SAT Attacks on ARX Ciphers with Automated Equations Generation

Michal Andrzejczak[1] and Wladyslaw Dudzic[2]

***Abstract*— We propose a novel and simple approach to algebraic attack on block ciphers with the SAT-solvers. As opposed to a standard approach, the equations for key expansion algorithms are not included in the formulas that are converted to satisfiability problem. The lack of equations leads to finding the solution much faster. The method was used to attack a lightweight block ciphers - SIMON and SPECK. We report the timings for round-reduced versions of selected ciphers and discuss the potential factors affecting the execution time of our attack.**

***Index Terms*—SAT attacks, cryptanalysis, ARX ciphers, SIMON, SPECK, FPGA.**

I. Introduction

Block ciphers are the essential elements in communication security, providing secrecy of exchanged information at expected security level. From time to time, a new cipher is proposed, usually offering some new functionality. For example, authenticated encryption, or a new property like smaller hardware requirements. The lightweight cryptography is a recently growing area of research. It'-s aim is to deliver secure ciphers suitable for embedded device development, especially Internet of Things (IoT) devices. Due to hardware limitations, lightweight cipher usually consists mostly of basic logic operations.

The proper cipher cryptanalysis is a key part of cipher development, as well as vital for proving the targeted security strength of the cipher. To date, many techniques for cryptanalysis have been introduced. Thus, it takes time to prove the security of the cipher, since every new cipher must resist against all known attacks.

Previously, several ciphers were broken by a new kind of attack, which was not known at the time of cipher development. For example, Data Encryption Standard (DES) was broken by algebraic cryptanalysis with SAT solvers [1]. Thus, effort put into searching for new methods and accelerations to known ones may be beneficial.

The recent development and improvements in SAT solvers have led to a new algebraic attack on block ciphers. The current state of SAT solvers lowers the total time of the key recovery attack for a new set of ciphers directly affecting cipher's security, especially the lightweight ones.

This work was not supported by any organization.

[1,2] Institute of Mathematics and Cryptology Military University of Technology Warsaw, Poland.

[1] E-mail: michal.andrzejczak@wat.edu.pl

[2] E-mail: wladyslaw.dudzic@wat.edu.pl

A. Contribution

In this paper we present our novel approach to constructing an SAT attack on lightweight block ciphers. We report the results for an SAT attack with our method on ARX ciphers: SIMON and SPECK [2]. The method for obtaining equations for describing the cipher, which are required for an algebraic attack is presented with several sources of equations being listed. We describe our attack approach, considering many factors influencing the time of the attack — several equations, the form of equations, the number of known and used plaintex-ciphertext pairs and the used SAT solver [3]. By analyzing our results, we propose the best approach to break SIMON and SPECK with algebraic cryptanalysis. This approach can be also applied to other ciphers.

II. Previous work

SIMON and SPECK resistance against differential and linear cryptanalysis has been thoroughly investigated in [4], [5] and [6]. SAT attacks are widely used in cryptography, often as a supporting method for other classical attacks like linear or differential cryptanalysis.

SIMON and SPECK have gained cryptanalyst's attention and as a result, several papers about security of the mentioned ciphers were published. In 2014, Curtois et.al [7] presented an algebraic attack combined with a truncated differentials attack on SIMON. They were able to conduct a practical and successful attack on nine rounds of SIMON. However, the attack is more distinguished and requires additional time spent on searching for proper truncated differentials. Found differentials are provided to a system of equations as a plaintext-ciphertext pairs. In the next step, an SAT attack is conducted.

The most recent results have been published by Ren and Chen [8]. They report the first zero-correlation linear attack and integral attack on 11 rounds of SPECK. To conduct the attack, they have used an SAT-based model to search for impossible differentials and zero-correlation linear hulls.

III. Algebraic cryptanalysis

Algebraic cryptanalysis is an attack method on a large subset of ciphers [9]. It consist of two main steps. The first step relies on converting the cipher into a system of polynomial equations, usually over $GF(2)$, but not limited to this particular ring. In the second step, the system of equations is being solved to obtain a proper secret key used for encrypting the provided plaintext-ciphertext pair. There

DOI: 10.36244/ICJ.2019.4.1

are several approaches for solving the system of equations, ranging from XL algorithm and Gröbner basis [10] to SAT-solvers. A brief overview of algebraic cryptanalysis is provided by Bard [11]. This technique is also used for cryptanalysis of the hash function [12].

