A Survey on Stream Ciphers for Constrained Environments

Sameeh A. Jassim
Department of Computer Sciences
University of Technology
Baghdad, Iraq
cs.19.22@grad.uotechnology.edu.iq,
prog85sameeh@gmail.com

Alaa K. Farhan
Department of Computer Sciences
University of Technology
Baghdad, Iraq
110030@uotechnology.edu.iq

2021 1st Babylon International Conference on Information Technology and Science (BICITS) | 978-1-6654-1573-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/BICITS51482.2021.9509883

***Abstract*— Lightweight ciphers are defined as symmetric ciphers. They could be categorized into stream and block ciphers. A stream cipher is faster and less complex than block ciphers so it is suitable with the Internet of Things (IoT). The IoT is composed of many interconnected constrained devices that share and exchange data and information among each other continuously. Therefore, IoT devices must ensure basic security characteristics to protect that information. In this paper, we will make a survey on a solution that used stream cipher in cryptography. This survey investigates a detailed flow of the stream ciphers such as algorithm design pattern, key size, internal state, throughput, the vulnerability in security, and the initial vectors for comparison among various types of stream ciphers from lightweight cryptographic solutions. The goals of this survey are to discover the most effective IoT protection solution and to look at lightweight cryptographic solutions by taking into account the constraints the IoT devices have, as well as how does researched symmetric key cryptographic solution analysis work. The conclusion is the Fruit stream cipher has good resistance to known attacks, whereas the Enocoro128 and F-FCSR stream ciphers have large throughputs, as well as a WG, Grain, and MICKEY-128 stream ciphers are faster and more suitable to constrained devices (e.g., IoT) than other studied algorithms.**

Keywords*— *Lightweight cryptography, Symmetric cryptography, IoT, lightweight stream ciphers

I. INTRODUCTION

This paper abstracts a state-of-the-art comparison among various and most commonly algorithms published in the lightweight symmetric stream cipher cryptographic field. However, depending on their implementations many researchers have provided different meanings of IoT, but in simple terms, IoT is a network of linked things, each with a specific identifier, capable of gathering and sharing data with or without human intervention over the Internet[1]. Lightweight stream cipher cryptographic algorithms are continuously studied and improved to meet the development in both hardware performance and software requirements [2]. Particularly, the lightweight cryptographic algorithms are designed for devices constrained in resources (e.g. Wireless Sensor Networks (WSN), RFID systems, smart cards, etc.)[3][4]. Also, stream ciphers are ideal for systems where the plaintext length is unknown or continuous, such as military applications and network streams. Wherever the cipher stream is installed in a protected environment and fed computers that are supposed to be operated in dangerous conditions[5]. Moreover, stream ciphers are usually fast, lightweight, and low-power consuming, giving an appealing alternative for devices that are constrained in resources[6].

In terms of operation, ciphers achieve good results utilizing an Arduino-based microcontroller. Also, lightweight asymmetric algorithms complex and not time-efficient. These algorithms are often made weak by the operands scale and the continuous advance in attack models[7]. Due to its rapid operations, which are mainly XORed and permutations, symmetric cryptography is more fitting for IoT applications. Hence, A key size is identical as the data is utilized by stream ciphers. Stream ciphers are symmetric ciphers that encrypt the stream of plain text bits to produce ciphertext with the corresponding keystream[8]. Therefore, the stream cipher with maximum security and minimum computational complexity may be called a lightweight stream cipher. There are several lightweight stream ciphers currently in existence, and each has special requirements and vulnerabilities[9]. The remaining of this paper is organized as follows: section two explains the lightweight stream ciphers, while the existing stream cipher algorithms are given in section three, and finally, the most important conclusion is summarized in section four.

II. LIGHTWEIGHT STREAM CIPHERS

Lightweight ciphers are defined as symmetric ciphers and could be categorized into stream and block ciphers[10]. However, the key size used in lightweight stream ciphers is equivalent to the data size used in cryptography operation. The ciphertext is acquired simply by XOR-ed the plaintext with the keystream. This operation is done bit by bit (one-time pad. They are considered to be theoretically more lightweight, as they utilize only bit operations. Compared to block ciphers, the stream ciphers are faster and simpler in hardware. Nonlinear feedback shift registers (NLFSRs) and linear feedback shift registers (LFSRs) are used to construct the stream ciphers[11]. It is commonly utilized in wireless networking, mobile phones, etc. Their big downside is the long period of the setup before first use. Besides, communication protocols exist do not use stream ciphers. They are still in the foreground, though, because of their hardware simplicity and speed. They are also used in applications where the size of plaintext is unclear or continuous[12].

A-A5/1 stream cipher

A5/1 stream cipher is the kind of private key cryptography utilized to encrypt the transmitted signal. In modern symmetric cryptography, it plays a significant role[13]. Due to its performance and hardware suitability. It is commonly utilized in practice. The plain text is combined with keystream to encrypt the data and produce ciphertext utilizing a binary linear function (XOR). These ciphers are

utilized to produce the keystream that is a pseudo-random binary sequence. Stream ciphers are created to be very rapid and quicker than the ciphers of the block[14]. Applications that utilize stream cipher to encrypt their data have an unknown plain text size[15]. A5/1 is utilized in the Global System for Mobile Communications (GSM) standard to give privacy to the voice and data communication of the customers[16].

