Arabian Journal for Science and Engineering (2021) 46:4015–4037
https://doi.org/10.1007/s13369-021-05358-4

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

Performance Evaluation of Lightweight Encryption Algorithms for IoT-Based Applications

Pejman Panahi[1] · Cüneyt Bayılmış[1] · Unal Çavuşoğlu[2] · Sezgin Kaçar[3]

Received: 7 May 2020 / Accepted: 13 January 2021 / Published online: 6 February 2021

Abstract
Today, all smartphones, notebooks, or other communication devices could connect to the cloud, so the data are accessible everywhere. When these devices are interconnected through the internet, they make an Internet of Things (IoT) network that exchanges data among network nodes and other services. IoT has a broad application area from smart applications to various industrial usages. However, the high volume of data transferred in the IoT network makes it crucial to implement mechanisms to transfer the data safe and secure. Enciphering is one of the best techniques to offer end-to-end security. Considering an IoT network, nodes have restricted resources, and applying classical cryptography methods are costly and not efficient, so lightweight block ciphers are one of the sophisticated solutions to overcome security drawbacks in this scope. In this paper, ten lightweight algorithms involve AES, PRESENT, LBlock, Skipjack, SIMON, XTEA, PRINCE, Piccolo, HIGHT, RECTANGLE tested to evaluate their performance for key factors such as memory usage (RAM and ROM), energy consumption, throughput, and execution time for both encryption and decryption modes over cloud transmission. We have done simulations using Raspberry Pi 3 and Arduino Mega 2560 as the leading devices in the IoT scope. As a result, this paper will help IoT developers to choose the right platform and enciphering algorithm to set up a secure network due to multiple factors like energy and memory usage, especially for software platforms.

Keywords Lightweight block ciphers · IoT · Cloud · Security · Raspberry pi 3 · Arduino mega 2560

1 Introduction

Today, we can see many applications of the Internet of Things (IoT) in the various aspects of human life. Smart homes, wearable devices, industrial smart monitoring systems are some samples of IoT. The first time, Kevin Ashton in 1999 within the context of an RFID project introduced an Internet of Things term. Back to 2009, when the Internet of Things (IoT) term first revealed as a group of interrelated devices capable of interconnecting wired or wireless, up to now we can see a growing progress for technologies, infrastructure, and using of IoT. Two prime factors are helping IoT overall growth: Fast internet connections and using the cloud. In terms of fast internet connections, the trend toward 5G networks (Or faster networks) will show its effect. Cloud-based solutions provide data mobility and collaboration, which is vital to create and maintain a scalable IoT network. IoT nodes have a certain physical size, processing power, and memory capacity because of limited battery considerations, so storing the data over real-time clouds services can show its outcome on nodes durability and cut downtime.

The crucial advantage of using the cloud is to reach the data from any point connected to the internet. Looking back to the high density of traffic transferred among IoT nodes and data centers, there is another endeavor called providing security. It is clear that the insecure network can lead to vulnerabilities and unwanted damage. One of the potential ways of protecting the data from unwanted access is to encrypt them with strong algorithms. Since devices interconnected on an IoT network have limited resources, especially energy as mentioned before, typical encryption algorithms cannot be scale for them. Classic encryption algorithms designed to use on desktops, phones,

✉ Pejman Panahi
pejman.panahi@ogr.sakarya.edu.tr

[1] Department of Computer Engineering, Sakarya University, Serdivan, Sakarya, Turkey

[2] Department of Software Engineering, Sakarya University, Serdivan, Sakarya, Turkey

[3] Department of Electrical and Electronics Engineering, Sakarya University of Applied Science, Serdivan, Sakarya, Turkey

tablets, and servers while lightweight block ciphers are a good out for providing end-to-end security over embedded systems and sensor networks. Embedded systems are using 4-, 8-, 16-, and 32-bit micro-controllers that deal with real-time applications. For sensor networks, different technologies associate power with timing requirements, number of gates, and applying cryptographic functions. All the mentioned features here lead to the use of lightweight cryptography instead of classical methods. The intrinsic motivation of using lightweight block cipher is because of the pervasive upcoming IT landscape and the factor that they consume less computing power [1, 2].

In this paper, we chose ten lightweight block ciphers and tested their performance for their power consumption, memory usage (RAM and ROM), throughput comparison, average execution time for encryption and decryption operations in different payloads. Software lightweight implementations help to keep processor needs and memory usage as low as possible. We picked the Raspberry Pi 3 and Arduino Mega 2560 as our IoT testbed and we have done data transform over the Dropbox cloud. Searching the literature, we can see many lightweight block ciphers in which categorized in disparate groups. Choosing a suitable algorithm regarding low resource consumption (energy, RAM and ROM) and provide an acceptable level of security is our motivation to compare chosen block ciphers in this paper. The main contributions of this paper are:

- Providing a fair comparison by software implementing of ten block ciphers over the cloud using an IoT platforms (Raspberry Pi 3 and Arduino Mega 2560) for metrics cited in the literature like power consumption, memory usages, throughput, execution time.
- Investigate and analyze the reported data from software implementations and ordering the mentioned algorithms in specific order regarding each evaluation metric.
- Also, we discuss the results for selecting the right algorithm with the best results for all the multiple metrics.
- This research will help the researchers to choose the right lightweight block cipher algorithm regarding the implemented platforms.

The rest of this paper is arranged as follows. In Sect. 2, we briefly explain lightweight block ciphers. Section 3 discusses previous related works. Section 4 represents the performance comparison of selected Lightweight Block Ciphers and our evaluation results over Raspberry Pi 3 and Arduino Mega 2560. Section 5 concludes this work.

2 Lightweight Block Ciphers

A lightweight cipher is a less demanding cryptographic method, concentrating on optimizing memory usage, power consumption and providing enough level of security. We could implement it in both hardware and software platforms. The former focuses on reducing the number of running cycles and memory footprints, and the later targeted at reducing energy consumption and memory usage. In terms of power and speed, the hardware solution provides better results. Searching the literature, we can see three groups of lightweight block cipher: Substitution-Permutation Network (SPN), Feistel Networks, and other designs. In terms of comparing Feistel against SPN structure, the round function (F) of SPN has to be invertible, but for Feistel, function F need not to be invertible. SPN provides relatively higher security in contrast to the Feistel network. On the other hand, SPN consumes more resources. In a Feistel structure, the encryption and decryption are similar and even identical in some cases. Combining natural features of mentioned ciphers led to new categories of block ciphers called hybrid. As mentioned before memory footprint is one of the supreme challenges in designing a lightweight block cipher, thus, many of the block ciphers have no S-box or tiny ones (4-byte). S-box plays a key role to resist against attacks. In this part, we have presented a brief review of the lightweight block ciphers used in this paper and the other algorithms are compared inside the Table 1.

1. AES—is an SPN cipher available in different encryption packages. The S-box of this cipher is a four-by-four matrix. After key expansion, depending on the preferred key length, it will apply the round function over plaintext and comprises four operations; SubBytes (A four-byte word is the SubByte return value), ShiftRows (State rows are cyclically shifted over different offsets.), MixColumns (State columns are expressed as polynomials over GF(2^8) and multiply by modulo $x^4 + 1$ with a constant polynomial c(x)), and AddRoundkey (Using a simple bitwise EXOR, a Round Key is applied to the State) [3].
2. RECTANGLE—is suitable for hardware and software environments and has SPN framework. There are 25 rounds in this cipher that include these steps: AddRoundKey (A simple bitwise XOR (Of round subkey) applied to the intermediate state), SubColumn (Parallel application of S-boxes to the four bits in the same column), ShiftRow (A left rotation applies to each row over different offsets). A sequence of twelve basic logical operations could build S-box for this cipher. P-layer composes three rotations [4].
3. PRESENT—the infrastructure of this block cipher is the inspiration resource for many lightweight block ciphers. It is a bit-oriented SPN structure. After generating round keys, the algorithm implements the following steps for each of 31 rounds: addRoundKey

Table 1 Comparison of different lightweight block ciphers

Name	Block size (bit)	Key size (bit)	Number of rounds	Structure	Implementation environment
AES	128	128, 192, 256	10, 12, 14	SPN	Hardware and software
KLEIN	64	64, 80, 96	12, 16, 20	SPN	Hardware and software
LED	64	64, 128	32, 48	SPN	Hardware and software
SKINNY	64, 128	64, 128, 192 128, 256, 384	32, 36, 40 40, 48, 56	SPN	Hardware and software
Mysterion	128, 256	128, 256	12, 16	SPN	Hardware and software
RECTANGLE	64	80, 128	25	SPN	Hardware and software
NOEKEON	128	128	16	SPN	Hardware and software
PICO	64	128	32	SPN	Hardware and software
GIFT	64/128	128	28,40	SPN	Hardware and software
QARMA	64/128	128/256	16/24	SPN	Hardware and software
Loong	64	64, 80, 128	16, 20, 32	SPN	Hardware and software
NVLC	64	80, 128	20	SPN	Hardware and software
BORON	64	80, 128	25	SPN	Hardware and software
Midori	64, 128	128	16, 20	SPN	Hardware
Zorro	128	128	24	SPN	Hardware
Fantomas/Robin	128	128	12/16	SPN	Hardware
PRIDE	64	128	20	SPN	Hardware
MANTIS	64	128	14	SPN	Hardware
mCrypton	64	64, 96, 128	12	SPN	Hardware
PRESENT	64	80, 128	31	SPN	Hardware
PRINCE	64	128	12	SPN	Hardware
OLBCA	64	80	22	SPN	Hardware
KHAZAD	64	128	8	SPN	Hardware
PRINTCIPHER	80/160	48/96	48/96	SPN	Hardware
SPARX and LAX	64 128	128 128, 256	24 32/40	SPN (ARX)	Software
Chaskey Cipher	128	128	8	ARX	Software
SPECK	32 48 64 96 128	64 72/96 96/128 96/144 128/192/256	22 22/23 26/27 28/29 32/33/34	ARX	Software
CHAM	64, 128	128, 256	16, 32	ARX	Hardware and software
HIGHT	64	128	32	Feistel (ARX)	Hardware and software

Table 1 (continued)

Name	Block size (bit)	Key size (bit)	Number of rounds	Structure	Implementation environment
RC2	64	8–1024	2, 16	Feistel	Rc2 (software),
RC5	32, 64, 128	0…2040	12	ARX	RC5 RC6 (hardware or software)
RC6	128	128, 192, 256	20	Feistel (II)	
Cast 5	64	40–128	12–16	Feistel	Software
Cast 6	128, 160, 192, 224, 256	128	48	GFN	
BRIGHT	64 128	80, 96, 128 128, 192, 256	32, 33, 34 35. 36.37	GFN	Software
Khudra	64	80	18	GFN	Hardware
QTL	64	64, 128	25, 31	GFN	Hardware
Piccolo	64	80, 128	25, 31	GFN	Hardware
CLEFIA	128	128, 192, 256	18, 22, 26	GFN	Hardware and software
LEA	128	128/192/256	24/28/32	GFN	Hardware and software
TWINE	64	80, 128	36	GFN	Hardware and software
Camellia	128	128, 192, 256	24	Feistel	Hardware and software
SIMECK	32, 48, 64	64, 96, 128	32, 36, 44	Feistel	Hardware and software
LiCi	64	128	31	Feistel	Hardware and software
KASUMI/MISTY1	64	128	8	Feistel	Hardware and software
LBlock	64	80	32	Feistel	Hardware and software
SEA	96	96	93	Feistel	Hardware and software
ANU	64	80, 128	25	Feistel	Hardware + software
BlowFish	64, 128	Up to 448	16	Feistel	Software
Twofish	128	128, 192, 256	16		
Seed	128	128	16	Feistel	Software
TEA	64	128	64	Feistel	Software
XTEA					
LILIPUT	64	80	30	Feistel	Software
NASE	Variable	Variable	Variable	Feistel	Software
ITUBEE	80	80	20	Feistel	Software
RoadRunneR	64	80, 128	10, 12	Feistel	Software
DES	64	56	16	Feistel	Hardware
G-DES	Variable	Variable	Variable(even)		
3DES	64	168, 112, 56	48		
DESL	64	56	16		
DESX	64	184	16		
DESLX	64	184	16		
KAMAR	128	128, 192, 256	16, 20, 32	Feistel	Hardware
Skipjack	64	80	32	Feistel	Hardware
HISEC	64	80	15	Feistel	Hardware

Table 1 (continued)

Name	Block size (bit)	Key size (bit)	Number of rounds	Structure	Implementation environment
GOST	64	256	32	Feistel	Hardware
DLBCA	32	80	15	Feistel	Hardware
SIMON	32	64	32	Feistel	Hardware
	48	72/96	36		
	64	96/128	42/44		
	96	96/144	52/54		
	128	128/192/256	68/69/72		
SFN	64	96	32	Feistel + SPN	Hardware and software
SIT	64	64	5	Feistel + SPN	Hardware and software
Iceberg	64	128	16	Involutional architecture	Hardware
PUFFIN	64	128	32	Involutional SPN	Hardware
PUFFIN2		80	34		
FeW	64	80/128	32	Balanced GFN + SPN	Software
μ^2	64	80	15	GFN + SPN	Hardware + software
LRBC	16	16	24	Feistel + SPN	Hardware
FLY	64	128	20	Bitsliced cipher	Hardware + software
KATAN/KTANTAN	32, 48, 64	80	254	Bivium	Hardware
TREYFER	64	64	32	Simple	Hardware
BKSQ	96, 144, 192	96	10, 14, 18	SQUARE cipher	Hardware

(Involves of the operation for $0 \leq j \leq 63$, $b_j \rightarrow b_j \oplus k_j^i$), sBoxLayer (Uses a single 4×4 S-box), pLayer (The ith bit of STATE is moved to position P(i) due to specific mapping). It is recommended to on hardware platforms [5].

