H2A, A NEW SECURE AND EFFICIENT CIPHER ALGORITHM FOR SMART CARDS

[1]HAZEM M. ELDEEB, [2]ABU B. SESAY,[3] HISHAM M. DAHSHAN, [4]ALAA EL-DIN R. SHEHATA

1,2Dept. of Electrical & Computer Eng., University of Calgary, Calgary, Canada
[3,4]Dept. of Communication, Military Technical College, Cairo, Egypt.
E-mail: [1]hazem.eldeeb@ucalgary.ca, [2]sesay@ucalgary.ca, [3]hdahshan@mtc.edu.eg, [4]alla rohiem@yahoo.co.uk

Abstract - Smart cards are computer-like devices with small size and very limited hardware and software resources. This paper introduces a new secure and small-sized cipher algorithm suitable for implementation on limited resources devices like smart cards. The proposed cipher algorithm has a high degree of cryptographic strength compared to the Advanced Encryption Standard finalists but with less number of rounds and smaller size. The building blocks of the proposed algorithm have been chosen/generated with very high-valued cryptographic properties. The proposed cipher has been tested for randomness, unpredictability and immunity against known linear and differential cryptanalysis attacks. The high performance of the algorithm has been evaluated by counting the number of CPU cycles consumed for one-byte encryption. Finally, and for the purpose of verification, the algorithm has been implemented on an open platform processor card to evaluate its suitability to work on smart cards.

Index Terms - Block cipher algorithms, smart cards, security attacks.

I. INTRODUCTION

A. Smart Card Overview

Today, smart cards are widely used in our daily life. Their technology is being used in many fields like credit cards, passports, health cards, ID cards, driving licenses, SIM cards for mobile phones, etc. Smart cards are originally known as integrated circuit cards (IC cards). The set of card functions could be extended in run time according to the system they work in and according to user requirements as well [1]. One of the most important advantages of smart cards is that they provide tamper-resistant storage for protecting sensitive information and parameters against unauthorized access and manipulation. The stored data could be accessed only by the serial interface and under the control of operating system. Another important feature in smart cards is that sensitive data can be kept encrypted with on-board cryptographic algorithms with cipher keys and sub-keys that never leave the card [2]. Smart cards that are programmed with cipher algorithms are named cryptographic smart cards. Cryptographic algorithms programmed on smart cards are subjected to different kinds of attacks. Some of these attacks are based on information extracted from the weaknesses found in the mathematical design of the algorithms (e.g. linear and differential cryptanalysis).

Some others attacks are based on the information collected from the implementation of cipher algorithms [3]. These collected information, could be exploited to reveal some information about the cryptographic keys and other secrets on the card (e.g. side-channel attacks).

B. Mathematical attacks on cryptographic smart cards

Linear cryptanalysis takes advantage of the high probability occurrences of linear expressions that involve plaintext bits, ciphertext bits and sub-key bits. Linear cryptanalysis depends on that the adversary must possess a large set of input plaintext blocks and their corresponding ciphertext blocks for a certain cipher key, and that is why it is called known plaintext attack [4].

Differential cryptanalysis takes advantage of the high probability occurrences of a certain difference in the ciphertext (□C), which occurs as result of some differences in the plaintext (□P). For an Ideal cipher algorithms with n-bits input, the probability that a particular input difference will result in a particular output difference is $1/2^n$ which is highly unlikely to happen [4].

C. Side-channel attacks on cryptographic smart cards

Power Analysis is a form of side-channel attacks. Power Analysis has two forms: Simple power analysis (SPA) and differential power analysis (DPA). SPA, analyzes the current consumption on a smart card chip over a period of time, while DPA is a statistical method which analyzes the power consumption to compute the intermediate values and data-dependent correlations [5]. Timing attacks is also is a non-invasive attack in which the attacker measures the non-constant time consumed by the smart card to perform on-board cryptographic operations. These measurements could be exploited to leak information about the secret key inside [5].

II. LITERATURE SURVEY

The resistance of cipher algorithms against side-channel attacks is affected by the method the cryptographic functions are implemented. For example, fixed shift or rotate operations do not leak information about the data being processed, while additions and multiplications could be used to partially/completely extract information about the data being processed. RC6 is one of the AES finalists. It has a 128-bit input block size and a variable key length (128/192/265) bit. The algorithm includes a variable rotation (a <<< b), for the w-bit word (a) to the left by the amount given by the least significant ($\log_2 w$) bits of (b). In most smart cards, processor supports only one-bit shift. So n-bit shift operation will be implemented as multiple single bit shifts (n-times). This multiple shift will subject the algorithm to timing attack [6]. As a counter measure, it is possible to perform a constant time rotation, but this comes with a performance penalty.