B- RC4

In the cryptography field, RC4 can be utilized as one of the most widespread symmetric encryption streams. It is sometimes referred to as ARCFOUR or ARC4. Ron Rivest of RSA security developed the algorithm in 1987 and then published it anonymously in 1994 for mail development [17]. The algorithm has several applications and has been used to confidentially encrypt file items via e-mail to protect many popular protocols such as WEP (Wireless Equivalent Privacy) and Secure Sockets Layer (SSL)/TLS[17]. This algorithm utilizes a variable key-size stream that is independent from plaintext, from 1 to 256 bytes[18]. RC4 has a hidden internal state that is a permutation of all the terms of $n = 2^n$ potential n bits, along with two indices in it.

C- Rabbit

In 2003, Rabbit was presented, depending on a series of coupled non-linear functions being iterated. Rabbit is distinguished by good efficiency in software[19]. The concept of Rabbit was inspired by the chaotic maps. The Rabbit algorithm could be defined briefly as follows: the input parameters are 64-bit initial value (IV) and 128-bit secret key, to produces an output block of 128 pseudo-random bits from an integration of the internal state bits for each iteration. The XORing used pseudo-random data and plaintext/ciphertext to achieve encryption/decryption operations. The internal state size is 513 bits, split into one counter carry bit, eight 32-bit counters, and eight 32-bit state variables. Eight coupled non-linear functions update the eight state variables. For the state variables, the counters guarantee a lower limit on the duration of the time[19].

D-Trivium

Trivium stream cipher is synchronous cryptography intended to establish a keystream of up to 2^{64} bits from an initial value of 80 bits (IV) and an 80-bit secret key. This method consists of two steps, as with most stream ciphers: the first step is the cipher internal state which is initialized by utilizing the IV and the key, after that the state is periodically modified and utilized to produce keystream bits[20].

E-Salsa

In 2005, the Salsa stream cipher was created. It utilized 128-bit initial vectors (IVs) and 256-bit keys[21]. To cover the tradeoff between performance and protection, three variants were proposed, accounting the various application requirements. In standard cryptographic applications, Salsa20/20 is reserved for cryptography, Whereas, the Salsa20/8 and Salsa20/12 versions provide minimal security but the operations are quicker. Its architecture depends on bit rotation, bitwise XOR, and basic additional operations modulo 232, that are executed efficiently in software[21].

F- Grain and Grain128a

The grain stream cipher is synchronous. It was created in 2004. It uses both the LFSR and a function of non-linear filtering. Thus, the LFSR guarantees less period of output balancing and a keystream. The function of filtering is known as an NFSR form and applies the cipher to nonlinearity. With the input of this NFSR to stabilize its state, the LFSR output is masked[22]. However, raising the length of the processing term also provides the ability to increase its speed; the number of bits can be increased at the cost of more hardware (we can expand the rate to 16 bits/cycle)[23]. Whereas Grain128a is a new version of Grain cipher, it utilizes 96-bit IVs, 128-bit keys, and up to 32 bits of tag size is variable. Grain-128a was created to provide higher level of security to sensitive applications. One bit is the minimum word length, and 32 bits is the maximum word length[24].

G- HC-128

There are two major versions of HC cipher, HC-128 (128 bit IVs and 128 bit key size) and HC-256 (256 bit IVs and 256 bit key size). There are two big secret tables utilized. Each table has elements with size 512 32 bit and performs on words of 32 bits. Thus, the element is upgraded utilizing an NLFSR function at each step, and a non-linear output filtering function produces a 32-bit output. Since three sequential steps could be calculated in parallel, HC is sufficient for modern superscalar microprocessors and parallel processing. At each step, the output functions and feedback could be executed at the same time [25].

H- F-FCSR

The design of the stream cipher is as follows. the architecture of LFSRs is replaced by Feedback with carrying Shift Registers (FCSRs). The key distinction lies in the computation of feedback between these two automata. While LFSRs utilize basic bitwise addition, FCSRs utilize carries of addition. Consequently, an FCSR's transformation function is non-linear, more specifically quadratic[26].

I- SNOW

The stream cipher SNOW was suggested in 2000. After that, a new version of SNOW called SNOW 2.0. was proposed in 2003, while SNOW 3G was introduced in 2010. The SNOW 3G utilizes two modules, a Finite State Machine (FSM) and a Linear Feedback Shift Register (LFSR) [27]. The LFSR consists of 16 phases, each phase carrying 32 bits, and a primitive polynomial over the finite field $GF(2^{32})$ which determines the feedback. The FSM is using three R1, R2, and R3 32-bit registers and utilizes two S1 and S2 substitution boxes. The addition operation modulo 2^{32} and exclusive OR are the mixed operations[28][29].

J- ACORN

ACORN utilizes 128-bit IVs and a 128-bit key. The length of the plaintext length and the associated data are smaller than 2^{64} bits. ACORN stream cipher is proposed for lightweight authenticated encryption. It is a bit-wise authenticated cipher (i.e. one piece of message is interpreted in a single stage) which is work efficiently both hardware and software, and it is easy to evaluate its authentication security. The bit-wise mechanism facilitates the application of light-weight hardware, therefore, the control circuit could be simplified greatly[30].

