IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 20, 2017, accepted October 20, 2017, date of publication October 25, 2017,
date of current version November 28, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2766234

Algebraic Side Channel Attack on Trivium and Grain Ciphers

ASIF RAZA KAZMI[1], MEHREEN AFZAL[1], MUHAMMAD FAISAL AMJAD[1], (Senior Member, IEEE), HAIDER ABBAS[1], (Senior Member, IEEE), AND XIAODONG YANG[2], (Senior Member, IEEE)

[1]National University of Sciences and Technology, Islamabad 44000, Pakistan
[2]School of Electronic Engineering, Xidian University, Xi'an, Shaanxi 710071, China

Corresponding authors: Haider Abbas (dr.h.abbas@ieee.org) and Xiaodong Yang (xdyang@xidian.edu.cn)

The work was supported in part by the National Natural Science Foundation of China under Grant 61671349.

ABSTRACT Solving a system of multivariate quadratic equations obtained through algebraic cryptanalysis is a nondeterministic polynomial time-complete problem. Owing to the trend of stream ciphers based on nonlinear update, the success of algebraic attacks has been limited to their reduced variants. On the other hand, side channel attacks (SCAs), although require a continued access to the target device for capturing leakages, are a potent threat against the stream ciphers. Algebraic SCA (ASCA) combines and solves equations obtained through algebraic cryptanalysis and partial SCA of cipher implementation. ASCA is successfully being applied against block ciphers since 2009; however, there is no existing published work on ASCA against stream ciphers as per our knowledge. In this paper, we propose an idea of mounting ASCA on stream ciphers, and we demonstrated it through the application of ASCA on trivium and grain stream ciphers.

INDEX TERMS Algebraic side channel attack, ASCA, stream ciphers, Crypto-1, Bivium-B, trivium, grain, cryptanalysis, SAT solver, CryptoMiniSAT 5.0, grain-of-salt.

I. INTRODUCTION

Algebraic attacks on stream ciphers, as introduced by Courtois and Meier in 2003, attempted to solve a system of multivariate polynomial equations obtained from the association of internal state bits and few output stream bits [1]. This subsequently led to the efficient retrieval of internal state bits or secret key in case of low-degree equations. Courtois also proved existence of low-degree annihilator functions against Boolean functions, which led to reduction of overall complexity of algebraic attacks. Numerous literatures on the subject can be found, such as in [2]–[6]. However, nonlinearity in the state update of stream ciphers through employment of nonlinear feedback shift register (NLFSR) or nonlinear combiner functions eventually became a common feature, thereby greatly increasing the degree of generated equations and decreasing the computational feasibility of their corresponding algebraic attacks. Therefore, it can be safely concluded that the success of algebraic attack against stream ciphers has been confined to toy ciphers and round-reduced variants of popular stream ciphers in spite of evolving techniques to resolve multivariate quadratic equation (MQ) problem such as relinearization [7], extended linearization [8], sparse extended linearization [9], Groebner bases [10], and Boolean satisfiability (SAT) solving. Stream ciphers based on LFSRs, such as Sfinks [11] and WG [12], were dropped in the second and third phases of Project eSTREAM [13], respectively. NLFSR-based stream ciphers proved to be highly resistant against algebraic attacks, as the degree of underlying algebraic equations increased with clocking of registers. The authors in [14] demonstrated that after about 80 equations against Grain v1 stream cipher were obtained through an algebraic attack, the degree of the equations rose to as high as 160. Clock-controlled stream ciphers, such as A5/1, possess impressive resistance against algebraic cryptanalysis as well [15]. Algebraic attacks against modern stream ciphers, therefore resort to guessing few bits and in some cases specific guessed bits lead to better results [16], [17].

Side Channel Analysis (SCA), on the other hand, has been more successful against stream ciphers when compared with algebraic attack; however, SCA is not as common as algebraic attack [18], [19]. In [18] and [20], the authors have mounted a successful differential power attack (DPA) both against Trivium and Grain ciphers. In [21], the authors demonstrated the concept of template attack by successfully applying

it against RC4 stream cipher. In [22], the susceptibility of eSTREAM ciphers towards SCA was comprehensively evaluated, and a theoretical approach, which shows that all eSTREAM ciphers were vulnerable to SCA, was adopted.

As recent popular stream ciphers based on nonlinear update functions are highly resistant against algebraic attacks but are reasonably vulnerable to SCA leads to a logical speculation that *some side channel leakage information from stream cipher implementation might augment the algebraic equations obtained from an algebraic attack in a way to make the solving easier*. In our study, we investigated this theory, which eventually turned out to be a valid one.

The rest of the paper is organized as follows. Works related to current research are briefly discussed in Section II. A generic model of our proposed attack is presented in Section III. We applied ASCA against a simple stream cipher as a proof of concept in Section IV. Thereafter, we used popular stream ciphers, such as Trivium and Grain, to demonstrate the efficacy of ASCA in Sections V and VI, respectively. The summary of our results is presented in Section VII. Lastly, the concluding remarks and future work possibilities are mentioned in Section VIII.

II. RELATED WORK

To our knowledge, our work is the first to deal with the application of ASCA against stream ciphers, although it can be related to researches on ASCA against block ciphers. Renauld and Standaert [23] proposed to combine algebraic cryptanalysis with SCA in 2009. They successfully applied ASCA against reduced and extended versions of the PRESENT cipher [24] and observed that the resolution time varied linearly with the number of rounds. To demonstrate the efficacy of ASCA, they assumed to extract exact values of hamming weights at the input and output of S-boxes in all rounds in a single trace. Resultantly, the algebraic equations, combined with hamming weight leakage equations of 8-, 16-, 24-, 32- and 64-round PRESENT ciphers, were resolved using SAT solvers in average times of 0.39, 0.98, 1.5, 2.5 and 7 seconds, respectively. From 2009 onwards, ASCA against block ciphers drew the interest of researchers, especially through the improvement of its error-tolerance. In [25], the authors applied ASCA on advanced encryption standard (AES) and highlighted that most of the notions existing for PRESENT cipher can be noted against unprotected implementation of AES on an 8-bit micro-controller. They recovered an AES key after observing a single encryption operation through ASCA. In [26], the authors emphasized that errors in side channel information to be incorporated in ASCA, makes the SAT problem unsatisfiable; therefore, they recommended the usage of pseudo-Boolean optimizers (PBOPTs) instead of SAT solvers for ASCA in the presence of errors. Optimizers also take into account some additional logical constraints during solving. Another error-tolerant technique that deals with inaccurate side channel measurements in ASCA, which is known as multiple deductions-based ASCA (MDASCA), was introduced in [27]. In [28], the authors presented a novel notion of algebraic immunity for designing ASCA resistant S-boxes. Furthermore, the authors in [29] presented an improved ASCA on AES; whereas in [30], in addition to optimizers and solvers comparison in terms of robustness and speed, the authors proposed to search for leakage models, aside from hamming weight, ASCA application. Moreover, in another study regarding ASCA on AES [31], the authors established an error rate threshold that can be tolerated using an optimizer; and they claimed to have recovered an AES key in ten hours with 20% error rate from an average of 100 measurements. In [32], the authors attacked an AES key using a template attack combined with ASCA in accessing an exceedingly restricted implementation device. Another study on error-tolerant ASCA used a constraint programming compiler called BEE (Ben-Gurion University Equi-propagation Encoder) [33]. Furthermore, in [34], as the authors were attacking AES, side channel information along with measurement noise were added into equation set obtained through algebraic attack, and solved as a PBOPT instance.

