A Note on SIMON-32/64 Security

John Matthew Macnaghten, James Luke Menzies and Mark Munro

Alba3 Group
Edinburgh, Scotland,
jmm-jlm-mm@protonmail.com

Abstract. This paper presents the results of a new approach to the cryptanalysis of SIMON-32/64, a cipher published by NSA in 2013 [4]. Our cryptanalysis essentially considers combinatorial properties. These properties allow us to recover a secret key from two plaintext/ciphertext pairs, in a time ranging from a few hours to a few days, with rather limited computing resources.
The efficiency of our cryptanalysis technique compared to all known cryptanalyses (including key exhaustive search) is a justification for not revealing the cryptanalysis techniques used. We have adopted a zero-knowledge-inspired method of proof which was initiated in [11].

1 Introduction

SIMON et SPECK are two lightweight block ciphers proposed by NSA in 2013 [4]. These ciphers have been developped to achieve the best possible performances in both software and hardware implementation. In 2014, these algorithms were proposed for inclusion in ISO/IEC 29192. As from their publication, these algorithms have elicited mistrust from the international cryptology community. The NSA is strongly suspected of having introduced backdoors. This suspicion motivated the IOC to reject Simon and Speck algorithms as part of the standardization process. Similarly, when the Linux community discovered that the SIMON algorithm had been introduced in the Linux 4.16 kernel (in the *fscrypt* functionality, a transparent encryption library of the system files), it was immediately removed from the next version (4.17).

In this paper, we will focus on the SIMON-32/64 version that allowed us to validate our combinatorial analysis technique. Although its key size is reduced, this algorithm seems to be of interest to the IoT and embedded systems community [25, 13] for real use.

Unlike statistical or algebraic cryptanalysis techniques, we consider particular combinatorial structures Σ whose core elements (blocks) are made up of particular bits from the secret key, the plaintext and the ciphertext. These structures are defined by very strong combinatorial

properties linking these core elements. In some cases, a block encryption system S may represent a more or less noisy version of one or more structures $\Sigma_1, \Sigma_2, \ldots$. By considering an appropriate combinatorial similarity measure d_c then if the values $d_c(S, \Sigma_i)$ are large enough, it is possible to find key bits knowing certain bits of plaintext and ciphertext by exploiting the combinatorial properties that link them within the elements of the Σ_i structures.

Our analysis technique is not public at this stage of development. There is still a lot of work to be done to obtain an optimized, more efficient and industry-level version, which would be transposable more generally to other cryptographic systems. However, since we know that an algorithm is no longer secure and can no longer be used, it is essential to make this information public, without necessarily revealing the method used. We have therefore chosen to be inspired by a proof technique derived from zero-knowledge protocols and initiated in [11]. Unfortunately, this idea is proposed in a relatively unfinished and complicated way. We have taken up the principle and adapted it to our needs. The principle is pretty simple: we choose randomly several pairs of plaintext/ciphertext (P_i, C_i) taken from a current reference text (so in fact P_i and C_i are blocks of plaintext) and we look for the keys K_i such that $C_i = E_{K_i}(P_i)$. The nature of the text and the choice of these non-random blocks of data ensures that the result can only be obtained by a particular and more efficient method than the exhaustive key search would.

Although the 64-bit size is below usual cryptographic security standards [10], an effective 64-bit exhaustive search in a limited time is totally out of reach for us. It is even more so if you want to repeat the attack several times. Such cryptanalysis on 64-bit keys might only be possible by a very few intelligence agencies in the world such as the NSA (USA), GCHQ (UK), FSO/Spetssvyaz (Federation of Russia) or GSD/3PLA (China).

Our main result is that we can find a 64-bit key in about three days (average time) on two Odroid MC1 clusters (8 Gb) [18] from two pairs of plaintext/ciphertext. To date the probability of success is still very low ($p = 0.025$) but we are optimistic about the possibility of significantly increasing it in the upcoming months. Indeed we have a second algorithm, which is theoretically proven, with a success rate of 0.25. It is not yet fully tested and executed because it requires a higher computing power, although it is still reasonable for operational cryptanalysis (which can be repeated over time).