4. Skipjack—has an 80-bit key size, and it suffers against common attacks (Because of key size). The block size of Skipjack is 64 bits (divided into four 16 bits word) and the round number is 32 and ranked as an unbalanced Feistel network. This is a simple cipher that has an 8×8 S-box and there are two alternating stepping rounds called Rule A and Rule B (inversion of Rule A adding minor positioning) inside the structure [6].
5. PRINCE—is based on FX construction and no actual key scheduling mechanism of this algorithm could describe as; It achieves two of its 64-bit keys within the 128-bit master key that acts as a whitening key and the third one during encryption is XORed in the internal state. Each round combines these steps: key addition (The 64 bits state is XORed with 64 bits subkey), one Sbox-layer (It uses a single four bitsSbox), a linear layer (In this layer a 64*64 matrix is multiplied by a 64-bit state), and round constant addition (The state is XORed with a 64-bit round constant) [7].
6. SIMON—is a balanced Feistel structure and optimized to use on hardware-oriented systems. Bitwise XOR, bitwise AND, and left circular shift to make the body of round function. The key schedule could be balance or not. The authors optimized SPECK to use on software platforms and is a member of ARX based Feistel networks. Key sizes are flexible for scenarios and start from 64 bit up to 256 bits. Unique values starting from 32 to 128-bit are selectable for a block. Round numbers could be 22–34. There are two rotations, adding, and XORing operations for the round function [8].
7. HIGHT—ARX based generalized Feistel is a category that this cipher belongs. HIGHT is based on simple operations: XOR, addition mod 28, bitwise rotation (left). There are two algorithms for a key schedule: WhiteningKeyGeneration (Uses eight whitening key bytes for initial and final transformations) and SubKeyGeneration (128 subkeys are used for one computation). It is common to run for hardware and software platforms [9].
8. Piccolo—is a hardware-oriented block cipher and its key sizes are available in Piccolo-80 and Piccolo-128. For the diffusion layer, it uses special half-word permutation and whitening technique. There is a 4×4 S-box and can be implemented using the following operations: four NOR, three XOR, and one XNOR gates [10].
9. LBlock—is capable of implementing on software and hardware environments. There is a 4×4 S-Box in LBlock. The key schedule mechanism applies two additional S-boxes. Encryption operation involves a 32-round iterative structure and the decrypting is the inverse operation of encryption [11].
10. TEA and its successor XTEA—are following the Feistel formation. Both Ciphers have 128 bits of key sizes and 64-bit of block sizes. TEA represents an uncomplicated key schedule mechanism. Comparing to XTEA, some improvements have been done include rearranging shifts, XORs and additions operations. Also, a more complex key schedule mechanism used for it. Both have key sizes of 128 bits and block sizes of 128 bits [12].

Table 1 shows a comparison using different metrics among available lightweight block ciphers [13–29]. RELATED WORKS.

Generally, searching the literature the performance evaluation studies are categorized into three groups include software, hardware, and software/hardware evaluation papers. In this part, we look at related works in the field of software evaluations. A lightweight encryption algorithm called NUCLEAR introduced to be used in 6LoWPAN networks. To evaluate the software performance of their algorithm, they selected the ARM 7 LPC2129 platform and compared the results in terms of memory usage, power consumption, throughput, and execution time among nine lightweight block ciphers [30]. Jaber H [31] evaluated SIMON, SPECK, TWINE, KLEIN, Piccolo block ciphers in terms of power consumption and memory usage (Flash and RAM) over Atmega128 microprocessor. A performance comparison for IoT devices among six lightweight block ciphers were made by [32]. They chose Raspberry Pi 3 and Beagle Bone black as testbeds. They evaluated the mentioned algorithms due to execution time in the Cipher Block Chaining (CBC) and Electronic codebook (ECB) modes. CBC is a famous block cipher mode operation in which before encrypting operation, each plaintext is XORed with the previous ciphertext. In ECB, the message is divided into blocks, and each block is individually encrypted. Software evaluation of five block ciphers over the Strong SA-1100 processor done by Johann G [33]. XTEA, Rijndael, Serpent, RC6, and Twofish algorithms performance assessed due to power consumption, throughput, code size, and memory footprint. Miroslav B [34] compared XTEA against AES regarding several payloads over Atmel Atmega128RFA1 microprocessor considering energy consumption, time, and throughput metrics. Voice recognition applied to compare AES, PRESENT, HIGHT, and KLEIN block ciphers performance in terms of memory consumption, speed, floating-point numbers. All the codes were written in MATLAB, executed, and tested in both MATLAB and Arduino UNO [35]. Lejla and colleagues [36] evaluated AES, CLEFIA, KLEIN, LED, mCrypton, PRESENT, and KATAN block ciphers regarding encryption time, power (static and dynamic), and energy consumption for the encryption mode. They used Cadence Encounter RTL Compiler and ModelSIM simulator as the main tools to implement their work. Software performance comparison between SIMON and SPECK block ciphers over an AVR 8-bit Atmel Atmega128 microcontrollers based on Flash, RAM, cost (speed, energy) metrics for an encryption operation measured in [37]. Raspberry Pi 2 and Arduino UNO used as testbeds to evaluate PRESENT, SIMON, SPECK, RECTANGLE, PRINCE, Pride, and LBlock algorithms in terms of RAM and ROM usages, Execution time, Clock/cycle for encryption/decryption and key schedule modes [38]. The authors [39] made a performance comparison among CLEFIA, PICCOLO, and TWINE block ciphers regarding Throughput, RAM and ROM, energy consumption, and execution time metrics for the encryption mode. The experiments were done on a STM32F401RE microcontroller and 512, 1024, 2048, 3072 bytes plain texts entered as input to test the algorithms. Memory usage (RAM and ROM) and energy consumption of TEA, HIGHT, KATAN, and KLEIN block ciphers measured in [40] using an Atmel Attiny45 microcontroller in encryption mode. In [41], the authors review the approaches of smart homes, available challenges with privacy, security, and possible solutions for the mentioned concerns. Zahra and NZ in [42] makes a comprehensive review about user's privacy protection in location-aware mobile clouding services. They address and analyze available approaches in providing security for the positioning services in the future. The in-vehicle communication, software implementing of cryptographic algorithms for AUTomotive Open System Architecture (AUTOSAR) has been done for several devices applying AES, KATAN, PRESENT, SPECK, MD5, SHA1, SHA2, SHA3, and Blake2 algorithms in [43]. Khaled [44] represented an analytical model to discover the minimum number of cloud resources to meet Service Level Objectives (SLO) response time. In [45], the authors used numerical samples to estimate the number of necessary virtual machines over cloud data centers to provide quality of service parameters. Table 2 represents a comprehensive comparison of previous related works and this study. Performance Evaluation of Lightweight Encryption Algorithms.

We picked ten algorithms due to block length, key sizes, popularity, their inner structure, and implementation possibility over software environments. Although some of the algorithms like PRESENT were hardware-oriented on first versions, they have been improved to be installed on software platforms later. Except for AES with 128 bits block length, all other algorithms have 64-bit length and in terms of key size, except LBlock and Skipjack in which have an 80-bit key size, the others have 128-bit key sizes. Table 3 compares the chosen ten lightweight block ciphers.

To evaluate the mentioned ten block ciphers over an IoT platform, we selected Raspberry Pi 3 and Arduino Mega 2560 as our testbed platforms and Dropbox cloud service as an encrypted file exchange platform. We have repeated each experiment for five iterations to get more precise results. The plaintext is encrypted into a file at sender side and the file transfers to the cloud and for decryption at the target side, the device reads the file from cloud and starts decrypting it. Since network parameters like delay and network speed may change during simulations, we ignored upload and download times to the cloud and all the measured values in this paper are based on local encryption and decryption times at sender or receiver. Figure 1 show a general view of the data exchange system and Fig. 2 represents the hardware layout of the evaluated system, respectively. As can be seen in Fig. 2, the Raspberry Pi 3 empowered by a 10,400 mA Power Bank and Arduino Mega is connected to notebook USB port. To control and execute commands, we connected the Raspberry Pi and Arduino to a MacBook through an ethernet and USB ports. For measuring power consumption in different scenarios, we used USB power meter. Pi 3 runs at 1.2 GHz and uses 1 Gb Ram and equipped with a 64-bit ARM Cortex A53 processor. It can be powered by a 5 V Micro USB or Power Bank. On the other hand, Arduino Mega 2560 has a 16 MHz ATmega2560 8-bit microcontroller, 256 KB Flash memory, 8 KB SRAM, and 4 KB EEPROM. The recommended input voltage for Arduino is 7–12 V.

Simulations have been implemented in C. For calculating the amount of power consumed for each algorithm, we used the following equations:

$$\text{Current Stream (A)} * \text{time (Seconds)} = \text{Charge (C)} \tag{1}$$

$$\text{Charge (C)} * \text{Voltage (V)} = \text{Energy (J)} \tag{2}$$

Figure 3 and 4 represent energy consumption behavior for an encryption operation for both devices. Raspberry Pi 3 and Arduino Mega 2560 consume remarkably the maximum energy for the PRESENT algorithm comparing to other block ciphers. That is because this block cipher was originally designed to be a hardware friend. Table 4 shows the consumed energy comparison for two devices. The lightweight block ciphers in the related columns are arranged from the lowest to the highest. XTEA, Skipjack, and RECTANGLE have the minimum amount of energy consumption for the Raspberry Pi 3. XTEA and Skipjack's simple structure is an important reason to explain their energy consumption. RECTANGLE has a bit-slice style. These features help in consuming less power in software

