

It is well known that the concatenation of two valid Walsh Hadamard vectors of dimension N results in a valid Walsh Hadamard vector of dimension $N + 1$. The building of higher dimension functions using step-by-step concatenation of their Walsh Hadamard Transform vectors forms the basis of Method 2. It is a trivial exercise to generate a complete list of all 4-variable (N=4) boolean functions and their characteristics. We use this as our starting pool for Method 2.

We define selection criteria independently for boolean functions, f and g, at each dimension N, for $N = 4$ to $targetN$, where $targetN$ is the dimension of the desired boolean functions. The selection criteria used for Method 2 are maximum Walsh Hadamard Transform vector value $WHTmax_N$, minimum non zero Walsh Hadamard Transform vector value $WHTmin_N$ and minimum degree of correlation immunity CI_N. It should be noted that the code for Method 2 automatically incorporates the balance property for any degree of correlation immunity ≥ 1.

We now describe the Method 2 algorithm:

1. Let L_4 be a set of T boolean functions, where N=4, that satisfy $WHTmax_{L_4}$, $WHTmin_{L_4}$ and CI_{L_4}. Let R_4 be a set of T boolean functions, where N=4, that satisfy $WHTmax_{R_4}$, $WHTmin_{R_4}$ and CI_{R_4}.

2. For $N = 5$ to $targetN$

    (a) Call the BUILD procedure(N, L_{N-1}, L_N)

    (b) Call the BUILD procedure(N, R_{N-1}, R_N)

BUILD procedure(N, S_{N-1}, S_N):

1. Select f where $f \in S_{N-1}$
2. Select g where $g \in S_{N-1}$
3. Concatenate the *WHTs* of f and g to form the *WHT* of an N-dimensional boolean function, h
4. Add h to the set S_N iff h satisfies $WHTmax_{S_N}$, $WHTmin_{S_N}$ and CI_{S_N}.
5. Return to Step 1. until the set S_N is of the desired size.

Many examples of the best known boolean function combinations of properties have been generated by this method, using a range of parameters for $WHTmax$, $WHTmin$, CI, and set sizes. The size of the sets are specified by the code. Generally, the larger the set size for a given N, the more $(N + 1)$-variable boolean functions satisfying the selection criteria are found. Clearly, it is more useful, to decrease the set size as N increases and tends to $targetN$, as the number of functions satisfying the selection criteria at each dimension N will be less than the number of functions in the set.

In Steps 1. and 2. of the BUILD procedure, a selection process to choose a function from the set occurs. Among the different selection functions which have been trialled for Method 2 are random selection and exhaustive pairing. Each of these processes is described below.

The random selection process allows a boolean function to be chosen randomly from each of the sets L_{N-1} and R_{N-1}. These two chosen boolean functions are concatenated to form an N-variable boolean function, which is retained if the selection criteria is met. Our experimental trials of Method 2 always employ random selection for at least $N = 5$ and 6 so that non-deterministic factors influence the computation.

The exhaustive pairing process may be used in the higher levels of Method 2 to ensure that all distinct pairings of boolean functions at $N - 1$ are tested for satisfaction of the selection criteria. This is a practical approach at this stage of Method 2 since the retained sets of boolean functions for higher N become smaller and exhaustive concatenation of pairs is not too computationally intensive. We typically used this process for the $targetN$ and $targetN - 1$ variable levels.

4.1 Results for Method 2

Previously, many different methods have had to be used (for example, [19], [18], [14], [8]) to generate these optimal boolean functions. We have demonstrated that Method 2 is, by itself, able to reproduce many examples of these best results within a single method. We present in Table 4 the best functions which our method was able to achieve in (N,m,deg,NL) notation, where N is the dimension, m is the degree of resilience, deg is the algebraic degree and NL represents the nonlinearity, of the function.

N	Optimal known functions
5	(5,1,3,12), (5,2,2,8)
6	(6,1,4,24), (6,2,3,24), (6,3,2,16)
7	(7,1,5,56), (7,2,4,56), (7,3,3,48), (7,4,2,32)
8	(8,1,6,112), (8,2,5,112), (8,3,4,112), (8,4,3,96), (8,5,2,64)
9	(9,1,7,240), (9,2,5,240), (9,3,5,224), (9,5,3,192), (9,6,2,128)

Table 4: Optimal combinations of properties currently known, which were able to be achieved by Method 2. Note, however, that (8,1,6,112) is not optimal for nonlinearity.