Rijndael is the Advanced Encryption Standard winner. It has an input block of size 128 bits and a variable key size (128/192/256) bits. In [7], Mangard showed that Rijndael cipher leaks the Hamming weight of intermediate results occurring during the key expansion process. This weakness could be utilized by applying simple power-analysis attack to significantly reduce the cipher key space. In [8] Leander et. al. proposed a new lightweight variant of DES cipher called DESL (DES Lightweight). In DESL, and in order to minimize the size of the algorithm, a single s-box was repeatedly used to replace eight different s-boxes in the original DES. However, the security strength of the algorithm is still limited by the short 56-bit cipher key [9]. With exhaustive search techniques on the key space, the cipher key can be retrieved in few weeks. Twofish is also one of the Advanced Encryption Standard finalists. The algorithm was designed by Schneier and Whiting with smart cards in mind [10]. The algorithm consumes in an optimized implementation around sixty bytes of random access memory. The algorithm has a relatively good performance and it uses fixed rotate and shift operations in order to resist timing attacks. In [11], Engels et. al presented Hummingbird, a new lightweight cipher algorithm for RFID tags. It has 16-bit block size, 256-bit key size, and 80-bit internal buffer. However, Hummingbird is valuable to cube attacks under the single-bit-leakage model [12].

III. CONSTRUCTION AND BUILDING BLOCKS

H2A is a 10-round, SPN-structured block cipher algorithm with 256-bit cipher key length and 128-bit input block length (see Fig.1). H2A was designed according to the approach of Wide Trail Strategy (WTS). In WTS, each round of the block cipher is composed of different bijective transformations, each with a specific functionality [13].

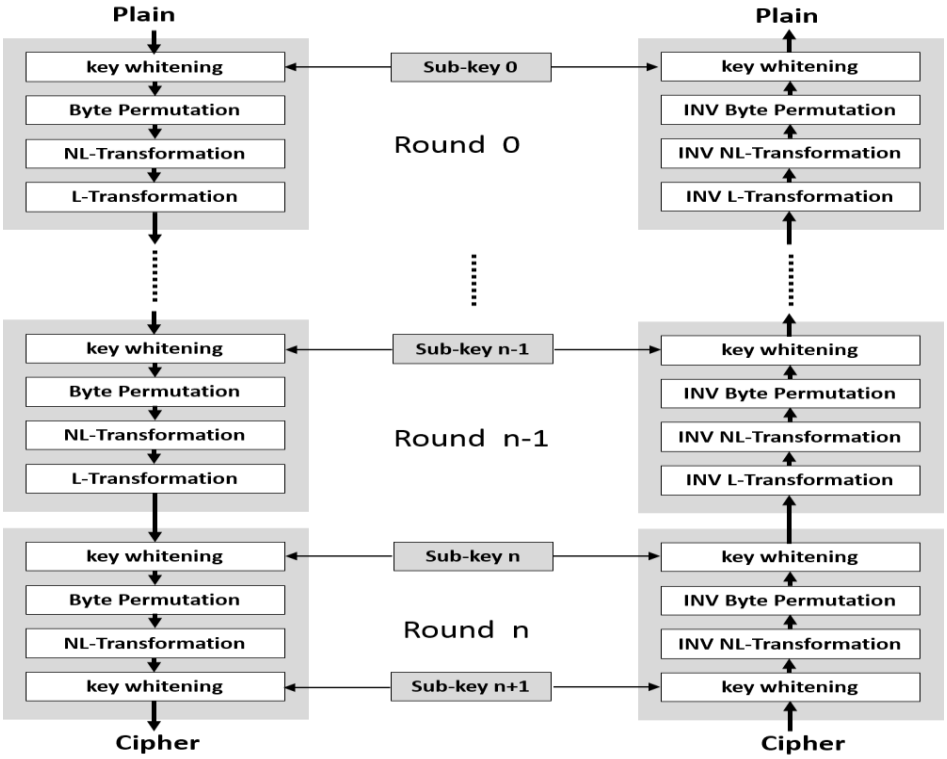


Fig.1. H2A general construction.

H2A round is constructed from sub-key derivation algorithm and four cascaded layers: key whitening, byte permutation, non-linear transformation/substitution box (s-box) and linear transformation (see Fig.2).

The key whitening layer introduces all the sub-keys to the cipher algorithm. The byte permutation and linear transformation are the main source of diffusion in the algorithm. They act together to guarantee a large number of active s-boxes after a few round. Non-linear transformation is the main source of non-linearity in the algorithm. Non-linear transformation in conjunction with the diffusion components, provide the algorithm with the required resistivity to linear and differential attacks.