Table 2 Previous works comparison

Paper	Platform	Evaluated metrics	Algorithms	Comparison method	Mode
Gaurav B [25]	ARM 7 LPC2129	Flash, RAM, execution time, throughput, number of cycles	PRESENT, LED, BORON, KLEIN, HUMMINGBIRD2, SIMON, SPECK, PICCOLO, TWINE, CLEFIA	Block lengths of algorithms	Encryption
Rutuja S [30]	ARM 7 LPC2129	Memory usage, power consumption, throughput, and execution time	PRESENT, TWINE, PICCOLO, AES, CLEFIA, NUCLEAR	Block lengths of algorithms	Encryption
Jaber H [31]	Atmega128 (AVR simulator)	Power consumption and memory usage (Flash and RAM)	SIMON, SPECK, TWINE, KLEIN, Piccolo	Block lengths of algorithms	Encryption
Kedar D [32]	Raspberry Pi 3, Beagle Bone Black	Execution time in CBC and EBC modes	AES, DES, TDES, Blowfish, Twofish, RC2	Files in MB (1, 2, 4, 8, 16, 32, 64, 128 MB)	Encryption
Michael A [46]	8-bit microcontroller (4 and 16 MHz)	RAM, Cycles, throughput, Energy	KATAN, TEA, PRESENT, SEA, AES	Block lengths of algorithms	Encryption
		Flash, SRAM, throughput	SIMON, SPECK, PRESENT, SEA, AES		
Johann G [33]	StrongARM SA-1100 (200 MHz)	Power consumption, throughput, code size, memory (RAM and ROM) footprint	RC6, Rijndael, Serpent, Twofish, XTEA	Block lengths of algorithms	Encryption and Decryption
Miroslav B [34]	Atmel ATmega128RFA1	Energy consumption, time, and Throughput	XTEA, AES	Payloads (1, 15, 16, 31, 32, 47, 48, 63, 64, 79, 80, 95, 96, 104 bytes)	Encryption and decryption
Mickael C [47]	MSP430	Cycle count, cycles/byte, RAM, ROM, combined metric	AES, CLEFIA, DESXL, HIGHT, IDEA, Noekeon, KATAN, KLEIN, KTANTAN, LBlock, LED, MCRYPTON, MIBS, PRESENT, Piccolo, SEA, Skipjack, TEA, TWINE, XTEA	Block lengths for each algorithm	Encryption and decryption
C.A. Lara N [48]	Qualcomm MSM8926 and Snapdragon 400 (D2303, D2306)	Power consumption, running time, throughput, memory usages	AES, PRESENT	Block lengths for each algorithm	Encryption and decryption
S. Kotel [49]	8-bit AVR microcontroller and 16-bit MSP microcontroller	Code size, RAM, execution time, efficiency	SIMON, SPECK, AES, PRESENT	Block lengths for each algorithm	Encryption and decryption
William D [50]	Xilinx Kintex-7 FPGA	Throughput, TP/A, cycles/block, cycles/bytes	AES, SIMON, SPECK, PRESENT, LED, TWINE	Each algorithm	Software versus hardware
Margi [19]	Arduino Uno, MATLB	Memory usage, speed, floating point numbers comparison	PRESENT, AES, HIGHT, Klein	Each algorithm	Software
Deepti S [51]	Intel (R) Core (TM) i5-2430 M CPU @ 2.40 GHz	Memory usage, cycles/byte, speed, execution time	BRIGHT, RoadRunneR, SPECK, HIGHT, SPARX	Block ciphers	Encryption and decryption
M. Razvi [52]	Sim-Panalyzer	Energy consumption, execution time, speed	AES, RC5	Payloads (16, 64, 256, 1024, 8192 bytes)	Encryption and decryption

Table 2 (continued)

Paper	Platform	Evaluated metrics	Algorithms	Comparison method	Mode
Lejla [36]	Cadence Encounter RTL Compiler and ModelSIM simulator	Encryption time, power (static and dynamic), energy	AES, CLEFIA, KLEIN, LED, mCrypton, PRESENT, KATAN	Block lengths	Encryption
Ray B [37]	Atmel ATmega128	Flash, RAM, speed, energy	SIMON, SPECK	Block lengths	Encryption
W.K.LEE [53]	ODROID-XU3	Execution time, energy efficiency, power consumption	AES, CLEFIA, SIMON, SPECK, PRESENT	64 MB plain text	Encryption
George H [17]	8-bit, 16-bit, 32-bit microcontrollers	RAM, ROM, latency, energy, throughput, software efficiency	AES, Camellia, IDEA, NOEKEON, KASUMI, GOST, HIGHT, PRESENT, CLEFIA, HB, Piccolo, TEA, XTEA, SEA, mCrypton, MIBS, TWINE, LED, Klein, KATAN, KTANTAN, ITUbee, SIMON, SPECK, PRINTcipher, LBlock, PRINCE, PRIDE, Zorro, Robin, Fantomas, LEA, DES	Key and block size of block ciphers	Software comparison versus hardware
Lukas [54]	Samsung Galaxy S i9000	Power consumption, execution time	Power consumption (AES, DES, TDES), Execution time (AES, Blosfish, Camellia, CAST 5/6, DES, DESede, GOST28147, IDEA, Noekeon, RC 2/4/5, Rijindal, SEED, Serpent, Skipjack, TEA, XTEA, Twofish)	64, 128, and 512-bit messages	Encryption and decryption
Tasnime O [38]	Raspberry Pi 2 and Arduino UNO	RAM, ROM, Execution time, Clock/cycle	PRESENT, SIMON, SPECK, RECTANGLE, PRINCE, Pride, LBlock	Block lengths of algorithms	Encryption and decryption, Keyschedule
Levent E [39]	STM32F401RE Microcontroller	Throughput, RAM and ROM, energy consumption, execution time	CLEFIA, PICCOLO and TWINE	512, 1024, 2048, 3072 bytes payloads	Encryption
Mojtaba A [40]	Atmel ATtiny45 microcontroller	RAM, ROM, energy consumption	TEA, HIGHT, KATAN, KLEIN	Block lengths of algorithms	Encryption
Murat Ç [55]	Atmel Atmega128 microcontroller	RAM, ROM, execution time, throughput	AES, SERPENT, Camellia, Cast5, MARS	Block lengths of algorithms	Encryption and decryption
Pal-Stefan M [43]	S08, S12, S12Z, RL78 D1A, TC1782, TC1797, MSP430, MPC5606B, iMX6, RH850 G3K, RH850 G3M	ROM, execution time	MD5, SHA1, SHA2, SHA3, Blake2, AES, KATAN, PRESENT, SPECK	8, 64, 576, 1536, 4096, long message	Encryption

Table 2 (continued)

Paper	Platform	Evaluated metrics	Algorithms	Comparison method	Mode
This study	Raspberry Pi 3 and Arduino Mega 2560 (Atmel ATmega 2560)	RAM, ROM, execution time, throughput, energy consumption	AES, PRESENT, LBlock, Skipjack, SIMON, XTEA, PRINCE, Piccolo, HIGHT, RECTANGLE	16, 64, 128, 256, 512, 1024, 2048 bytes payloads	Encryption and decryption

Table 3 Comparing selected lightweight block ciphers

Name	Block size (bit)	Key size (bit)	Structure
LBlock	64	80	Feistel
Skipjack	64	80	Feistel
XTEA	64	128	Feistel
HIGHT	64	128	Feistel
Piccolo	64	128	Feistel
SIMON	64	128	Feistel
RECTANGLE	64	128	SPN
PRESENT	64	128	SPN
PRINCE	64	128	SPN
AES	128	128	SPN

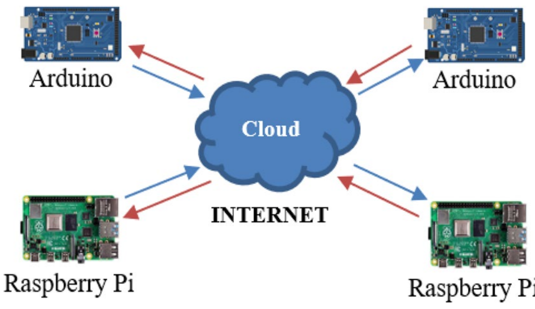

Fig. 1 Data exchange schema for test systems

applications. For the Arduino Mega 2560, Piccolo and PRESENT have the highest amount of measured power consumption. The mentioned algorithms are designed to be used in hardware platforms and this explains why there are consuming the maximum energy in the software simulations. Skipjack, RECTANGLE, and HIGHT have the lowest amount of consumed power for Arduino Mega 2560. Despite the HIGHT has been designed for hardware platforms, it represents a good performance in terms of encryption power consumption. Arduino Mega 2560 has lower processing and memory features comparing to Raspberry Pi 3, and this affects the algorithm's order in Table 4. For Arduino Mega 2560 chart, since the PRESENT algorithm values are significantly higher than other block ciphers, we limited the chart scale to 250. In this case, the PRESENT algorithm energy consumption for 512, 1024, and 2048 seems equal. As a comparison, the real values of the PRESENT algorithm for mentioned payloads, for example, are nine times bigger than Piccolo algorithm energy consumption for the mentioned payloads.

Figure 5 and 6 shows the measured energy consumption for decryption operation for two devices. Table 5 helps in a better understanding of the algorithm's arrangement from lowest to the highest for decryption. We can see a similar situation to encryption operation in terms of power

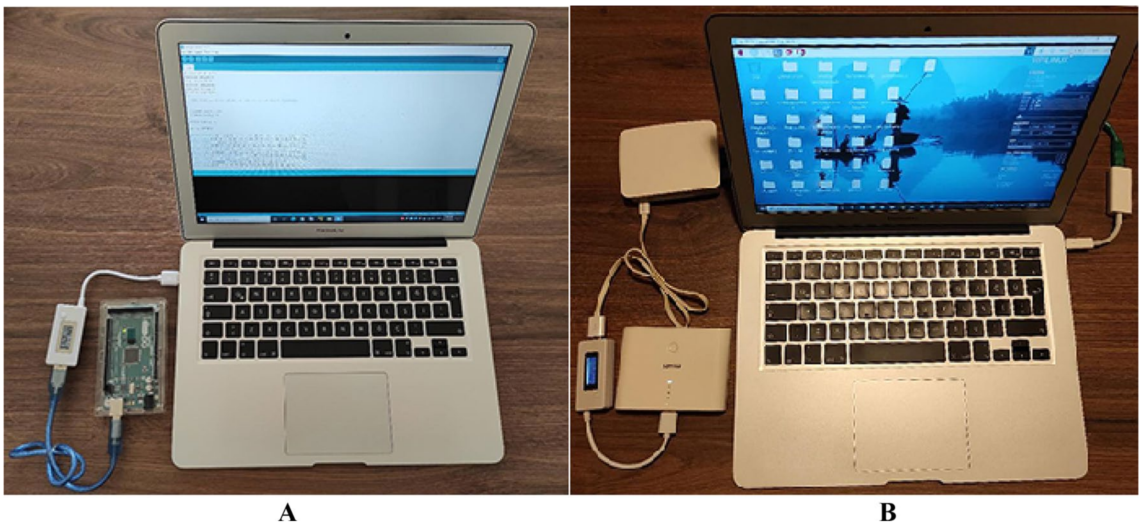

Fig. 2 Hardware layout of tested system (**a** using Arduino Mega—**b** using Raspberry Pi)

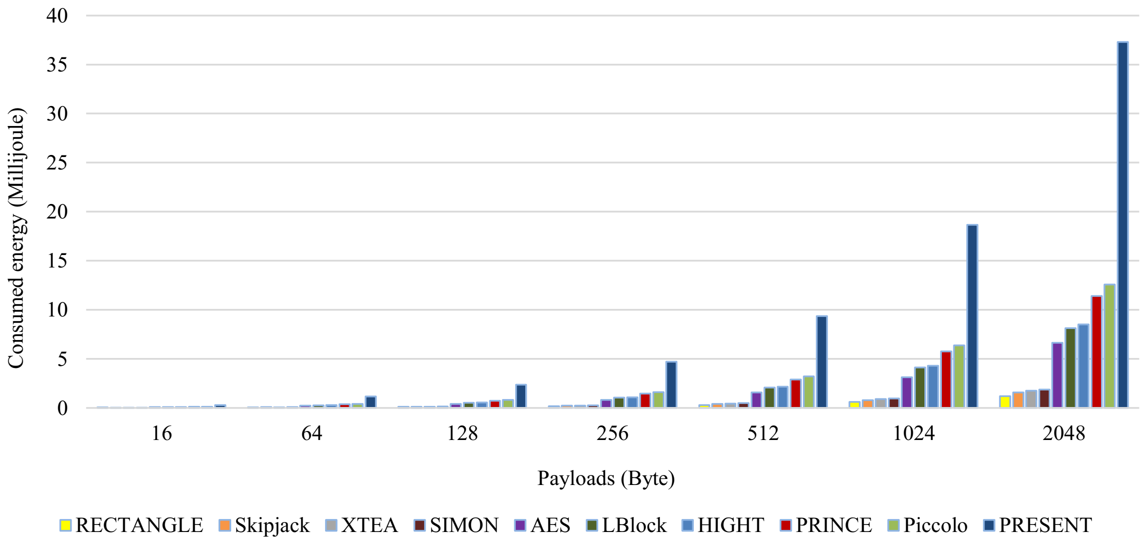

Fig. 3 Consumed energy for encryption (Raspberry Pi 3)

consumption for decryption mode. For the Arduino Mega graph, the vertical axis has been limited to 250, just like the encryption graph to visualize other algorithms better.

