I. J. Computer Network and Information Security, **2021, 4, 14-25**
Published Online August 2021 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2021.04.02

Modified Key Derivation Function for Enhanced Security of Speck in Resource-Constrained Internet of Things

Roman Alex F. Lustro
Information and Communication Technology Department, Isabela State University, Philippines
E-mail: romanalex.f.lustro@isu.edu.ph

Received: 01 April 2021; Revised: 22 April 2021; Accepted: 07 May 2021; Published: 08 August 2021

Abstract: Randomness is an imperative component in every cryptographic algorithm to guarantee secret keys are unpredictable and secured against all forms of attacks. Speck generated sequence is non-random, a clear sign that it falls beyond the acceptable success rate when tested in statistical analysis. Thus, this study resolves the non-randomness by integrating a novel key derivation function that uses elementary operators designed for lightweight application. This design aims not to compromise performance when implemented on software and hardware. As a result, the modified Speck successfully passed the NIST SP 800 - 22 and die harder v3.31.0 Statistical Test Analysis as no p-value is flagged as failed during testing. Hence, making modified Speck cryptographically secured. Nevertheless, a 1.06% decrease in the figure of merit of the modified Speck still makes it worthier in a resource-constrained Internet of Things application as contrasted to Speck because it is proven to be beyond cryptographically secured.

Index Terms: Communications technology, Computer networks, Cyberspace, Network security, Internet of Things, Cryptographic protocols, Wireless communication, Wireless networks.

1. Introduction

The Internet of Things (IoT) is a growing platform and expands exponentially. This newfound innovation has had a remarkable effect on people's lives and the business sector as it empowers companies to automate processes and decrease employment costs [1]. It improved service quality, making it economical for production and goods delivery. Besides, IoT offer industries with an instantaneous guise on how their systems work by providing visions into everything from machine performance to distribution channel [2]. More industries realize the capability of interconnected devices to keep them viable as it continues to improve. Similarly, it is anticipated to have remarkable growth to reach over 28 billion in the next years to come. However, the volume of data gathered by IoT devices in monitoring activities anytime and anywhere can create a distressing risk to people's privacy and security [1]. The illicit access and unethical use of personal data in resource-constrained devices are among the possible threats that can harm end-users.

Lightweight cryptography exhibits a significant part in the security of resource-constrained devices. Several innovative and inexpensive lightweight ciphers have been introduced to offer solid security. At the moment, notable lightweight block ciphers are recommended, such as PRESENT[3], PRINT [4], EPCBC[5], MIBS [6], LED [7], Piccolo [8], LBlock [9], TWINE [10], KLEIN [11], PRINCE [12], ITUbee [13], RECTANGLE [14], QTL [15], GIFT [16], Skinny [17] and Simon and Speck [18]. The Substitution-Permutation Network (SPNs) and Feistel-type structures are the two main classical designs of these lightweight ciphers.

Among this Feistel – type structure is the Speck lightweight block ciphers. It is the algorithm that bears ten instances of various block**s** and key sizes. Speck was created to ensure security towards common adversaries that can suitably encipher and decipher intensive data. Its key scheduling uses a one-up counter to eliminate slide issues. Similarly, SPECK provides exemplary performance in memory usage and code size in software. Also, it has the best throughput on 64 – bit processors thus outperforming other noteworthy lightweight block ciphers [19-21].

However, the randomness performance of Speck is beyond the acceptable success rate, a clear manifestation that the algorithm's sequence is not secured [22]. Since Speck requires manual input of key by the user, the use of such pre-set public string intensifies predictability level and offers cryptanalyst partial understanding of the key, therefore jeopardizing security [23-27]. Hence, this study resolves the non-randomness issue of Speck through the integration of a novel key derivation function that uses elementary operators, designed for lightweight applications to ensure performance applicability during software and hardware implementation.

2. Related Works

The indication of feeble security parameters classically implemented these days has made IoT security a hot button issue [28]. Regrettably, no established and steady security solutions exist at this time. The newness of the problem and the instability of the hardware and software devices call for further research [29]. Various attempts have been made towards the effective characterization of IoT systems to cope with this issue. Survey papers [30-32] have been published that tackle reviews of enabling technologies, IoT applications, architectural requirements, device platforms, and network topologies which proposed different perspectives of IoT systems, etc.

This leads to the development of lightweight cryptographic algorithms that are intended for low–resource environment and must be able to cater less overhead, minimal power consumption, and acceptable security level [33]. It is more thought-provoking to precisely define low-cost due to its compelling dependence on the software and hardware platforms [34, 35]. To classify a cipher as lightweight, key scheduling, operations, key-size, and block-size are the main properties [36]. Hence, lightweight cipher means simplifying the key schedule by applying elementary operations and increasing the number of encryption and decryption rounds.

A comprehensive review on the implementations of lightweight block ciphers' essential security, performance metrics, and sources inaccuracies and deviations were discussed in [37]. The energy /bit metric was designated as the utmost suitable metric for energy-constrained designs. A survey evaluation showed PRESENT as the top-performing cipher in various metrics. It has a decent reference for hardware implementation in terms of cost, speed, and balanced efficiency [38]. However, when evaluated in terms of clock-cycle-per block, throughput, Figure of Merit (FOM), and Gate Equivalent (GE), Simon, Piccolo, and Speck is the best lightweight block ciphers in hardware implementation.

Among the three, Speck has the best value in terms of energy requirement and consumer memory when implemented in ATmega128, WSN [10], and FELICS [39]. It has the best performance optimized for hardware and software implementations [45, 40, 41]. The design is sophisticated and straightforward, that is easy to find efficient implementations [42]. It operates using basic arithmetic operations that can adaptively encrypt and decrypt a large amount of data. [43]. It uses one-up counter to eliminates slide issues during key schedule. It provides security against traditional adversaries. However, Speck resistance against common attacks was not discussed by the design team and left the task of analyzing the security of their constructions to the research community [44, 45]. No security evaluation was provided and no analysis of its cryptographic strength was given [43]. In the same way, the first and last rounds of Speck do nothing cryptographically as it omits plaintext and ciphertext key whitening operations [46]. Key whitening improves brute force attack complexity by intensifying the actual size of the key without much modification in the algorithm to increase security. Moreover, Speck has been subjected to cryptanalysis to verify the utmost number of rounds that would be vulnerable to attack [19, 43, 41]**,** and [47]. Likewise, Speck sequence is beyond the acceptable success rate when tested in NIST Statistical Test Suite, a clear sign that the generated sequence is non-random[22].

Randomness is an integral component of every cryptography algorithm. This should guarantee that generated secret keys are random, secured against known attacks, anonymity, privacy, and unpredictable [48]. Also, randomness performs multiple roles to safeguard the suitable strength of the algorithms [49]. Hence, random numbers suitability is essential to all computer systems security [50].