Another line of research that can be related to our work is the exploitation of side channel leakage information from cipher implementation. SCA targets the implementation of ciphers by exploiting power leakage, electromagnetic radiations, execution time, photonic emissions, etc. Timing analysis is possible if there is an existing relationship between execution time of algorithm and its internal states [35]. A cipher might be susceptible to timing attack if its algorithm has some conditional branch instructions or table look ups. In case of power analysis, hamming weight or hamming distance information are captured from the leaked power traces. Algorithmic noise due to hardware implementation can also be advantageous to an attacker. Simple power analysis (SPA) takes advantage of the relationship between instant power consumption and internal states in one or few power traces [36]. DPA exploits the difference in power consumption due to variation in the data being processed with same key. Albeit, it may not be practically possible to restart the operation of stream cipher, researchers have used scenarios wherein frequent resynchronization is needed [37]. In template attacks, the adversary acquires an exact replica of the encryption device that executes the stream cipher [21]. This is a reasonable assumption, as standard micro-controllers are used quite often for this purpose. In the first phase, separate templates of typical signal are captured and recorded along with the associated noise against all possible key values, using the replica device. In the second phase, a single leakage trace is required from the actual device, to match with recorded template to reach the exact key. Instead of performing this classification process on the entire key space *extend-and-prune* strategy is recommended to be used iteratively. The issues that are related to template attacks have been discussed in detail in [38].

In this paper, we propose the application of ASCA against stream ciphers for the first time. We demonstrated the application of ASCA, through the transformation of partial side channel leakage information from implementation of target

ciphers into conjunctive normal form (CNF) clauses; and we added these said clauses to those obtained from conventional algebraic attack technique and were subsequently inputted to SAT solver. The SAT solving results of ASCA against Crypto-1, Bivium-B, Trivium and Grain v1 stream ciphers using CryptoMiniSAT 5.0 lead to successful results –0.158, 11.531, 21.54 and 28.25 seconds, respectively –which, by far, are the most satisfactory results ever published using a modest Linux desktop computer.

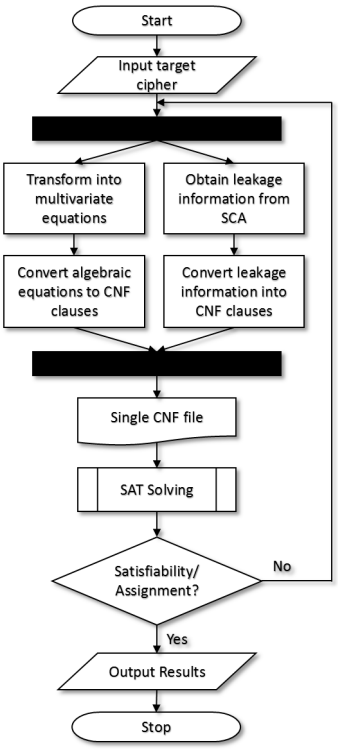

FIGURE 1. Attack Model.

III. PROPOSED ATTACK METHODOLOGY

The sequence of ASCA steps on stream ciphers can be summarized through a flowchart, as shown in Fig 1. A brief introduction of the underlying concept is also presented in [39].

Sole algebraic attack has offline and online phases of execution. In the offline phase, target stream cipher is transformed into a system of multivariate equations; whereas in the online phase, the adversary needs sufficient output stream bits that depend on the unknown variables, to solve algebraic equations. Normally, stream ciphers have an initialization phase wherein no output bit is generated. Generally, algebraic attacks target complete internal state bits and subsequently reach the secret key through backtracking. However, secret key can also be directly targeted prior to initialization. In the former case, the number of unknowns would be greater, but the equations would be of lesser degree; whereas in the latter case, the equations would be of higher degree but with lesser unknowns.

In ASCA methodology the equations obtained from algebraic attack phase are not solved until additional equations from partial SCA are added into them.

In a full SCA, sufficient leakage information is obtained to find the complete key. However, partial SCA, as constituent of ASCA, aims to extract only adequate information, thus making the solving of the system of equations obtained earlier through algebraic attack tractable. Analyzing the leakage pattern of the cipher implementation to select leakage points or models that extract optimal information with minimal access to the device can make the overall attack more efficient.

There may be a case in which in spite of side channel information addition, the overall system of equations/clauses is not solved in reasonable time (a threshold of 3,600 seconds was considered in our study). In such a scenario, more information from SCA can be extracted, leakage points/model can be altered, and algebraic attack part can be revisited by capturing more output stream bits.

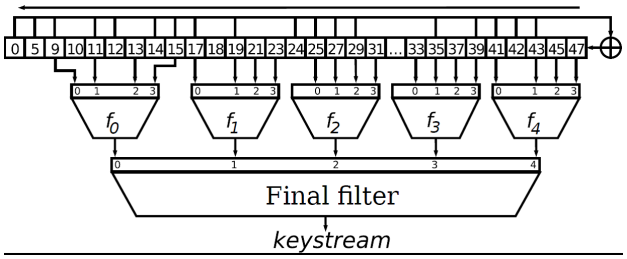

FIGURE 2. Structure of Crypto-1 Stream Cipher.

IV. ASCA AGAINST CRYPTO-1 AS PROOF OF CONCEPT

Crypto-1 stream cipher, as shown in Fig 2 [40], consists of a single 48-bit LFSR, which is initialized with a 48-bit key. Then, five 4-bit sets from LFSR are fed into five three-degree nonlinear functions $(f_0, f_1, .., f_4)$. If the input bits to these functions are denoted by a, b, c, and d, then, the functions are as follows:

$$\begin{aligned} f_0 = f_3 &= a \oplus b \oplus b.a \oplus c.a \oplus c.b \oplus d.a \\ &\quad \oplus d.c \oplus d.c.a \oplus d.c.b \\ f_1 = f_2 = f_4 &= b.a \oplus c \oplus c.a \oplus c.b \oplus c.b.a \\ &\quad \oplus d.a \oplus d.b \oplus d.c.a \oplus d.c.b \end{aligned}$$

The outputs of these five functions are fed into a four-degree nonlinear final filter function, which generates the key stream:

$$\begin{aligned} &f_0 \oplus f_2.f_0 \oplus f_3.f_0 \oplus f_3.f_1.f_0 \oplus f_3.f_2.f_1 \\ &\quad \oplus f_4 \oplus f_4.f_0 \oplus f_4.f_1.f_0 \oplus f_4.f_2.f_1.f_0 \oplus f_4.f_3 \\ &\quad \oplus f_4.f_3.f_0 \oplus f_4.f_3.f_1 \oplus f_4.f_3.f_2.f_1 \end{aligned}$$

Although Crypto-1 has already been broken down in 200 seconds through an algebraic attack on a PC, as demonstrated in [40], it has been chosen in our study as an easy primary target to demonstrate as to how partial side channel information based on hamming weight leakage model can improve the solving time in ASCA as compared to algebraic attacks.

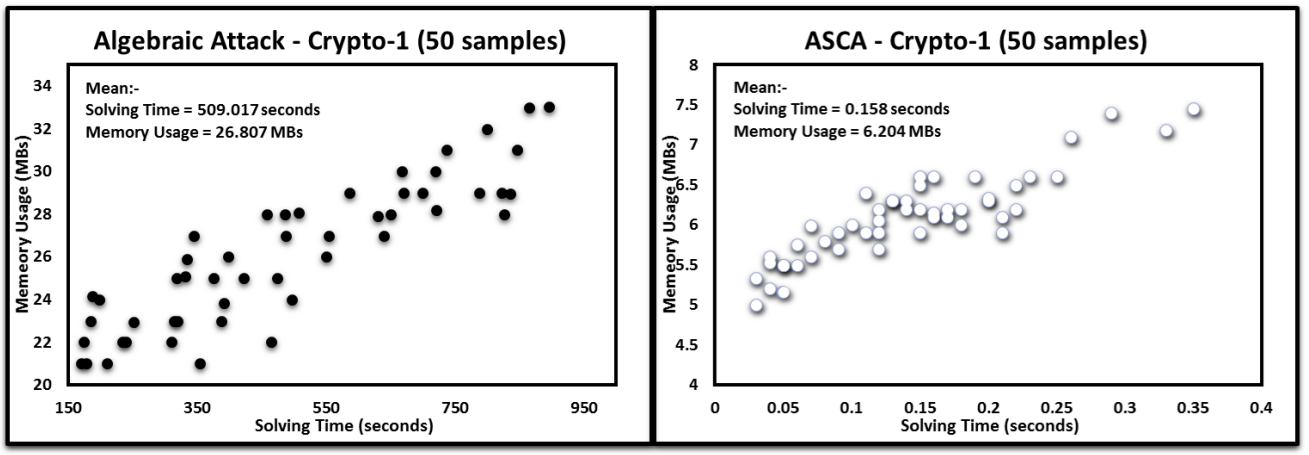

FIGURE 3. A Comparison of Algebraic Attack and ASCA on Crypto-1 Stream Cipher.