The set sizes for the starting pool of 4-variable functions are dependent on the number of functions which satisfy the selection criteria specified at that level. As the selection criteria can be varied for each program execution, these 4-variable set sizes will differ accordingly. For our experimental runs, typical set sizes for the lower dimensional functions (eg $N = 5$ and 6) were between 30,000 and 50,000. Set sizes for the higher dimensional functions ($N = 7$, 8 and 9) were typically between 1,000 to 10,000 but usually tending to 1,000 for the *targetN*-variable set. It is, however, a flexible parameter of the Method and may be decreased or increased beyond these values to achieve a balance between a desired number of final functions and the execution time.

Example truth tables (in hex notation) of some of the best known functions that have been obtained by Method 2 are listed below. We also set out below tables containing the typical parameters at each level of the concatenation process, $N = 4$ to *targetN*, which were used to achieve these results. The reader should note, however, that most of the property combinations are able to be achieved by a number of different sets of parameters.

Example 2: (7,2,4,56)

6369d82d56ac8b71499bb5c27a64863d

N	$WHTmax_N$	CI_N
4	8	0
5	8	1
6	16	2
7	16	2

Example 3: (8,1,6,112)

`54aa27d9eb324c562299f3ac4fe5341c813dc2f69f29f8054d9d78292cb2e356`

N	$WHTmax_N$	CI_N
4	8	0
5	8	0
6	16	1
7	32	1
8	32	1

Example 4: (9,2,5,240)

```
92cd3d629e31c16ea51d96b8d2a64b5692cd3d629e31c16ea51d96b8d2a64b56
d83c2769c32796d849b567a8ba49586727c3d8963cd86927b64a985745b6a798
```

N	$WHTmax_N$	$WHTmin_N$	CI_N
4	8	0	0
5	8	0	1
6	16	0	2
7	16	16	2
8	32	32	2
9	32	0	2

A comprehensive summary of much of the work which has been done to date on correlation immune and resilient functions can be found in [17]. The majority of recent research in the area of resilient functions have had to perform a linear transformation on the resulting boolean function in order to obtain some degree of correlation immunity. Method 2 uses concatenation of selected boolean functions to achieve the desired degree of correlation immunity and no linear transformations are required.

We compare our results with the directed search for resilient functions in [15]. The best resilient functions found were (8,1,6,112) and (9,1,6,236). Method 2 equals or surpasses the results obtained by this directed search. For example, Method 2 is able to find 9-variable resilient boolean functions with a nonlinearity of 240, whereas the directed search proposed in [13] found 9-variable resilient functions with nonlinearity at most 236. The authors of [15] report taking up to two weeks to generate these results, compared with a few hours at most for $N = 9$ and up to a few minutes for $N = 8$ for Method 2 to produce 1000 functions each with these properties.

In [3], resilient functions were generated using Simulated Annealing. Again, Method 2 was able to generate resilient functions that were at least equal to, and in some instances with better properties than those found by [3]. For example, Method 2 was easily able to achieve many (7,1,5,56) functions and 9-variable resilient functions with nonlinearity 240. It should be noted, however, that Method 2 does not measure autocorrelation in its computation.

Method 2 is more suited to discovering functions where $targetN \leq 9$. The reason for this is that as the concatenation process commences from a pool of $N = 4$ boolean functions, as N increases, a larger $targetN$ will become increasingly computationally intensive. Another limitation of Method 2 is that for larger $targetN$ and small m, functions which possess multi-valued Walsh Hadamard Transform vector values are increasingly difficult to generate. Regardless, Method 2 has demonstrated the ability to successfully and quickly generate the majority of currently known functions with optimal combinations of properties, and a large number of them.

5 Conclusions

We have developed two simple heuristic methods to enable the rapid generation of many different boolean functions with good or optimal cryptographic properties. We have demonstrated that Method 1 successfully generates optimal high nonlinearity, low autocorrelation, even-dimensional functions for $6 \leq N \leq 14$, some of which are the best known for these combination of properties. The computational effort required for Method 1 ranged from a few seconds for $N = 6$ to as little as 30 minutes for $N = 14$. We have also shown that Method 2 enables the generation of highly nonlinear resilient functions with a range of degrees of resilience for each N in the experiments for $N \in \{5,6,7,8,9\}$. The majority of experimental runs for Methods 1 and 2 were performed on a Pentium III 700 MHz PC (typically for $N \leq 8$) and a Pentium III 1000 MHz PC (typically for $N \geq 8$).