IV. SIMON AND SPECK CIPHERS

SIMON and SPECK [2] ciphers are members of the ARX ciphers family. The only operations used in ARX ciphers are simple logic operations: AND, rotate and XOR. The ciphers are lightweight and suitable for implementation on constrained devices. SIMON and SPECK both are Feistel type ciphers.

A. SIMON

SIMON's round function is as follows. For a round key k, the round function is a two stage Feistel map $R_k : GF(2)^n \times GF(2)^n \leftarrow GF(2)^n \times GF(2)^n$ defined as:

$$R_k(x, y) = (y \oplus f(x) \oplus k, x), \tag{1}$$

where $f(x) = (Sx \& S^8 x) \oplus S^2 x$, and S^j is a left circular shift by j elements. The SIMON cipher family has several possible data block and key sizes. For each possible combination the number of rounds also varies. All versions with parameters of SIMON ciphers are listed in Tab. I.

B. SPECK

The SPECK family of block ciphers is constructed only from bitwise XOR, addition modulo 2^n and similar to SIMON family, left circular shift S^j by j positions. The round function is the map $R_k : GF(2)^n \times GF(2)^n \leftarrow GF(2)^n \times GF(2)^n$ defined as:

$$R_k(x, y) = ((S^{-\alpha}x + y) \oplus k, S^{\beta}y \oplus (S^{-\alpha} + y) \oplus k), \tag{2}$$

where k is a round key and $\alpha = 7$ and $\beta = 2$ for block size $n = 32$ and $\alpha = 8$ and $\beta = 3$ otherwise. The specification of all block ciphers from SPECK family is presented in Tab. I

TABLE I
POSSIBLE VARIANTS OF SIMON AND SPECK BLOCK CIPHER

block size	key size	word size	# of SIMON rounds	# of SPECK rounds
32	64	16	32	22
48	72	24	36	22
	96		36	23
64	96	32	42	26
	128		44	27
96	128	48	52	28
	144		54	29
128	128	64	68	32
	192		69	33
	256		72	34

SIMON and SPECK have a low multiplicative complexity, which is a one of the measurement units of non-linearity [13]. Thus, algebraic cryptanalysis seems to be promising.

V. SAT ATTACK ON SIMON AND SPECK

We present a known plaintext attack for lightweight block ciphers. The attack partially belongs to the algebraic cryptanalysis family. The proposed attack starts with obtaining proper algebraic equations describing a chosen cipher, which is necessary for a key recovery attack. Classical algebraic cryptanalysis methods try to solve given equations with XL algorithm [14] or Grobner basis [10]. There are also attempts to minimize the number of variables by using external tools [15] [16]. In these algorithms, the attacker usually tries to gather some additional information from equations or tries to decrease the degree of equations and number of variables before solving the system of equations. In our approach, we do not solve the equations directly. Instead, we convert the equations to a satisfiability problem and we try to find a key's values that are valid for used pairs of plaintext and ciphertext. There are several factors affecting the execution time and probability of success. We consider them to find the best approach for attacking SIMON and SPECK with SAT—solvers. Compared with other SAT attacks on SIMON and SPECK our method does not require puting an additional effort into selecting a proper plaintext and does not require any additional tools for minimizing the number of variables. Used pairs are picked up at random and the equations are taken as they were produced by different compilers.

A. Attack model

In our attack we use two different approaches for constructing the attack model.

In the first scenario, equations for the cipher and key expansion algorithms are used by an SAT-solver. The found keys are usually valid if the number of used pairs is larger than two.

For the second scenario, equations for the key expansion algorithm are not included in the system of equations converted to a satisfiability problem. With this approach, the round keys are found in reduced time, compared to the first scenario, but the round keys might be unrelated. The unrelated keys are not valid, and they are a random keys that can not be a result of a key expansion algorithm. The probability of finding a valid key depends mostly on the number of plaintext-ciphertext pairs.