K- Sablier

Sablier stream has built-in authentication as a hardware-efficient cipher. Sablier utilizes a modern internal structure to produce the keystream from an 80-bit IVs and an 80-bit key, opposite the standard LFSR-based stream ciphers and the normal nonlinear/linear shift registers merged structure in Trivium and Grain. In Sablier only bitwise intra-word rotation, bitwise logical, and bitwise xor are utilized[31]. In

restricted hardware environments, it can be applied successfully and the speed of encryption is about 16 times quicker than Trivium in hardware[32].

L- Sosemanuk

SOSEMANUK is a cipher for a synchronous stream. The size of the IV is 128 bits, while, the range size of the key is (128 to 256) bits. The protection standard given, however, is identical, at 128 bits, to different sizes of the key. It utilizes the same concepts and procedures of the architectural principles of the SNOW 2.0 cipher and Serpent block cipher. As it has a quicker IV initialization step, SOSEMANUK is better performing than SNOW 2.0 and requires fewer static data. It utilizes and functions on 32-bit terms with an LFSR and an FSM. A 24-round serpent is utilized to accomplish the FSM and the LFSR for the setup process. The FSM's four output words are fed into Serpent's third S-Box at the keystream generation level, afterwards, it is XOR-ed with the LFSR output words[33].

M- ALE

ALE is an AES-based, lightweight, authenticated encryption algorithm called ALE (Authenticated Lightweight Encryption) that is both software and hardware are efficient[34]. It is a nonce-based online single-pass scheme that maintains data memory alignment. It has a secret internal state of 256 bits reliant on both key and nonce[35][36].

N- MICKEY

MICKEY (Mutual Erratic Clocking KEYstream generator) implements nonlinearity in addition to several innovative strategies to assurance time and pseudo-randomness. It utilizes an NFSR and a Galois LFSR with sporadic clocking. It utilized 80 bits key size, and from 0 to 80 bits, the IV will vary. We can create 240 keystream bits from each (key and IV) pair, and up to 240 various IVs of a similar length could be utilized for each key[37].

O- CHACHA

ChaCha stream cipher has a 256-bit depending on the Salsa20 cipher, ChaCha has conjectural and greater per-round diffusion improved cryptanalysis strength relative to Salsa20. The essence of the ChaCha (and Salsa20) features is a hash function that maps 64 input bytes and output keystream with irreversible and a special 64 bytes[38]. The encryption and decryption operations are achieved by XORing the input data into the keystream. The probability of output block creation at random locations and the auto-adapted constant time for processing stream blocks are two useful features of ChaCha[39].

P- Enocoro

Enocoro is a stream cipher algorithm suggested in 2007 by Watanabe et al. It comprises two algorithms named Enocoro-80 and Enocoro-128v1.1, the key lengths of which are 80 bits and 128 bits respectively[40]. Enocoro utilizes 64-bit IVs and uses an S-box byte-oriented architecture that works well in both hardware and software. For each pair IV and key, it generates one byte for each round and up to (264) bytes[40][41].

Q- A2U2

A2U2 stream cipher was created for the constraint resource for the printed RFID tag environment. The region occupied for protection in this application field must be about 500 GE, whereas the power usage is reduced to lesser tens of Ws. To allow interactions with a big number of tags in real-time, throughput should also be appropriate. A2U2 is a cipher for synchronous streams that utilize 56-bit keys[42]. A2U2 uses short-length registries backed by lightweight usable blocks and reutilizes components of hardware to reach a limited hardware area. Its execution is firmly dependent on effective concepts of hardware architecture implemented by the block cipher KATAN. More precisely, as suggested by KATAN, it utilizes a mixture of two NSFRs and an LFSR-based counter. During the initialization process, the LFSR acts as a counter and then begins to act as an LFSR. Every NFSR's feedback feature provides the other NFSR with feedback. In comparison, in the filter functions and feedback, A2U2 uses irregular shifts[43].

R- Quavium

Quavium is a flexible Trivium extension, Quavium is suggested and supported like Trivium the 80 bits IV, internal state (288-bits), and key (80-bits) sizes. It depends on Trivium-like four-round SHRs and primitive polynomials of the k-order. In coupling relation, Quavium utilizes four Trivium-like SHRs, rather than the three SHRs in the sequence attachment used in the original Trivium. For either two or three rounds, it could also work, as the relation of the pairing preserves the characteristic polynomials primitiveness[44].

S- WG-8

WG-8 stream cipher is a type of the Welch Gong family. However, WG-8 has a 20-phase LFSR with an 80-bits initialization vector and an 80-bits key size. It has two operating stages, the setup stage, and the running process. The cipher contains LFSR accompanied by transformations of the Welch-Gong with feedback polynomial, that produces sequences of bits with proven properties of randomness. It has better performance than most ciphers and fewer memory requirements. It has attacked resistor as well as it has strong randomness. This offers strong productivity and consumes the less power. But it was found that the key recovery attack was not secure[45].