Although the research in this paper has focussed on the development of heuristic techniques, it should be noted that the algebraic construction of boolean functions can provide functions with optimal cryptographic properties that are, for large N, generally better than those able to be discovered by heuristic techniques. However, typically algebraic constructions are unable to provide a large number of functions with these optimal properties. Heuristic techniques, on the other hand, are capable of generating a significantly larger set of boolean functions with good properties. Both algebraic constructions and heuristic techniques are important ways in which good boolean functions can be generated.

Future work in this area could involve further experimentation to extend the heuristic methods proposed in this paper. For example, adapting Method 1 to enable the generation of odd-dimensional boolean functions with these properties is one avenue of further research. Also, the careful selection of starting bent functions for Method 1, as opposed to random selection, may be useful in generating balanced boolean functions with maximal nonlinearity, as conjectured by Dobbertin [5]. With regard to Method 2, a logical continuation to the work would involve investigations into ways of generating a pool of starting N-variable functions ($N > 4$) to begin the concatenation process in order to achieve $targetN$ boolean functions where $targetN$ > 9. Finally, an obvious future direction is the development of new heuristic techniques for the generation of optimal boolean functions to be utilised in the design of cryptographic cipher systems.

Appendix

Example of a 6-variable boolean function (in hex notation) from Table 1 with NL = 26, deg = 5, AC_{max} = 16 and σ = 6,784

```
a4ebd7d8b0b6808d
```

Example of a 8-variable boolean function (in hex notation) from Table 1 with NL = 116, deg = 7, AC_{max} = 16 and σ = 89,728

```
7eb4719b4da742a8bbe124ce18fa17fd7e6b716c4d58c2572b3e3431180d1702
```

Example of a 10-variable boolean function (in hex notation) from Table 1 with $NL = 488$, $deg = 9$, $AC_{max} = 40$ and $\sigma = 1{,}272{,}448$

```
b024e3e6e571b6a3bf2bece9ea76b9bc8317d0d5d64285808c18cfd8d94d8a8fb0
dbe311e58eb64cbfd4ec06ea81b94383e8d02ad6bd857f0ce75f25d9b28a70b024
1c19e571494cbf2b1306ea7e06438317272ad6427a7d8c182025d94d7570b0db1c
e6e50e49b3bfd413e9ea8146bc82e82fd556bd7a808ce720dad992758f
```

Example of a 12-variable boolean function (in hex notation) from Table 1 with $NL = 2000$, $deg = 11$, $AC_{max} = 64$ and $\sigma = 18{,}757{,}120$

```
2c63a66c3ce83d0d7936f33969bde8582c63a66cc317c2f2793ef339964297a723
6ca96333e732027639fc3666b26757236ca963cc18cdfd7639fc36994d9ca82c9c
a6933c1f3df279c9f3c6694268a72c9ca693c3e8c20d79c9f3c696bd97582393a9
9c331832fd76c6fcc9664d67a82393a99ccce7cd0276c6fcc999b29cd72c635993
3ce8e2f279360cc669bd97a72c635993c3173d0d79360cc696426858236c569c33
e7cdfd763903c966b298a8236c569ccc183202763903c9994d675f2c9c596c3c17
c20d79c90c3969429f582c9c596cc3e83df279c90e3996bd68a7239356633318cd
0277d60336664d9857239b5663cce732fd76c6033699b267a81f50955f0fdb8e3e
4a17c00a5a8e5b6b1f50955ff024f1c14a15c00aa571a494105f9a5120d4013145
0acf055581546c105f9a70ff2bfece450acf25aa7eab9b1faf95a00f240ec14afa
c0f55a715b941faf95a0f0dbf1be4afac0f5a58ea46b10a09aaf002b01ce45f5cf
fa557e549b10a09aafffd4fe31c7f5cffaaa81ab641f506aa00fdbf1c14b053ff5
5a8ea4941f506aa0f2240e3e4a053ff5a5715b6b105f65af00d4fece450b30fa55
81ab9b105f65afff2b0131450a30faaa7e54641faf6a5f0f24f1be4efa3f0a5a71
a46b1faf6a5ff0db0ec14afa3f0aa58e5b9410b06550002bfe3145f5b005557eab
6410a06550ffd401ce45f53005aa81549b
```

References

[1] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4:3–72, 1991.