A cryptographically secured pseudo-random number generator (PRNGs) is the key to safeguarding sensitive data. It plays a significant part in network security applications and the like [51]. The seed or the entropy is the principal input for PRNG. It is the sequence of numbers used to generate random values. If the entropy or seed can be predicted then the PRNG fails [52]. In this regard, a lot of researchers ventured into the development of a lightweight random number generator that fits in this resource-constrained environment. A study revealed that using data from six smartphone sensors as a source slightly improves randomness, but failed to pass any known batteries of the test [52]. Similarly, another sensor-based RNG uses the noise generated from sensor data to produce random values [53]. The randomness of SensoRNG was evaluated using statistical analysis and implies that it can produce high-quality random values with additional overhead. However, sensor-based RNGs cannot recurrently produce on demand a single sequence of random values. In addition, a competitive yet affordable True Random Number Generator TRNG based on Micro Controller Unit (MCU) improved the efficiency of cryptographic algorithms deployed in the Internet of Things [54]. It successfully passed the NIST statistical test suite. But, TRNG based on MCU uses ASIC boards or additional FPGA in producing TRNG features to generate random numbers base on hardware making the system costly and difficult to apply in software.

The use of predetermined public string may increase the predictability and exposes to a cryptanalyst partial knowledge of the key, thus jeopardies the security. The aim of Key Derivation Functions (KDFs) is to create unpredictable and random secret keys. So, [55] proposed KDF that is completely key-dependent that cryptanalyst has to precisely foresee completely the elements in the key string. A Quasigroup-based key metadata expansion and reduction functions are integrated with the design. The algorithm shows a significant response that any modification in the input will have a definite impact on the output. This eliminates chosen plaintext and the related message attack. Furthermore, [56] simulated the key derivation function of Trivium, Sosemanuk, and Rabbit stream ciphers. The developed KDF demonstrated efficiency advantages as compared to the frequently employed KDFS THAT are designed on block ciphers and hash functions. However, a stream cipher is secured only if the original cipher is secured. Also, [57] applied a novel output expansion technique on Message Authentication Code (MAC) with feedback mode. It adds security to the original MAC in handling arbitrary-length-output messages. As a result, XKDF is cryptographically secured if the character set is complicated enough and the material key size is longer than six.

As discussed above, several approaches to the development of lightweight cryptographic algorithms have been proposed by various researchers. This only shows that the research community is really serious in addressing the Internet of Things security issues.

3. Research Method

The study was performed in three phases namely, modification, implementation, and evaluation. The modification of the algorithm was achieved by developing a novel key derivation function that uses elementary operators as presented in the next section. The modified Speck was assessed in terms of statistical analysis to evaluate the randomness of the binary sequence generated. The encryption and decryption time were also assessed to determine the amount of time consumed. The (.ino) file of the modified Speck was imported and implemented in the two resource-constrained devices namely: ATmega 328P and ATmega 2560 to measure its performance.

3.1. Modification

The modified Speck was created based on the original design. A key is needed to encipher inputs. The key derivation is performed in a one-way execution. In the same way the input is manipulated to suit the required size, the input fixed-key size is applied in key derivation along with the generated value of the 64-bit tempgenkey function. Hence, the input plaintext serves as a seed in the key derivation function process as presented below. The generated key from the KDF serves as the key material in the key scheduling and encryption process respectively**.** The formula for the encryption process is shown in (1).

```
Key Derivation
PreKey ← P1 ⊕ P2|| P2 ⊕ P3|| P3 ⊕ P4|| P4 ⊕ P1'
Q ← PreKey = 0..n
for p = 0..a
$k \leftarrow P_{a+1} >>> a^{ŋ}$
end for

Key Expansion
for i = 0..T-2
$\ell[i+m-1] \leftarrow (k[i] + S^{-\alpha}\ell[i]) \oplus i$
$k[i+1] \leftarrow S^{\beta} k[i] \oplus \ell[i+m-1]$
end for

Encryption
for i = 0..T-1
$x \leftarrow S^{-\alpha}x + y) \oplus k[i]$
$y \leftarrow S^{\beta}y \oplus (S-^{\alpha}x + y) \oplus k[i]$
end for
```

Where:
n = word size (32, 48, 64)
m = number of key words (must be 3 or 4 if n = 32, 2 or 3 if n = 48, 2 or 3 or 4 if n = 64)
T = number of rounds = 26 or 27 if n = 32, m = 3 or 4
= 28 or 29 if n = 48, m = 3 or 4
= 32, 33, or 34 if n = 64, m = 3 or 4
(α, β, ŋ) = (8, 3, 7)
x,y = plaintext
r,p = tempkeygen
ℓ[m-2].. ℓ[0], k[0] = key words
Q = rlut

Key Scheduling

The derived key from the KDF serves as the key material and is divided into two ($x_{2i}+1$, x_{2i}).

1. Rotate (x_{2i}+1) to the left by α
2. Perform addition modulo 2^n from the result of the rotated x_{2i}+1 (from step i) with x_{2i}
3. Rotate (x_{2i}) to the right by β
4. Perform XOR from the result of the addition modulo 2^n, (from step ii) with the key from x_{2i}.
5. Perform XOR from the result of (step iv) with the result of (step iii).
6. This served as the input key in the encryption process.

Note that α = 8 and β = 3 respectively for both encryption and key scheduling processes. This paper uses Speck's original key expansion and encryption technique. Except for the key derivation technique, which is the novel contribution of this paper.

Encryption process

User Input Plaintext
The input plaintext is split into two (x,y)

1. Rotate (x) to the left by α
2. Perform addition modulo 2^n from the result of the rotated x (from step i) with y
3. Rotate (y) to the right by β
4. Perform XOR from the result of the addition modulo 2^n, (from step ii) with the initial key from x.
5. Perform XOR from the result of (step iv) with the result of (step iii)

$$R_k(x,\, y) \;=\; ((S^{-\alpha}x \,+\, y) \oplus k,\; S^{\beta}y \oplus (S^{-\alpha}x \,+\, y) \oplus k) \tag{1}$$

Note that the input plaintext from the user is of arbitrary length. To achieve the desired block size, a padding technique is used by appending zeros.

Decryption process

The ciphertext served as an input to generate the plaintext and key. To reverse engineer this process, the *x* that was rotated to the left by *α* during encryption is rotated to the right by *α*, the y that was rotated to the right by *β* is shifted to the left by *β*. The addition modulo 2^n is changed to subtraction modulo 2^n. The formula for decryption is stated in (2).

$$R^{-1}{}_k(x,\, y) \;=\; (S^{\alpha}((x \oplus k) \,-\, S^{-\beta}(x \oplus y)),\; S^{-\beta}(x \oplus y)) \tag{2}$$

3.2. Implementation

The Speck and modified Speck are implemented in two resource-constrained IoT devices. The ATmega 328P and ATmega2560 are the subjects of the implementation to measure the throughput, efficiency, and the figure of merit performances of the three variants.

Software

NetBeans IDE version 8.1 is used for the coding of the algorithm. The NIST SP 800 – 22 and DieHarder version 3.31.0 are used for Statistical Test Analysis. The java Running Average Power Limit (jRAPL) [22] for measuring power consumption.

Hardware

The performance of Speck and modified Speck are simulated in a personal laptop computer operating on Intel (R) Core (TM) i5 – 7200U CPU with 2.50 GHz 2.70 GHz Processor, 4 GB RAM Memory and 64- bit. An Arduino Uno with ATmega 328P Processor, 32 (kB) Flash Memory, 2 (kB) SRAM, 16 MHz Clock Speed, 1(kB) EEPROM, and 5V Voltage Level. Similarly, Arduino Mega ATmega 2560 Processor, 16 MHz Clock Speed, 4 (kB) EEPROM, 256 (kB) Flash Memory, 8 (kB) SRAM, and 5V Voltage Level.