This article is organized as follows. Section 2 briefly presents the structure of the algorithms of the SIMON family as well as the best attacks

known about the different members of this family. In Section 3, we present a new vision of encryption systems which is useful for our zero-knowledge cryptanalysis proof. This vision is to be discussed in regard to some interesting combinatorial properties. It also shows that some parameter configurations are strongly to be avoided. In Section 4, we give the main cryptanalysis results we obtained for SIMON-32/64. Finally, in Section 5 we conclude and present the future developments of our work.

2 SIMON Family and Known Attacks

# Rounds	Key size	Data blocksize
32	64	32
36, 36	72, 96	48
42, 44	96, 128	64
52, 54	96, 144	96
68, 69, 72	128, 192, 256	128

Table 1. SIMON family parameters

SIMON uses a very simple round function which is iterated over many rounds (the number of rounds is actually significantly larger than for usual block ciphers). The system exhibits a very compact structure that supposedly simplifies the analysis. Basically SIMON is a two-branch balanced Feistel network. A plaintext to be encrypted P is processed by the round function for a certain number of rounds, and finally the ciphertext C is output. The round function uses different keys referred to as round keys, derived from the original key (key schedule). All round functions are designed using only modular Addition, Rotation and XOR. Round function definition and key expansions are given in [4]. Data blocksize usually equals to $32, 48, 64, 96$ or 128 bits. Table 1 summarizes the different block and key sizes, in bits, and corresponding number of rounds. SIMON supports a variety of block and key sizes: 32-, 48-, 64-, 128-bit blocs with 64-, 96-, 128-, 256-bit key. The number of rounds depends on the selected parameters. Our study presently focuses on SIMON-32/64. Table 2 sums up the known attacks on SIMON-32/64 version.

No attack takes into account the full version of SIMON-32/64 so far (27 rounds at most). The time and data complexity is still to high to allow effective and repeated cryptanalyses over time.

Attack type	# attacked rounds	Computation	Data	Success probability	Ref.
Related-Key Linear Crypt.	27	$2^{61.34}$	2^{60}	0.5	[15]
Differential Crypt.	18	2^{46}	$2^{31.2}$	0.632	[1]
Related-key Rectangle	18	$2^{54.55}$	$2^{30.86}$	≈ 1.0	[1]
Impossible differential	13	$2^{50.10}$	2^{30}	≈ 1.0	[1]
Differential Crypt.	16	$2^{26.48}$	$2^{29.4}$	-	[2]
Impossible differential	14	$2^{44.18}$	$2^{33.29}$	-	[2]
Differential	21	$2^{55.25}$	2^{31}	0.51	[26]
Differential	22	$2^{58.76}$	2^{32}	0.31	[19]
Linear	23	$2^{61.84}$	$2^{31.19}$	0.27	[5]
Integral	21	2^{63}	2^{31}	1.0	[27]
Integral	22	2^{63}	2^{31}	1.0	[12]
Integral	24	2^{63}	2^{32}	1.0	[6]
Impossible differential	20	$2^{62.8}$	2^{32}	-	[8]
Zero correlation	21	$2^{59.4}$	2^{32}	-	[24]
Meet-in-the-middle	18	$2^{62.57}$	8	1.0	[23]
Correlated sequence	27	$2^{62.94}$	3	1.0	[22]
Algebraic attack	11	30" to 3'	50	≈ 0.2	[21]

Table 2. SIMON-32/64 known attacks

In the case of SIMON-32/54 the size of the key is relatively small (64 bits). What is the feasibility of an attack by exhaustive testing on the key? The only reference for this magnitude of cryptanalysis is [9]. In 2002, a 64-bit RC4 key was obtained in 1,754 days (300,000 participants). Today, to keep up with the evolution of computing power since 2002, it is necessary to divide by 2^{10}, or about 3 days with the same number of participants.