In the offline phase, an algebraic attack was launched on the cipher, and 50 random samples of 50-bit output stream were utilized though grain-of-salt tool. It took an average of 509.017 seconds and 26.807 MBs of memory to solve the SAT problem composed of 24, 636 CNF clauses and to correctly assign values to unknown variables through CryptoMiniSat 5.0 on a Linux VM furnished with 2 processors and 2.8 GB memory. In the online phase, a template-like SCA, same as in [23], was simulated to extract hamming weight leakage at the inputs of filter functions f_0, f_1, f_2, f_3, f_4 and final filter function. Algebraic equations can be obtained from hamming weight information through the procedure given in [28]; however, in our study, this information was directly converted to CNF clauses as needed by the SAT solver. For example, for four inputs of function f_0 at time t, if hamming weight $HW(t) = k$, then product of any $k + 1$ or more bits will always be zero; and the value of k can be 0, 1, 2, 3, and 4 (i.e. $|k| = 5$). The possible CNF clauses for various values of k, where variables 1, 2, 3, and 4 were used as input bits to the function f_0, are displayed in Table 1.

Over a thousand CNF clauses thus acquired from SCA were added into the clauses obtained from algebraic attack, and the combined CNF file was inputted to CryptoMiniSAT 5.0 for satisfiability/assignment.

The experimental results of ASCA against the same 50 random samples of 50-bit output stream on the same machine as used for algebraic attack show that it took an average of 0.158 seconds and 6.204 MBs of memory to successfully complete the attack. A comparison drawn between ASCA and algebraic attack on Crypto-1 stream cipher, as demonstrated in Fig 3, clearly indicates the superiority of the former over the latter.

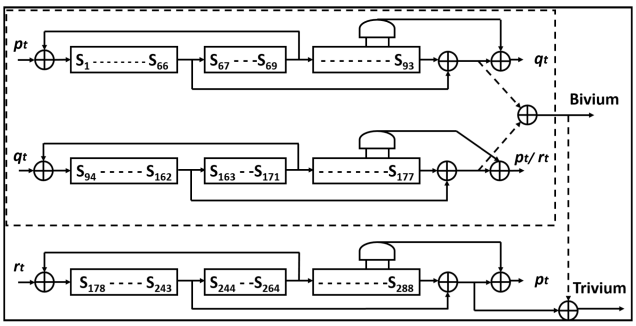

FIGURE 4. Structure of Bivium-B and Trivium.

TABLE 1. Converting HW(t) into CNF clauses.

HW(t)	Implication	CNF Clauses
k=0	all bits are zero	−1 0, −2 0, −3 0, −4 0
k=1	OR of any two or more bits is one	−1 −2 0, −1 −3 0, −1 −4 0, −2 −3 0, −2 −4 0, −3 −4 0, −1 −2 −3 0, −1 −2 −4 0, −1 −3 −4 0, −2 −3 −4 0, −1 −2 −3 −4 0, 1 2 3 4 0
k=2	OR of any three or more bits is one	1 2 3 0, 1 2 4 0, 1 3 4 0, 2 3 4 0, 1 2 3 4 0, −1 −2 −3 0, −1 −2 −4 0, −1 −3 −4 0, −2 −3 −4 0, −1 −2 −3 −4 0
k=3	OR of any four or more bits is one	1 2 0, 1 3 0, 1 4 0, 2 3 0, 2 4 0, 3 4 0, 1 2 3 0, 1 2 4 0, 1 3 4 0, 2 3 4 0, 1 2 3 4 0, −1 −2 −3 −4 0
k=4	all bits are one	1 0, 2 0, 3 0, 4 0,

V. ASCA AGAINST BIVIUM-B AND TRIVIUM

Trivium stream cipher [41] has an internal state of 288 bits, composed of three nonlinear registers of 93, 84, and 111 bits, as illustrated in Fig 4. For initialization, the cipher is clocked $4 * 288$ times; after the 80 bit secret key is loaded into the first register, 80-bit IV is loaded into the second register. The remaining states are loaded with zeros, except the ones in last three bits of the third register. The output stream is produced from a linear output function, which combines six internal state bits. The pseudocode of Trivium is shown in Algorithm 1.

Algorithm 1 Trivium Pseudocode

1 **for** $i \leftarrow 1$ **to** N **do**
2 $\quad t_1 \leftarrow s_{66} \oplus s_{93}$;
3 $\quad t_2 \leftarrow s_{162} \oplus s_{177}$;
4 $\quad t_3 \leftarrow s_{243} \oplus s_{288}$;
5 $\quad z_i \leftarrow t_1 \oplus t_2 \oplus t_3$;
6 $\quad t_1 \leftarrow t_1 \oplus s_{91}.s_{92} \oplus s_{171}$;
7 $\quad t_2 \leftarrow t_2 \oplus s_{175}.s_{176} \oplus s_{264}$;
8 $\quad t_3 \leftarrow t_3 \oplus s_{286}.s_{287} \oplus s_{69}$;
9 $\quad (s_1, s_2, ..., s_{93}) \leftarrow (t_3, s_1, ..., s_{92})$;
10 $\quad (s_{94}, s_{95}, ..., s_{177}) \leftarrow (t_1, s_{94}, ..., s_{176})$;
11 $\quad (s_{178}, s_{179}, ..., s_{288}) \leftarrow (t_2, s_{178}, ..., s_{287})$;
12 **end**

Algebraic attacks on Trivium in [42]–[48] targeted complete internal state bits instead of key. The best overall time and data complexity were found to be $2^{42.5}$ and 2^{12}, respectively, on an average computer in [42] that targeted a round-reduced variant of 625-round Trivium.

Bivium-B stream cipher, a reduced variant of Trivium (Fig 4), has an internal state of 177 bits, composed of two nonlinear registers of 93 and 84 bits, initialized with 80-bit secret key and 80-bit IV, respectively, with zeros at remaining states. The cipher is clocked $4 * 177$ times before the production of the output stream from a linear output function, which combines 4 bits out of 177 internal states [44]. The pseudocode of Bivium-B is shown in Algorithm 2.

Algorithm 2 Bivium-B Pseudocode

1 **for** $i \leftarrow 1$ **to** N **do**
2 $\quad t_1 \leftarrow s_{66} \oplus s_{93}$;
3 $\quad t_2 \leftarrow s_{162} \oplus s_{177}$;
4 $\quad z_i \leftarrow t_1 \oplus t_2$;
5 $\quad t_1 \leftarrow t_1 \oplus s_{91}.s_{92} \oplus s_{171}$;
6 $\quad t_2 \leftarrow t_2 \oplus s_{175}.s_{176} \oplus s_{69}$;
7 $\quad (s_1, s_2, ..., s_{93}) \leftarrow (t_2, s_1, ..., s_{92})$;
8 $\quad (s_{94}, s_{95}, ..., s_{177}) \leftarrow (t_1, s_{94}, ..., s_{176})$;
9 **end**

In [49], the authors have claimed to break Bivium-B in an approximately $2^{36.5}$ seconds using a Xeon E5345 @2.33GHz computer with the help of grain-of-salt tool to generate CNF clauses from pure algebraic attack and to solve them using CryptoMiniSat [50]. To our knowledge, this is the most efficient algebraic attack against Bivium-B that has ever been reported.