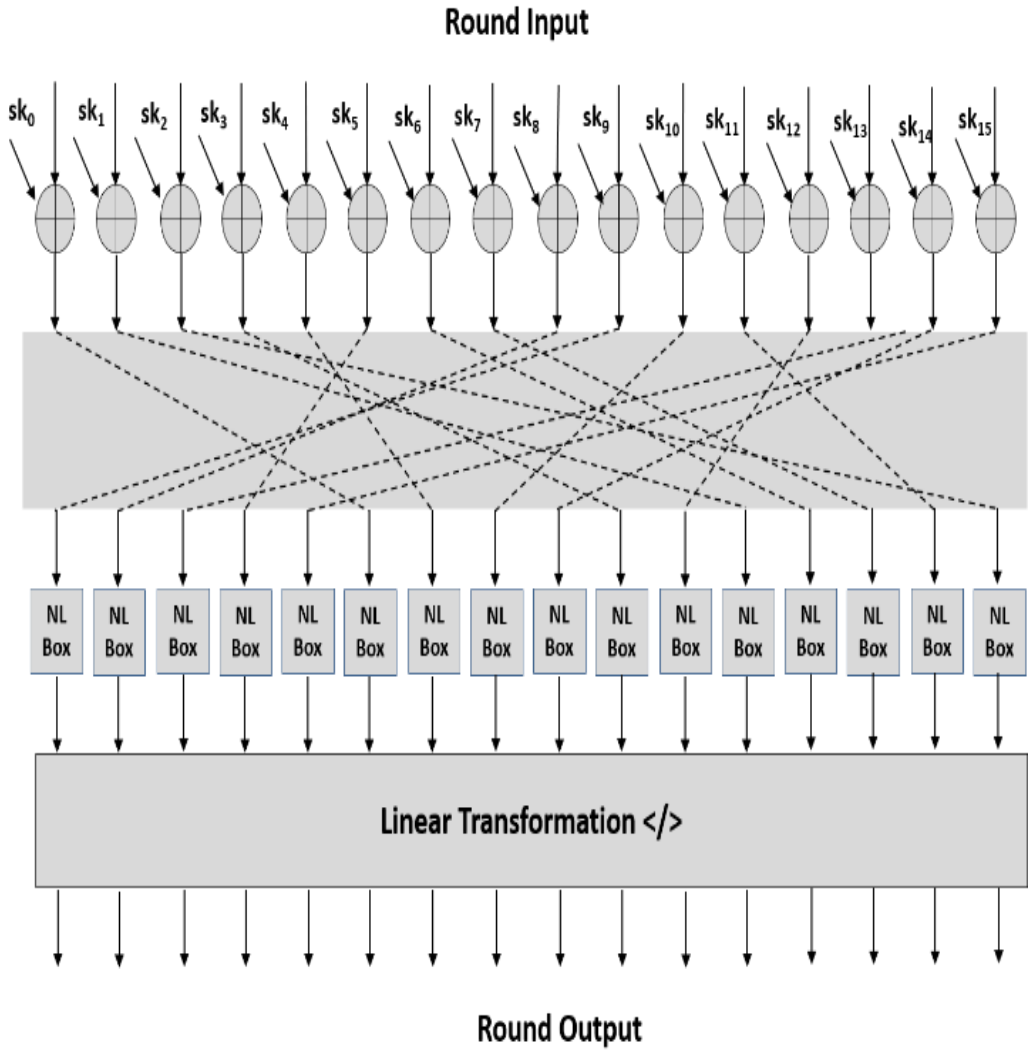


Fig.2. H2A one -round.

A. Sub-key Derivation

Sub-Key derivation (Key expansion) algorithm is the process of cipher key expansion to generate all the set of round keys (sub-keys) required for the encryption/ decryption process. It must be impossible for the adversary to recover the cipher key knowing one or more of the round keys. In H2A, the sub-Key derivation algorithm is simply a non-linear pseudo-random generator (NLPRNG). NLPRNG consists of two 128-bit feedback shift register (R1, R2) which constitute its internal state. R1 is a non-linear feedback shift register, which uses an s-box for the non-linear feedback, while R2 is a linear feedback shift register with non-sparse primitive polynomial acting as a feedback polynomial (see Fig.3.).

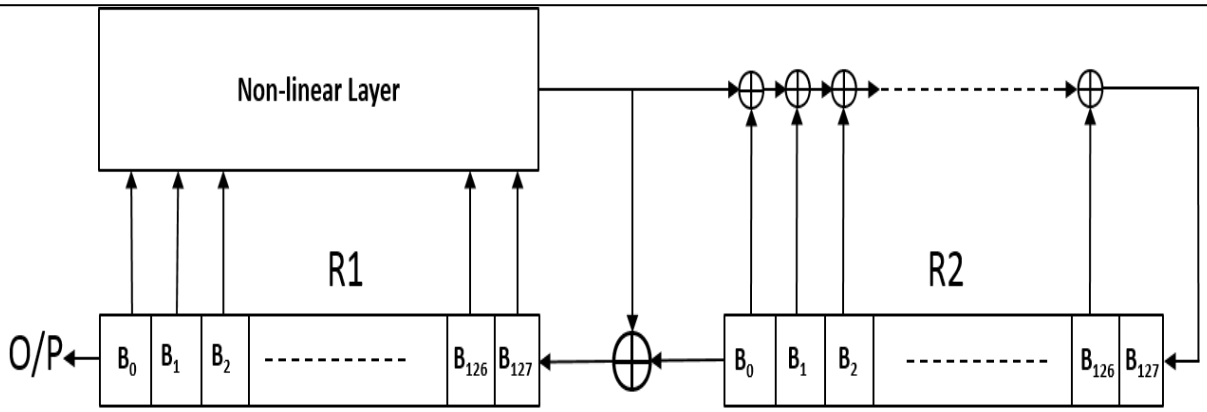


Fig.3. Sub-key derivation algorithm.