3 A New Model for Block Cipher Systems

Throughout this section we use the following notations. We denote the iterated logarithm (base e) by $\ln^{(.)}$, *i.e.* $\ln^{(1)} = \ln(x)$ and $\ln^{(k+1)} = \ln(\ln^{(k)}(x)$ for all $k \geq 1$; $f << g$ means $f = o(g)$ and $f >> g$ means $f = \omega(g)$. We use the term $polyln(x)$ to denote the class of functions $\bigcup_{k\geq 1}((\ln(x))^k)$. We say that an event $\mathcal{E}$ occurs with high probability if $P[\mathcal{E}] = 1 - o(1)$.

Let us model a block cipher system E as a function of $\mathcal{P}\times\mathcal{K} \to \mathcal{C}$ where $\mathcal{P}$ (respectively $\mathcal{K}$ and $\mathcal{C}$) describes the plaintext space (respectively key and ciphertext spaces). Let us set, in the general case, $|\mathcal{P}| = |\mathcal{C}| = 2^{|C|}$ and

$|\mathcal{K}| = 2^{|K|}$ where $|C|, |P|$ and $|K|$ denotes the ciphertext, plaintext and the key size respectively. For most block cipher systems, $|C| = |P| = |K|$.

The binary representation of the elements of these spaces allows us to move on to the following notion for a system of block encryption S:

$$\begin{aligned} \mathbb{F}_2^{|P|} \times \mathbb{F}_2^{|K|} &\to \mathbb{F}_2^{|C|} \\ (P, K) &\mapsto C \end{aligned} \tag{1}$$

When selecting a key K, we then consider the restriction to the next bijection:

$$\begin{aligned} \mathbb{F}_2^{|P|} &\to \mathbb{F}_2^{|C|} \\ P &\mapsto C = E_K(P) \end{aligned} \tag{2}$$

In the case of our protocol of cryptanalysis with zero-knowledge proof, we use an opposite view. We fix a pair of plaintext block and ciphertext block (P, C). We then consider the next application:

$$\begin{aligned} \mathbb{F}_2^{|K|} &\to \mathbb{F}_2^{|C|} \\ K &\mapsto C = E_K(P) \end{aligned} \tag{3}$$

This application is no longer a bijection because it is neither surjective nor injective. Let us look at its properties. We get the following proposal.

Proposition 1. *Let us consider a fixed plaintext block P. The probability that there is no C such as $C = E_K(P)$ regardless of the key K is $p = (1 - \frac{1}{|\mathcal{C}|})^{|\mathcal{K}|}$.*

Proof. We have to interpret Function 3 when we go through all the keys $K \in \mathcal{K}$ as a "bins and balls" problem [20]. The value $E_K(P)$ represents a ball and we look at which bin C it is assigned to, under the assumption that Function 3 behaves strictly randomly (minimum expected minimum property for an encryption system).

Let $X_i = X_i(m, n)$ be the random variable which counts the number of balls in the i-th bin whenever we throw m balls independently and uniformly at random into n bins. Clearly, Y_i is a binomially distributed random variable. Equivalently we have $X_i \sim \mathcal{B}(m, \frac{1}{n})$. We then have $P[X_i = k] = \binom{m}{k}(\frac{1}{n})^k(1 - \frac{1}{n})^{m-k}$. Hence the result by taking $k = 0$, $m = |\mathcal{K}|$ and $n = |\mathcal{C}|$. QED

From Proposition 1, we can now discuss the result depending on the respective value of m and n. In the rest of the paper we will set $m = 2^{|K|}$ and $n = 2^{|C|}$.