T- Sprout

Sprout's architecture is adopted from Grain 128a. The sizes of both the Grain families LFSR and NLFSR were minimized to half of their values also the functions were modified slightly[46]. Except for the inclusion of a circular key element to combine key bits for each clock, the design principle of output generation and input functions is almost retained. Therefore, the cost of an area stream cipher is approximately 800 GE, while Grain requires 1162 GE for the equivalent 80-bit security standard[47][48].

U- Plantlet

Plantlet is built to fulfill the following design aims: 80-bit low-area stream cipher, shorter internal state thus retaining the degree of protection. This achieves high efficiency even though the key is permanently stored and continuously read during computation from re-writable non-volatile memory while being hardware-friendly and independent of the option of underlying non-volatile memory technology. A stronger variant of Sprout is Plantlet. In specific, it inherits from Sprout the general framework, adopts continuous key engagement, but at the same time imposes patches for vulnerabilities found, e.g. greater function of the round key and prevent the all-zero states[49].

W- Fruit

Armknetcht et al., in FSE 2015, suggested a new stream cipher architecture method. With each round of a bit generation keystream, this approach requires repetitive utilize of key bits. The probability of developing stream ciphers where the internal state size is considerably smaller than twice. The key size was shown by this proposal. They suggested a modern cipher, called Sprout, depending on this concept. But Sprout rapidly verified vulnerable in an exhaustive search[50]. The new concept utilized in Sprout, however, presented a new direction in stream cipher design, leading to the suggestion of many modern ciphers with limited internal state sizes. The fruit is an alternative recently suggested cipher in this context since the key size and state size are both 80. Till now, no attack on this cipher has occurred[51].

X- Lizard

The Lizard's design was inspired by the stream ciphers Grain family. Lizard's internal state is spread over two interconnected feedback shift registers (FSRs). But note that whereas Grain utilizes one NFSR and one LFSR, both are identical in length, Lizard instead utilizes two NFSRs of various lengths. As in Grain, besides the two FSRs, the third important building block is a nonlinear output function, which holds inputs from both shift registers and is often utilized as a portion of the state initialization algorithm[52].

Y- Espresso

Among the lightweight ciphers below 1500 GE, ESPRESSO is designed to be the quickest. It has to collect the benefit of NLFSR's Fibonacci configuration and Galois configuration. It utilizes an initialization vector with a length of 96-bit and a 128-bit key together in the configuration stage. It is composed of an NLFSR with a length of 256 bit and a non-linear 29 vector function. It has low propagation delays and could be formally analyzed. Both speed optimization and hardware size were considered for their design. It has, thus, minimized the footprint of hardware and expanded throughput. It is created specifically for applications of 5G with improved service quality, as well as it provides a few milliseconds of minimum latency[53].

Z- Modified RC4

There are two steps of the RC4 algorithm's main work: the key generation step and the encryption step. For a new key, both steps must be done. Key generation is the first step in RC4, and the most complicated. Two state variables like S1 (initial with between 0 and 255) and S2 (fill with the selected key) are used in key generation. The second step is carried out by conducting several operations on the S1 and S2, such as (swapping, modulo, and other formulas). The encryption method is carried out after generating a stream bit of key, XOR-ed bit with a bit of plaintext to create the ciphertext, and the cipher-text is XOR-ed with key-stream to decrypt the plaintext. Since RC4 has two phases: KSA and PRGA, the suggested improvements in the KSA process are based on a linear equation with a certain prime number, allowing the main generator function[17].

WG-29

Two main building blocks consist Grain-128AEAD. The first is a pre-output generator that is designed utilizing an NFSR, an LFSR, and a pre-output function, whereas the second one is an authenticator generator which is comprised of a shift register and accumulator. The architecture is very identical to Grain-128a but has been changed to endorse AEAD and to make larger authenticators[54].

A4

A4 is proposed in 2020 as a modern lightweight stream cipher, utilizing one Feedback with Carry Shift Register (FCSR) and LFSR. In various applications where safe communication among parties is a priority, A4 highly guarantees security to a large degree and is also easy to enforce. The LFSR serves as a clock to guarantee the primary degree of security. The LFSR seed value is pseudo-randomly taken from a seedbox composed of 256 values with a length of 128 bits each. It clocks the FCSR that produces the keystream for server and client-side messages to be encrypted and decrypted, respectively[2].

III. EXISTING STREAM CIPHER ALGORITHMS

The latest stream cipher algorithms have been studied by more than twenty-seven symmetric LWC algorithms proposed by numerous scholarly, proprietary, and government agencies with an emphasis on cost savings (memory, computing power, GE), energy consumption) and better performance of hardware and software (latency, throughput)[6]. Low computing energy in low-end devices makes the encryption/decryption processes more complex to apply at the level of the system in the resource-constrained environment. Thus, the metric of throughput plays a crucial role in guaranteeing the system's efficiency. The throughput of the chosen stream cipher was verified by producing large keystreams. In software, the throughput could be calculated by computing the average plaintext amount processed per CPU clock cycle at a frequency equal to 4 MHz. Whereas in hardware, it could be calculated in terms of plaintext processed bits per second (per time unit) at a frequency equal to 100 kHz[6]. In this study, throughput was demonstrated in kilobytes per second (kbps). Table I. shown a summary of the most common stream ciphers. The key size, internal state (IS), initialization vector (IV), algorithm design pattern, throughput, and vulnerability of algorithms. While Table II. illustrates the categories of each algorithm into a particular class according to the structure type utilized.is a summary of the most common stream ciphers.