B. Number of pairs

The solution found by an SAT-solver in the second scenario is not always the valid key, used for encryption. The probability of success depends mostly on the number of used pairs. In theory, only one pair should be needed to solve the equations. This is true, the solution for an SAT-problem is found even for one pair and it is done very quickly. However, without equations for a key expansion included to the model, it is possible to find invalid unrelated round keys that solve the satisfiability problem. The probability of finding a valid key increases with the number of used pairs.

The number of pairs also affects the execution time. In Fig. 1 , Fig. 2 and Fig. 3 the time required for solving an SAT problem depens on the number of used pairs. For simple problems, additional pairs extended the running time. There is a visible cross point, where adding more pairs extend the time required for finding solution. For five rounds of SPECK, the

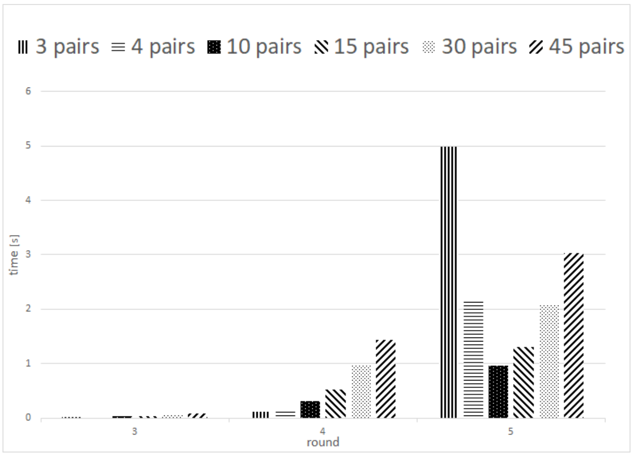

Fig. 1. Average time of attack using Lingeling and different number of pairs of SPECK 32/64 (with key expand algorithm and handmade equations)

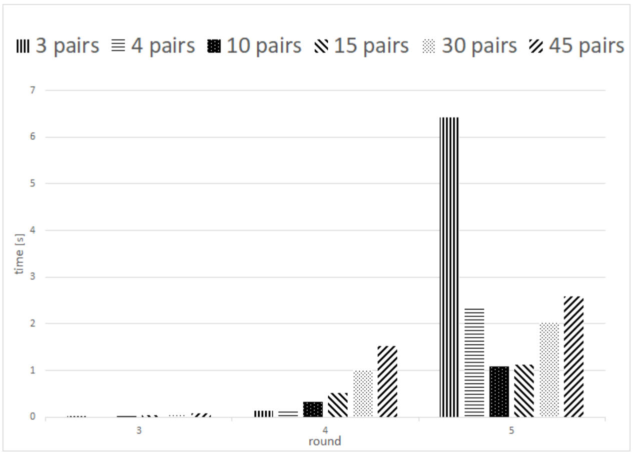

Fig. 2. Average time of attack using Lingeling and different number of pairs of SPECK 32/64 (without key expand algorithm and handmade equations)

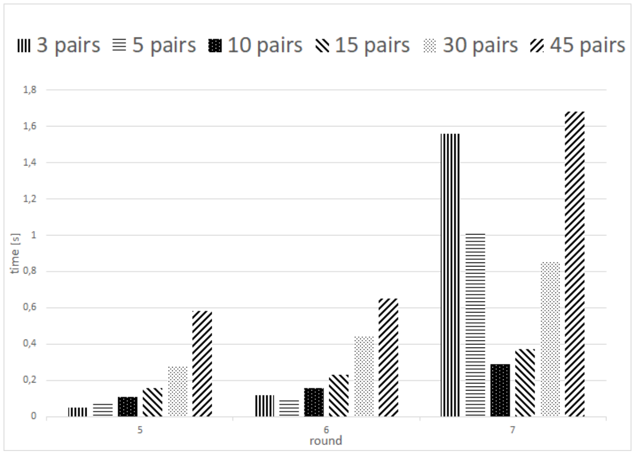

Fig. 3. Average time of attack using Lingeling and different number of pairs of SIMON 32/64 (without key expand algorithm and handmade equations)

attack works the best for only 10 pairs. For complex problems, the situation is the opposite.