Corollary 1. *Whenever the key and ciphertext block sizes are equal* ($m = n$)*, only* 0.6321 *of the ciphertext blocks can be obtained from a fixed plaintext block when going through all the keys. Whenever the size of the keys is larger than that of the ciphertext block* ($m >> n$) *then all encrypted blocks can be obtained with a probability close to 1, all the more quickly as* m *is high compared to* n*.*

Proof Straightforward from Proposition 1 QED

Now that we have determined the probability with which a block of ciphertext can be obtained from a fixed plaintext block by running through the space of keys, it is necessary to evaluate the number of collisions of Function 3. Indeed, for our proof technique to be admissible, each ciphertext block C must be obtained only from a limited number of pairs $(P, K_{i_1}), (P, K_{i_2}), (P, K_{i_3}), \ldots (P, K_{i_r})$ for a fixed block of plaintext P.

It is possible to use the formula used in the proof of Proposition 1 which allows to calculate $P[X_i = k]$ for different values of k. However, only the maximum number of collisions does really matter. The larger the r value is, the less difficult it is to find a K key for a given (P, C) pair. We use the general Theorem 1 due to [20] in the context of the balls-into-bins model.

Theorem 1. *Let* M *be the random variable that counts the maximum number of balls in any bin, if we throw* m *balls independently and uniformly at random into* n *bins. Then* $P[M > k_\alpha] = o(1)$ *if* $\alpha > 1$ *and* $P[M > k_\alpha] = 1 - o(1)$ *if* $0 < \alpha < 1$ *where,*

$$k_\alpha = \begin{cases} \frac{\ln(n)}{\ln(\frac{n.\ln(n)}{m})}\left(1 + \alpha.\frac{\ln^{(2)}(\frac{n.\ln(n)}{m})}{\ln(\frac{n.\ln(n)}{m})}\right) & \text{if } \frac{n}{polyln(n)} \leq m << n.\ln(n), \\ (d_c - 1 + \alpha).\ln(n), & \text{if } m = c.n.\ln(n) \\ & \text{for some constant } c, \\ \frac{m}{n} + \alpha\sqrt{2.\frac{m}{n}.\ln(n)}, & \text{if } n.\ln(n) << m \leq n.polyln(n), \\ \frac{m}{n} + \sqrt{\frac{2m.\ln(n)}{n}.\left(1 - \frac{1}{n}.\frac{\ln^{(2)}(n)}{2\ln(n)}\right)}, & \text{if } m >> n.(\ln(n))^3. \end{cases}$$

Here d_c *denotes a suitable constant depending only on* c*.*

Proof. See [20] for the proof. QED

The application of Theorem 1 to the study of collisions of the function 3 makes it possible to establish the following proposition.

Proposition 2. *When the key size* $|K|$ *is equal to the ciphertext block size* $|C|$*, then the maximum number of collisions is* $r = \frac{\ln(n)}{\ln^{(2)}(n)}.(1 + o(1))$ *with a high probability.*

$$\textit{When } |K| = 2.|C| \textit{ then } r = \frac{m}{n} + \sqrt{\frac{2m.\ln(n)}{n}.\left(1 - \frac{1}{\alpha}.\frac{\ln^{(2)}(n)}{2\ln(n)}\right)}.$$

Proof. By applying Theorem 1. For the case $|K| = |C|$, the reader will refer to [20, Section 4]. The proof is relatively technical and cannot be deduced directly from Theorem 1.

For the case $|K| = 2.|C|$, we have $m >> n.(\ln(n))^3$. The last case of Theorem 1 applies directly. QED

Before applying these results to Simon-32/64, we compared the theory with reality by considering two smaller algorithms: the miniAES [7] and mSimon32. The latter algorithm has been designed on purpose for this study. It is a reduced version of Simon-32/64 accepting a 32-bit key. In both cases (see Table 3), these algorithms present extreme results that are not compatible with the theory. This suggests that these two algorithms have strong combinatorial irregularities.