To analyze the amount of memory (RAM and ROM) usage by each block cipher, we used Atmel Studio 7 [56, 57]. Figure 7 and 8 are about encryption RAM usage for our tested systems. Arduino Mega 2560 has a considerably lower SRAM size (8 KB) compared to 1 GB RAM of Raspberry Pi 3. Table 6 sums up the encryption RAM usage for all payloads between two devices. For Raspberry Pi 3, XTEA and RECTANGLE have the minimum RAM usage against XTEA and Piccolo for Arduino Mega 2560. Considering the highest RAM occupation, Skipjack and AES have the maximum amount of RAM usage For Raspberry Pi 3. For Arduino Mega, HIGHT and PRESENT reaching the peak points. Our tested devices have different hardware specifications. Also, the infrastructure of tested algorithms and the targets that they designed affects the simulation results for encryption and decryption modes.

We can see changes in decryption RAM usage in Fig. 9 and 10. Looking at Table 7, for Raspberry Pi 3, there is no change in the order of the algorithms compared to encryption mode. For Arduino Mega 2560, XTEA occupies the minimum amount of RAM for decryption mode, and PRESENT reaches the peak just like the encryption mode. In contrast to an encryption operation, there are some changes in the algorithm sequences for Arduino Mega.

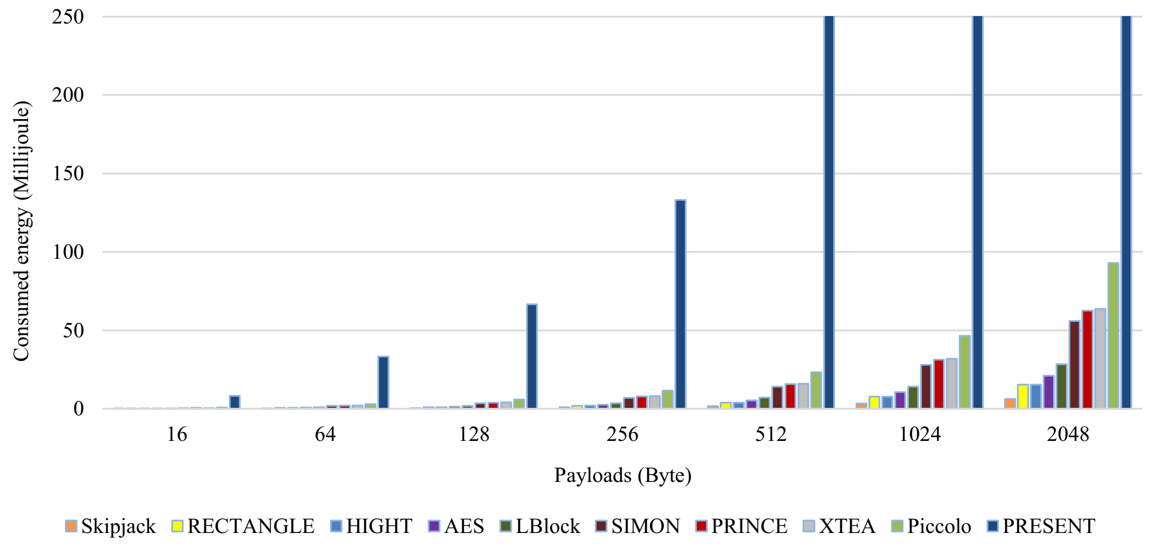

Fig. 4 Consumed energy for encryption (Arduino Mega 2560)

Table 4 Consumed energy comparison for encryption operation for Raspberry Pi 3 and Arduino Mega 2560

Device	Payloads	Algorithms lists from the lowest to the highest, respectively
Raspberry Pi 3	16	XTEA, Skipjack, SIMON, RECTANGLE, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
	64–128	RECTANGLE, XTEA, Skipjack, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
	256–2048	RECTANGLE, Skipjack, XTEA, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	64–2048	Skipjack, RECTANGLE, HIGHT, AES, LBlock, SIMON, PRINCE, XTEA, Piccolo, PRESENT

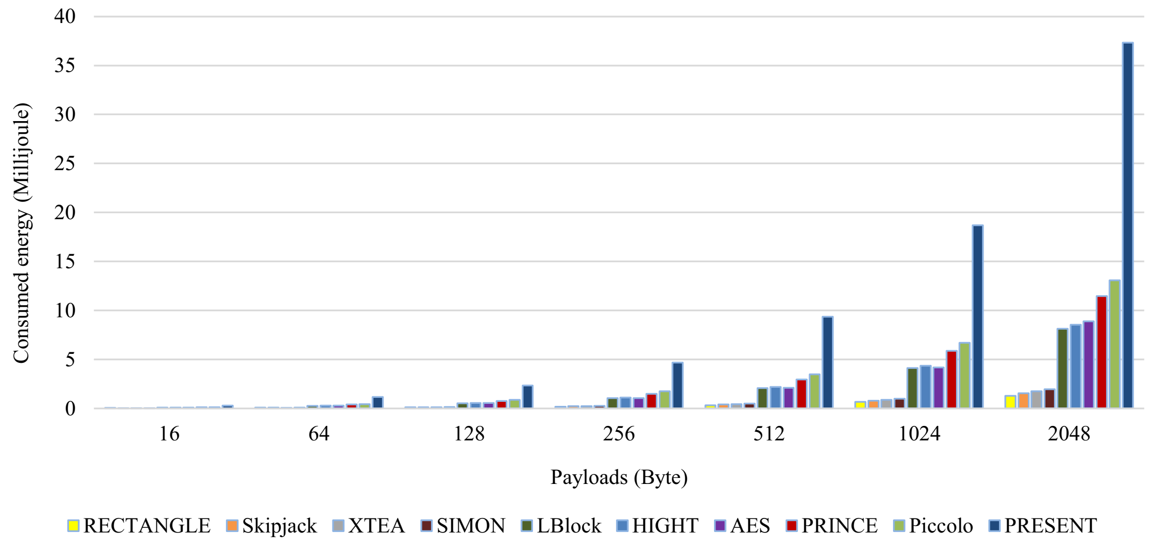

Fig. 5 Consumed energy for decryption (Raspberry Pi 3)

Raspberry Pi 3 is considerably stronger than Arduino Mega 2560 considering hardware features. Figure 11 and 12 provide a good comparison among ten lightweight block ciphers between two devices. Table 8 summarizes the algorithm's order from the lowest to the highest ROM usage. The peak ROM usage in bytes for encryption operation belongs to the RECTANGLE algorithm for Raspberry Pi 3 and for Arduino Mega 2560, AES occupied the highest

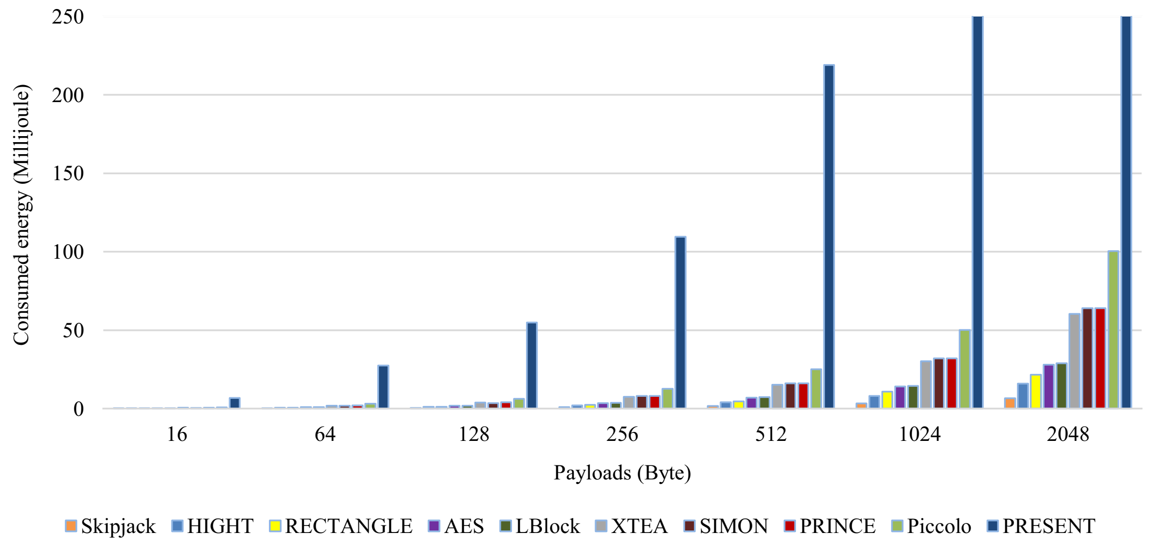

Fig. 6 Consumed energy for decryption (Arduino Mega 2560)

Table 5 Consumed energy comparison for decryption operation for Raspberry Pi 3 and Arduino Mega 2560

Device	Payloads	Algorithms lists from the lowest to the highest, respectively
Raspberry Pi 3	16	XTEA, SIMON, Skipjack, RECTANGLE, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	64	XTEA, Skipjack, RECTANGLE, SIMON, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	128–2048	RECTANGLE, Skipjack, XTEA, SIMON, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16	Skipjack, HIGHT, RECTANGLE, LBlock, AES, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	64–128	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	256–2048	Skipjack, HIGHT, RECTANGLE, AES, LBlock, XTEA, SIMON, PRINCE, Piccolo, PRESENT

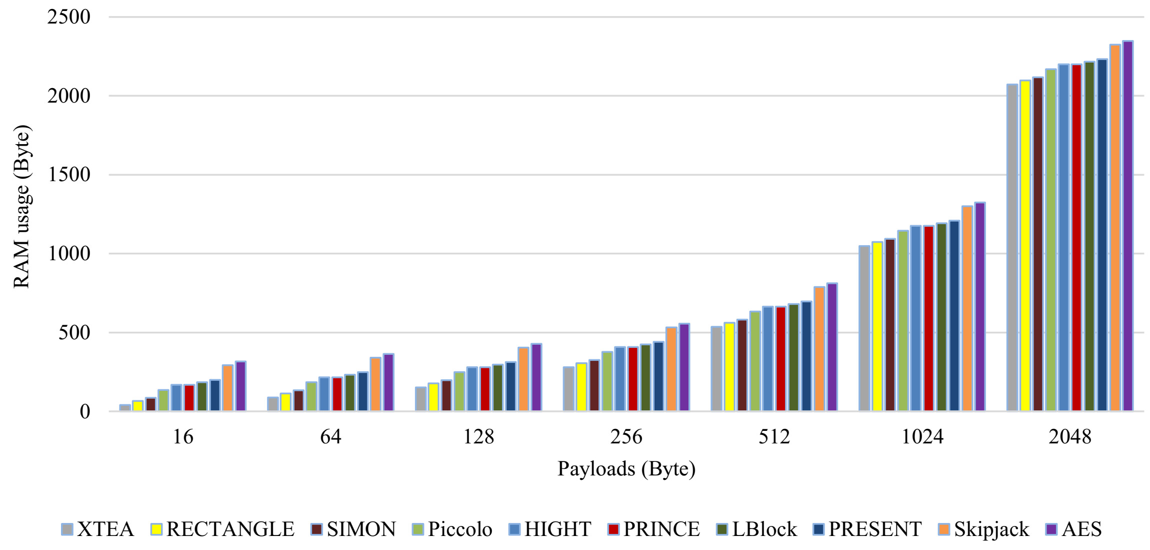

Fig. 7 Encryption RAM usage (Raspberry Pi 3)

ROM. SIMON and XTEA for Raspberry Pi 3 and SIMON and PRESENT for Arduino Mega have the lowest ROM usage.