The algorithm runs 200 clock for warming in order to guarantee that the contents of the two registers have been non-linearly changed. The algorithm runs ((#rounds+1) * 16) to generate the set of sub-keys. The process of sub-keys generation takes place before the start of encryption/decryption process, and all the required sub-keys are stored in the smart card programmable memory (EEPROM).

B. Key Whitening

A basic requirement for any cryptosystems is that every plaintext key bit should influence every ciphertext bit. To achieve this influence in H2A, a simple bit-wise exclusive-OR is applied on each byte of 128-bit round key k^i with each byte of the 128-bit round input block B_j (byte by byte).

$B_j \rightarrow B_j \oplus K_j^i$ $\quad 0 \le i \le n$, n: number of rounds.

$K_j^i = K_0^i\ K_1^i\ K_2^i \ldots K_{15}^i$ $\quad 0 \le j \le 15$, j: byte index.

In order to ensure the effectiveness of each round of the algorithm, a sub-key is xored with the output of the last round. This last sub-key will prevent the adversary from ignoring the last round when attacking the algorithm backwardly.

C. Byte Permutation

In the byte permutation (P-box), a byte permutation is applied on the 16-byte input block. It takes the outputs of key whitening and permutes the bytes before feeding them into the S-boxes of the current round. In H2A, the design of byte permutation layer, guarantees that each adjacent 4 bytes of the input are distributed into four different linear transformations (see Table I).

Table I Byte permutation table

Bit Position (i)	0	1	2	3	4	5	6	7
P(i)	3	7	11	15	2	6	10	14
MDS affected	1st	2nd	3rd	4th	1st	2nd	3rd	4th

Bit Position (i)	8	9	10	11	12	13	14	15
P(i)	1	5	9	13	0	4	8	12
MDS affected	1st	2nd	3rd	4th	1st	2nd	3rd	4th

D. Non-liner Byte multiplier (s-box)

The resistance of block cipher to differential and linear cryptanalysis; is tied directly to its non-linear substitutions (s-boxes) [14]. S-box is a form of multi-output Boolean function. It is a mapping of m-bit inputs to n-bit outputs. The components of multi-output Boolean interact and affect each other. Although each component function has certain cryptographic properties, the multi-output function constructed from them, does not necessarily take on similar properties [14]. Power function is a form of mathematically constructed multi- output function (permutation). These functions are based on a bijective transformation ($\alpha \rightarrow \alpha^d$) from n-bit input α to n-bit output α^d for an integer exponent d. Power functions with GCD (d, 2^n - 1) = 1, are acting as a permutation.

H2A cipher utilizes a power function-based s-box represented in the form of look-up table of 256 elements. Each byte of the 16-byte input block from the previous round is raised to a mathematically computed exponent d. The exponent d is derived from famous Dobbertin & Niho function, which gives maximum non-linearity [15].

For a function with 8-bit dimension (n=8):

d = $\sum_{i=0}^{n/2} 2^{ik}$: n=2t , n ≡ 0 [4] , gcd (k,n) = 1 , 0 < k < t

Since that n=8, and 8 mod 4 =0 therefore t=4.

We have chosen k=3 such that the Greatest Common Divisor GCD (k, n) = GCD (3, 8) =1 .

The exponent (d) = $\sum_{i=0}^{4} 2^{3i} = 4681 \ mod \ p(x)$,

P $(x) = x^8 + x^4 + x^3 + x^2 + 1$ (irreducible polynomial)

Taking advantage of the equivalency classes property in power functions with maximum nonlinearity, in which, every power function defined by exponent d is corresponding to another functions with exponent 2^i.d mod (2^n – 1) with (i < n). These corresponding functions are also functions with the same linear properties and maximum nonlinearity.

Also by taking advantage of the linear equivalence in which composition with non-degenerated linear functions does not affect the linear properties. An affine transformation with the value $(x^6 + x^4 + x\ 1)$ has been applied on the function to get the function in its final form.

E. Linear transformation

Diffusion component is used to realize the principle of diffusion by ensuring that every plaintext bit influences every ciphertext bit. For any linear transformation (T), the branch number (β) is a measure of its diffusion power. Branch number (β) is the minimum number of nonzero bytes in the input vector (a) and output vector F (a) when the input bytes are not all zero (byte hamming weight) (WH) ≠0). The branch number (β) of a linear transformation (T) is defined as:

$$(\beta) = Min_{a \neq 0} [W_H (a) + W_H(F(a))]$$

An active byte is a non-zero byte that involved in linear/ differential expression with high probability bias. Diffusion component that built on the basis of Maximal Distance Separable (MDS) codes is considered the best since it guarantees large number of active S-boxes in the context of differential/linear cryptanalysis. MDS code is a matrix with dimension (k x k) such that the minimum distance between any two code-words of length (n) differ in at least d places (β) = d = n- k + 1.