Large systems of equations can be solved more efficiently with more pairs available. This dependency works for both attack models.

This is probably the case where some of the SAT-solvers are able to utilize additional information taken from a larger number of pairs, which results in finding the solution in a shorter timeframe. There are also solvers not able to utilize this additional information and their running time increases with the number of pairs

C. Equations type

The same cipher can be described by sets of a different algebraic equations. The main differences between two sets are: the number of equations, the maximal algebraic degree and the number of clauses. The equations can be obtained automatically by proper software tools. We have developed a method for extracting algebraic equations from software and hardware implementations. Thus, we are able to process different equations describing the same cipher. In the experiments we used equations from several sources including: handwritten, generated by hardware synthesis tools, generated by C compiler and generated by Cryptol. Handwritten equations seems to be the most natural and readable for human. The hardware equations are describing every logic cell in the implementation of the cipher in FPGAs, so the structure of the cipher is hidden. It is similar to equations from C and Cryptol, where the cipher is translated by compiler to computer readable format.

The idea of using an equation taken from hardware tools was earlier explored by Courtois et al. [1] to conduct an SAT attack on DES block cipher. In 2012, during SHA-3 competition, Homsirikamol et al. [15] developed a similar tool to obtain hardware equations describing SHA-3 final candidates and evaluating their security margin.

D. Numbers of clauses and variables

Fig.4 presents the increase of the number of clauses for SPECK cipher with every additional round. Fig.5 presents

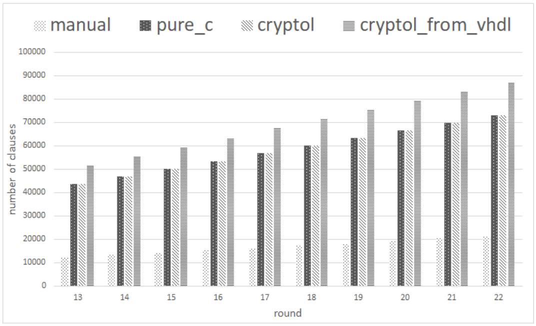

Fig. 4. Number of clauses in CNF equation, depending on the type of equations generator and number of SPECK rounds (without key expand algorithm)

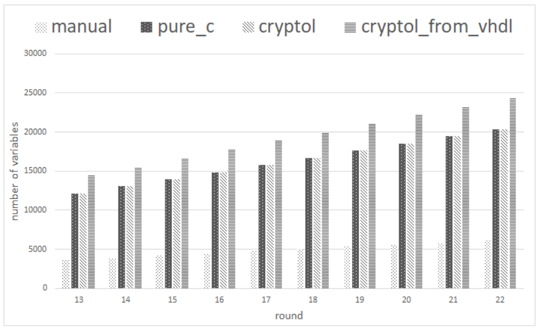

Fig. 5. Number of variables in CNF equation, depending on the type of equations generator and number of SPECK rounds for two pairs (without key expand algorithm)

the number of variables obtained from conversion tools to describe the round reduced SPECK cipher. Both numbers have a linear dependency on the number of rounds. The numbers describing the hardware and software equations increase more than handwritten ones. However, according to our experiments, the system of equations with larger number of variables and clauses can be solved faster than smaller ones. In some cases, an SAT solver can utilize the additional information hidden under the equations.

VI. RESULTS

We report the best results obtained among three SAT—solvers: Cadical, Lingeling and Treengeling using AMD FX(tm)-8300 CPU clocked with 3531 MHz. The results were obtained by attacking the smallest ciphers from the SIMON and SPECK families with limited computation time set to maximum one hour. The hardware equations are taken from synthesizing the cipher design into an FPGA board by a free-of-charge version of Intel Quartus 18.1. For every logic cell in the FPGA design, an appropriate equation is given. Our targeted FPGA family was Intel Cyclone V. Equations from C implementation are obtained by translating the and-inverted-graphs (AIG) [17] produced by clang version 3.6.0 C compiler. Equations from Cryptol [18] implementation are obtained in a similar way. The reported results are taken as average of 40 runs of each experiment. To make the comparison fair, the used key was random and the plaintext-ciphertext pairs were the same in every experiment.