3.3. Evaluation

Data

A total of 54 million ciphertexts were generated using the three variants from the modified Speck for the statistical test analysis depending on the data category tested.

Statistical Test Analysis

The NIST SP 800-22 and DieHarder v3.31.0 are two of the stringent tests used to assess the randomness of a cryptographic algorithm. The NIST statistical test suite is selected because it covers extensive metrics of randomness.

Consequently, DieHarder v3.31.0 test does not offer precise metric for pass or fail. Only the p - values for pass must be in the range of [>=0.025 - <=0.975], for weak from the range of p-value [>=0.000, < 0.025 and > 0.975, <=0.999], and p-value from the range of [>=0, <=0.000 and >=0.999, <=1] (i.e., close to 0 or 1) implies fail.

Combined Metrics (CM)

The encryption and decryption are performed ten times to measure the average combined metrics. Then, take the product of the code size$_{\text{bits}}$ and encryption/decryption cycle count. The block sizes refer to each algorithm's variant. The evaluation of performance on encryption and decryption time is the ratio of code size$_{\text{bits}}$ multiplied by encryption cycle count$_{\text{ms}}$ together with the algorithm variant block size$_{\text{bits}}$ as depicted in (3).

$$CM = \frac{codesize[bits] \times encryption / decryptioncyclecount[ms]}{blocksize[bits]} \tag{3}$$

Software Performance

The software performance**s** of Speck and modified Speck are measured in throughput$_{\text{mbps}}$ using (4).

$$throughput_{mbps} = \frac{blocksize_{bits}}{executime_{ns}} \times frequency_{mhz} \tag{4}$$

Hardware Performance

The energy per bit consumption of the ciphers power was computed in terms of consumption$_{joules}$, execution time$_{ns}$, frequency$_{\text{MHz}}$, and block size$_{\text{bits}}$. The java Running Average Power Limit (jRAPL) is used to obtain the power consumption. The energy efficiency performance is measured using (5).

$$efficiency_{energy/bit} = \left(\frac{power_{joule} \times executiontime_{ns}}{frequency_{mhz} \times blocksize_{bit}} \right) \tag{5}$$

Figure of Merit

The obtained value $P_{i,d}$ from (6) combines the three parameters of M = {RAM consumption, code size, execution time}. For the performance parameters $P_{i,d}$ are computed in every implementation $_i$, and device $_d$.

$$P_{i,d} = \sum_{m \in M} W_m \frac{V_{i,d,m}}{\min\left(V_{i,d,m}\right)} \tag{6}$$

The cost of the metric *m* on device *d* and implementation *I is denoted by* $V_{i,d,m}$; The relative weight of metric *m* and min_i *(*$V_{i,d,m}$*)* is the minimum value of the metric *m* from full execution of all treated ciphers on the same identified platform *d is refered to as* W_m. Where: $W_m = 1$ then select the implementation with least $P_{i,d}$ in every target device and cipher. The Figure of Merit (FOM) is computed for every candidate cipher and the two selected implementations i_1, i_2, using (7) as the average performance value on the two devices. The FOM is the result of overall performance indicating higher throughput and lower efficiency.

$$FigureofMerit(FOM)_{i_1,i_2} = \frac{P_{i_1 atmega228p+} P_{i_2 atmega2560}}{2} \tag{7}$$

4. Results and Discussion

The result of the test performed using the three data categories of the three variants of the modified Speck is shown in Table 1. As observed, all the tests performed were above the acceptable success rate of 99.60% based on the 5% significance level. For the non – overlapping, the acceptable success rate is 99.88% because of the 148-subtest performed compared to the others which have only one test. With the above result, all the three variants for the modified Speck successfully passed the randomness test on SKA, SPA, and PCC.

4.1. Randomness Analysis of the modified Speck using NIST SP 800 – 22

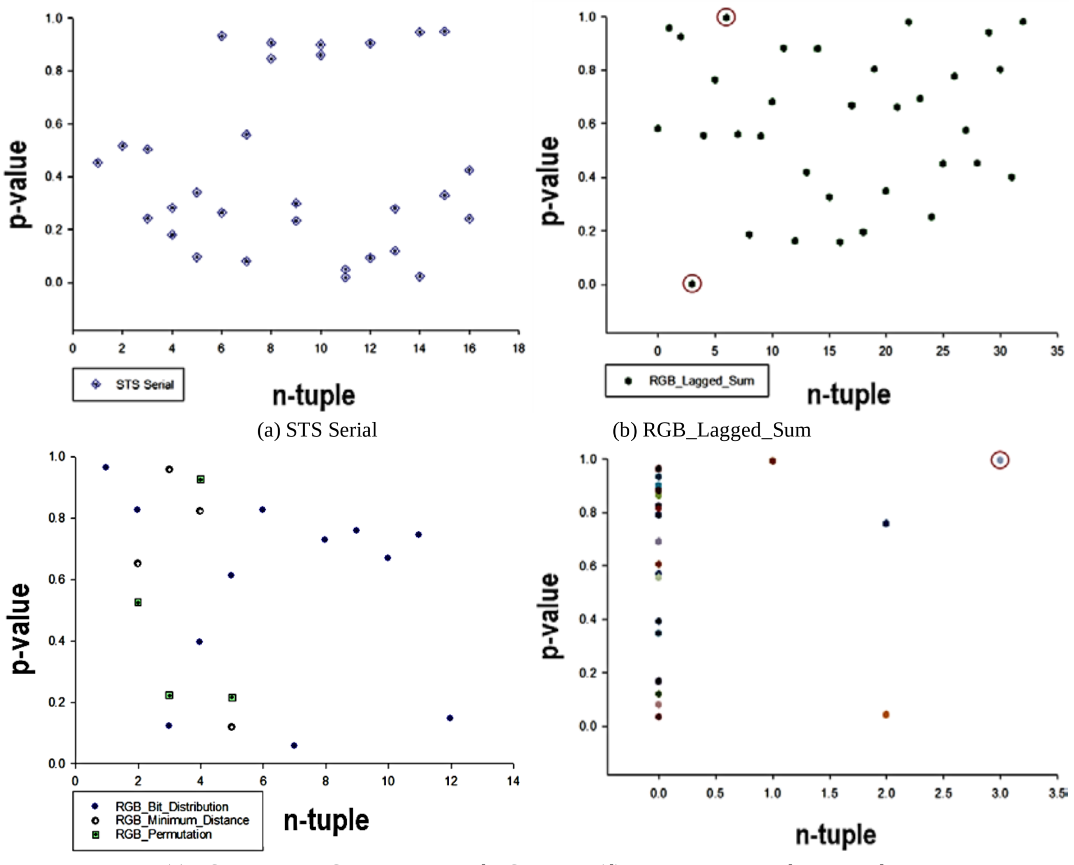
Table 1. The randomness of the modified speck over speck using NIST STS SP 80 – 22

Data Categories	Test Selection	Speck Chew, et al., [22] (%)	Acceptable Success Rate (%)	Modified Speck (%)
		128/128 variant		
LDP	Runs	99.5	99.6	99.9
SPA	Overlapping	99	99.6	99.8
		128/192 variant		
SKA		99.76	99.81	99.82
SPA	Random Excursion Variant	99.76	99.81	99.84
RPRK		99.76	99.81	99.86
LDK	Overlapping	99.4	99.6	100
		128/256 variant		
SKA	Random Excursion Variant	99.8	99.81	99.82
PCC	Random Excursion	99.74	99.77	99.86
HDK	Spectral DFT	99	99.6	99.9
LDP	Non – Overlapping	99.86	99.88	99.9
HDP		99.86		99.9

All the variants of the modified Speck are above the acceptable success rate making it, 22.2%, 44.4%, and 55.6% respectively better than the three variants of Speck. Thus, making the modified Speck cryptographically secured.