Our ASCA on Bivium-B/Trivium targeted 80-bit secret key instead of the complete state of cipher after the initialization, while capturing 80 output bits. The success of ASCA on Bivium-B/Trivium stream ciphers or any other cipher is largely dependent on the availability of leakage points in the hardware implementation of the cipher so that maximum possible side channel information could be extracted through minimum exposure. By looking at the design of both ciphers, the SCA that is similar to the method employed against Crypto-1 cipher appears to offer no advantages. One may attempt to capture hamming weight of registers at each round in a template-like attack and convert them into algebraic equations or CNF clauses; however, due to the large size of the registers, entropy of hamming weights could be very high as well [51]. Therefore, the equations based on hamming weights of registers would not extensively reduce the overall complexity of ASCA. In [22], the authors assessed the susceptibility of Trivium over SCA, and they concluded that its power consumption can be easily described using a hamming distance model. They also highlighted that there is almost no possibility of recovering internal states after the initialization phase, as the key spreads into all the registers; thus, useful side channel information can only be extracted in the initialization phase. This is applicable to Bivium-B as well, as it has the same inherent structure. Moreover, the authors suggested that a DPA can be mounted while focusing on the s_{94} input of second register. The contents of this flip-flop after each round are given by following equation from pseudocode (line 8 for Bivium-B and line 10 for Trivium as above):

$$s_{94}(i+1) = s_{66}(i) \oplus s_{91}(i).s_{92}(i) \oplus s_{93}(i) \oplus s_{171}(i)$$

As we subjected Bivium-B and Trivium ciphers to a known IV attack; therefore only the unknown on the right hand side of the above mentioned equation was $s_{66}(i)$, corresponding to the 66^{th} bit of secret key in round 1. In a DPA, two hypothetical power consumptions of the second register, using both (0 and 1) values for s_{94}, can help in determining its correct value with correlation coefficient. Consequently, an additional equation or few CNF clauses from SCA were obtained, and it can augment the system of equations or CNF clauses obtained through classic algebraic attack.

For the next round, s_{94} will assume a new value, but the rest of the bits of second register were known, thus DPA can be continued in a similar manner and the correct values for s_{94} in the subsequent equations of the succeeding rounds can be placed. How many values of s_{94} or how many round-equations are needed? We aimed to involve all 80 secret key bits in these equations and proceeded until round 66, which is adequate for both Bivium-B and Trivium. The resulting equations based on following expressions were acquired and were either equal to 1 or 0, depending on the value of s_{94} as per DPA outcome:

Round 1: $k_{66} \oplus 0 * 0 \oplus 0 \oplus iv_{78}$
Round 2: $k_{65} \oplus 0 * 0 \oplus 0 \oplus iv_{77}$
Round 3: $k_{64} \oplus 0 * 0 \oplus 0 \oplus iv_{76}$
Round 4: $k_{63} \oplus 0 * 0 \oplus 0 \oplus iv_{75}$
Round 5: $k_{62} \oplus 0 * 0 \oplus 0 \oplus iv_{74}$
.
.
Round 11: $k_{56} \oplus 0 * 0 \oplus 0 \oplus iv_{68}$
Round 12: $k_{55} \oplus k_{80} * 0 \oplus 0 \oplus iv_{67}$

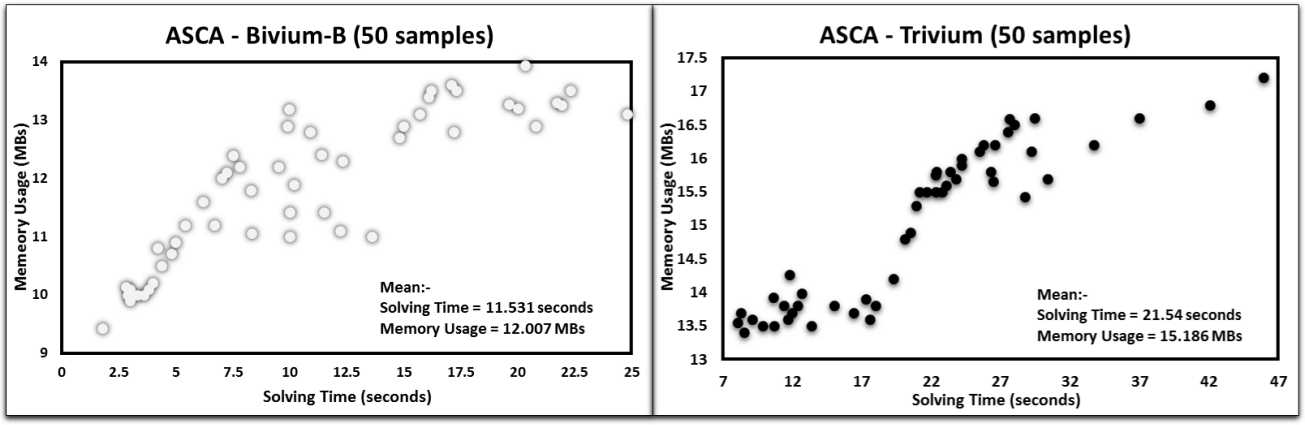

FIGURE 5. ASCA Results against Bivium-B and Trivium Stream Ciphers.

Round 13: $k_{54} \oplus k_{79} * k_{80} \oplus 0 \oplus iv_{66}$
Round 14: $k_{53} \oplus k_{78} * k_{79} \oplus k_{80} \oplus iv_{65}$
Round 15: $k_{52} \oplus k_{77} * k_{78} \oplus k_{79} \oplus iv_{64}$
.
.
.
Round 62: $k_5 \oplus k_{30} * k_{31} \oplus k_{32} \oplus iv_{17}$
Round 63: $k_4 \oplus k_{29} * k_{30} \oplus k_{31} \oplus iv_{16}$
Round 64: $k_3 \oplus k_{28} * k_{29} \oplus k_{30} \oplus iv_{15}$
Round 65: $k_2 \oplus k_{27} * k_{28} \oplus k_{29} \oplus iv_{14}$
Round 66: $k_1 \oplus k_{26} * k_{27} \oplus k_{28} \oplus iv_{13}$

For SCA, only equations from the first 66 rounds were utilized; however, for algebraic attack, the complete initialization phase ($4 * 177$ rounds in case of Bivium-B and $4 * 288$ rounds in case of Trivium) was needed to be included, aside from the 80 additional rounds for 80 output bits. This was done to employ variables introduced for initial state bits in the new equations, which demonstrate the relationship between output and state bits. The inclusion of initialization phase in the algebraic attack poses the disadvantage of dealing with high-degree equations but, at the same time, has an advantage of targeting lesser unknown variables of secret key bits only instead of complete state bits. In ASCA, as leakage information was obtained in the initialization phase, leakage information has to be included in the algebraic attack phase as well.

The CNF clauses obtained from both algebraic and SCA phases were combined and were inputted into CryptoMiniSat 5.0 for satisfiability/assignment.

The results of ASCA against Bivium-B and Trivium, as summarized in Fig 5 and Table 2, were obtained using a Linux machine with Intel Core i3 CPU M350 @2.27GHz and 6 GB memory. It took 11.531 and 21.54 seconds, along with 12.007 and 15.186 MBs of memory, to successfully complete the attack on Bivium-B and Trivium, respectively, while targeting 80-bit secret key on 50 random samples and making use of 80 output stream bits, on the average.