# of rounds\# of pairs	3	4	>= 5
5	∼0,23	∼0.62	∼1,00
6	∼0,05	∼0.47	∼1,00
7	∼0,00	∼0.62	∼1,00
8	∼0,00	∼0.59	∼1,00
9	∼0,00	∼0.58	∼1,00
10	∼0,00	∼0.61	∼1,00
$\geq$ 11	∼0,00	?	?

TABLE II
EXPERIMENTAL PROBABILITY OF SUCCESS ATTACK FOR SIMON 32/64 (WITHOUT KEY EXPAND ALGORITHM)

All of our experiments took less than one hour. After one hour of computation, the tasks were terminated. With this approach, we failed to perform a successful attack on six round of SPECK and 8 rounds of SIMON with the first attack model, where key expansion algorithm is included into system of equations.

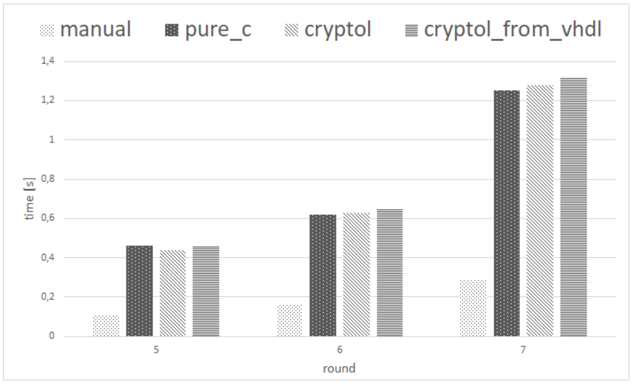

Fig. 6. Average time of attack using Lingeling for ten pairs of SIMON (without key expand algorithm)

In Tab. II and Tab. IV we report probability of success for SAT attack on SIMON and SPECK when equations from the key expand algorithm are not included. In Tab. III we report probability of success for SAT attack on SPECK when equations from key expand algorithm are included.

For the SIMON algorithm, with less than 4 pairs, the probability of finding the valid key is decreasing with every additional round of encryption and becomes negligible even for a small number of rounds. Extending the system of equations with additional pairs increases the probability of success. Moreover, only 5 pairs are required to find the valid key in the scenario, where equations for a key expansion are not included into the system of equations. This is an important observation that allows for the successful execution of a known plaintext attack with limited access to an encryption device.

Our attack for SIMON works the best for 10 pairs. The best obtained results were for handwritten equations. The equations taken from software and hardware compilers were more than three times higher.

The worst results were obtained for equations taken from HDL implementation. This might come from a high complexity of compilers and a specific form of logic element build, where only several pins of input can be mapped to up to two pins of output.

Similar results for SPECK are shown in Fig. 11. However, in Fig. 10 the results for an attack on SPECK with four pairs is presented. For a given setting, the best results were obtained for the equations taken from hardware implementation and the hand written equations provided the worst results. This is the opposite to most of the other experiments. Hardware equations were also the best for an attack on SIMON with 30

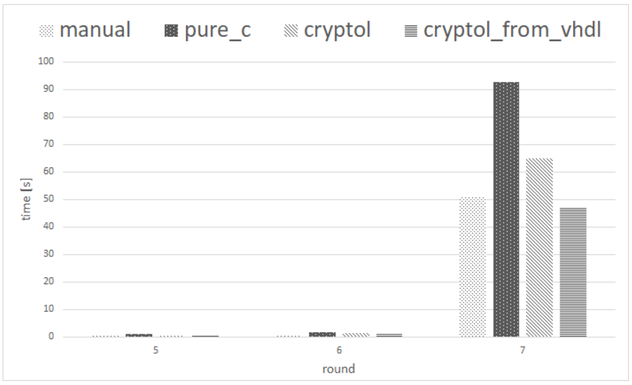

Fig. 7. Average time of attack using Cadical for thirty pairs of SIMON (without key expand algorithm)