4.2. Randomness Analysis of the modified Speck using Die Harder v3.31.0

The Die Harder is utilized to further assess the randomness performance of the modified Speck. The results for variant 128/128, 128/192, and 198/256 is shown in Fig. 1, 2, and 3 respectively. The encircled ones indicate that p – values are weak but do not mean that they failed.



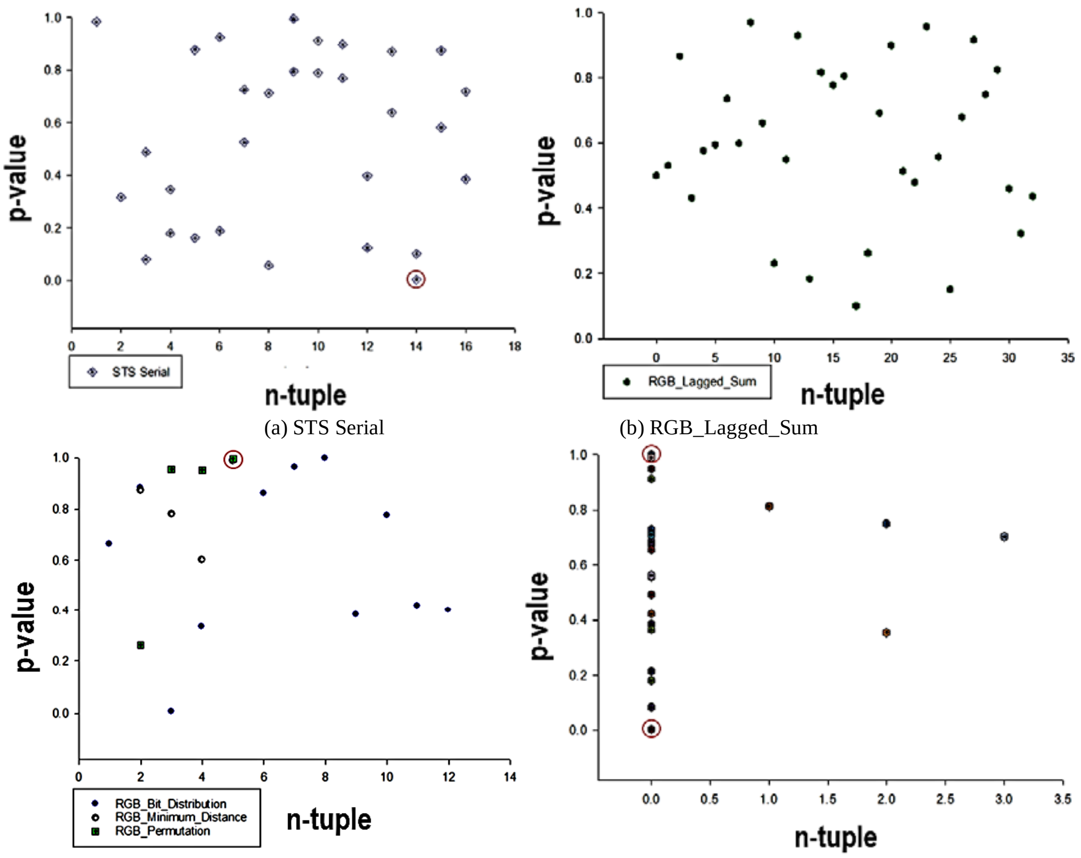
(a) STS Serial (b) RGB_Lagged_Sum

(c) RGB BitDist; RGB Min _Dist and RGB Perm (d) Remaining test with one p-value

Fig.1. Die Harder randomness test results of modified Speck 128/128 variant

As shown in Fig. 1, all the p – values passed the randomness testing on subplot (a) STS Serial test. On contrary, two tests were flagged as weak with a p-value of 0.00165 on n-tuple 3 and 0.99700 on n-tuple 6 respectively on subplot

(b) RGB_Lagged_Sum. On the other hand, subplot (c) shows good randomness, as all the test is within the passing range of p-value >=0.025 - <=0.975. While 24 individual tests revealed that the diehard 3dsphere test was flagged as weak with a p-value of 0.99502 on subplot (d). Take note that, the test flagged as weak is not considered as failed in DieHarder testing.

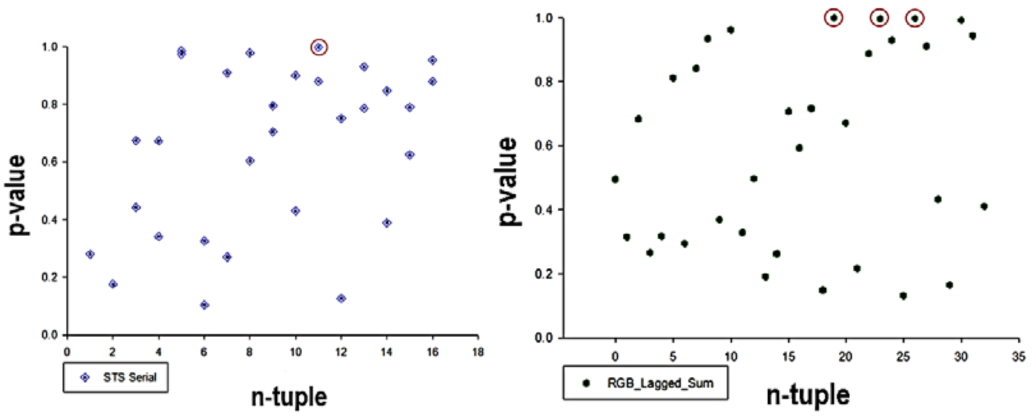


(a) STS Serial (b) RGB_Lagged_Sum

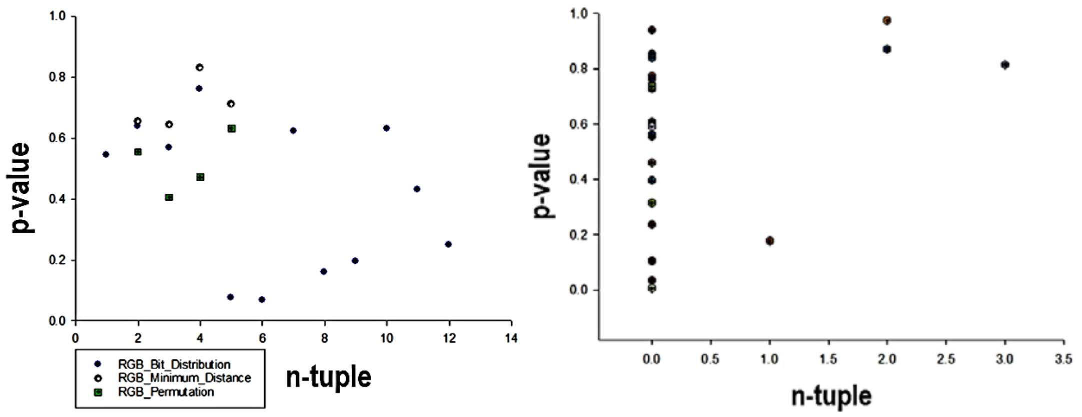
(c) RGB BitDist; RGB Min _Dist and RGB Perm (d) Remaining test with one p-value