TABLE I. :A SUMMARY OF THE MOST COMMON STREAM CIPHERS

Cipher	Year	Key size (bit)	Internal state (bit)	IV (bit)	Algorithm Design Pattern	Average of Throughput at 100 kHz (kbps)	Vulnerable to:
RC4[18]	1987	40-256	$N= 2^n$	-	Add, Rotate, and XOR (ARX)	227.14[55]	A related key attack.
A5/1[16]	1987	64	64	22	LFSR		Clocking control mechanism and fixed tap bits for 3 FSRs
Rabbit[19]	2003	128	513	64	Chaotic Table + simple arithmetic	236.8	Exhaustive key search
Trivium[20]	2005	80	288	80	Three shift registers (3SHR)	86[56]	Guess and Determine+algebraic + Resynchronization attacks
Salsa[21]	2005	128/256	512	128	ARX	160.46[55]	Power analysis attacks+ differential attacks
Grain[23]	2005	80	160	64	LFSR+NFSR	100[56]	Weak Key-IV
Grain128a[24]	2011	128	256	96	LFSR + NFSR	123[56]	Weak Key-IV
HC-128[25]	2008	128	512	128	Two large tables	148.98[55]	Easy to recover master key from subkey p and q
F-FCSR[26]	2005	80	196	80	FCSRs utilize addition with carries	800[57]	Resynchronization attack

SNOW 2.0[29]	2003	128/256	576	128	LFSR + FSM	304[58]	Weaknesses in the design
SNOW 3G[27]	2010	128	608	128	LFSR + FSM	106.84[55]	Related-key key recovery attacks
ACORN[30]	2014	128	293	128	6 LFSR	88.7	DPA attack
Sablier[32]	2014	80	208	80	ARX		Key recovery attack
Sosemanuk[33]	2008	128/256	384	64/128	LFSR + FSM	362.2[59]	Linear cryptanalysis
ALE[34]	2014	128	Two 128	128	SPN	121.9[60]	Compromised by cryptanalysis
MICKEY[37]	2008	80/128	200/320	0-80/0-128	NFSR + Galois LFSR	100[57]	Differential fault attack
CHACHA[39]	2008	256	512	64	ARX		Classical fault Analysis techniques
Enocoro80[41]	2008	80	176	64	PRNG	224[61]	Cryptanalysis
Enocoro128[40]	2009	128	176	64	PRNG	800[61]	Cube attacks
A2U2[43][42]	2011	56	95	64	LFSR + 2NLFSR	50[56]	Ultra-efficient chosen-plaintext attack
Quavium[44]	2012	80	288	80	4 Trivium-like SHR		No cryptanalysis results are presented
WG-8[45]	2013	80	160	80	WG + LFSR	100[61]	Key recovery attack
Sprout[46]	2015	80	89	70	Counter Reg + NLFSR + LFSR	100[51]	Key recovery is possible from partial information of the IS
Plantlet[49]	2016	80	110	90	Counter + LFSR + NLFSR	100[49]	TMD tradeoff attacks
Fruit[51]	2016	80	80	70	LFSR + NLFSR	100[51]	Resistance to Known Attacks[62]
Lizard[52]	2017	120	121	64	NLFSR		TMD tradeoff attacks
Espresso[53]	2015	128	256	96	Galois structure NLFSR		Clock gating and power gating
Modified RC4[17]	2020	8-2048	-	0 -256 + chosen key	Swapping, modulo and other formulas		No cryptanalysis results are presented
WG-29[54]	2019	128	256 (128 LFSR + 128 NFSR)	96	LFSR+NFSR		Fault attack
A4[2]	2020	128	16*16	-	LFSR + FCSR		No cryptanalysis results are presented

TABLE II. STRUCTURE OF STREAM CIPHER ALGORITHMS

Structure (Technique) Type	Algorithms
ARX	RC4, Salsa, Sablier, CHACHA
LFSR	A5/1, ACORN
Chaotic Table + simple arithmetic	Rabbit
SHR	Trivium, Quavium
LFSR+NFSR	Grain, Grain128a, MICKEY, A2U2, Fruit, WG-29
Two large tables	HC-128
FCSR	F-FCSR,
LFSR + FSM	SNOW 2.0, SNOW 3G, Sosemanuk
SPN	ALE
PRNG	Enocoro80, Enocoro128
WG + LFSR	WG-8
Counter + NLFSR + LFSR	Sprout, Plantlet,
NLFSR	Espresso, Lizard
Swapping, modulo, and other formulas	Modified RC4
LFSR + FCSR	A4

Also, Table III. illustrates the total time (clock ticks) for encrypted one block, encryption speed (cycles/byte), and encryption speed (Mbps) for various stream cipher algorithms implemented on CPU speed with 3001.6 MHz (Intel(R), Pentium(R) 4, CPU 3.00 GHz) and cache size 1024 KB[63].