Figure 13 and 14 illustrate ROM usage for each of ten block ciphers regarding different payloads for decryption operation for Raspberry Pi and Arduino Mega. In Fig. 13,

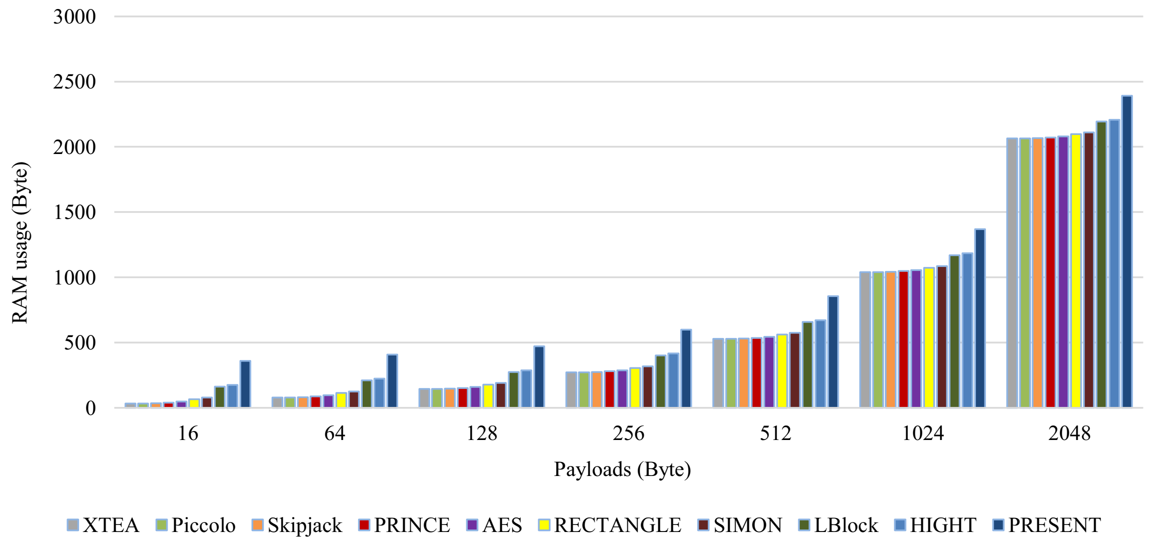

Fig. 8 Encryption RAM usage (Arduino Mega 2560)

Table 6 Sorted algorithms for encryption RAM usage (Arduino Mega 2560 vs Raspberry Pi 3)

Device	Payloads	Algorithms lists from the lowest to the highest, respectively
Raspberry Pi 3	16–2048	XTEA, RECTANGLE, SIMON, Piccolo, HIGHT, PRINCE, LBlock, PRESENT, Skipjack, AES
Arduino Mega 2560	16–2048	XTEA, Piccolo, Skipjack, PRINCE, AES, RECTANGLE, SIMON, LBlock, HIGHT, PRESENT

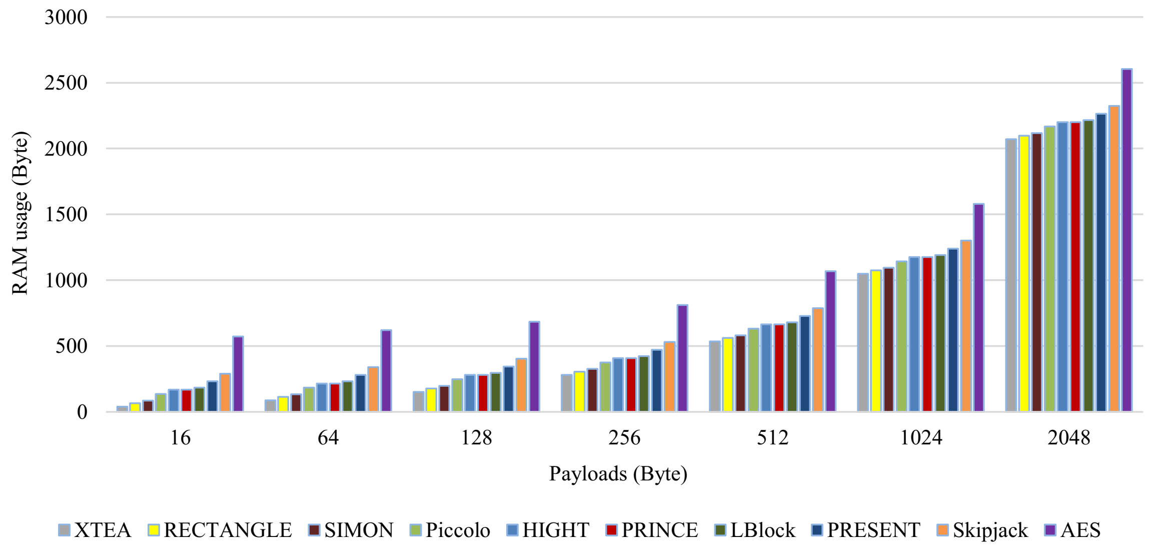

Fig. 9 Decryption RAM usage (Raspberry Pi 3)

RECTANGLE, and Fig. 14, AES considerably have a higher amount of ROM usage compared to other block ciphers from the beginning. Interestingly, the rate of increase for ROM usage follows a specific pattern for both devices. SIMON and XTEA have the minimum ROM usage for both devices in decryption mode Table 9.

We can illustrate the average encrypting execution times for Raspberry Pi 3 and Arduino Mega 2560 in Fig. 8 and 9. For all payloads, PRESENT is in the first rank with the highest encrypting execution time for both devices. Because the PRESENT running times after 512 bytes are high, the vertical axis in Fig. 9 has been limited to 0.9 s to show other algorithms running times clearly. Looking at Table 10, we

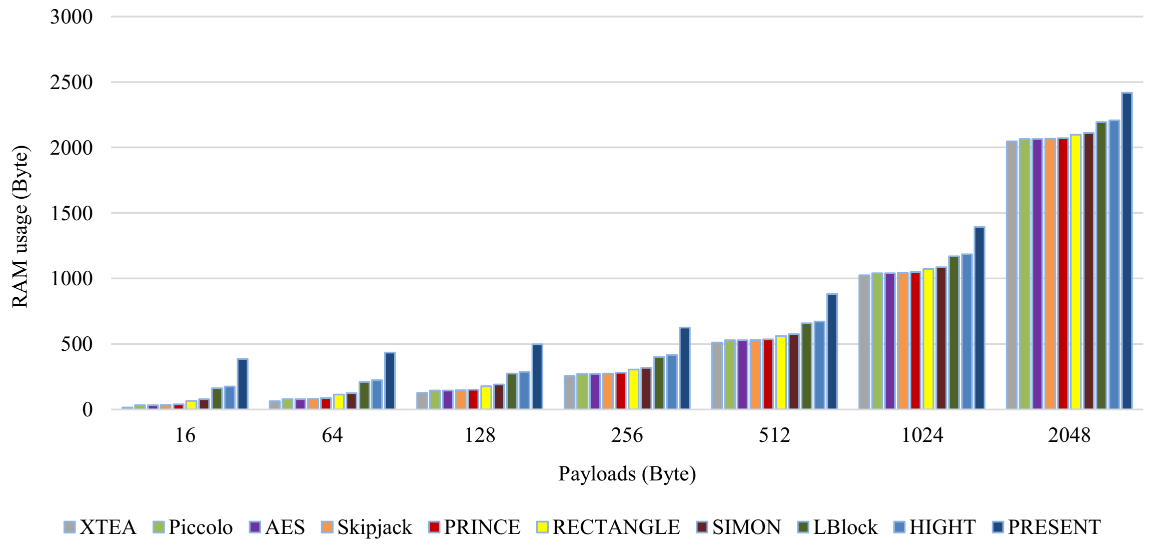

Fig. 10 Decryption RAM usage (Arduino Mega 2560)

Table 7 Sorted algorithms for decryptin RAM usage (Arduino Mega 2560 vs Raspberry Pi 3)

Device	Payloads	Algorithms lists from the lowest to the highest, respectively
Raspberry Pi 3	16–2048	XTEA, RECTANGLE, SIMON, Piccolo, HIGHT, PRINCE, LBlock, PRESENT, Skipjack, AES
Arduino Mega 2560	16–2048	XTEA, Piccolo, AES, Skipjack, PRINCE, RECTANGLE, SIMON, LBlock, HIGHT, PRESENT

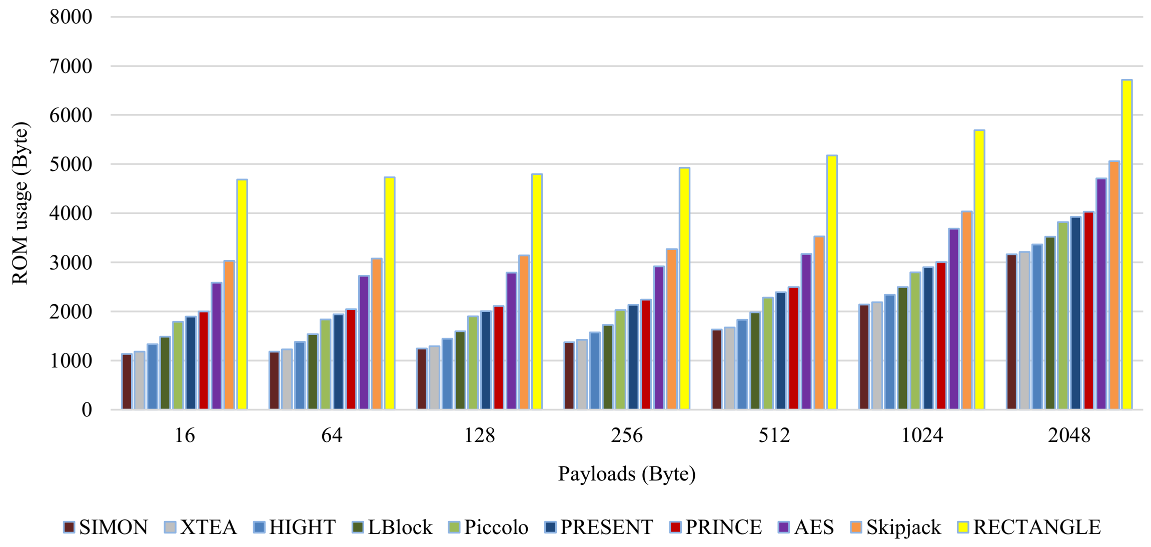

Fig. 11 Encryption ROM usage (Raspberry Pi 3)

can see the ascending order of ten algorithms by average encryption execution time for each device (Figure 15, 16).