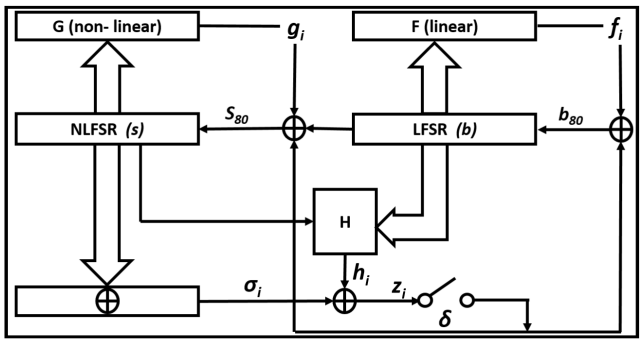

FIGURE 6. Structure of Grain Stream Cipher.

TABLE 2. ASCA Results against Bivium-B and Trivium.

Bivium-B		Trivium	
Solving Time (sec)	# of Samples	Solving Time (sec)	# of Samples
$t < 5$	12	$t < 10$	5
$5 \leq t < 10$	13	$10 \leq t < 20$	15
$10 \leq t < 15$	10	$20 \leq t < 30$	25
$15 \leq t < 20$	8	$30 \leq t < 40$	3
$20 \leq t < 25$	7	$40 \leq t < 50$	2

VI. ASCA AGAINST GRAIN

Grain v1 cipher, illustrated in Fig 6, is composed of an 80-bit NLFSR (denoted as s with state bits $s_0, s_1, \ldots, s_{79}$) and an 80-bit LFSR (denoted as b with state bits $b_0, b_1, \ldots, b_{79}$) which are loaded with the 80-bit secret key and 64-bit IV plus remaining 1's, respectively [52]. The output h_i of a three-degree nonlinear function H, which obtains five input bits from both registers, is XORed with a linear function σ_i to generate the output stream z_i. During initialization phase, which lasts for 160 rounds, $\delta = 1$, hence no output stream is generated and z_i is XORed with the feedback functions of both registers. The following equations govern the operation of the cipher:

- Feedback functions of s and b:
  $s_{80} = b_0 \oplus g_i \oplus z_i.\delta$
  $b_{80} = f_i \oplus z_i.\delta$
- Six-degree nonlinear function g_i:
  $s_{62} \oplus s_{60} \oplus s_{52} \oplus s_{45} \oplus s_{37} \oplus s_{33} \oplus s_{28} \oplus s_{21} \oplus s_{14} \oplus s_9 \oplus s_0 \oplus s_{63}s_{60} \oplus s_{37}s_{33} \oplus s_{15}s_9 \oplus s_{60}s_{52}s_{45} \oplus s_{33}s_{28}s_{21} \oplus s_{63}s_{45}s_{28}s_9 \oplus s_{60}s_{52}s_{37}s_{33} \oplus s_{63}s_{60}s_{21}s_{15} \oplus s_{63}s_{60}s_{52}s_{45}s_{37} \oplus s_{33}s_{28}s_{21}s_{15}s_9 \oplus s_{52}s_{45}s_{37}s_{33}s_{28}s_{21}$
- Linear function f_i:
  $b_{62} \oplus b_{51} \oplus b_{38} \oplus b_{23} \oplus b_{13} \oplus b_0$
- Linear function σ_i:
  $s_1 \oplus s_2 \oplus s_4 \oplus s_{10} \oplus s_{31} \oplus s_{43} \oplus s_{56}$
- Three-degree nonlinear function h_i:
  $b_{25} \oplus s_{63} \oplus b_3b_{64} \oplus b_{46}b_{64} \oplus b_{64}s_{63} \oplus b_3b_{25}b_{46} \oplus b_3b_{46}b_{64} \oplus b_3b_{46}s_{63} \oplus b_{25}b_{46}s_{63} \oplus b_{46}b_{64}s_{63}$
- Output stream z_i:
  $\sigma_i \oplus h_i$

However, no adequate researches on algebraic cryptanalysis of Grain v1 have been accomplished. Although Grain v1 has been subjected to other attacks, such as those in [53]–[55], only one instance of pure algebraic attack against Grain v1 has been published [17], which claimed to have solved Grain v1 equations in $2^{80.70}$ seconds, using the Groebner bases technique.

Similar to the attacks on other ciphers in previous sections, Grain v1 was converted into CNF clauses through the help of grain-of-salt tool using 80 output stream bits [49]. The initialization phase, which consisted of 160 rounds, was included in the attack, as we intended to directly target the secret key instead of targeting complete 160 state bits.

In [22], the authors mentioned that Grain v1 is susceptible to SPA in case of bit-serialized implementations through the subsequent measurement of key bits hamming distances after resetting to a defined state at key setup. However, in [18], Fischer *et al.* mounted a successful DPA with chosen IV while attacking the key setup process and thereby learning the key bits iteratively. The authors in [22] also mentioned that a DPA could possibly be mounted specifically before or after the output function based on hamming distance model.

A. UNSUCCESSFUL ATTEMPT

In this attempt in a known IV scenario, new incoming bit of LFSR in each round was targeted knowing that all 80 bits of the LFSR b are initially known at round 0. A correlation DPA can measure the value of new bit in the subsequent rounds through all known LFSR bits in the previous round, which are as follows:

- From the equations given above, it can be observed that in round 1, b_{80} would be equal to:
  $b_{62} \oplus b_{51} \oplus b_{38} \oplus b_{23} \oplus b_{13} \oplus b_0 \oplus s_1 \oplus s_2 \oplus s_4 \oplus s_{10} \oplus s_{31} \oplus s_{43} \oplus s_{56} \oplus b_{25} \oplus s_{63} \oplus b_3b_{64} \oplus b_{46}b_{64} \oplus b_{64}s_{63} \oplus b_3b_{25}b_{46} \oplus b_3b_{46}b_{64} \oplus b_3b_{46}s_{63} \oplus b_{25}b_{46}s_{63} \oplus b_{46}b_{64}s_{63}$
- All the bits coming from LFSR with suffix b were known from the previous round, therefore the value of b_{80} becomes:
  $s_1 \oplus s_2 \oplus s_4 \oplus s_{10} \oplus s_{31} \oplus s_{43} \oplus s_{56} \oplus s_{63} \oplus b_{64}s_{63} \oplus b_3b_{46}s_{63} \oplus b_{25}b_{46}s_{63} \oplus b_{46}b_{64}s_{63} \oplus k_1$
- b_{80} can be simplified further as:
  $s_1 \oplus s_2 \oplus s_4 \oplus s_{10} \oplus s_{31} \oplus s_{43} \oplus s_{56} \oplus s_{63}.k_2 \oplus k_1$
- If the value of b_{80} was measured as b through correlation DPA, then one of the two possible equations can be obtained for each round during the initialization phase as either:
  - For $k_2 = 1$: $s_1 \oplus s_2 \oplus s_4 \oplus s_{10} \oplus s_{31} \oplus s_{43} \oplus s_{56} \oplus s_{63} = b \oplus k_1$
  - For $k_2 = 0$: $s_1 \oplus s_2 \oplus s_4 \oplus s_{10} \oplus s_{31} \oplus s_{43} \oplus s_{56} = b \oplus k_1$
- After the initialization phase, the value of b_{80} was only determined from LFSR bits as under:
  $b_{62} \oplus b_{51} \oplus b_{38} \oplus b_{23} \oplus b_{13} \oplus b_0$
  Therefore side channel equations targeting b_{80} would only be useful until the end of initialization phase.

One-hundred sixty equations thus obtained through SCA were added into the CNF clauses already obtained. However, none of the 50 samples resulted into a satisfiable solution within our time limit, which is 3,600 seconds.