TABLE III. SHOWS A COMPARISON AMONG VARIOUS TYPES OF STREAM CIPHER SPEEDS

Cipher	Profile	Key	IV	Total time for Encrypted one block	Average Encryption speed (cycles/byte)	Average Encryption speed (Mbps)
HC-128	SW	128	128	672150	287.56	83.51
HC-256	SW	256	128	648908	288.47	83.24
SNOW-2.0	SW	128	128	723218	9.08	2643.31
SNOW-2.0	SW	256	128	727605	9.55	2514.17
SOSEMANUK	SW	128	64	738135	14.63	1641.40
SOSEMANUK	SW	256	128	738285	13.64	1759.94
Rabbit	SW & HW	128	64	726330	9.97	2409.19
TRIVIUM	HW	80	64	733883	14.05	1709.57
TRIVIUM	HW	80	80	736222	14.19	1692.39
RC4	SW	256	0	721313	78.18	307.14
RC4	SW	128	0	719355	78.07	307.60
Salsa20	SW & HW	128	64	737715	16.27	1475.78
Salsa20	SW & HW	256	64	738157	16.01	1499.84
F-FCSR-8	SW	128	128	706185	76.49	313.93
F-FCSR-H	HW	128	128	639435	106.47	225.54
MICKEY	HW	80	80	4846455	1272.40	18.87
MICKEY-128	HW	128	128	5455327	1503.35	15.97
Grain	HW	80	64	16732072	4340.44	5.53
WG	HW	128	128	91223775	22427.07	1.07

As shown in Tables I. the WG, Grain, and MICKEY-128 ciphers have a large total time for encrypted one block, and also the average encryption speed (cycles/byte) is much greater than the other tested ciphers but the average encryption speed (Mbps) is minimum values than others.

IV. CONCLUSION

Recently, stream ciphers have gained more interest by many researchers those have evaluated the secure stream cipher architectures by utilizing of IoT and smart devices, for example, RFID tags, smartphones, sensors, and actuators. These allow the collecting and exchange of secret data. Therefore, in this work, we discussed more than twenty-seven stream cipher algorithms and provide a clear accurate definition and classification of stream cipher algorithms with comparison among the most common algorithms used. Comparison is based on important features like the key size, internal state, an initialization vector, and algorithm design pattern, as well as other computational capabilities. They are like the total time for encrypted one block and encryption speed in cycles/byte and Mbps for a various stream cipher. Finally, we concluded that the Fruit stream cipher has good resistance to known attacks, and the Enocoro128 and F-FCSR stream ciphers have large throughputs, as well as a WG, Grain, and MICKEY-128 stream ciphers are faster and more suitable to constrained devices (e.g., IoT) than other studied algorithms. So, they must be given more attention by the researchers to update them to meet the development in both hardware performance and software constrained requirements as to well as analyze them against the well-known attacks.

REFERENCES

[1] S. A. JASSIM and W. K. Awad, "SEARCHING OVER ENCRYPTED SHARED DATA VIA CLOUD DATA STORAGE.," J. Theor. Appl. Inf. Technol., vol. 96, no. 12, 2018.

[2] N. A. Mohandas, A. Swathi, R. Abhijith, A. Nazar, and G. Sharath, "A4: A lightweight stream cipher," 2020, doi: 10.1109/ICCES48766.2020.09138048.

[3] J. R. Naif, G. H. Abdul-Majeed, and A. K. Farhan, "Secure IOT System Based on Chaos-Modified Lightweight AES," 2019, doi: 10.1109/ICOASE.2019.8723807.

[4] N. A. Hasan, and A. K. Farhan, "Security Improve in ZigBee Protocol Based on RSA Public Algorithm in WSN," Eng. Technol. J., vol. 37, no. 3, pp. 67–73, 2019, doi: 10.30684/etj.37.3B.1.

[5] H. M. Al-Mashhadi, H. B. Abdul-Wahab, and R. F. Hassan, "Secure and time efficient hash-based message authentication algorithm for wireless sensor networks," in 2014 Global Summit on Computer & Information Technology (GSCIT), 2014, pp. 1–7.

[6] V. A. Thakor, M. A. Razzaque, and M. R. A. Khandaker, "Lightweight Cryptography Algorithms for resource-constrained IoT devices: A Review, Comparison and Research Opportunities," IEEE Access, 2021, doi: 10.1109/ACCESS.2021.3052867.

[7] S. G. Abbas, M. Husnain, U. U. Fayyaz, F. Shahzad, G. A. Shah, and K. Zafar, "IoT-Sphere: A Framework To Secure IoT Devices From Becoming Attack Target And Attack Source," in 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), 2021, pp. 1402–1409.

[8] F. Dridi, S. El Assad, W. El Hadj Youssef, M. Machhout, and R. Lozi, "The design and fpga-based implementation of a stream cipher based on a secure chaotic generator," Appl. Sci., 2021, doi: 10.3390/app11020625.

[9] H. Lee, "Home IoT resistance: Extended privacy and vulnerability perspective," Telemat. Informatics, vol. 49, p. 101377, 2020.

[10] R. Mathews and D. V Jose, "Analysis of Lightweight Cryptographic Algorithms for Internet of Things," Available SSRN 3734786, 2020.