Alg.	$r^t_{\max}$ (theory)	$r^o_{\max}$ (observed)	# of cases where $r^t_{\max} < r^o_{\max}$
Mini-AES	5	9	39
mSimon32	8	14	4,824

Table 3. Evaluation of Mini-AES and mSimon32 with respect to Proposition 2. Results are average values computed on few hundreds of plaintexts

Let us now apply Proposition 2 SIMON-32/64. So we have two cases:

- Either we consider a single 32-bit ciphertext block for a twice as large key. In this case, $r = 2^{32} + 2^{18}$. The maximum number of collisions is very high.
- Or we consider two blocks of data simultaneously (concatenated), and $|K| = 2.|C|$. So the maximum number of possible collisions is $11 < r < 12$. Note that by considering the Binomial distribution formula, we get Table 4 for 99.9% of the keys, with for each value the *Known-plaintext Unicity Distance* [17, p. 235, 7.35].

In the subsequent section, in the case of SIMON-32/64, we consider a 64-bit data block composed of two **different** 32-bit blocks (see Section 4).

4 Application to SIMON-32/64

Now that the theoretical framework has been defined, we present the results of a four-month full-scale experiment. However, it is important to

r	$P[C_i = r]$	KPUD
0	0.36787944117144232160	-
1	0.36787944117144232159	2
2	0.18393972058572116079	3
3	0.06131324019524038691	3
4	0.01532831004881009672	3
5	0.00306566200976201934	3

Table 4. Collision probability for a fixed C and Known-plaintext Unicity Distance (KPUD)

mention that the chosen proof protocol requires testing more fixed (P, C) pairs than in the case of plaintext/ciphertext pairs for which we are sure they are actually the product of encryption/decryption by at least one K key. Indeed, according to Table 4, about $P[r = 0] = 0.36788$ of the ciphertext blocks cannot be obtained from a fixed plaintext block P when you go through all the keys. Algorithm 1 describes our general procedure. The reference text used is the King James' Bible [14].

Input : A public reference text $\mathcal{T}$, N the number of pairs (P_i, C_i) of 64-bit datablocks to decrypt
Output: At least one key K_i found such that $C_i = E_{K_i}(P_i)$ whenever K_i does exist

for $i \leftarrow 1$ **to** N **do**
  Draw at random a 64-bit block P_i and a 64-bit block C_i in $\mathcal{T}$
  if P_i *has the form* (P_i', P_i') **then**
    Continue
  end
  Find at least one key K_i such that $C_i = E_{K_i}(P_i)$
  Print P_i, K_i, C_i
end

Algorithm 1: Decryption Algorithm of randomly chosen meaningful pairs (P_i, C_i)

Testing this analysis procedure on plaintext blocks that we have encrypted using known keys (hence we are sure that the key does exist) allowed us to determine the probability of success $p_s = 0.025$. From Proposition 1, the probability of success of Algorithm 1 is therefore $p_s^{\text{alg}_1} = p_s \times (1 - P[r = 0]) = 0.016$. With our second theoretical algorithm (still not fully tested), the success rate would amount to 0.1580.

The condition in Line 3 is important because in the case where P is in the form (P', P') it is very easy to find several K_i keys according to Proposition 2.

As an example, Table 5 shows some examples found in a few hours of calculation.

Clair (text - Hexa)	Crypto (text - Hexa)	K	# keys found
YHWHYHWH	YHWHYHWH	0x0019F32123313010	
0x5948574859485748	0x5948574859485748	0x1257930310310132	8,846
		0x7386E21032220307	
		0x801D720E84B8077E	
		0xE009E842A2A2855F	
		0xF63C412201020001	
” ”	********	0x800826680374D290	
0x2020202020202020	0x2A2A2A2A2A2A2A2A	0xAF0035E6FDD8C589	11,060
		0xE0271B23198503CB	
		0xC016A3911A903C4F	
		0xC1D4D9597868496B	

Table 5. Examples of K for different $P = (P', P')$ and $C = (C', C')$ (computation time 24 hours) - YHWH is the Tetragrammaton, in the Latin script, is the four-letter biblical name of the God of Israel