Figure 17 and 18 represent the average time spent for decryption for our testbed systems. For two charts, we can consider the PRESENT block cipher with the highest execution times values, especially beginning from 128 bytes. The execution time has been limited to 0.9 s in the second chart to view other algorithms running times easier. Table 11 summarizes the algorithm's order for decryption operation considering running time. PRINCE, Piccolo, and PRESENT have hardware-oriented architecture and have the highest running times respectively for software simulations. On the other hand, Skipjack and HIGHT have a simple structure and have the minimum running times among other algorithms

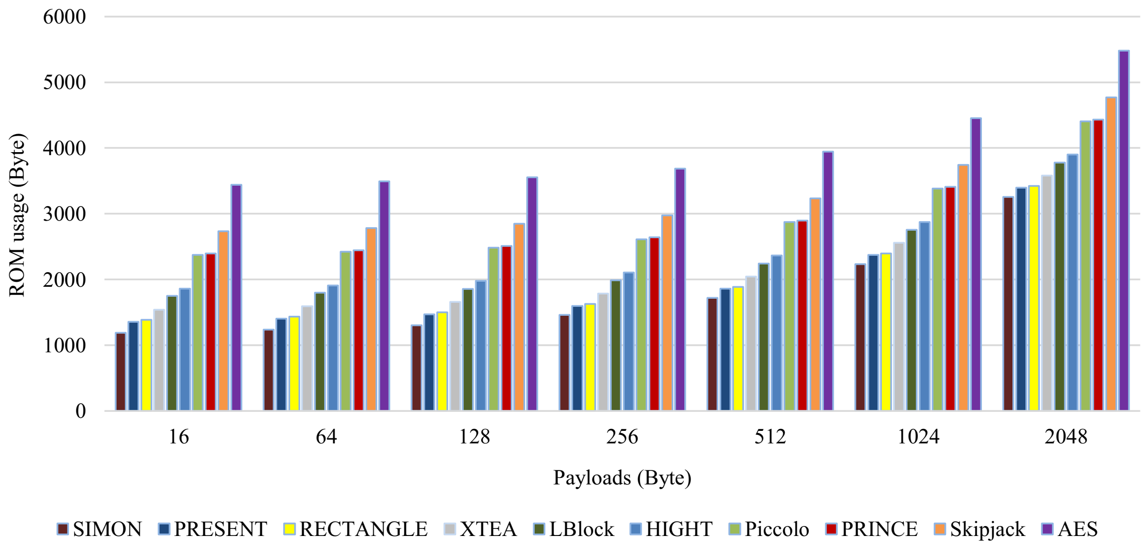

Fig. 12 Encryption ROM usage (Arduino Mega 2560)

Table 8 Tested algorithms order by ROM usage for encryption

Device	Payloads	Algorithms lists from the lowest to the highest, respectively
Raspberry Pi 3	16–2048	SIMON, XTEA, HIGHT, LBlock, Piccolo, PRESENT, PRINCE, AES, Skipjack, RECTANGLE
Arduino Mega 2560	16–2048	SIMON, PRESENT, RECTANGLE, XTEA, LBlock, HIGHT, Piccolo, PRINCE, Skipjack, AES

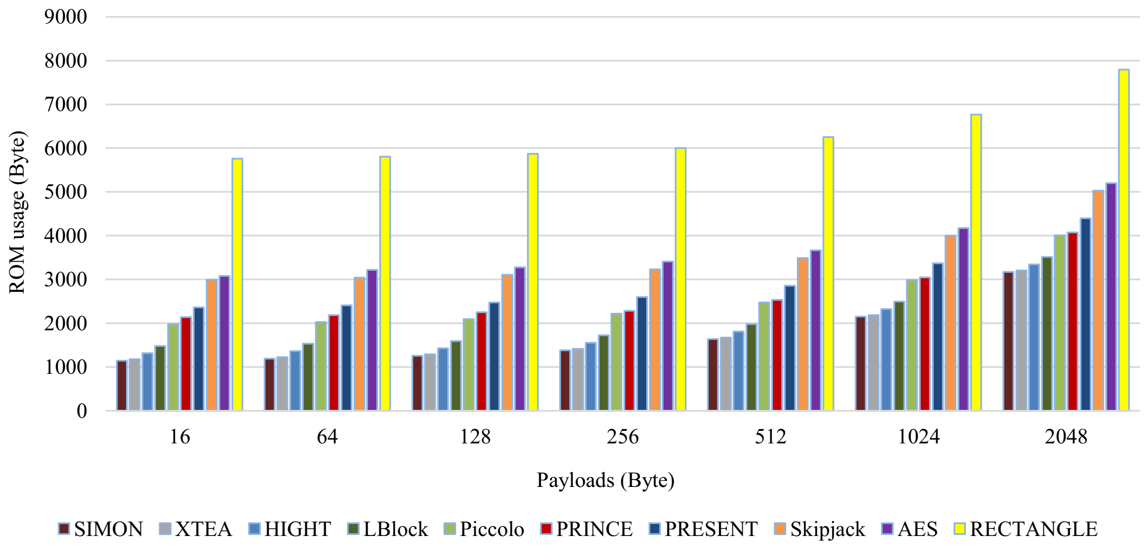

Fig. 13 Decryption ROM usage (Raspberry Pi 3)

for Arduino Mega. For Raspberry Pi 3, XTEA and RECTANGLE are faster than other lightweight block ciphers (Table 12, 13).

The average encryption throughputs are shown in Fig. 19 and 20 for Raspberry Pi 3 and Arduino Mega 2560. Throughput is being calculated using the following equation:

$$\text{Throughput} = \frac{\text{Number of Bytes}}{\text{End Time} - \text{Start Time}} \quad (3)$$

For encryption mode, for 16 bytes, XTEA achieves the peak value among other block ciphers for Raspberry Pi 3. From 64 bytes, RECTANGLE takes the highest throughput

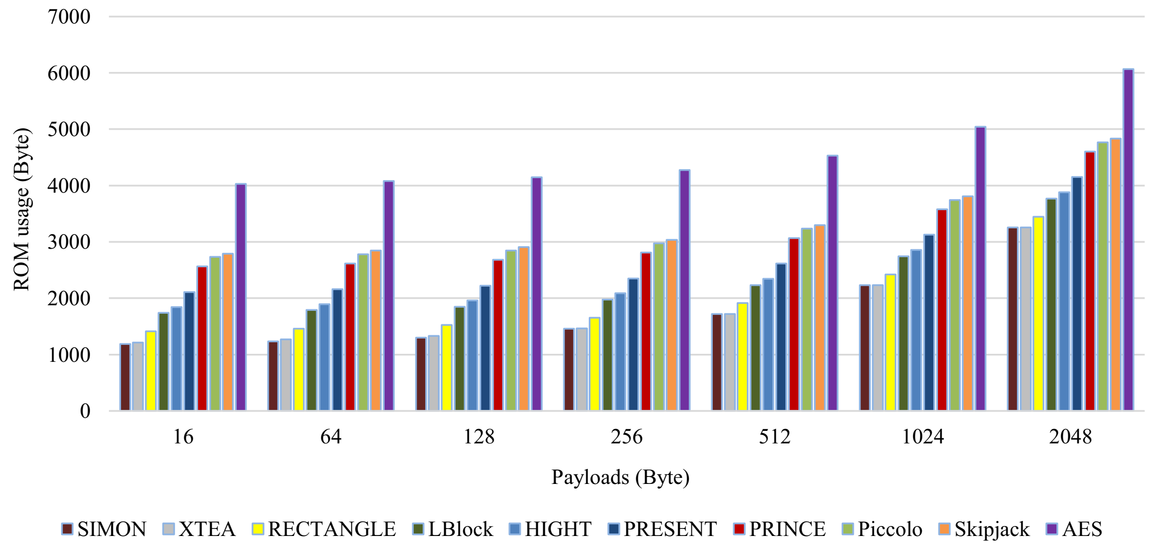

Fig. 14 Decryption ROM usage (Arduino Mega 2560)

Table 9 Tested algorithms order by ROM usage for decryption

Device	Payloads	Algorithms lists from the lowest to the highest, respectively
Raspberry Pi 3	16–2048	SIMON, XTEA, HIGHT, LBlock, Piccolo, PRINCE, PRESENT, Skipjack, AES, RECTANGLE
Arduino Mega 2560	16–2048	SIMON, XTEA, RECTANGLE, LBlock, HIGHT, PRESENT, PRINCE, Piccolo, Skipjack, AES

Table 10 Algorithms order due to encryption execution time

Device	Payloads	Algorithms lists from the lowest to the highest, respectively
Raspberry Pi 3	16	XTEA, Skipjack, SIMON, RECTANGLE, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	64–128	RECTANGLE, XTEA, Skipjack, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
	256–2048	RECTANGLE, Skipjack, XTEA, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	64–2048	Skipjack, RECTANGLE, HIGHT, AES, LBlock, SIMON, PRINCE, XTEA, Piccolo, PRESENT

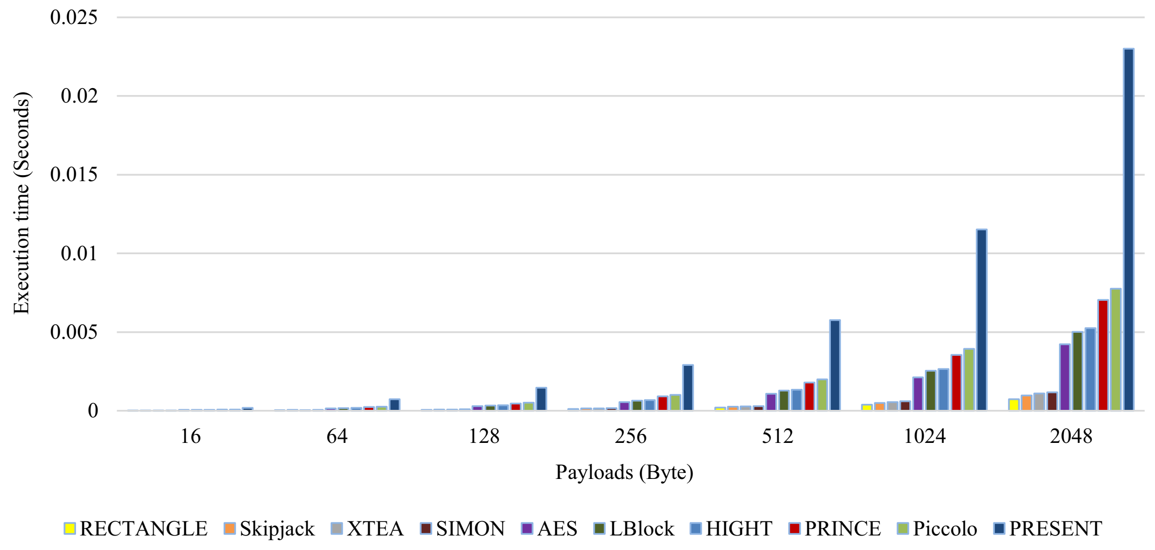

Fig. 15 Average encrypting execution time (Raspberry Pi 3)

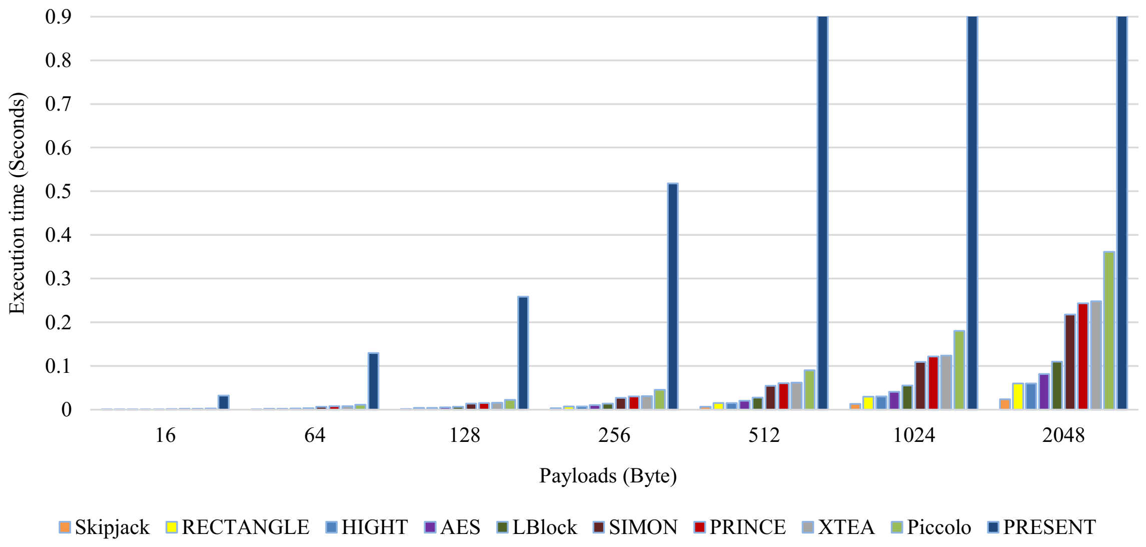

Fig. 16 Average encrypting execution time (Arduino Mega 2560)