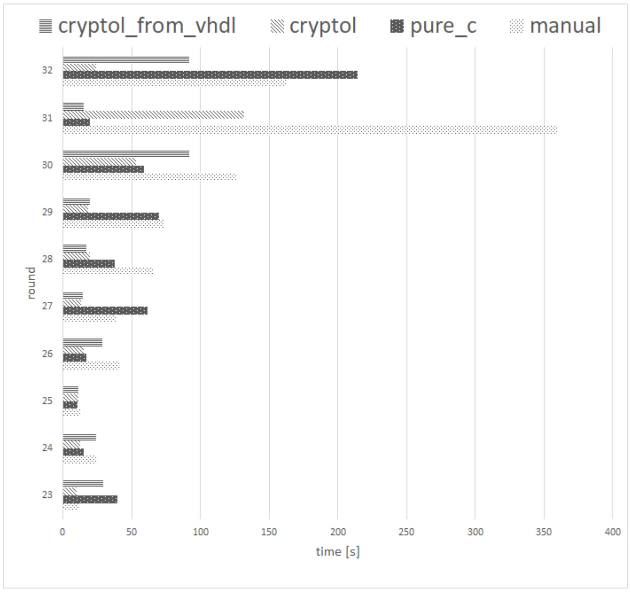

Fig. 8. Average time of attack using Cadical for three pairs of SIMON (without key expand algorithm)

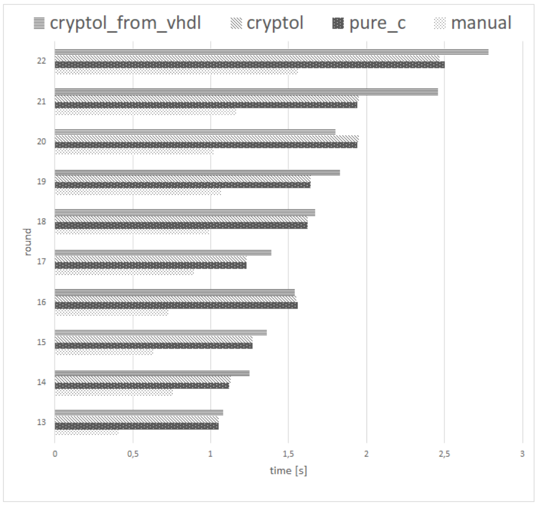

Fig. 9. Average time of attack on SPECK using Cadical for two pairs (without key expand algorithm)

pairs. The results of this attack are presented in Fig. 7. The common aspect of the two attacks is the Cadical SAT solver used for finding a solution. The proper solver and equations type selection might be crucial for obtaining the best results.

Our attack for SPECK also works the best for 10 pairs. We were able to attack up to 6 rounds with 10 pairs using Lingeling SAT to obtain a valid key in less than 600 seconds.

In Fig. 9 the time of attack for two pairs on full SPECK with Cadical is presented. The same as for SIMON, the Cadical SAT—solver was the best for two pairs and Lingeling is the best for more than 10 pairs.

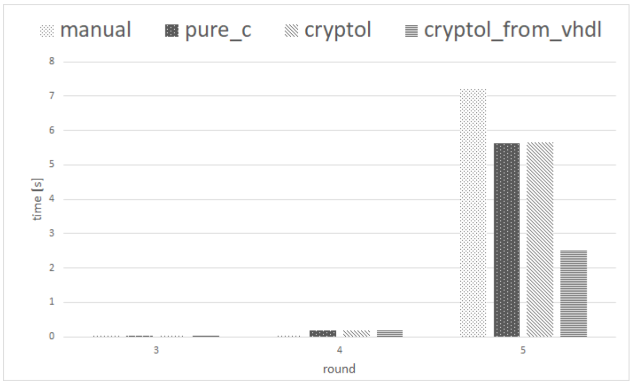

Fig. 10. Average time of attack on SPECK using Cadical for four pairs (without key expand algorithm)