Fig.2. DieHarder randomness test results of modified Speck 128/192

Only one out of the 30 STS serial tests are flagged as weak with a p-value of 0.00477, the rest of the test passed the randomness testing as shown on subplot (a) based on Fig. 2. While on subplot 3, rgb_bitbist and rgb_permutations p – values of 0.99839 and 0.99696 were implying that the bit sequences tested are weak. Similarly, diehard_bitstream and diehard_craps on subplot (d) with 0.99905 and 0.00281 p–values respectively are flagged as weak as they are out of the acceptable passing range of >=0.025 - <=0.975 p-values. On contrary, the RGB_Lagged _Sum on subplot (b) shows that all the 33-test performed are within the acceptable passing range implying that the bit sequences provide good randomness.



(a) STS Serial (b) RGB_Lagged_Sum



(c) RGB BitDist; RGB Min _Dist and RGB Perm (d) Remaining test with one p-value

Fig.3. DieHarder randomness test results of modified Speck 128/256 variant

The p-value distribution on the different tests performed is demonstrated in Fig. 3. It can be noticed that most p – values on the four subplots (a, b, c, & d) are within the acceptable passing range except for STS Serial n-tuple 11 with 0.99769 p-values and RGB_Lagged_Sum 19, 23, & 26 with 0.99969 0.99686 and 0.99777 p – values respectively which are flagged with weak randomness.

4.3. Encryption and Decryption Combined Metrics

The test was performed ten times for the three variants of Speck and modified Speck for the encryption cycle count, as shown in Fig. 4. The average encryption time served as input for the encryption cycle count.

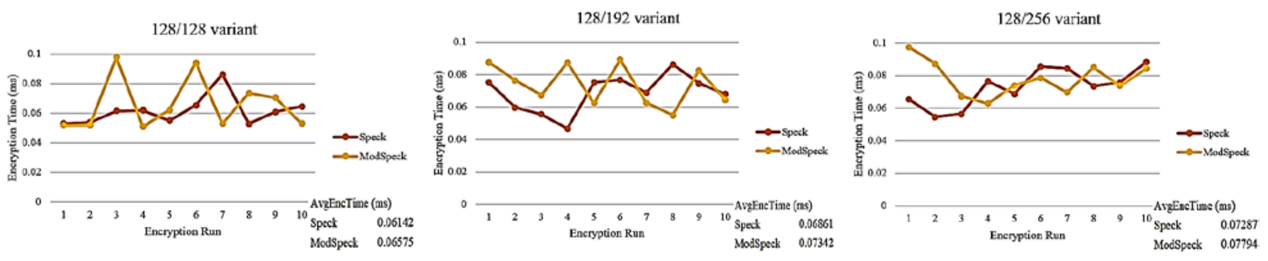


Fig.4. Comparison on Encryption Average Run Times of Speck and ModSpeck

As shown in Fig. 4 all the variants of Speck achieved lesser encryption time. This is because no key derivation function is performed as compared to the modified Speck.

Table 2. Encryption combined metrics performance evaluation

Parameters	**128/128 Variant**		**128/192 Variant**		**128/256 Variant**	
	Speck	**ModSpeck**	**Speck**	**ModSpeck**	**Speck**	**ModSpeck**
Encryption Cycle Count$_{ms}$	0.06142	0.06575	0.06361	0.06787	0.08779	0.09264
Code Size$_{bits}$	145392	151264	145392	151264	145392	151264
Block Size$_{bits}$	128	128	128	128	128	128
Combined Metrics$_{ms}$	69.77	77.70	72.25	80.21	99.72	109.48
Change (%)		**-11.37**		**-11.01**		**-9.79**

An increase in the encryption cycle count and code size in the modified Speck resulted in a bigger combined metric as shown in Table 2. The average increase variance of 10.72% in the combined metrics on the encryption process was achieved. This is the result of the additional key derivation function in the modified Speck.

Meanwhile, decryption was implemented in original and modified Speck ten times as shown in Fig. 5.

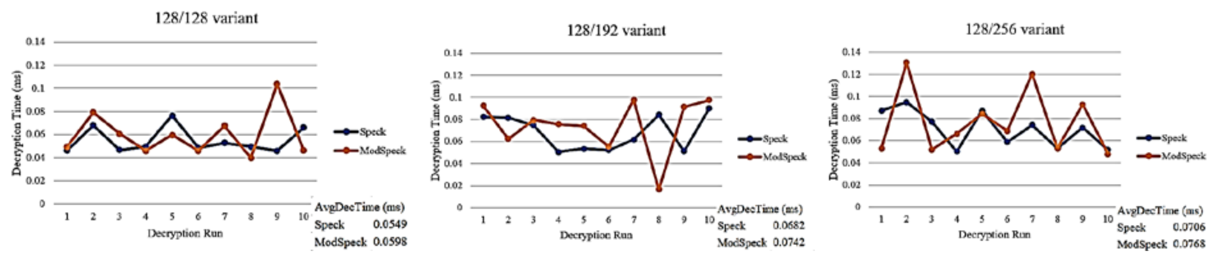


Fig.5. Comparison on Decryption Average Run Times of Speck and ModSpeck

Speck achieved minimal average decryption time as compared to modified Speck as presented in Fig. 5. Likewise, the decryption performance evaluation in terms of combined metrics is presented in Table 3.

Table 3. Decryption combined metrics performance evaluation

Parameters	128/128 Variant		128/192 Variant		128/256 Variant	
	Speck	ModSpeck	Speck	ModSpeck	Speck	ModSpeck
Decryption Cycle Count$_{ms}$	0.05488	0.05975	0.06819	0.07422	0.07059	0.07682
Code Size$_{bits}$	145392	151264	145392	151264	145392	151264
Block Size$_{bits}$	128	128	128	128	128	128
Combined Metrics$_{ms}$	62.34	70.61	77.46	87.71	80.18	90.78
Change (%)		**-13.27**		**-13.24**		**-13.22**

The amount of time consumed to decrypt a ciphertext is lesser than the time consumed to encrypt having the same code and block size, this is because no key derivation technique was performed during decryption as reflected in Table 3. Hence, a lesser decryption average variance of -13.24 % is attained.

4.4. The figure of merit (FOM) performance of the different lightweight block ciphers

The overall performance evaluation in terms of the figure of merit of the different lightweight block ciphers implementation in resource-constrained devices is presented in Table 4.