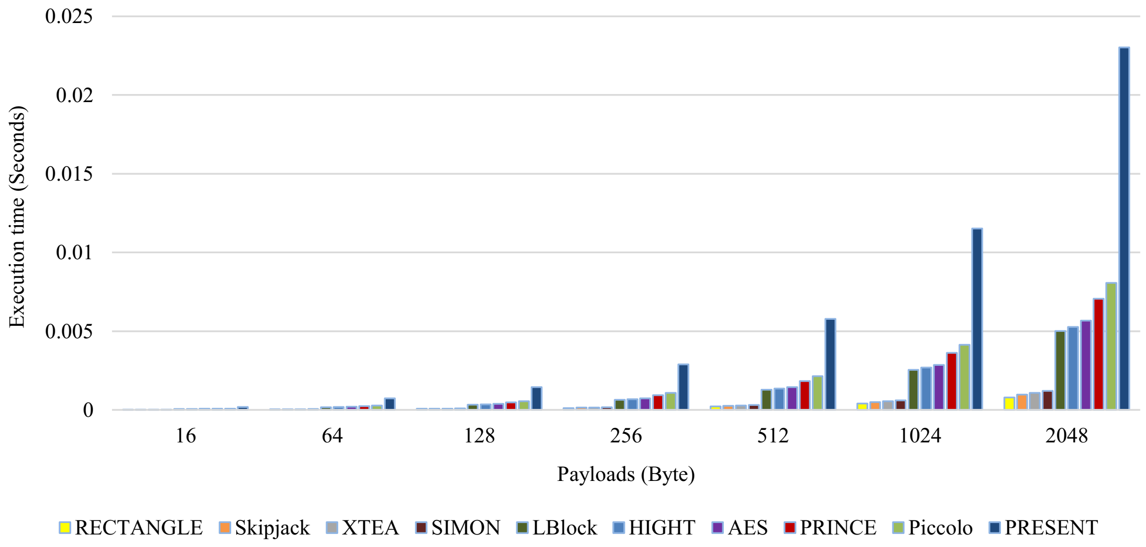

Fig. 17 Average decrypting execution time (Raspberry Pi 3)

for Raspberry Pi 3. For Arduino Mega 2560, Skipjack has the maximum encryption throughput for all payloads. For both devices, PRESENT has the lowest encryption throughput.

Figure 21 shows the average decryption throughput for Raspberry Pi 3. RECTANGLE, Skipjack, SIMON, and XTEA has significantly higher throughputs than other algorithms. PRINCE, Piccolo, and PRESENT encryption throughput values are the smallest ones among others. For Arduino Mega 2560, Skipjack decryption throughput value is substantially bigger than other algorithms, HIGHT and RECTANGLE are in the second and third positions. As expected, Piccolo and PRESENT have the minimum decryption throughput values for both devices (Fig. 22).

3 Conclusion and Future Works

Internet of Things is composed of interrelated devices that exchange data among themselves. Many of the IoT devices are sending sensitive information that should be accessible only by legal authorities. Therefore, securing the data is a significant priority for an IoT network. To provide end-to-end security for low-power devices, we can apply lightweight block ciphers. In this article, we chose ten lightweight block

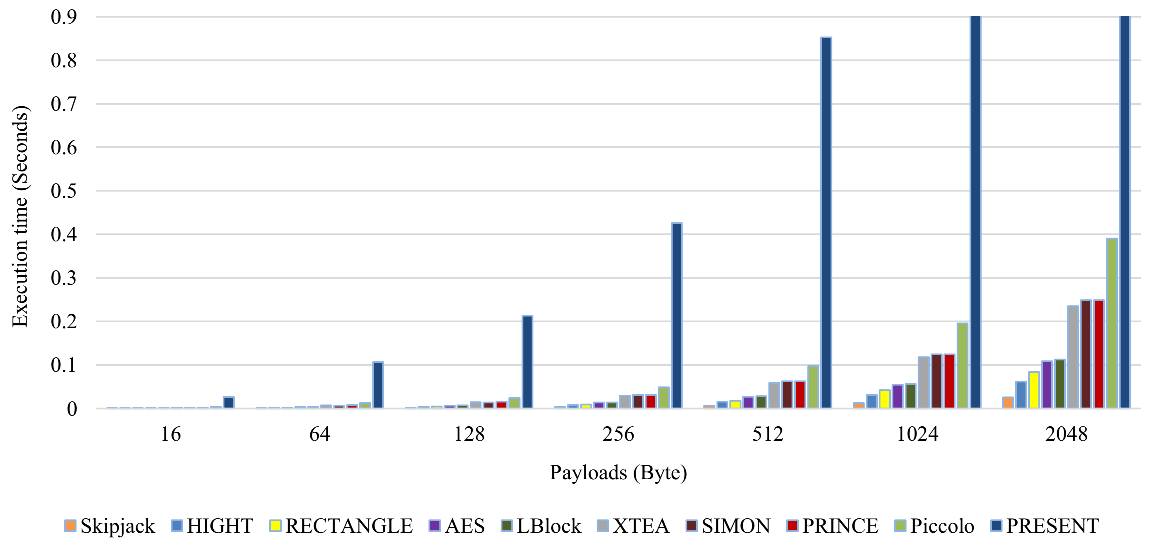

Fig. 18 Average decrypting execution time (Arduino Mega 2560)

Table 11 Algorithms order due to decryption execution time

Device	Payloads	Algorithms lists from the lowest to the highest, respectively
Raspberry Pi 3	16	XTEA, SIMON, Skipjack, RECTANGLE, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	64	XTEA, RECTANGLE, Skipjack, SIMON, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	128–2048	RECTANGLE, Skipjack, XTEA, SIMON, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16	Skipjack, HIGHT, RECTANGLE, LBlock, AES, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	64–2048	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT

Table 12 Algorithms sequence due to the average encryption throughput

Device	Payloads	Algorithms lists from the highest to the lowest, respectively
Raspberry Pi 3	16	XTEA, SIMON, Skipjack, RECTANGLE, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	64–128	RECTANGLE, XTEA, Skipjack, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
	256–2048	RECTANGLE, Skipjack, XTEA, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16–2048	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT

Table 13 Algorithms sequence due to the average decryption throughput

Device	Payloads	Algorithms lists from the highest to the lowest, respectively
Raspberry Pi 3	16	XTEA, SIMON, Skipjack, RECTANGLE, LBlock, HIGHT, AES, PRINCE, Piccolo, PRESENT
	64	RECTANGLE, XTEA, Skipjack, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
	128–2048	RECTANGLE, Skipjack, XTEA, SIMON, AES, LBlock, HIGHT, PRINCE, Piccolo, PRESENT
Arduino Mega 2560	16	Skipjack, HIGHT, RECTANGLE, LBlock, AES, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	64–128	Skipjack, HIGHT, RECTANGLE, AES, LBlock, SIMON, XTEA, PRINCE, Piccolo, PRESENT
	256–2048	Skipjack, HIGHT, RECTANGLE, AES, LBlock, XTEA, PRINCE, SIMON, Piccolo, PRESENT

ciphers and tested their performance over Raspberry Pi 3 and Arduino Mega 2560 devices. We measured encryption/decryption functions performance for different payloads due to memory usage (RAM and ROM), execution time, throughput, and energy consumption.

This paper will help readers to select the right platform and enciphering algorithm due to multiple factors like energy and memory usage, especially for software platforms. As future work, we can expand results for more block ciphers, compare their performance with other enciphering techniques such as stream ciphers, and do tests over new IoT testbeds.

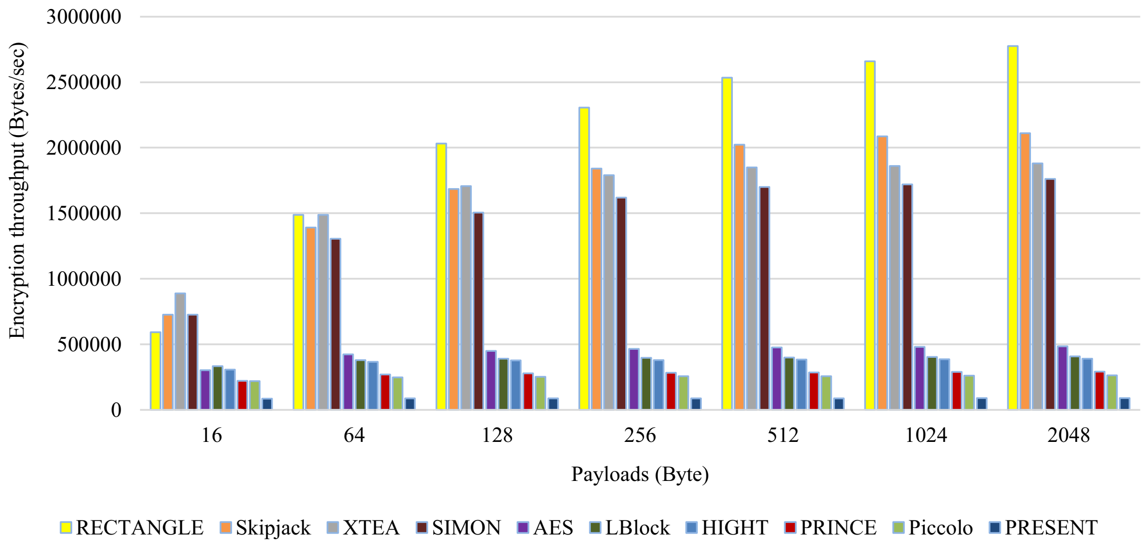

Fig. 19 Average encrypting throughput (Raspberry Pi 3)

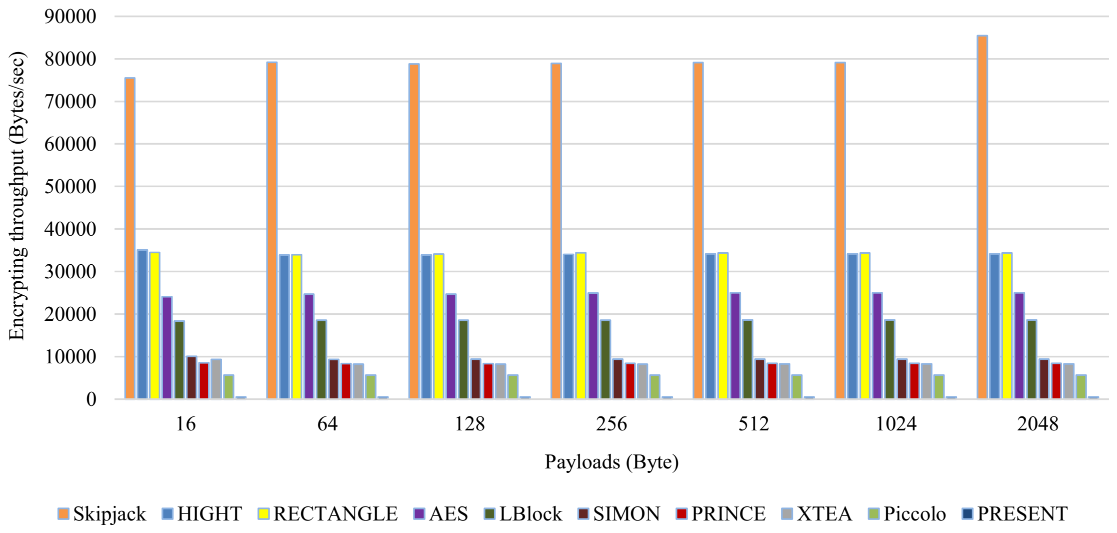

Fig. 20 Average encrypting throughput (Arduino Mega 2560)