The Linear transformation used in H2A is Hadamard matrix of dimension (4 X4) which has the MDS property by construction. In general, Hadamard matrix has the form:

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha+\alpha^2 \\ \alpha & 1 & \alpha+\alpha^2 & \alpha^2 \\ \alpha^2 & \alpha+\alpha^2 & 1 & \alpha \\ \alpha+\alpha^2 & \alpha^2 & \alpha & 1 \end{pmatrix}$$

Hadamard matrix has two interesting properties:

- The matrix is entirely defined by the first row only.
- $H * H^{-1} = I\, c^2$: Such that:

c = $1 + \alpha + \alpha^2 + \alpha+\alpha^2$, I: the identity matrix.

If α has chosen such that c=1 ⇒ $H * H^{-1} = I$.

In H2A, the value of α has been chosen =3,

The working field (GF) has a dimension (n= 8) with irreducible polynomial p(x) equals to:

$$p(x) = x^8 + x^4 + x^3 + x^2 + 1$$

$$c = 1 + 3 + 5 + 6 \ \ mod\ (x^8 + x^4 + x^3 + x^2 + 1)$$

$$= 1 + (1 + x) + (1 + x^2) + (x + x^2) \ mod\ (x^8 + x^4 + x^3 + x^2 + 1) = 1$$

This means $H * H^{-1} = I$ which means that $H = H^{-1}$

$$H_{H2A} = H_{H2A}^{-1} = \begin{pmatrix} 1 & 3 & 5 & 6 \\ 3 & 1 & 6 & 5 \\ 5 & 6 & 1 & 3 \\ 6 & 5 & 3 & 1 \end{pmatrix}$$

IV. SECURITY ANALYSIS

A. Statistical Evaluation

In order to test the randomness property of the proposed cipher algorithm, the NIST statistical package has been used [16]. A set of 15 tests has been applied on an arbitrarily long binary sequences produced by the cipher algorithm operating in Cipher block chaining (CBC) mode of operation.

A file of size 8 Megabyte in binary format has been produced by the proposed cipher algorithm, with the input plaintext block set to zeros and non-zero random value for the initialization vector (IV). Table II shows the values of the test parameters.

Table II: Statistical Tests parameters

Test Parameter	value
Block Frequency Test - block length(M)	128
Non-Overlapping Template Test - block length(m)	9
Overlapping Template Test - block length(m)	9
Approximate Entropy Test - block length(m)	10
Serial Test - block length(m)	16
Linear Complexity Test - block length(M)	500