[2] J.A. Clark and J.L. Jacob. Two-stage optimisation in the design of boolean functions. In *Australasian Conference in Information Security and Privacy (ACISP 2000)*, volume 1841 of *Lecture Notes in Computer Science*, pages 242–254. Springer Verlag, 2000.

[3] John A Clark, Jeremy L Jacob, Susan Stepney, Subhamoy Maitra, and William Millan. Evolving boolean functions satisfying multiple criteria. In *Third International Conference on Cryptology in India (INDOCRYPT 2002)*, volume 2551 of *Lecture Notes in Computer Science*, pages 246–259. Springer Verlag, 2002.

[4] E Dawson, A Clark, J Goliç, W Millan, L Penna, and L Simpson. The LILI-128 keystream generator. In *Proceedings of the First NESSIE Workshop*, 2000.

[5] Hans Dobbertin. Construction of bent functions and balanced boolean functions with high nonlinearity. In *Fast Software Encryption*, volume 1008 of *Lecture Notes in Computer Science*, pages 61–74. Springer Verlag, December 1994.

[6] Jovan Dj. Goliç. Modes of operation of stream ciphers. In *Selected Areas in Cryptology (SAC'2000)*, volume 2012 of *Lecture Notes in Computer Science*, pages 233–247. Springer Verlag, August 2000.

[7] Subhamoy Maitra. Highly nonlinear balanced boolean functions with good local and global avalanche characteristics. *Information Processing Letters*, 83(5):281–286, September 2002.

[8] Subhamoy Maitra and Enes Pasalic. Further constructions of resilient boolean functions with very high nonlinearity. *IEEE Transactions on Information Theory*, 48(7):1825–1834, July 2002.

[9] M. Matsui. Linear cryptanalysis method for DES cipher. In *Advances in Cryptology - EUROCRYPT'93*, volume 765, pages 386–397. Springer Verlag, May 1993.

[10] R.L. McFarland. A family of difference sets in non-cyclic groups. *Journal of Combinatorial Theory*, 15:1–10, 1973.

[11] William Millan, Andrew Clark, and Ed Dawson. Smart hill climbing finds better boolean functions. In *Selected Areas in Cryptology (SAC'97)*, pages 50–63, August 1997.

[12] William Millan, Andrew Clark, and Ed Dawson. Heuristic design of cryptographically strong balanced boolean functions. In *Advances in Cryptology - EUROCRYPT'98*, volume 1403, pages 489–499. Springer Verlag, 1998.

[13] National Institute of Standards and Technology. Advanced encryption standard, fips-publication 197. In *http://csrc.nist.gov/CryptoToolkit/aes/*, November 2001.

[14] E. Pasalic, S. Maitra, T. Johansson, and P. Sarkar. New constructions of resilient and correlation-immune boolean functions achieving upper bound on nonlinearity. In *Workshop on Coding and Cryptography - WCC 2001*, volume 6 of *Electronic Notes in Discrete Mathematics*. Elsevier Science, 2001.

[15] Enes Pasalic and Thomas Johansson. Further results on the relation between nonlinearity and resiliency for boolean functions. In *Proceedings of the 7th IMA Conference on Cryptography and Coding*, volume 1746 of *Lecture Notes in Computer Science*, pages 35–44. Springer, 1999.

[16] G Rose. A stream cipher based on linear feedback over gf(2^8). In *Third Australasian Conference on Information Security and Privacy*, volume 1438 of *Lecture Notes in Computer Science*, pages 135–146. Springer Verlag, 1998.

[17] Bimal Roy. A brief outline of research on correlation immune functions. In *Australasian Conference in Information Security and Privacy (ACISP 2002)*, volume 2384 of *Lecture Notes in Computer Science*, pages 379–394. Springer Verlag, July 2002.

[18] P. Sarkar and S. Maitra. Nonlinearity bounds and constructions of resilient boolean functions. In *Advances in Cryptology - CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 515–532. Springer Verlag, 2000.

[19] Palash Sarkar and Subhamoy Maitra. New directions in design of resilient functions. In *http://eprint.iacr.org/2000/009*. Cryptology ePrint Archive, March 2000.

[20] T Siegenthaler. Correlation-immunity of nonlinear combining functions for cryptographic applications. *IEEE Transactions on Information Theory*, IT-30(5):776–780, 1984.

[21] Xian-Mo Zhang and Yuliang Zheng. GAC - the criterion for global avalanche characteristics of cryptographic functions. *Journal of Universal Computer Science*, 1(5):316–333, 1995.

(Received 12 Dec 2002)