Algorithm 1 was run on two Odroid MC1 clusters [18] for nearly 120 days with the following parameters: reference text [14] and $N = 1,000$. We found 19 keys confirming the probability $p_s^{\text{alg}_1}$. The result is presented in Table 6. It should be noted that if we consider Table 4, we should have

P (hexa)	C (hexa)	K
”ving tee” (0x76696E6720746565)	”y will I” (0x792077696C6C2049)	0xF76C496859696B5A
” itfor t” (0x206974666F722074)	”way, for” (0x0005801333213003)	0x0005801333213003
”oaring o” (0x6F6172696E67206F)	”a, the g” (0x612C207468652067)	0x59BC08C69E3927AE
”et of hi” (0x6574206F66206869)	”ur the d” (0x7572207468652064)	0x93D444F3D0951DB5
”mthee. {” (0x6D746865652E207B)	”t] infam” (0x745D20696E66616D)	0x000C801220001031
”ons: {1:” (0x6F6E733A207B313A)	”t] he wo” (0x745D20686520776F)	0x8C908C41B1F6CA37
”ey forth” (0x657920666F727468)	”shak. {1” (0x7368616B2E207B31)	0x2D3448495B5B596B
”sar, Che” (0x7361722C20436865)	”seed be ” (0x7365656420626520)	0xE90C0C73B2E5E907
”een Esth” (0x65656E2045737468)	”rgetteth” (0x7265674746574680)	0x34AC4979785A7B58
”ave [him” (0x617665205B68696D)	”ns in ju” (0x6E7320696E206A75)	0xB28488C79E18249E
”e flesh.” (0x6520666C6573682E)	”nor dead” (0x6E6F722064656164)	0x4F108C4183C6FB34
”re enemi” (0x726520656E656D69)	”k uponhi” (0x6B2075706F6E6869)	0x2D30C84A4849584A
”h meinto” (0x68206D65696E746F)	”h; thats” (0x683B207468617473)	0x0009801311321010
”de you s” (0x646520796F752073)	”fessed t” (0x6665737365642074)	0x4F140C7081F5DB27
”er. {2:3” (0x65722E207B323A33)	”can say ” (0x63616E2073617920)	0xE65CC84B686B5949
”fferedbu” (0x6666657265646275)	”ax hotag” (0x617820686F746167)	0xA04488C59E1B15AE
”ame: {25” (0x616D653A207B3235)	”LORDhard” (0x4C4F524468617264)	0x45900C40A2F5C926
”re; [but” (0x72653B205B627574)	” for] go” (0x20666F725D20676F)	0x0001011212222301

Table 6. Cryptanalysis results for the reference text [14] and $N = 1,000$ ($\approx$ 120 days)

found several keys for some pairs. This is not the case. Our hypothesis is that the result of Proposition 2 assumes that the blocks of plaintext and ciphertext are uniformly distributed variables. However, in our case, the

choice for P and C does not correspond to this assumption, which may explain the fact that we are only in cases where $r = 1$.

5 Conclusion and Future Work

The other versions of the SIMON family also seem to have the same weaknesses that can be exploited by our combinatorial analysis technique. However, it is important to remain cautious and to validate this through experiments. This is the subject of our current work. This should also improve the probability of success that we still estimate much too weak too weak (practical success rate of 0.025 and theoretical success rate [not fully tested yet] of 0.25). Many parameters, configurations and structures are possible and we are far from having explored them all. The goal is that what is still only a Proof-of-Concept algorithm evolves into a product mature enough for industrial use (cryptanalysis, evaluation and certification).

However, our results allow us to state that the SIMON-32/64 algorithm should no longer be considered secure. The attack we are able to carry out makes it possible – albeit with a still low probability of success – to cryptanalyze it in a finite time compatible with effective cryptanalysis requirements.