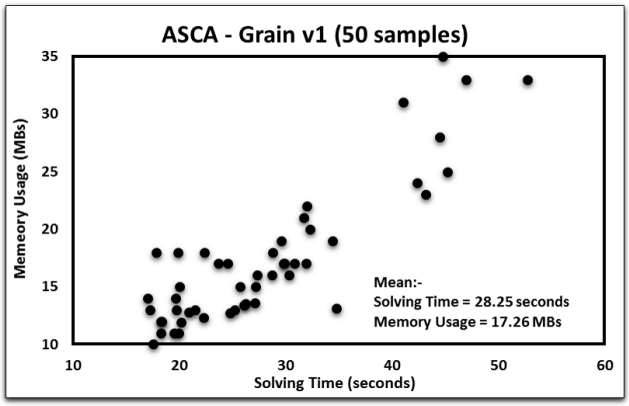

FIGURE 7. ASCA Results against Grain v1 Stream Cipher.

TABLE 3. ASCA results against Grain v1.

Solving Time (sec)	# of Samples	Memory Usage (MBs)	# of Samples
$10 \leq t < 20$	12	$10 \leq m < 15$	21
$20 \leq t < 30$	21	$15 \leq m < 20$	18
$30 \leq t < 40$	9	$20 \leq m < 25$	5
$40 \leq t < 50$	7	$25 \leq m < 30$	2
$50 \leq t < 60$	1	$30 \leq m < 35$	4

B. SUCCESSFUL ATTEMPT

As per our attack model, shown in Fig 1 in Section III, in case of unsuccessful or inefficient solving, one must revisit either of the constituent attack phases. In this case, we changed the leakage points in partial SCA phase and captured the leakage at the input of function h_i. This kind of Grain v1 susceptibility was also pointed out by authors in [22]; however, no study regarding practical SCA on Grain v1 exploiting this leakage has been published so far. We simulated a template-like SCA at the input of h_i using a hamming weight leakage model [21].

TABLE 4. Comparison of full SCA and partial SCA.

Cipher	Full SCA Reference	Partial SCA (this work)		
		Leakage points	**Leakage model/ Simulated Attack Type**	**CNF clauses**
Crypto-1	Nil	Input of filter functions	Hamming weight, Template attack, Single trace	1234 (average)
Bivium-B	Nil	1st NLFSR	Known IV, Correlation DPA targeting 1st NLFSR	66 XOR clauses
Trivium	Correlation DPA, Hamming distance model, 550 traces [20]; CPA, Resynchronization phase, chosen IV, 256 traces [56]; Chosen IV, DPA, Resynchronization phase, theoretical [18]	1st NLFSR	Known IV, Correlation DPA targeting 1st NLFSR	66 XOR clauses
Grain v1	Correlation DPA, Hamming distance model, 2600 traces [20]; Chosen IV, DPA, 256 traces per IV [18]	LFSR	Known IV, Correlation DPA targeting LFSR	160 XOR clauses
		Input of function h_i	Hamming weight, Template attack, Single trace	240 XOR clauses

A similar technique was applied by authors in [23] while applying ASCA against PRESENT block cipher when they captured the hamming weight leakage at the input and output of its S-boxes through template attacks.

Input bits at function h_i during various rounds are:

Round 1: $b_3, b_{25}, b_{46}, b_{64}, s_{63}$

Round 2: $b_4, b_{26}, b_{47}, b_{65}, s_{64}$

Round 3: $b_5, b_{27}, b_{48}, b_{66}, s_{65}$

...

Hamming weight information of these bits can either be converted to algebraic equations as explained in [28] or be directly converted to CNF clauses as explained in Section IV. Here, it was directly converted into CNF clauses and was added to the previously obtained CNF clauses from algebraic attack. Moreover, at this instance, the CryptoMiniSAT 5.0 was able to give satisfiability/assignment, in an average of 28.25 seconds using 17.26 MBs of memory for 50 samples using the same machine as used for Trivium above. The summary of results is presented in Fig 7 and Table 3.

VII. ANALYSIS OF RESULTS

The successful application of ASCA against popular stream ciphers, such as Trivium and Grain, as demonstrated above, proves the efficacy of our proposed technique and model. Lone algebraic attacks were never successful against the full versions of either Trivium or Grain stream ciphers, and it can be attributed to the high complexity of the generated MQ problem. This high complexity is the consequence of built-in nonlinearity in the design of such stream ciphers. Algebraic attack phase, as a constituent of ASCA against Trivium and Grain, was mounted until the formulation of multivariate quadratic equations. The resultant MQ problem was computationally infeasible for solving using any existing mathematical method or tool. The main advantage of ASCA over pure algebraic attack is the additional equations obtained from SCA. SCA is only possible if leakage information from a cipher implementation is available. It is next to impossible to implement a stream cipher without any sort of leakage. This leakage information, once converted into algebraic equations or CNF clauses and included into overall MQ problem or SAT problem, may successfully render ASCA as computationally feasible.

Tables 4 and 5 summarize the results of our experiments on Crypto-1, Bivium-B, Trivium and Grain v1 and show the comparison with full-blown SCA and lone algebraic attacks respectively.

Table 4 highlights the comparison between the previously launched full-blown SCA with partial SCA information for ASCA against the stream ciphers under discussion. The SCA leakage model we employed in our study is as per the previous works. However, the extent of partial SCA information, which can be ascertained from the number of CNF clauses obtained in the process, is less as compared to complete SCA, which is unable to singularly find the secret key or internal state. The limited side channel information in our study is satisfactory. Traditional SCA targets complete key bits by demanding continuous access to the implementation device to acquire a greater number of power traces. However, partial SCA, as part of ASCA, needs limited access to the hardware device to acquire single or few traces as it has an easier target of a few relations translating into algebraic equations or CNF clauses. These supporting equations or clauses further defined and simplified the system of equations/clauses obtained through algebraic attack.

TABLE 5. Comparison of Lone Algebraic Attacks and ASCA.

Cipher	Known Algebraic Attack (seconds)	ASCA (this work)			
		Known Output bits	Init. Phase Included?	Combined CNF: # of Var, Average # of (Algebraic+SCA clauses)	Solving Time (sec)
Crypto-1	200 [57]	50	No	1541, 24636+1234	0.158
Bivium-B	$2^{36.5}$ [49]	80	Yes	3006, 5282+66 XOR	11.531
Trivium	$2^{42.5}$, 625 round-reduced version [42]	80	Yes	6975, 12640+66 XOR	21.54
Grain v1	$2^{80.7}$ [17]	80	Yes	4549, 16490+160 XOR	>3600
		80	Yes	4549, 16490+240 XOR	28.25

The comparison of lone algebraic attacks with the ASCA on the stream ciphers is presented in Table 5. It elucidates the colossal reduction of the complexity of overall ASCA, in which only a few CNF clauses or algebraic equations from partial SCA were added into those obtained from lone algebraic attack. The addition of equations/clauses from partial SCA not only makes the system of equations excessively defined but it also simplified the equations in terms of degree and, occasionally, in terms of the number of unknowns. There may be a case where partial SCA information is inadequate and thus the MQ problem or SAT problem would remain intractable. In such cases, either the amount of partial SCA information may be increased or the SCA leakage points/model may be changed altogether; and this was amply demonstrated while attacking Grain cipher, wherein the first ASCA attempt was not able to yield results within the time limit.