The tested binary file has passed the statistical test with significance level equals to 95% (α=0.05). Fig.4. illustrates the summary of the results with Average calculated test statistic ≤ threshold value for each test.

```
------------------------------------------------------------------------------
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES
------------------------------------------------------------------------------
   generator is <H2A.txt>
------------------------------------------------------------------------------
 C1  C2  C3  C4  C5  C6  C7  C8  C9 C10  P-VALUE  PROPORTION  STATISTICAL TEST
------------------------------------------------------------------------------
  5   3   3   9   6   6   9  10   9   4  0.275709     63/64      Frequency
 11   5   4  11   5   3   7   6   6   6  0.275709     64/64      BlockFrequency
  5   4   4   8  10  10   7   4   5   7  0.468595     64/64      CumulativeSums
  5   9   3   5  10   5   7   3   7  10  0.299251     63/64      CumulativeSums
  4   6   4   2  11   6   6   8   9   8  0.275709     64/64      Runs
  5  10   5   7   2   8   5   7   9   6  0.500934     64/64      LongestRun
  6   6   9   5   7   5   6   5  12   3  0.378138     64/64      Rank
 17  10   3   9   0   5   7   4   3   6  0.000078 *   61/64      FFT
  4  10   9   8   3   8   7   5   7   3  0.378138     64/64      NonOverlappingTemplate
  7   5   7   8  10   4   8   5   6   4  0.739918     64/64      OverlappingTemplate
  5   6   8   6  12   5   4   4   6   8  0.437274     64/64      Universal
  8   5   7   9   3   3   6   7   6  10  0.500934     63/64      ApproximateEntropy
  2   6   5   3   6  10   3   3   4   4  0.162606     46/46      RandomExcursions
  3   8   3   4   5   4   2   9   4   4  0.213309     46/46      RandomExcursionsVariant
  0   3   3   1   2   5   8   9  12  21  0.000000 *   64/64      Serial
  2   1   6   1   3   4   9  11  14  13  0.000020 *   64/64      Serial
  8   8   5   6  12   5   4   4   4   8  0.324180     64/64      LinearComplexity


- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
The minimum pass rate for each statistical test with the exception of the
random excursion (variant) test is approximately = 60 for a
sample size = 64 binary sequences.

The minimum pass rate for the random excursion (variant) test
is approximately = 43 for a sample size = 46 binary sequences.

For further guidelines construct a probability table using the MAPLE program
provided in the addendum section of the documentation.
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Fig.4. Statistical tests result.

B. S-Box Analysis.

In general, the security strength of any cipher algorithm is premised on the cryptographic strength of the non-linear component (s-box). Bad linear and differential properties of the s-box; negatively affect the immunity of the cipher algorithm against linear and differential attacks. As a basic step of the security evaluation, the s-box of the proposed cipher algorithm has been studied and analyzed. The cryptographic properties like maximum Walsh, maximum autocorrelation, differential uniformity and algebraic degree have been measured.

- The non-linearity (NL) of an s-box is of n variables refers to the minimum hamming distance between the s-box function $f(x)$ and any affine/linear function. Non-linearity is calculated by computing the maximum value of Walsh-Hadamard Transform $W(u)$

$$\forall u \in \{0,1\}^n \quad W(u) = \sum_{x \in \{0,1\}^n} (-1)^{f(x) \oplus f(x \oplus u)}$$

$$NL = 2^{n-1} - \frac{1}{2} \max |w_f(w)| = 2^7 - \frac{1}{2} * 32 = 112$$

- Differential uniformity is another important property of the s-box, which is related to the differential immunity. Differential uniformity calculates the probability that a particular output difference □Y of the s-box occurs given a particular input difference □X is $(1/2)^n$ where n is the number

of bits of the input X. Differential uniformity is calculated by finding the maximum value of difference distribution table D(f) .

$$T_f(u,v) = \#\{x \in \frac{E_n}{f(x)} \oplus f(x \oplus v) = u\}$$

$$D(f) = max\ (T_f(u,v))$$

Such that:

u, v are input and its corresponding output difference.

T_f is the difference table of the s-box.

$f(x)$ is the s-box function from the field E_n to the field E_m.

The difference distribution table has been constructed and the maximum value of the table is 16.

- The Auto-correlation (AC) is derived from the power spectrum of the WHT. It gives an indication about the imbalance of the first order derivatives of an s-box. Imbalanced derivatives of an s-box could be exploited by an adversary to mount a useful linear attack.
- Algebraic Degree is another important property of an s-box. S-boxes with low algebraic degree are subjected to successful cryptographic attacks.

Table II shows the cryptographic properties of the s-box of the proposed cipher algorithm compared to the advanced encryption standard.

Table II: Properties of the cipher algorithm

	Walsh maximum	Auto-correlation maximum	Differential uniformity	Algebraic degree
H2A	32	48	16	5
AES	32	32	4	7

C. Linear Evaluation.

In general, linear cryptanalysis takes advantage of the high probability occurrences of linear expressions with considerable probability bias (ε_i). These expressions involve plaintext bits (X_i), ciphertext bits (Y_j) and sub-key bits (K_k).

$$X_1 \oplus .. \oplus X_n \oplus Y_1 \oplus .. \oplus Y_m \oplus K_1 \oplus ... \oplus K_l = ½ + 2^{n-1} \prod_{i=1}^{n} \varepsilon_i \quad (1)$$

Linear cryptanalysis depends on that the adversary must acquire a large set of input plaintext blocks and their corresponding ciphertext blocks for a certain cipher key, such that ε_i is the probability bias after i rounds.