We will also apply our analysis technique to other recent algorithms of larger key sizes. At this point, we are able to easily process and analyze any algorithm with key size up to 64 bits. If we note $\mathcal{C}(n)$ the complexity of our attack depending on the key size n, hithertho it seems that $2^n - \mathcal{C}(n)$ grows with n. Equally, this would indicate that for larger key sizes, the gain over exhaustive research is greater.

We are also working on several open issues related to the formalization of Section 3. In particular, is it possible to have a distinguisher to identify pairs P, C for which $r = 0$ (Proposition 2 and Table 4)? It would also be interesting if this distinguisher could be discriminating enough with respect to the different values of r.

From a more general point of view, we are convinced that it is important for cryptanalysts to have the conditions to publish proof of cryptanalysis without revealing the techniques used. The motivations can be of several kinds: protection of industrial know-how, legal limitations or responsible disclosure. It would be interesting and useful if the authors of cryptographic algorithms made systematically pairs of plaintext/ciphertext blocks for different keys, under their responsibility and control, available to the cryptology community, as they usually do with vector sets to val-

idate the implementation of their algorithms. A set of of pairs (P_i, C_i) produced by a few millions of different keys K_i would be very useful.

Acknowledgement

We would like to thank Oleg Ivanovich Popov for his insightful comments and advice as well as his constant support during the discussions we had with him.

References

1. F. Abed, E. List, S. Lucks and J. Wenzel. *Differential cryptanalysis of round-reduced SIMON and SPECK*. In: Cid C., Rechberger C. (eds.), FSE 2014. LNCS, vol. 8540, p. 525545. Springer, Heidelberg, 2014.
2. H. A. Alkhzaimi and M. M. Lauridsen. Cryptanalysis of the SIMON Family of Block Ciphers. Cryptology ePrint Archive, Report 2013/543, 2013.
3. L. Armasu. *NSA-Designed SPECK Algorithm to Be Removed From Linux 4.20*. Tom's Hardware, 4 Sep. 2018. `https://www.tomshardware.com/news/nsa-speck-removed-linux-4-20,37747.html`
4. R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks and L. Wingers. *The SIMON and SPECK Families of Lightweight Block Ciphers*. Cryptology ePrint Archive, Report 2013/404, 2013.
5. H. Chen and X. Wang. *Improved linear hull attack on round-reduced SIMON with dynamic key-guessing techniques*. In: Peyrin T. (eds), FSE 2016. LNCS, vol. 9783, p. 428449. Springer, Heidelberg, 2016)
6. Z. Chu, H. Chen, X. Wang, X. Dong and L. Li. *Improved integral attacks on SIMON32 and SIMON48 with dynamic key-guessing techniques*. Security and Communication Networks, 2018. `https://doi.org/10.1155/2018/5160237`
7. R. Chung-Wei Phan. *Mini Advanced Encryption Standard (Mini-AES): A Testbed for Cryptanalysis Students*, Cryptologia, 26(4), p. 283-306, 2002.
8. P. Derbez and P.-A. Fouque. *Automatic search of meet-in-the-middle and impossible differential attacks*. In: Robshaw M., Katz J. (eds), Crypto 2016. LNCS, vol. 9815, p. 157184. Springer, Heidelberg, 2016.
9. Distributed.net. *RC5-64 HAS BEEN SOLVED!*. 25 Sep. 2002. `http://www1.distributed.net/images/9/92/20020925_-_PR_-_64_bit_solved.pdf`
10. Ecrypt-CSA. *Algorithms, Key Size and Protocols Report*. Project 645421 - H2020-ICT-2014, 28 Feb. 2018. `http://www.ecrypt.eu.org/csa/documents/D5.4-FinalAlgKeySizeProt.pdf`
11. E. Filiol. *Zero-knowledge-like Proof of Cryptanalysis of Bluetooth Encryption*. Cryptology ePrint Archive, Report 2006/303, 2006.
12. K. Fu, L. Sun and M. Wang. *New integral attacks on SIMON*. IET Information Security 11(5), p. 277286, 2016.
13. J. Hosseinzadeh and M. Hosseinzadeh. *A Comprehensive Survey on Evaluation of Lightweight Symmetric Ciphers: Hardware and Software Implementation*. Advances in Computer Science: an International Journal, [S.l.], p. 31-41, Jul. 2016. ISSN 2322-5157. Available at: `http://www.acsij.org/acsij/article/view/529`.
14. King James' Bible. `https://www.gutenberg.org/cache/epub/10/pg10.txt`