Table 4. Speck and Modspeck overall performance evaluation

ALGORITHM		ATMega 328P						ATMega 2560						Figure of Merit (FOM)
		Global Variables (kb)	Code Size (kb)	RAM (bits)	Execution Time (Cycles)	Efficiency (mbps)	Throughput (energy/bit)	Global Variables (kb)	Code Size (kb)	RAM (bits)	Execution Time (Cycles)	Efficiency (mbps)	Throughput (energy/bit)	
128	ModSPECK	0.535	18.22	3136	11355	**0.18**	**0.79**	0.535	20.52	3136	10690	**0.19**	**1.08**	**9.34**
	SPECK	0.397	17.95	3120	10353	**0.20**	**0.56**	0.397	20.37	3120	10797	**0.19**	**1.07**	**9.25**
	Simon	0.397	16.92	3040	10787	**0.19**	**0.66**	0.397	17.68	3040	10543	**0.19**	**1.04**	**9.05**
192	ModSPECK	0.535	18.22	3136	13010	**0.16**	**0.91**	0.535	20.52	3136	12270	**0.17**	**1.77**	**9.50**
	SPECK	0.397	17.95	3120	12846	**0.16**	**0.71**	0.397	20.37	3120	11343	**0.18**	**1.39**	**9.40**
	Simon	0.397	16.92	3040	10435	**0.20**	**0.75**	0.397	17.68	3040	10432	**0.20**	**1.30**	**9.03**
256	ModSPECK	0.535	18.22	3136	15226	**0.13**	**1.10**	0.535	20.52	3136	14816	**0.14**	**2.26**	**9.73**
	SPECK	0.397	17.95	3120	14898	**0.14**	**0.85**	0.397	20.37	3120	13766	**0.15**	**2.05**	**9.62**
	Simon	0.397	16.92	3040	10234	**0.20**	**0.87**	0.397	17.68	3040	10221	**0.20**	**1.16**	**9.01**

The Speck can process an average of 5.71% higher number of megabits per second on three variants when compared with modified Speck while Simon is 12.57% better when compared with the latter on the ATmega 328P as depicted in Table 4. Similarly, the modified Speck is 4.07% slower when contrasted with Speck and 1.81% slower compared to Simon on the ATmega 2560 implementation. This is because of the increase in ram consumption, code size, and execution time associated with the integration of a key derivation function. Similarly, the modified Speck consumes an average 34.98% higher total of energy/bit on three variations when contrasted with Speck and 25.80% on Simon on the ATmega 328P. Likewise, the modified Speck exhausts an average of 12.84% higher amount of energy/bit on three variants when contrasted with Speck and 44.65% on Simon on the ATmega 2560 implementation. Hence, there is a tradeoff between the cipher’s security and its hardware implementation due to the increase in energy usage.

The balance concerning performance, cost, and security are the primary considerations in the design of a lightweight cryptographic algorithm. Performance and cost, performance and security, cost and security, or two of these three principles must be achieved. But, achieving all three at the same time is a challenge [58-61]. Hence, this research work was able to achieve better security at the same cost without compromising performance. The minimal increase in the figure of merit still makes Modified Speck a better choice for resource-constrained IoT implementation because it is more cryptographically secured.

5. Conclusion

Although there are some studies conducted to exploit the security strength of Speck in terms of linear and differential cryptanalysis, none of them attempted to address the non-randomness, which is a very fundamental requirement to all cryptographic algorithms. In this study, Speck was modified by integrating a novel key derivation function in its processes using basic operators intended for lightweight applications to address its non-randomness. The NIST SP 800 – 22 and Die Harder v3.31.0 were used to evaluate statistical analysis. The tests revealed that the modified Speck successfully passed all the 9 data categories tests in NIST and the 30 tests in Die Harder v3.31.0. Similarly, modified Speck performance was determined in ATmega 328P and ATmega 2560. The enhanced Speck turned out well a competitive number of processed bits completed per second. The hardware performance shows that the improved algorithm achieved an able energy/bit when compared with Speck. More importantly, the figure of merit exhibited commendable performance as compared to Speck.

References

[1] M. M. Ogonji, G. Okeyo, and J. M. Wafula, “A survey on privacy and security of Internet of Things,” *Comput. Sci. Rev.*, vol. 38, p. 100312, 2020, doi: 10.1016/j.cosrev.2020.100312.

[2] M. U.Farooq, M. Waseem, A. Khairi, and S. Mazhar, “A Critical Analysis on the Security Concerns of Internet of Things (IoT),” *Int. J. Comput. Appl.*, vol. 111, no. 7, pp. 1–6, 2015, doi: 10.5120/19547-1280.

[3] G. Sravya, M. O. V. P. Kumar, G. Merlin Sheeba, K. Jamal, and K. Mannem, “Hardware lightweight design of PRESENT block cipher,” *Mater. Today Proc.*, vol. 33, no. xxxx, pp. 4880–4886, 2020, doi: 10.1016/j.matpr.2020.08.435.

[4] L. Knudsen, G. Leander, A. Poschmann, and M. J. B. Robshaw, “PRINTcipher: A Block Cipher for IC-Printing,” in *Cryptographic Hardware and Embedded Systems, CHES 2010*, 2010, pp. 16–32, doi: 10.1007/978-3-642-15031-9_2.

[5] P. Yang, C. Wu, and W. Zhang, “Automatic Security Analysis of EPCBC against Differential Attacks,” *Procedia Comput. Sci.*, vol. 107, no. Icict, pp. 176–182, 2017, doi: 10.1016/j.procs.2017.03.075.

[6] M. H. F. Sereshgi, M. Dakhilalian, and M. S. Shakiba, “Biclique cryptanalysis of MIBS-80 and PRESENT-80 block ciphers,” *Secur. Commun. NETWORKS*, vol. 9:, no. October 2016, pp. 27–33, 2016, doi: 10.1002/sec.

[7] A. R. Raza, K. Mahmood, M. F. Amjad, H. Abbas, and M. Afzal, “On the efficiency of software implementations of lightweight block ciphers from the perspective of programming languages,” *Futur. Gener. Comput. Syst.*, vol. 104, pp. 43–59, 2020, doi: 10.1016/j.future.2019.09.058.

[8] P. Li *et al.*, “Efficient implementation of lightweight block ciphers on volta and pascal architecture,” *J. Inf. Secur. Appl.*, vol. 47, pp. 235–245, 2019, doi: 10.1016/j.jisa.2019.04.006.

[9] Y. Wei, Y. Rong, and X. A. Wang, “New differential fault attack on lightweight cipher LBlock,” *Proc. - 2016 Int. Conf. Intell. Netw. Collab. Syst. IEEE INCoS 2016*, pp. 285–288, 2016, doi: 10.1109/INCoS.2016.32.

[10] A. G. Bafghi, “Software Implementation And Evaluation Of Lightweight Symmetric Block Ciphers Of The Energy Perspectives And Memory,” *Int. J. Eng. Educ.*, vol. 9, no. 2, pp. 1–6, 2017.

[11] P. Singh, B. Acharya, and R. K. Chaurasiya, “High Throughput Architecture for KLEIN Block Cipher in FPGA,” *IEMECON 2019 - 9th Annu. Inf. Technol. Electromechanical Eng. Microelectron. Conf.*, pp. 64–69, 2019, doi: 10.1109/IEMECONX.2019.8877021.