In order to evaluate the attack complexity, first, the linear properties of the s-box have been studied and analyzed. Second, the probability bias (ε) for each linear expressions has been computed according to (1). A complete enumeration of the results has been tabulated in the linear approximation table (LAT). From LAT, the highest linear expression probability in the table equals to $\frac{32}{256}$. The success of the attack is based on magnitude of the bias and not on the sign of the bias. In another words, an affine approximation can be used equivalently to linear approximation. Third, the number of active s-boxes for an arbitrary number of rounds has been computed and enumerated (see Table III).

Table III: active s-boxes after four rounds

	Round 1	Round 2	Round 3	Round 4
active s-boxes/round	1	7	16	5
Total	1	8	25	29

In [17], Matsui showed that, the number of known plaintext pairs, which the adversary must possess to exploit the found bias and partially recover the sub-key, is proportional to □□^{2}. Let (N_{kp}) represents the number of known plaintexts possessed by the adversary N_{kp}□ □1/□^{2}.

From LAT, the highest probability equals to $\frac{32}{256}$

To calculate the probability bias, we substitute in (1)

$$½ + 2^{n-1} \prod_{i=1}^{n} \varepsilon_i$$

$$\frac{32}{256} = \frac{128}{256} \pm \varepsilon \qquad \Rightarrow \qquad \varepsilon = \frac{1}{8} = 2^{-3}$$

By applying the Piling-Up Lemma on the value of bias

$$\varepsilon_{1,2,...,n} = 2^{n-1} \prod_{i=1}^{n} \varepsilon_i$$

The total bias after 4 rounds $= 2^{-3*29} = 2^{-87}$

$N_{kp} = 2^{3*29*2} = 2^{174}$ pairs of known plaintext, which is considered infeasible for the adversary to acquire and process in terms of memory space and processing power.

D. Differential Evaluation

In the differential evaluation, the aim of the adversary is to find a differential characteristic of the overall cipher algorithm with reasonable probability. Let(N_{cp}) represents the number of chosen plaintext pairs, which must be acquired by the adversary to mount a successful differential attack. Let D_p represents the differential characteristic probability for r rounds.

$D_p = \prod_{i=1}^{\sigma} \beta_i$ Such that:

σ is the number of active s-boxes for r rounds,

βi is the differential characteristic for the i-th active s-box.

The minimum number of chosen plaintext pairs required to mount the attack is: N_{cp}□ □c/D_p , such that c is some constant [18].

After four round: $D_p = \prod_{i=1}^{\sigma} \beta_i = 2^{-6*29} = 2^{-174}$

N_{cp}□c/2^{-174} □2^{174}

2^{174} pairs of chosen plaintext pairs and their corresponding ciphertext pairs are highly infeasible for the adversary to acquire and process.

E. Design Security considerations

The resistance to power analysis attacks and timing attacks must be taken into consideration when designing a new cipher algorithm for hardware devices. The proposed cipher algorithm does not slip any information about the cipher key during the key derivation process. The key derivation process runs a flat number of CPU clocks for any value of the cipher key with no variable shift/ rotate or modular add/subtract operations.

Due to the fact that multiplication operation is hard to protect against timing attacks, the utilized s-box, which includes byte multiplication, is represented in the form of masked precomputed look-up table. There are no multiplication operations take place during encryption/decryption process.

For Target platform smart cards with little higher resource (e.g. 32-bit processor and several kilobytes of programmable memory), a stronger security could be achieved by implementing the complete proposed cipher in the form of precomputed tables (T-tables). The linear transformation matrix could be combined with the s-box layer in the form of four (one for each row in the matrix) different look-up tables, each of one kilobyte size.

In the same way, the byte permutation layer can be added to the precomputed arrays by re-ordering the bytes inside each array according the byte permutation table and without increasing the size of precomputed tables.

V. PERFORMANCE ANALYSIS

A. Implementation

In order to evaluate the performance of the proposed cipher algorithm, H2A has been coded in embedded C language for Atmel AVR and programmed on a processor smart card. An open platform card "Prussian card" has been chosen to implement and test the proposed cipher algorithm. The chosen card is ATMEL AVR (processor AT90S8515A) and it has the following specs:

- SRam: 512 bytes
- EEPROM: 512 bytes
- Max Freq: 8 MHz
- I/O Ports: 32
- Programmable Timer.
-Low-Power Idle Modes.
-Programmable Serial UART
-Master/slave Serial interface
-External and Internal Interrupt
-On-Chip Analog Comparator

A cross-platform IDE (e.g. Code::Blocks for windows operating system) that supports multiple compilers including GCC, Clang and AVR, has been used to develop and compile the code. An Executable and Linking Format (ELF) file has been created after the compilation process. The elf file has been downloaded on the card flash memory using "Infinity USB smart card programmer.

The sub-keys are generated by the key expansion algorithm and kept in the erasable programmable memory (EEPROM) using the command "EEMEM". This operation takes place before the encryption/decryption process. The command "PROGMEM" has been used to keep the S-box look-up table in the flash memory at the programming time.

Smart card needs a card reader/card acceptance device (CAD) to communicate with any user/application. The communication between the card and the reader is performed via APDU (Application Protocol Data Units). APDU is a message structure/communication unit between the smart card and the card reader. APDU carries a command from the reader to the card and brings the response from the card to the reader. An APDU from the reader to the card is called command APDU (C-APDU), and an APDU from the card to the reader is called response APDU (R-APDU).