VIII. CONCLUSIONS AND FUTURE WORK

ASCA against stream ciphers is proposed in our study and applied on Crypto-1, Bivium-B, Trivium, and Grain v1 to successfully recover the secret key. It can be observed that in the application of ASCA, the information obtained through SCA can be quite small, as it was obtained from only a few samples. This is different from traditional SCAs wherein the complete key is targeted with little to no knowledge on algorithm. The selection or availability of leakage points in the implementation of stream cipher and the judicious utilization of side channel leakage information into MQ problem or SAT problem developed from algebraic attack are central to the success of ASCA. This necessitates thorough knowledge of the target cipher algorithm. On the other hand, it is of great importance to note that incorrect side channel leakage information would definitely render the problem unresolvable. Therefore, error-tolerant techniques, as in the case of ASCA on block ciphers, are equally essential for successful ASCA against stream ciphers. Furthermore, the possibility of ASCA on stream ciphers necessitates the research on countermeasures that could thwart it.

REFERENCES

[1] N. T. Courtois and W. Meier, "Algebraic attacks on stream ciphers with linear feedback," in *Advances in Cryptology—EUROCRYPT* (LNCS). Warsaw, Poland: Springer, 2003, pp. 346–359.
[2] F. Armknecht, "Improving fast algebraic attacks," in *Proc. Int. Workshop Fast Softw. Encryption*, 2004, pp. 65–82.
[3] F. Armknecht and M. Krause, "Algebraic attacks on combiners with memory," in *Proc. Annu. Int. Cryptol. Conf.*, 2003, pp. 162–175.
[4] N. T. Courtois, "Higher order correlation attacks, XL algorithm and cryptanalysis of toyocrypt," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, 2002, pp. 182–199.
[5] N. T. Courtois, "Fast algebraic attacks on stream ciphers with linear feedback," in *Proc. Annu. Int. Cryptol. Conf.*, 2003, pp. 176–194.
[6] N. T. Courtois, "Algebraic attacks on combiners with memory and several outputs," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, 2004, pp. 3–20.
[7] A. Kipnis and A. Shamir, "Cryptanalysis of the HFE public key cryptosystem by relinearization," in *Proc. Annu. Int. Cryptol. Conf.*, 1999, pp. 19–30.
[8] N. Courtois, A. Klimov, J. Patarin, and A. Shamir, "Efficient algorithms for solving overdefined systems of multivariate polynomial equations," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, 2000, pp. 392–407.
[9] N. T. Courtois and J. Pieprzyk, "Cryptanalysis of block ciphers with overdefined systems of equations," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2002, pp. 267–287.
[10] B. Buchberger and F. Winkler, *Gröbner Bases and Applications*, vol. 251. Cambridge, U.K.: Cambridge Univ. Press, 1998.
[11] A. Braeken, J. Lano, N. Mentens, B. Preneel, and I. Verbauwhede, "SFINKS: A synchronous stream cipher for restricted hardware environments," in *Proc. SKEW-Symmetric Key Encryption Workshop*, vol. 55. 2005, p. 72.
[12] Y. Nawaz and G. Gong, "The WG stream cipher," ECRYPT Stream Cipher Project, Tech. Rep. 2005, 2005, vol. 33. [Online]. Available: http://cr.yp.to/streamciphers/wg/desc.pdf
[13] M. Robshaw and O. Billet, Eds., *New Stream Cipher Designs*. Berlin, Germany: Springer-Verlag, 2008.
[14] M. Afzal and A. Masood, "Resistance of stream ciphers to algebraic recovery of internal secret states," in *Proc. 3rd Int. Conf. Converg. Hybrid Inf. Technol. (ICCIT)*, vol. 2. Nov. 2008, pp. 625–630.
[15] S. Al Hinai, L. M. Batten, and B. Colbert, "Mutually clock-controlled feedback shift registers provide resistance to algebraic attacks," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, 2007, pp. 201–215.
[16] P. Datta, D. Roy, and S. Mukhopadhyay, "A probabilistic algebraic attack on the grain family of stream ciphers," in *Proc. Int. Conf. Netw. Syst. Secur.*, 2014, pp. 558–565.
[17] M. Afzal and A. Masood, "Algebraic cryptanalysis of a NLFSR based stream cipher," in *Proc. 3rd Int. Conf. Inf. Commun. Technol., From Theory Appl. (ICTTA)*, Apr. 2008, pp. 1–6.
[18] W. Fischer, B. M. Gammel, O. Kniffler, and J. Velten, "Differential power analysis of stream ciphers," in *Proc. Cryptographers' Track RSA Conf.*, 2007, pp. 257–270.
[19] C. Rechberger and E. Oswald, "Stream ciphers and side-channel analysis," in *Proc. ECRYPT Workshop, SASC-State Art Stream Ciphers*, 2004, pp. 320–326.

[20] D. Strobel, I. C. Paar, and M. Kasper, "Side channel analysis attacks on stream ciphers," Lehrstuhl Embedded Secur., Masterarbeit Ruhr-Univ. Bochum, Bochum, Germany, Tech. Rep., 2009. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.182.1943&rep=rep1&type=pdf and https://pdfs.semanticscholar.org/845d/583597c934e2671731c664f036ec977fbc23.pdf

[21] S. Chari, J. R. Rao, and P. Rohatgi, "Template attacks," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, 2002, pp. 13–28.

[22] B. Gierlichs *et al.*, "Susceptibility of eSTREAM candidates towards side channel analysis," in *Proc. SASC*, 2008, pp. 123–150.

[23] M. Renauld and F.-X. Standaert, "Algebraic side-channel attacks," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, 2009, pp. 393–410.

[24] A. Bogdanov *et al.*, "PRESENT: An ultra-lightweight block cipher," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, 2007, pp. 450–466.

[25] M. Renauld, F. X. Standaert, and N. Veyrat-Charvillon, "Algebraic side-channel attacks on the aes: why time also matters in DPA," in *Cryptographic Hardware and Embedded Systems—CHES* (Lecture Notes in Computer Science), vol 5747, C. Clavier and K. Gaj, Eds. Berlin, Germany: Springer, 2009.

[26] Y. Oren, M. Kirschbaum, T. Popp, and A. Wool, "Algebraic side-channel analysis in the presence of errors," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, 2010, pp. 428–442.

[27] X. Zhao *et al.*, "MDASCA: An enhanced algebraic side-channel attack for error tolerance and new leakage model exploitation," in *Proc. Int. Workshop Constructive Side-Channel Anal. Secure Design*, 2012, pp. 231–248.

[28] C. Carlet, J.-C. Faugère, C. Goyet, and G. Renault, "Analysis of the algebraic side channel attack," *J. Cryptograph. Eng.*, vol. 2, no. 1, pp. 45–62, May 2012.

[29] M. S. E. Mohamed, S. Bulygin, M. Zohner, A. Heuser, M. Walter, and J. Buchmann, "Improved algebraic side-channel attack on AES," *J. Cryptograph. Eng.*, vol. 3, no. 3, pp. 139–156, Sep. 2013.

[30] Y. Oren, M. Renauld, F.-X. Standaert, and A. Wool, "Algebraic side-channel attacks beyond the Hamming weight leakage model," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, 2012, pp. 140–154.

[31] Y. Oren and A. Wool, "Tolerant algebraic side-channel analysis of AES," IACR Cryptol. ePrint Arch., Tech. Rep., 2012, p. 92. [Online]. Available: https://eprint.iacr.org/2012/092.pdf

[32] Y. Oren, O. Weisse, and A. Wool, "Practical template-algebraic side channel attacks with extremely low data complexity," in *Proc. 2nd Int. Workshop Hardw. Archit. Support Secur. Privacy*, 2013, p. 7.

[33] L. Song, L. Hu, S. Sun, Z. Zhang, D. Shi, and R. Hao, "Error-tolerant algebraic side-channel attacks using BEE," in *Proc. Int. Conf. Inf. Commun. Secur.*, 2014, pp. 1–15.