[12] S. Takemoto, Y. Nozaki, and M. Yoshikawa, “Evaluation of the hiding-countermeasure PRINCE using differential power analysis,” *2019 IEEE 8th Glob. Conf. Consum. Electron. GCCE 2019*, pp. 164–165, 2019, doi: 10.1109/GCCE46687.2019.9015513.

[13] J. Liu, W. Li, and G. Bai, “Efficient hardware implementation of ITUbee for lightweight application,” *2017 12th Int. Conf. Internet Technol. Secur. Trans. ICITST 2017*, pp. 372–376, 2018, doi: 10.23919/ICITST.2017.8356424.

[14] D. Sehrawat and N. S. Gill, “Lightweight Block Ciphers for IoT based applications: A Review,” *Int. J. Appl. Eng. Res.*, vol. 13, no. 5, pp. 2258–2270, 2018.

[15] L. Li, B. Liu, and H. Wang, “QTL: A new ultra-lightweight block cipher,” *Microprocess. Microsyst.*, vol. 45, pp. 45–55, 2016, doi: 10.1016/j.micpro.2016.03.011.

[16] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim, and Y. Todo, “GIFT: A small present: Towards reaching the limit of lightweight encryption,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10529 LNCS, pp. 321–345, 2017, doi: 10.1007/978-3-319-66787-4_16.

[17] B. Nallathambi and K. Palanivel, “Fault diagnosis architecture for SKINNY family of block ciphers,” *Microprocess. Microsyst.*, vol. 77, p. 103202, 2020, doi: 10.1016/j.micpro.2020.103202.

[18] R. Beaulieu, D. Shors, J. Smith, and S. Treatman-clark, “The simon and speck families of lightweight block ciphers,” *Cryptol. ePrint Arch.*, no. National Security Agency. USA, pp. 1–42, 2013, doi: 10.1145/2744769.2747946.

[19] J. Abed, Farzaneh; List, Eik; Lucks, Stefan; Wenzel, “Differential Cryptanalysis of Round - Reduced SIMON and SPECK,” 2014.

[20] J. Ren and S. Chen, "Cryptanalysis of Reduced-Round SPECK," *IEEE Access*, vol. 7, pp. 63045–63056, 2019, doi: 10.1109/ACCESS.2019.2917015.
[21] A. D. Dwivedi, P. Morawiecki, and G. Srivastava, "Differential Cryptanalysis of Round-Reduced SPECK Suitable for Internet of Things Devices," *IEEE Access*, vol. 7, no. c, pp. 16476–16486, 2019, doi: 10.1109/ACCESS.2019.2894337.
[22] A. A. Chew, L C N; Shah, I N M; Adbullah, N A N; Zawawi, N H A; Rani, H A; Zakaria, "Randomness Analysis on Speck Family Of Lightweight Block Cipher Cryptography Development Department , Cyber Security Malaysia ," *Int. J. Cryptol. Res.*, vol. 5, no. June 2013, pp. 44–60, 2015.
[23] J. M. McGinthy and A. J. Michaels, "Session Key Derivation for Low Power IoT Devices," *Proc. - 4th IEEE Int. Conf. Big Data Secur. Cloud, BigDataSecurity 2018, 4th IEEE Int. Conf. High Perform. Smart Comput. HPSC 2018 3rd IEEE Int. Conf. Intell. Data Secur.*, pp. 194–203, 2018, doi: 10.1109/BDS/HPSC/IDS18.2018.00050.
[24] P. Zacek, D. Malanik, P. Holbikova, R. Jasek, and L. Kralik, "Using the SHA-3 to derive encryption keys based on key-file," *Proc. - 2018 2nd Eur. Conf. Electr. Eng. Comput. Sci. EECS 2018*, pp. 348–351, 2018, doi: 10.1109/EECS.2018.00070.
[25] M. Indrasena Reddy, A. P. Siva Kumar, and K. Subba Reddy, "A secured cryptographic system based on DNA and a hybrid key generation approach," *BioSystems*, vol. 197, p. 104207, 2020, doi: 10.1016/j.biosystems.2020.104207.
[26] B. Lac, A. Canteaut, J. J. A. Fournier, and R. Sirdey, "Thwarting Fault Attacks against Lightweight Cryptography using SIMD Instructions," *Proc. - IEEE Int. Symp. Circuits Syst.*, vol. 2018-May, 2018, doi: 10.1109/ISCAS.2018.8351693.
[27] B. Seok and C. Lee, "Fast implementations of ARX-based lightweight block ciphers (SPARX, CHAM) on 32-bit processor," *Int. J. Distrib. Sens. Networks*, vol. 15, no. 9, 2019, doi: 10.1177/1550147719874180.
[28] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT Mirai and Other Botnets-2017-Computer," *Comput. 50*, pp. 80–84, 2017.
[29] W. Z. Khan, M. Y. Aalsalem, and M. K. Khan, "Five acts of consumer behavior: A potential security and privacy threat to Internet of Things," *2018 IEEE Int. Conf. Consum. Electron. ICCE 2018*, vol. 2018-Janua, pp. 1–3, 2018, doi: 10.1109/ICCE.2018.8326124.
[30] I. Yaqoob, E. AhmEd, I. abakEr T. HashEm, abdElmuTTlIb I. abdalla AhmEd, muhammad I. abdullah ganI, and M. GuIzanI, "Internet of thIngs ArchItecture: requIrements, And open chAllenges," *IEEE Wirel. Commun.*, vol. 20, no. 3, pp. 10–16, 2017, doi: 10.1109/MWC.2017.1600421.
[31] E. Borgia, "The internet of things vision: Key features, applications and open issues," *Comput. Commun.*, vol. 54, pp. 1–31, 2014, doi: 10.1016/j.comcom.2014.09.008.
[32] L. Da Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4. 2014, doi: 10.1109/TII.2014.2300753.
[33] X. Fan, K. Mandal, and G. Gong, "WG-8: A Lightweight Stream Cipher for Resource-Constrained Smart Devices," in *International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, 2013, pp. 617–632, doi: 10.1007/978-3-642-37949-9_54.
[34] S. Kerckhof, F. Durvaux, C. Hocquet, D. Bol, and F. X. Standaert, "Towards green cryptography: A comparison of lightweight ciphers from the energy viewpoint," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7428 LNCS, pp. 390–407, 2012, doi: 10.1007/978-3-642-33027-8_23.
[35] T. Eisenbarth *et al.*, "Compact Implementation and Performance Evaluation of Block Ciphers in ATtiny Devices," in *International Conference on Cryptology in Africa*, 2012, pp. 172–187, doi: 10.1007/978-3-642-31410-0_11.
[36] M. Cazorla, K. Marquet, and M. Minier, "Survey and benchmark of block ciphers for wireless sensor networks," 2013.
[37] B. J. Mohd, T. Hayajneh, and A. V. Vasilakos, "A survey on lightweight block ciphers for low-resource devices: Comparative study and open issues," *J. Netw. Comput. Appl.*, vol. 58, pp. 73–93, 2015, doi: 10.1016/j.jnca.2015.09.001.
[38] J. H. Zadeh, "Evaluation of Lightweight Block Ciphers in Hardware Implementation : A Comprehensive Survey," 2016.
[39] S. Kotel, F. Sbiaa, M. Zeghid, M. Machhout, A. Baganne, and R. Tourki, "Performance Evaluation and Design Considerations of Lightweight Block Cipher for Low-Cost Embedded Devices," 2016.
[40] A. V. Duka and B. Genge, "Implementation of SIMON and SPECK lightweight block ciphers on programmable logic controllers," *2017 5th Int. Symp. Digit. Forensic Secur. ISDFS 2017*, 2017, doi: 10.1109/ISDFS.2017.7916501.
[41] H. Tupsamudre, S. Bisht, and D. Mukhopadhyay, "Differential fault analysis on the families of SIMON and SPECK ciphers," *Proc. - 2014 Work. Fault Diagnosis Toler. Cryptogr. FDTC 2014*, pp. 40–48, 2014, doi: 10.1109/FDTC.2014.14.
[42] A. Bossert, S. Cooper, and A. Wiesmaier, "A comparison of block ciphers SIMON , SPECK , and KATAN." 2017, [Online]. Available: https://www.cdc.informatik.tu-darmstadt.de/fileadmin/user_upload/Group_CDC/Documents/Lehre/Seminar_IoT/2016-09-05_TR_SimonSpeckKatan.pdf.
[43] A. Biryukov, A. Roy, and V. Velichkov, "Differential analysis of block ciphers Simon and Speck," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8540, no. June, pp. 546–570, 2015, doi: 10.1007/978-3-662-46706-0_28.
[44] M.-J. O. Saarinen and D. Engels, "A Do-It-All-Cipher for RFID : Design Requirements (Extended Abstract)," *IACR Cryptol. ePrint Arch.*, vol. 2012, 2012, [Online]. Available: http://eprint.iacr.org/2012/317.pdf.
[45] F. Abed, E. List, S. Lucks, and J. Wenzel, "Cryptanalysis of the Speck Family of Block Ciphers," pp. 1–14, 2013.
[46] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," *Proc. 52nd Annu. Des. Autom. Conf. - DAC '15*, pp. 1–6, 2015, doi: 10.1145/2744769.2747946.
[47] Y. Huo, F. Zhang, X. Feng, and L. P. Wang, "Improved differential fault attack on the block cipher SPECK," *Proc. - 2015 Work. Fault Diagnosis Toler. Cryptogr. FDTC 2015*, pp. 28–34, 2016, doi: 10.1109/FDTC.2015.15.
[48] R. Kip, "Why is randomness important in cryptography? - Quora," 2016. https://www.quora.com/Why-is-randomness-important-in-cryptography (accessed Feb. 02, 2018).
[49] K. Marton, A. Suciu, and I. Ignat, "Randomness in digital cryptography: A survey," *Rom. J. Inf. Sci. Technol.*, vol. 13, no. 3, pp.