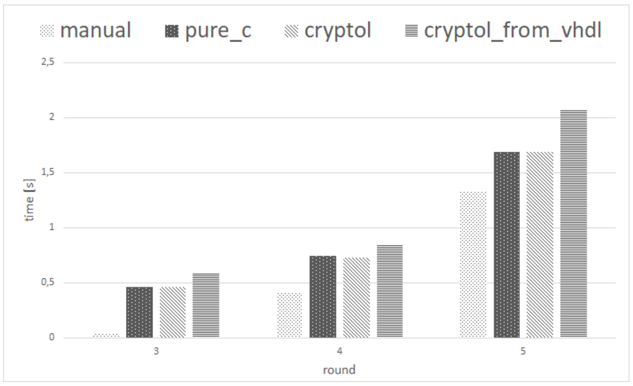

Fig. 11. Average time of attack on SPECK using Lingeling for ten pairs (without key expand algorithm)

# of rounds\# of pairs	1	2	>= 3
3	∼0,00	∼0.05	∼1,00
4	∼0,00	∼0.12	∼1,00
5	∼0,00	∼0.68	∼1,00
6	∼0,00	?	?
7	∼0,00	?	?
≥ 8	?	?	?

TABLE III
Experimental probability of success attack for SPECK 32/64 (with key expand algorithm)

# of rounds\# of pairs	2	3	>= 4
3	∼0,68	∼0.95	∼1,00
4	∼0,07	∼0.85	∼1,00
5	∼0,00	∼0.88	∼1,00
6	∼0,00	∼0.86	∼1,00
≥ 7	∼0,00	?	?

TABLE IV
EXPERIMENTAL PROBABILITY OF SUCCESS ATTACK FOR SPECK 32/64 (WITHOUT KEY EXPAND ALGORITHM)

VII. CONCLUSION

As for now, we have focused on research about the best parameters to set for an attack, rather than attacking the highest number of rounds. We have checked the influence of several parameters on the solving time of an satisfiability problem.

Considering all mentioned factors, we were able to successfully break up to 6 rounds of SPECK cipher and up to 10 rounds of SIMON cipher using our novel approach on a single-core CPU. In Tab. V we report our best results. For aforementioned ciphers, the best approach is to use the second attack scenario, where the key expansion algorithm is not included.

	number of rounds	number of pairs	equations type	SAT	time [s]
SPECK	6	10	cryptol	Lingeling	451,29
SPECK	6	10	vhdl	Lingeling	588,74
SIMON	10	10	cryptol	Lingeling	8,98
SIMON	10	10	manual	Lingeling	5,78

TABLE V
OUR THE BEST RESULTS OF SAT ATTACK ON SPECK 32/64 AND SIMON 32/64 (WITHOUT KEY EXPAND ALGORITHM)

Presented results are only a fraction of those obtained from experiments. We have shown that many factors affect the success rate and time required for SAT attack on SPECK and SIMON ciphers. Further exploration of mentioned factors like source of equations, used SAT—solver, number of pairs, number of rounds and attack model may lead to even better results affecting the claimed security of the aforementioned ciphers or even to a full break of these ciphers.

Our new approach to an SAT attack offers a significant speed-up when compared to the standard method. It also decreases the amount of work required for preparing the attack. Moreover, the attack can be applied to other ciphers and the preparation costs of the attack are very low. Thus, our tool seems to be a good choice for a plug-and-play attack on the initial security strength evaluation. Our method can be also combined with other types of attacks as reported in [7], [19]. Using pairs selected with linear or differential cryptanalysis for an SAT attack without key expansion algorithm is a promising idea for further research and obtaining even better results.

REFERENCES

[1] Nicolas T. Courtois and Gregory V. Bard. Algebraic cryptanalysis of the data encryption standard. In Steven D. Galbraith, editor, *Cryptography and Coding*, pages 152–169, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg, DOI: 10.1007/978-3-540-77272-9_10.

[2] J. Smith S. Treatman-Clark B. Weeks L. Wingers R. Beaulieu, D. Shors. The simon and speck families of lightweight block ciphers. 2016, DOI: 10.1145/2744769.2747946.

[3] Armin Biere. *CADICAL, LINGELING, PLINGELING, TREENGELING and YALSAT Entering the SAT Competition 2017*. 2017.

[4] Alex Biryukov, Arnab Roy, and Vesselin Velichkov. Differential analysis of block ciphers simon and speck. pages 546–570, 03 2014, DOI: 10.1007/978-3-662-46706-0_28.