B. H2A Memory usage

In order to evaluate the memory usage of the algorithm, the command "size H2A.elf" has been used. The command "size", lists the section sizes and the total size for the target elf file.

Table IV illustrates the memory usage of H2A. The total code size is equal to 4.252 Kbyte, which is considered very small code size and suitable for implementation on a device with limited memory space like the smart card.

Table IV: active s-boxes after four rounds

text	data	bss	dec	Hex	filename
4282	202	40	4252	11ac	H2A.elf

C. H2A speed evaluation

The aim of this section is to measure the execution time and throughput of the proposed cipher algorithm when implemented on limited –resource platforms like smart cards. Due to the fact that the algorithm speed is directly proportional to the speed of the platform processor, the most accurate way to measure the speed is counting the number of CPU cycles consumed for one byte of encryption. There are many ways to count the consumed CPU cycles, in this paper, RDTSC (read-time stamp counter) instruction has been used. The effect of out-of-order-execution, data cache and Instruction cache has been taken into consideration. H2A algorithm consumes 570 CPU cycles for the encryption of one byte.

If the clock speed equals to 20 MHZ, then:

$$Byte\ Encryption\ Time\ (seconds) = \frac{Number\ of\ CPU\ Cycles}{Freceuency\ (HZ)}$$

$$Bit\ Encryption\ Time\ (seconds) = \frac{570}{8 * 20 * 10^6} = 3.5 * 10^{-6}$$

According to the above equations, H2A exhibits a high throughput (304 k bit/sec) on a processor with very low clock frequency (20 MHZ).

CONCLUSION

In this paper, a novel secure, compact and high-speed cipher algorithm H2A for smart cards has been introduced. The high level of randomness of the proposed cipher algorithm has been successfully tested using NIST suite for statistical tests. The main non-linear component of the proposed cipher algorithm (s-box) has been studied and analyzed to assure its security strength. The proposed cipher algorithm has been proven highly resistive to linear and differential attacks.

The algorithm has a compact size and high throughput, which makes it suitable for limited resources devices like smart cards.

REFERENCES

[1] Wolfgang Rankl and Wolfgang Effing, Smart Card Handbook, 4th ed., Munich, Germany: John Wiley & Sons Ltd, 2010.

[2] Wolfgang Rankl, Smart Card Applications (Design Models for using and programming smart card). Munich, Germany: John Wiley & Sons Ltd, 2007.

[3] Savari, M., & Montazerolzohour, M. (2012, June). All about encryption in smart card. In Cyber Security, Cyber Warfare and Digital Forensic (CyberSec), 2012 International Conference on (pp. 54-59), IEEE.

[4] Wang, Y., Wu, W., Guo, Z., & Yu, X. (2014, June). Differential cryptanalysis and linear distinguisher of full-round Zorro. In International Conference on Applied Cryptography and Network Security (pp. 308-323). Springer, Cham.

[5] Fontana, G., Donatiello, S., & Di Sirio, G. (2014). U.S. Patent No. 8,804,949. Washington, DC: U.S. Patent and Trademark Office.

[6] Charbathia, S., & Sharma, S. (1831). A Comparative Study of Rivest Cipher Algorithms. International Journal of Information & Computation Technology. ISSN 0974-2239 Volume, 4, 1831-1838.

[7] Mangard, S. (2002, November). A simple power-analysis (SPA) attack on implementations of the AES key expansion. In International Conference on Information Security and Cryptology (pp. 343-358). Springer, Berlin, Heidelberg.

[8] Leander, G., Paar, C., Poschmann, A., & Schramm, K. (2007, March). New lightweight DES variants. In International Workshop on Fast Software Encryption (pp. 196-210). Springer, Berlin, Heidelberg.

[9] Kumar, S., Paar, C., Pelzl, J., Pfeiffer, G., & Schimmler, M. (2006, October). Breaking ciphers with COPACOBANA–a cost-optimized parallel code breaker. In International Workshop on Cryptographic Hardware and Embedded Systems (pp. 101-118). Springer, Berlin, Heidelberg.

[10] Schneier, B., & Whiting, D. (1998, September). Twofish on smart cards. In International Conference on Smart Card Research and Advanced Applications (pp. 265-276). Springer, Berlin, Heidelberg.

[11] Engels, D., Fan, X., Gong, G., Hu, H., & Smith, E. M. (2010, January). Hummingbird: ultra-lightweight cryptography for resource-constrained devices. In International Conference on Financial Cryptography and Data Security (pp. 3-18). Springer, Berlin, Heidelberg.

[12] Fan, X., & Gong, G. (2011, July). On the security of Hummingbird-2 against side channel cube attacks. In Western European Workshop on Research in Cryptology (pp. 18-29). Springer, Berlin, Heidelberg.

[13] Daemen, J. (1995). Cipher and hash function design strategies based on linear and differential cryptanalysis (Doctoral dissertation, Doctoral Dissertation, March 1995, KU Leuven).

[14] Carlet, C. (2010). Boolean functions for cryptography and error correcting codes. Boolean models and methods in mathematics, computer science, and engineering, 2, 257-397.

[15] Dobbertin, H. (1999). Almost perfect nonlinear power functions on GF (2^n): The Niho case. Information and Computation, 151(1-2), 57-72.

[16] MARTON, K., & SUCIU, A. (2015). On the interpretation of results from the NIST statistical test suite. SCIENCE AND TECHNOLOGY, 18(1), 18-32.

[17] Matsui, M. (1993, May). Linear cryptanalysis method for DES cipher. In Workshop on the Theory and Application of of Cryptographic Techniques (pp. 386-397). Springer, Berlin, Heidelberg.

[18] Lai, X., Massey, J. L., & Murphy, S. (1991, April). Markov ciphers and differential cryptanalysis. In Workshop on the Theory and Application of Cryptographic Techniques (pp. 17-38). Springer, Berlin, Heidelberg.

★★★