[11] S. M. Sim, "Differential Power Analysis on (Non-) Linear Feedback Shift Registers.," IACR Cryptol. ePrint Arch., vol. 2020, p.349, 2020.

[12] W. K. Alzubaidi and S. H. Shaker, "Secure Routing Scheme for Clustered Wireless Sensor Network (WSN)."

[13] S. B. Sadkhan and Z. Hamza, "Proposed Enhancement of A5/1 stream cipher," in 2019 2nd International Conference on Engineering Technology and its Applications (IICETA), 2019, pp. 111–116.

[14] H. B. A. Wahab and M. A. Mohammed, "Improvement A5/1 encryption algorithm based on sponge techniques," in 2015 World Congress on Information Technology and Computer Applications (WCITCA), 2015, pp. 1–5.

[15] S. B. Sadkhan and Z. Hamza, "Cryptosystems used in IoT-current status and challenges," 2017, doi: 10.1109/CRCSIT.2017.7965534.

[16] A. Biryukov, A. Shamir, and D. Wagner, "Real time cryptanalysis of A5/1 on a PC," 2001, doi: 10.1007/3-540-44706-7_1.

[17] S. M. Kareem and A. M. S. Rahma, "A Modification on Key Stream Generator for RC4 Algorithm," Eng. Technol. J., 2020, doi: 10.30684/etj.v38i2b.404.

[18] S. Fluhrer, I. Mantin, and A. Shamir, "Weaknesses in the key scheduling algorithm of RC4," 2001, doi: 10.1007/3-540-45537-x_1.

[19] M. Boesgaard, M. Vesterager, T. Pedersen, J. Christiansen, and O. Scavenius, "Rabbit: A new high-performance stream cipher," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2003, doi: 10.1007/978-3-540-39887-5_23.

[20] C. De Canniere and B. Preneel, "TRIVIUM Specifications," ECRYPT Stream Cipher Proj. Rep., 2005.

[21] D. J. Bernstein, "The salsa20 family of stream ciphers," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2008, doi: 10.1007/978-3-540-68351-3_8.

[22] H. M. Al-Mashhadi, H. B. AbdulWahab, and R. F. Hassan, "Chaotic Encryption Scheme for Wireless Sensor Network's Message," in Proc World Symp Comput Netw Inf Sec, 2014, pp. 116–120.

[23] M. Hell, T. Johansson, and W. Meier, "Grain: A stream cipher for constrained environments," Int. J. Wirel. Mob. Comput., 2007, doi: 10.1504/IJWMC.2007.013798.

[24] M. Ågren, M. Hell, T. Johansson, and W. Meier, "Grain-128a: A new version of Grain-128 with optional authentication," Int. J. Wirel. Mob. Comput., 2011, doi: 10.1504/IJWMC.2011.044106.

[25] H. Wu, "The stream cipher HC-128," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2008, doi: 10.1007/978-3-540-68351-3_4.

[26] F. Arnault and T. P. Berger, "F-FCSR: Design of a new class of stream ciphers," 2005, doi: 10.1007/11502760_6.

[27] G. Orhanou, "SNOW 3G Stream Cipher Operation and Complexity Study," Contemp. Eng. Sci., 2010.

[28] A. S. Hamad and A. K. Farhan, "Image Encryption Algorithm Based on Substitution Principle and Shuffling Scheme," Eng. Technol. J., vol. 38, no. 3B, pp. 98–103, 2020.

[29] P. Ekdahl and T. Johansson, "A new version of the stream cipher SNOW,"Lect. Notes Comput. Sci.(including Subser.Lect. Notes Artif. Intell.Lect.Notes Bioinformatics),2003,doi:10.1007/3-540-36492-7_5.

[30] H. Wu, "ACORN: A Lightweight Authenticated Cipher (v1)."

[31] J. M. Hassan and F. A. Kadhim, "New S-Box Transformation Based on Chaotic System for Image Encryption," in 2020 3rd International Conference on Engineering Technology and its Applications (IICETA), 2020, pp. 214–219.

[32] X. Feng and F. Zhang, "Cryptanalysis on the authenticated cipher sablier," 2014, doi: 10.1007/978-3-319-11698-3_15.

[33] C. Berbain et al.,"Sosemanuk, a fast software-oriented stream cipher," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect.Notes Bioinformatics),2008, doi: 10.1007/978-3-540-68351-3_9.

[34] A. Bogdanov, F. Mendel, F. Regazzoni, V. Rijmen, and E. Tischhauser, "ALE: AES-based lightweight authenticated encryption," 2014, doi: 10.1007/978-3-662-43933-3_23.

[35] M. S. Fadhil, A. K. Farhan, M. N. Fadhil, and N. M. G. Al-Saidi, "A New Lightweight AES Using a Combination of Chaotic Systems," 2020, doi: 10.1109/IT-ELA50150.2020.9253099.

[36] A. M. S. Rahma and A. M. Abbas, "A modified Matrices Approach in Advanced Encryption Standard Algorithm," Eng. Technol. J., vol. 37, no. 3B, pp. 86–91, 2019.