15. J.-K. Lee, B. Koo and W.-H. Kim. Related-Key Linear Cryptanalysis on SIMON. Cryptology ePrint Archive, Report 2018/152, 2018.
16. K. McCarthy. *ISO blocks NSA's latest IoT encryption systems amid murky tales of backdoors and bullying*. The Register, 25 Apr. 2018. `https://www.theregister.co.uk/2018/04/25/nsa_iot_encryption/`
17. A. J. Menezes, P. C. van Oorschot and S. A. Vanstone. Handbook of Applied Cryptography. CRC Press Series on Discrete Mathematics and its Applications, Boca raton, 1997.
18. Odroid MC1 Cluster. `https://www.hardkernel.com/shop/odroid-mc1-my-cluster-one-with-32-cpu-cores-and-8gb-dram/`
19. K. Qiao, L. Hu and S. Sun. *Differential analysis on SIMECK and SIMON with dynamic key-guessing techniques*. In: Camp O., Furnell S., Mori P. (eds), ICISSP 2016. LNCS, vol. 691, p. 6485. Springer, Cham, 2016.
20. M. Raab and A. Steger. *Balls into Bins A Simple and Tight Analysis*. In: Luby M., Rolim J. and Serna M. (eds). Randomization and Approximation Techniques in Computer Science, p. 159-170. Springer, Berlin, Heidelberg, 1998.
21. H. Raddum. *Algebraic Analysis of the SIMON Block Cipher Family*. In: Lauter K. and Rodríguez-Henríquez, F. (eds), Progress in Cryptology - LATINCRYPT 2015, p. 157-169. Springer, Cham, 2015.
22. R. Rohit and G. Gong. Correlated Sequence Attack on Reduced-Round SIMON-32/64 and SIMECK-32/64. Cryptology ePrint Archive, Report 2018/699, 2018.
23. L. Song, L. Hu, B. Ma and D. Shi. *Match box meet-in-the-middle attacks on the SIMON family of block ciphers*. In: Eisenbarth T., Öztörk E. (eds), LightSec 2014. LNCS, vol. 8898, p. 140151. Springer, Cham, 2014.
24. L. Sun, K. Fu and M. Wang. *Improved zero-correlation cryptanalysis on SIMON*. In: Lin D., Wang X., Yung M. (eds), Inscrypt 2015. LNCS, vol. 9589, p. 125143. Springer, Cham, 2014.
25. S. Taneja and M. Alioto. *Ultra-Low Power Crypto-Engine Based on Simon 32/64 for Energy- and Area-Constrained Integrated Systems*. ArXiv Preprint 1811.08507, 26 Nov. 2018. `http://arxiv.org/abs/1811.08507`
26. N. Wang, X. Wang, K. Jia and J. Zhao. *Differential attacks on reduced SIMON versions with dynamic key-guessing techniques*. Cryptology ePrint Archive, Report 2014/448, 2014. `https://eprint.iacr.org/2014/448.pdf`
27. Q. Wang, Z. Liu, K. Varc, Y. Sasaki, V. Rijmen and Y. Todo, Y. *Cryptanalysis of reduced-round SIMON32 and SIMON48*. In: Meier W., Mukhopadhyay D. (eds), Indocrypt 2014. LNCS, vol. 8885, p. 143160. Springer, Cham, 2014.