[5] Farzaneh Abed, Eik List, Jakob Wenzel, and Stefan Lucks. Differential cryptanalysis of round-reduced simon and speck. 03 2014, DOI: 10.1007/978-3-662-46706-0_27.

[6] Qingju Wang, Zhiqiang Liu, Kerem Varici, Yu Sasaki, Vincent Rijmen, and Yosuke Todo. Cryptanalysis of reduced-round simon32 and simon48. pages 143–160, 12 2014, DOI: 10.1007/978-3-319-13039-2_9.

[7] Nicolas Courtois, Theodosis Mourouzis, Guangyan Song, Pouyan Sepehrdad, and Petr Susil. Combined Algebraic and Truncated Differential Cryptanalysis on Reduced-round Simon:. In *Proceedings of the 11th International Conference on Security and Cryptography*, pages 399–404. SCITEPRESS - Science and and Technology Publications, DOI: 10.5220/0005064903990404.

[8] J. Ren and S. Chen. Cryptanalysis of Reduced-Round SPECK. 7:63045–63056, DOI: 10.1109/ACCESS.2019.2917015.

[9] Martin Albrecht. Algorithmic algebraic techniques and their application to block cipher cryptanalysis. 4:120–141, 04 2011, DOI: 10.1007/978-3-642-22497-3_9.

[10] Carlos Cid and Ralf-Philipp Weinmann. Block ciphers: Algebraic cryptanalysis and grobner bases. In Massimiliano Sala, Shojiro Sakata, Teo Mora, Carlo Traverso, and Ludovic Perret, editors, *Grobner Bases, Coding, and Cryptography*, pages 307–327. Springer Berlin Heidelberg, DOI: 10.1007/978-3-540-93806-4_17.

[11] Gregory V. Bard. Algebraic cryptanalysis. DOI: 10.1007/978-0-387-88757-9.

[12] Ilya Mironov and Lintao Zhang. Applications of sat solvers to cryptanalysis of hash functions. volume 2006, pages 102–115, 01 2006, DOI: 10.1007/11814948_13.

[13] Joan Boyar, Magnus Find, and Rene Peralta. Four Measures of Nonlinearity. In Paul G. Spirakis and Maria Serna, editors, *Algorithms and Complexity*, pages 61–72. Springer Berlin Heidelberg, DOI: 10.1007/978-3-642-38233-8_6.

[14] Gregory Bard. Algorithms for solving linear and polynomial systems of equations over finite fields with applications to cryptanalysis.

[15] Ekawat Homsirikamol, Pawel Morawiecki, Marcin Rogawski, and Marian Srebrny. Security Margin Evaluation of SHA-3 Contest Finalists through SAT-Based Attacks. In Agostino Cortesi, Nabendu Chaki, Khalid Saeed, and Slawomir Wierzchon, editors, *Computer Information Systems and Industrial Management*, pages 56–67. Springer Berlin Heidelberg, DOI: 10.1007/978-3-642-33260-9_4.

[16] Nicolas T. Courtois, Pouyan Sepehrdad, Petr Susil, and Serge Vaudenay. ElimLin Algorithm Revisited. In Anne Canteaut, editor, *Fast Software Encryption*, volume 7549, pages 306–325. Springer Berlin Heidelberg, DOI: 10.1007/978-3-642-34047-5_18.

[17] Biere Armin. *The AIGER And-Inverter Graph (AIG) Format*. 2007.

[18] Inc Galois. *Cryptol The Language of Cryptography*. 2018.

[19] Ludovic Perret Jean-Charles Faugere and Pierre-Jean Spaenlehauer. Algebraic-differential cryptanalysis of DES. 07 2010,

Michal Andrzejczak In 2016 he obtained a master's degree in computer science with a specialization in cryptology from the Military University of Technology. He is currently a PhD student on MUT and his research interests include FPGAs and its modern applications to cryptography.

Wladyslaw Dudzic In 2015 he obtained a master's degree in computer science with a specialization in cryptology from the Military University of Technology. He is currently a PhD student on MUT and his research interests include cryptography and its modern applications, design of cryptographic algorithms (in particular block ciphers), linear and diffenerial cryptanalysis, SAT solvers and their applications in algebraic cryptanalysis.