[37] S. Babbage and M. Dodd, "The MICKEY stream ciphers," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), 2008, doi: 10.1007/978-3-540-68351-3_15.

[38] M. S. Mahdi, R. A. Azeez, and N. F. Hassan, "A proposed lightweight image encryption using ChaCha with hyperchaotic maps," Period. Eng. Nat. Sci., vol. 8, no. 4, pp. 2138–2145, 2020.

[39] D. J. Bernstein, "ChaCha, a variant of Salsa20," Work. Rec. SASC, 2008.

[40] D. Watanabe, T. Owada, K. Okamoto, Y. Ig, and T. Kaneko, "Update on Enocoro stream cipher," 2010, doi: 10.1109/ISITA.2010.5649627.

[41] D. Watanabe, K. Ideguchi, J. Kitahara, K. Muto, H. Furuichi, and T. Kaneko, "Enocoro-80: A hardware oriented stream cipher," 2008, doi: 10.1109/ARES.2008.84.

[42] M. A. Abdelraheem, J. Borghoff, E. Zenner, and M. David, "Cryptanalysis of the light-weight cipher A2U2," 2011, doi: 10.1007/978-3-642-25516-8_23.

[43] M. David, D. C. Ranasinghe, and T. Larsen, "A2U2: A stream cipher for printed electronics RFID tags," 2011, doi: 10.1109/RFID.2011.5764619.

[44] Y. Tian, G. Chen, and J. Li, "Quavium - a new stream cipher inspired by Trivium," J. Comput., 2012, doi: 10.4304/jcp.7.5.1278-1283.

[45] X. Fan, K. Mandal, and G. Gong, "WG-8: A lightweight stream cipher for resource-constrained smart devices," 2013, doi: 10.1007/978-3-642-37949-9_54.

[46] F. Armknecht and V. Mikhalev, "On lightweight stream ciphers with shorter internal states," 2015, doi: 10.1007/978-3-662-48116-5_22.

[47] D. D. Salman and R. A. Azeez, "Key Generation from Multibiometric System Using Meerkat Algorithm," Eng. Technol. J., vol. 38, no. 3B, pp. 115–127, 2020.

[48] M. F. Esgin and O. Kara, "Practical cryptanalysis of full sprout with TMD tradeoff attacks," 2016, doi: 10.1007/978-3-319-31301-6_4.

[49] V. Mikhalev, F. Armknecht, and C. Müller, "On Ciphers that Continuously Access the Non-Volatile Key," IACR Trans. Symmetric Cryptol., 2017, doi: 10.46586/tosc.v2016.i2.52-79.

[50] Y. Todo, W. Meier, and K. Aoki, "On the Data Limitation of Small-State Stream Ciphers: Correlation Attacks on Fruit-80 and Plantlet," 2020, doi: 10.1007/978-3-030-38471-5_15.

[51] V. Aminghafari and H. Hu, "Fruit: ultra-lightweight stream cipher with shorter internal state.," IACR Cryptol. ePrint Arch., 2016.

[52] M. Hamann, M. Krause, and W. Meier, "LIZARD – A Lightweight Stream Cipher for Power-constrained Devices," IACR Trans. Symmetric Cryptol., 2017, doi: 10.46586/tosc.v2017.i1.45-79.

[53] E. Dubrova and M. Hell, "Espresso: A stream cipher for 5G wireless communication systems," Cryptogr. Commun., 2017, doi: 10.1007/s12095-015-0173-2.

[54] M. Hell, T. Johansson, W. Meier, J. Sönnerup, and H. Yoshida, "Grain-128AEAD-A lightweight AEAD stream cipher," NIST Light. Cryptogr. Round, vol. 1, 2019.

[55] P. Alexandrov Nikolov, "Analysis and Design of a Stream Cipher," Máster Universitario en Ciberseguridad, 2019.

[56] M. David, Lightweight cryptography for passive RFID tags. 2012.

[57] T. Good and M. Benaissa, "ASIC hardware performance," in New stream cipher designs, Springer, 2008, pp. 267–293.

[58] C. Lauradoux, "Throughput/code size tradeoff for stream ciphers," State Art Stream Ciphers-SASC, 2007.

[59] G. Meiser, T. Eisenbarth, K. Lemke-Rust, and C. Paar, "Efficient implementation of eSTREAM ciphers on 8-bit AVR microcontrollers," 2008, doi: 10.1109/SIES.2008.4577681.

[60] M. Agrawal, J. Zhou, and D. Chang, "A survey on lightweight authenticated encryption and challenges for securing industrial IoT," in Advanced Sciences and Technologies for Security Applications, 2019.

[61] C. Manifavas, G. Hatzivasilis, K. Fysarakis, and Y. Papaefstathiou, "A survey of lightweight stream ciphers for embedded systems," Security and Communication Networks. 2016, doi: 10.1002/sec.1399.

[62] V. A. Ghafari and H. Hu, "Fruit-80: A secure ultra-lightweight stream cipher for constrained environments," Entropy, 2018, doi: 10.3390/e20030180.

[63] "eSTREAM Optimized Code HOWTO." https://www.ecrypt.eu.org/stream/perf/ (accessed Feb. 25, 2021).