[34] Y. Oren and A. Wool, "Side-channel cryptographic attacks using pseudo-boolean optimization," *Constraints*, vol. 21, no. 4, pp. 616–645, Oct. 2016.

[35] P. C. Kocher, "Timing attacks on implementations of Diffie–Hellman, RSA, DSS, and other systems," in *Proc. Annu. Int. Cryptol. Conf.*, 1996, pp. 104–113.

[36] P. C. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proc. Annu. Int. Cryptol. Conf.*, 1999, pp. 388–397.

[37] J. Lano, N. Mentens, B. Preneel, and I. Verbauwhede, "Power analysis of synchronous stream ciphers with resynchronization mechanism," in *Proc. ECRYPT Workshop, SASC–State Art Stream Ciphers*, 2004, pp. 327–333.

[38] C. Rechberger, "Side channel analysis of stream ciphers," Tech. Rep., 2004. [Online]. Available: https://pdfs.semanticscholar.org/ed53/d2ed8600eeb946cc17c4029bd92808298177.pdf

[39] A. R. Kazmi, M. Afzal, M. F. Amjad, and A. Rashdi, "Combining algebraic and side channel attacks on stream ciphers," in *Proc. Int. Conf. Commun. Technol. (ComTech)*, Apr. 2017, pp. 138–142.

[40] K. Nohl, "Cryptanalysis of crypto-1," Dept. Comput. Sci., Univ. Virginia, Charlottesville, VA, USA, White Paper, 2008. [Online]. Available: http://www.proxmark.org/files/Documents/13.56%20MHz%20-%20MIFARE%20Classic/Cryptanalysis.of.Crypto-1.pdf

[41] C. De Cannière, "Trivium: A stream cipher construction inspired by block cipher design principles," in *Information Security—ISC* (Lecture Notes in Computer Science), vol. 4176, S. K. Katsikas, J. López, M. Backes, S. Gritzalis, and B. Preneel, Eds. Berlin, Germany: Springer, 2006.

[42] F.-M. Quedenfeld and C. Wolf, "Advanced algebraic attack on Trivium," in *Proc. Int. Conf. Math. Aspects Comput. Inf. Sci.*, 2015, pp. 268–282.

[43] C. McDonald, C. Charnes, and J. Pieprzyk, "An algebraic analysis of trivium ciphers based on the Boolean satisfiability problem," in *Proc. 4th Int. Workshop Boolean Funct., Cryptogr. Appl.*, 2008, pp. 173–184.

[44] H. Raddum, "Cryptanalytic results on Trivium," eSTREAM, ECRYPT Stream Cipher Project, Tech. Rep., 2006, vol. 39, p. 2006. [Online]. Available: http://www.ecrypt.eu.org/stream/papersdir/2006/039.ps

[45] T. E. Schilling and H. Raddum, "Analysis of Trivium using compressed right hand side equations," in *Proc. Int. Conf. Inf. Secur. Cryptol.*, 2011, pp. 18–32.

[46] I. Simonetti, J.-C. Faugère, and L. Perret, "Algebraic attack against Trivium," in *Proc. 1st Int. Conf. Symbolic Comput. Cryptogr. (SCC)*, vol. 8. 2008, pp. 95–102.

[47] S.-G. Teo, K. K.-H. Wong, H. Bartlett, L. Simpson, and E. Dawson, "Algebraic analysis of Trivium-like ciphers (poster)," in *Proc. 12th Austral. Inf. Secur. Conf.*, vol. 149. 2014, pp. 77–81.

[48] M. Afzal and A. Masood, "Experimental results on algebraic analysis of trivium and tweaked trivium," in *Global E-Security* (Communications in Computer and Information Science), vol. 12, H. Jahankhani, K. Revett, and D. Palmer-Brown, Eds. Berlin, Germany: Springer, 2008.

[49] M. Soos, "Grain of salt—An automated way to test stream ciphers through SAT solvers," *Proc. Tools*, vol. 10. 2010, pp. 131–144.

[50] M. Soos. (2009). *CryptoMiniSat—A Sat Solver for Cryptographic Problems*. [Online]. Available: http://www. msoos.org/cryptominisat4

[51] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: Wiley, 2012.

[52] M. Hell, T. Johansson, A. Maximov, and W. Meier, "The grain family of stream ciphers," in *New Stream Cipher Designs*. Springer, 2008, pp. 179–190.

[53] S. Banik, S. Maitra, and S. Sarkar, "A differential fault attack on the grain family of stream ciphers," in *Proc. Int. Workshop Cryptograph. Hardw. Embedded Syst.*, 2012, pp. 122–139.

[54] S. Karmakar and D. R. Chowdhury, "Fault analysis of grain-128 by targeting NFSR," in *Proc. Int. Conf. Cryptol. Africa*, 2011, pp. 298–315.

[55] H. Zhang and X. Wang, "Cryptanalysis of stream cipher grain family," IACR Cryptol. ePrint Arch., Tech. Rep., 2009, p. 109. [Online]. Available: https://eprint.iacr.org/2009/109.pdf

[56] Y. Jia, Y. Hu, F. Wang, and H. Wang, "Correlation power analysis of Trivium," *Secur. Commun. Netw.*, vol. 5, no. 5, pp. 479–484, May 2012.

[57] N. T. Courtois, K. Nohl, and S. O'Neil, "Algebraic attacks on the crypto-1 stream cipher in MiFare classic and oyster cards," IACR Cryptol. ePrint Arch., Tech. Rep., 2008, p. 166. [Online]. Available: https://eprint.iacr.org/2008/166.ps

ASIF RAZA KAZMI received the master's degree in information security from the College of Signals, National University of Sciences and Technology, Pakistan. His areas of interest are cryptology, malware analysis, and vulnerability exploitation/ defense.

MEHREEN AFZAL received the Ph.D. degree from the National University of Sciences and Technology, Pakistan. She is currently associated with the College of Signals, National University of Sciences and Technology. Her areas of interests are information security and cryptology.

MUHAMMAD FAISAL AMJAD (SM'16) received the Ph.D. degree in computer science from the University of Central Florida, USA, in 2015. He is currently an Assistant Professor with the Department of Electrical Engineering, National University of Sciences and Technology, Pakistan, where he is also associated with the Center of Data and Text Engineering and Mining. His current research focusses on the application of machine learning and game theoretic techniques in the domains of IoT and network security, digital forensics, and malware analysis. He specializes in dynamic spectrum access and defense against security vulnerabilities in cognitive radio networks and wireless sensor and ad hoc networks.

HAIDER ABBAS (SM'15) received the M.S. degree in engineering and management of information systems and the Ph.D. degree in information security from the KTH-Royal Institute of Technology, Stockholm, Sweden, in 2006 and 2010, respectively. His professional career consists of activities ranging from research and development and industry consultations (government and private), through multi-national research projects, research fellowships, doctoral studies advisory services, international journal editorships, conferences/workshops chair, invited/keynote speaker, technical program committee member, and reviewer for several international journals and conferences. He is currently a Cyber Security Professional, an Academician, a Researcher, and an Industry Consultant who took professional trainings and certifications from the Massachusetts Institute of Technology, USA; Stockholm University, Sweden; the Stockholm School of Entrepreneurship, Sweden; IBM, USA; and the EC Council. He is also an Adjunct Faculty and Doctoral Studies Advisor at the Florida Institute of Technology, USA.

In recognition of his services to the international research community and excellence in professional standing, he has been awarded one of the youngest Fellows of the Institution of Engineering and Technology, U.K.; a fellow of the British Computer Society, U.K.; and a fellow of the Institute of Science and Technology, U.K.

XIAODONG YANG (SM'17) has published over 30 peer-reviewed journal papers in highly ranked journals. His research interests include body area networks and information security. He has a global collaborative research network in the field of information security, body area networks, and health informatics. He is on the editorial board of several prestigious journals.