219–240, 2010.
[50] J. Graham-Cumming, “Why secure systems require random numbers,” 2013. https://blog.cloudflare.com/why-randomness-matters/ (accessed Feb. 02, 2018).
[51] O. Jallouli, M. Abutaha, S. El Assad, M. Chetto, A. Queudet, and O. Deforges, “Comparative study of two pseudo chaotic number generators for securing the IoT,” *2016 Int. Conf. Adv. Comput. Commun. Informatics, ICACCI 2016*, pp. 1340–1344, 2016, doi: 10.1109/ICACCI.2016.7732234.
[52] L. M. Dinca and G. Hancke, “Behavioural sensor data as randomness source for IoT devices,” 2017, doi: 10.1109/ISIE.2017.8001568.
[53] K. Wallace, K. Moran, E. Novak, G. Zhou, and K. Sun, “Toward Sensor-Based Random Number Generation for Mobile and IoT Devices,” *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1189–1201, 2016, doi: 10.1109/JIOT.2016.2572638.
[54] G. Souaki and K. Halim, “Random number generation based on MCU sources for IoT application,” *Proc. - 3rd Int. Conf. Adv. Technol. Signal Image Process. ATSIP 2017*, 2017, doi: 10.1109/ATSIP.2017.8075524.
[55] A. H. Disina, S. Jamel, Z. A. Pindar, and M. M. Deris, “All-or-Nothing Key Derivation Function Based on Quasigroup String Transformation,” *ICISS 2016 - 2016 Int. Conf. Inf. Sci. Secur.*, pp. 1–5, 2017, doi: 10.1109/ICISSEC.2016.7885839.
[56] C. W. Chuah, E. Dawson, and L. Simpson, “Key Derivation Function : The SCKDF Scheme,” in *28th IFIP TC 11 International Conference*, 2013, pp. 125–138, doi: 10.1007/978-3-642-39218-4_10.
[57] X. Chen, X. Li, Y. Chen, P. Li, J. Xing, and L. Li, “A modified PBKDF2-based MAC scheme XKDF,” in *IEEE Region 10 Annual International Conference, Proceedings/TENCON*, 2016, vol. 2016-Janua, no. 4, doi: 10.1109/TENCON.2015.7373109.
[58] M. Pourghasem, E. G. Sheikhloo, and R. E. Atani, “Light Weight Implementation of Stream Ciphers for M-Commerce Light Weight Implementation of Stream Ciphers for M-Commerce Applications,” no. November 2014, 2014.
[59] C. Pei, Y. Xiao, W. Liang, and X. Han, “Trade-off of security and performance of lightweight block ciphers in Industrial Wireless Sensor Networks,” *Eurasip J. Wirel. Commun. Netw.*, vol. 2018, no. 1, 2018, doi: 10.1186/s13638-018-1121-6.
[60] Lara-Niño, C. Andrés, M. S. Miguel, and D. P. Arturo, “An evaluation of AES and present ciphers for lightweight cryptography on smartphones,” *2016 Int. Conf. Electron. Commun. Comput. CONIELECOMP 2016*, pp. 87–93, 2016, doi: 10.1109/CONIELECOMP.2016.7438557.
[61] G. Hatzivasilis, K. Fysarakis, I. Papaefstathiou, and C. Manifavas, “A review of lightweight block ciphers,” *J. Cryptogr. Eng.*, vol. 8, no. 2, pp. 141–184, 2018, doi: 10.1007/s13389-017-0160-y.

Authors’ Profiles

Roman Alex F. Lustro earned his Doctor in Information Technology at the Technological Institute of the Philippines. He finished his Master in Information Technology at the University of La Salette. Likewise, obtained his Bachelor of Science in Information Technology at Isabela State University.

At present, Dr. Lustro is an assistant professor II at Isabela State University teaching capstone projects, system analysis and design, software engineering, and database management system. In addition, he is also holding an office as the program chair of the Bachelor of Science in Information Technology, responsible for administrative and academic-related matters.

He is also a researcher who has published researches in Scopus-indexed journals and a presenter in international conferences. His topics include cryptography, artificial intelligence, design pattern, and data mining.

How to cite this paper: Roman Alex F. Lustro, "Modified Key Derivation Function for Enhanced Security of Speck in Resource-Constrained Internet of Things", International Journal of Computer Network and Information Security(IJCNIS), Vol.13, No.4, pp.14-25, 2021. DOI: 10.5815/ijcnis.2021.04.02