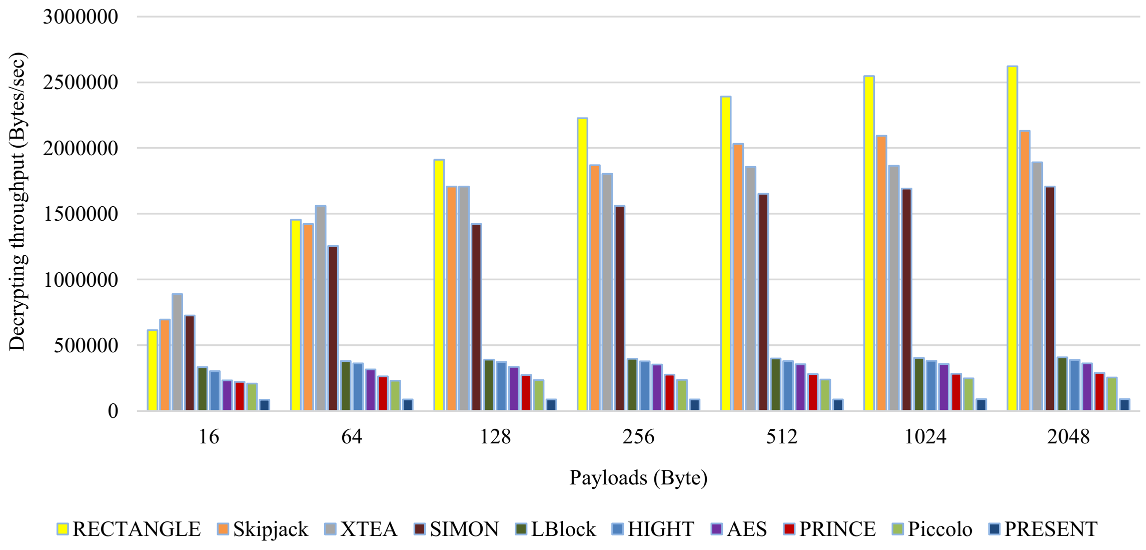

Fig. 21 Average decryption throughput (Raspberry Pi 3)

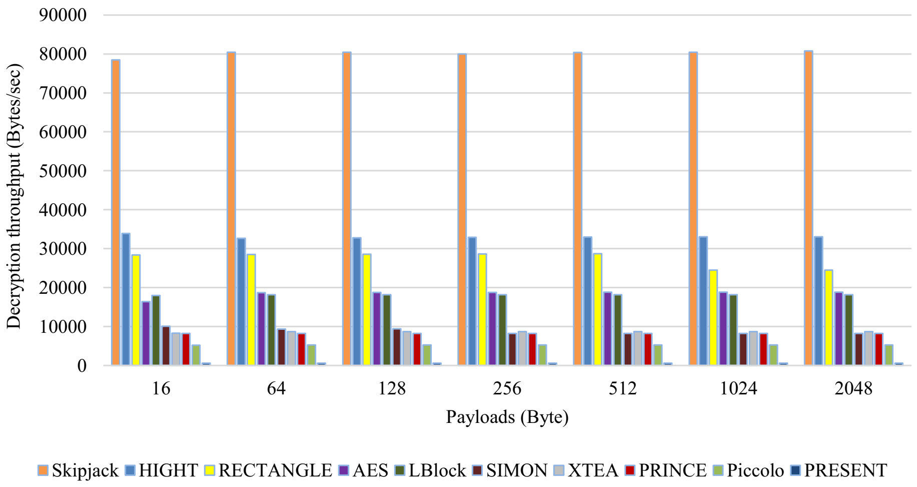

Fig. 22 Average decryption throughput (Arduino Mega 2560)

References

1. Vaidya, B.; Mouftah, H.T.: IoT applications and services for connected and autonomous electric vehicles. Arab. J. Sci. Eng. **45**, 2559–2569 (2020)
2. Mahapatra, S.N.; Singh, B.K.; Kumar, V.: A survey on secure transmission in internet of things: taxonomy, recent techniques, research requirements, and challenges. Arab. J. Sci. Eng. **45**(8), 6211–6240 (2020)
3. Daemen, J.; Rijmen, V.: AES Proposal: Rijndael. NIST AES Proposal (1998). http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf
4. Zhang, W.; Bao, Z.; Lin, D.; Rijmen, V.; Yang, B.; Verbauwhede, I.: RECTANGLE: a bit-slice lightweight block cipher suitable for multiple platforms. Sci. China Inf. Sci. **58**(12), 1–15 (2015)
5. Bogdanov; Andrey; et al. "PRESENT: An ultra-lightweight block cipher." International Workshop on Cryptographic Hardware and Embedded Systems. Springer, Berlin, (2007)
6. Kim; Jongsung; Raphael C-W; Phan. "A cryptanalytic view of the NSA's Skipjack block cipher design." International Conference on Information Security and Assurance. Springer, Berlin, Heidelberg, 2009.
7. Borghoff; Julia; et al. "PRINCE–a low-latency block cipher for pervasive computing applications." International Conference on the Theory and Application of Cryptology and Information Security. Springer, Berlin, Heidelberg, (2012)
8. Beaulieu; Ray; et al. "SIMON and SPECK: Block Ciphers for the Internet of Things." IACR Cryptol. ePrint Arch: 585 (2015)
9. Hong; Deukjo; et al. "HIGHT: A new block cipher suitable for low-resource device." International workshop on cryptographic hardware and embedded systems. Springer, Berlin, Heidelberg, (2006)
10. Shibutani; Kyoji; et al. "Piccolo: an ultra-lightweight blockcipher."International workshop on cryptographic hardware and embedded systems. Springer, Berlin, Heidelberg, (2011)
11. Wu; Wenling; Lei Zhang; "LBlock: a lightweight block cipher." International Conference on Applied Cryptography and Network Security. Springer, Berlin, Heidelberg, (2011)
12. Moon; Dukjae; et al. "Impossible differential cryptanalysis of reduced round XTEA and TEA." International Workshop on Fast Software Encryption. Springer, Berlin, Heidelberg, (2002)

13. Lakshmi, M.S.; Srikanth, V.: A study on light-weight cryptography algorithms for data security in IOT. Int J Eng Technol **7**(2.7), 887–890 (2018)
14. Tausif, M.; Ferzund, J.; Jabbar, S.; Shahzadi, R.: Towards designing efficient lightweight ciphers for internet of things. KSII Trans Int Inf Syst **11**(8), 4006–4024 (2017)
15. Singh, P.; Acharya, B.; Chaurasiya, R.K.: A comparative survey on lightweight block ciphers for resource constrained applications". Int J High Perform Syst Archit **8.4**, 250–270 (2019)
16. Rana, S.: A survey paper of lightweight block ciphers based on their different design architectures and performance metrics. Int J Comput Eng Inf Technol **11**(6), 119–129 (2019)
17. Hatzivasilis, G.; Fysarakis, K.; Papaefstathiou, I.; Manifavas, C.: A review of lightweight block ciphers. J Cryptograp Eng **8**(2), 141–184 (2018)
18. Sadkhan, S.B.; Salman, A.O.: A survey on lightweight-cryptography status and future challenges. In: 2018 International Conference on Advance of Sustainable Engineering and its Application (ICASEA), pp. 105–108. IEEE (2018)
19. Shah; Ankit; Margi Engineer. "A survey of lightweight cryptographic algorithms for iot-based applications." Smart innovations in communication and computational sciences. Springer, Singapore, (2019). 283–293
20. Sehrawat, D.; Gill, N.S.: Lightweight block ciphers for IoT based applications: a review. Int J Appl Eng Res **13.5**, 2258–2270 (2018)
21. Dutta; Indira Kalyan; Bhaskar G.; Magdy B. "Lightweight cryptography for internet of insecure things: a survey." 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, (2019)
22. Yeoh; Wei-Zhu; Je Sen The; Mohd Ilyas Sobirin Bin Mohd Sazali. "µ 2: A Lightweight Block Cipher." computational science and technology. Springer, Singapore, 281–290 (2020)
23. Patil; Anita; Soumi B.; Gautam B. "A survey on securing smart gadgets using lightweight cryptography." Proceedings of International Conference on Wireless Communication. Springer, Singapore, (2020)
24. Biswas, A.; Majumdar, A.; Nath, S.; Dutta, A.; Baishnab, K.: Lrbc: a lightweight block cipher design for resource constrained iot devices. J. Ambient Intell. Human. Comput. (2020). https://doi.org/10.1007/s12652-020-01694-9
25. Bansod, G.; Pisharoty, N.; Patil, A.: BORON: an ultra-lightweight and low power encryption design for pervasive computing. Front Inf Technol Elect Eng **18**(3), 317–331 (2017)
26. Bansod, G.; Patil, A.; Sutar, S.; Pisharoty, N.: ANU: an ultra lightweight cipher design for security in IoT. Sec Commun Netw **9**(18), 5238–5251 (2016)
27. Sehrawat, D.; Gill, N.S.: Performance evaluation of newly proposed lightweight cipher, BRIGHT. Int. J. Intell. Eng. Syst. **12**(4), 71–80 (2019). https://doi.org/10.22266/ijies2019.0831.08
28. Al-Rahman, S.A.; Sagheer, A.; Dawood, O. NVLC: new variant lightweight cryptography algorithm for internet of things. In 2018 1st Annual International Conference on Information and Sciences (AiCIS) (pp. 176–181). IEEE. (2018)
29. Liu, B.T.; Li, L.; Wu, R.X.; Xie, M.M.; Li, Q.P.: Loong: a family of involutional lightweight block cipher based on SPN structure. IEEE Access **7**, 136023–136035 (2019)
30. Salunke, R.; Bansod, G.; Naidu, P.: Design and implementation of a lightweight encryption scheme for wireless sensor nodes. In: Arai, K., Bhatia, R., Kapoor, S. (eds.) Intelligent Computing. CompCom 2019. Advances in Intelligent Systems and Computing, vol. 998. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-22868-2_41
31. Hosseinzadeh; Jaber; Abbas Ghaemi Bafghi.; "Software implementation and evaluation of lightweight symmetric block ciphers of the energy perspectives and memory." arXiv preprint arxiv:1706.03909 (2017)
32. Singh; Praneet; Kedar Deshpande. "Performance evaluation of cryptographic ciphers on IoT devices." arXiv preprint arxiv:1812.02220 (2018)
33. Grossschadl; Johann; Stefan Tillich; Christian Rechberger; Michael Hofmann; Marcel Medwed. "Energy evaluation of software implementations of block ciphers under memory constraints." In 2007 Design, Automation and Test in Europe Conference and Exhibition, pp. 1–6. IEEE, (2007)
34. Botta; Miroslav; Milan Simek; Nathalie Mitton. "Comparison of hardware and software based encryption for secure communication in wireless sensor networks." 2013 36th International Conference on Telecommunications and Signal Processing (TSP). IEEE, (2013)
35. Engineer, Margi; Ankit Shah. "Performance analysis of lightweight cryptographic algorithms simulated on arduino UNO and MATLAB using the voice recognition application." In 2018 International Conference on Circuits and Systems in Digital Enterprise Technology (ICCSDET), pp. 1–7. IEEE, (2018)
36. Batina; Lejla; Amitabh D.; Barış E.; Elif B.K.; Nele M.; Christof P.; Ingrid V.; Tolga Y. "Dietary recommendations for lightweight block ciphers: power, energy and area analysis of recently developed architectures." In International Workshop on Radio frequency Identification: Security and Privacy Issues, pp. 103–112. Springer, (2013)
37. Beaulieu; Ray; Douglas Shors; Jason Smith; Stefan Treatman-Clark; Bryan Weeks; Louis Wingers. "The SIMON and SPECK block ciphers on AVR 8-bit microcontrollers." In International Workshop on Lightweight Cryptography for Security and Privacy, pp. 3–20. Springer, Cham, (2014)
38. Omrani; Tasnime; Rhouma Rhouma; Layth Sliman. "Lightweight cryptography for resource-constrained devices: a comparative study and rectangle cryptanalysis." In International Conference on Digital Economy, pp. 107–118. Springer, Cham, (2018)
39. Ertaul; Levent; Sachin K.R. "Performance analysis of CLEFIA, PICCOLO, TWINE Lightweight Block Ciphers in IoT Environment." In proceedings of the International Conference on Security and Management (SAM), The Steering Committee of The World Congress in Computer Science, computer engineering and applied computing (WorldComp), pp. 25–31. (2017)
40. Alizadeh; Mojtaba; Mazleena S.; Mazdak Z.; Jafar S.; Sasan K. "Security and performance evaluation of lightweight cryptographic algorithms in RFID." Kos Island, Greece, pp.45–50. (2012)
41. Almusaylim, Z.A.; Zaman, N.: A review on smart home present state and challenges: linked to context-awareness internet of things (IoT). Wireless Netw. **25**(6), 3193–3204 (2019)
42. A. Almusaylim, Z.; Jhanjhi, N.: Comprehensive review: privacy protection of user in location-aware services of mobile cloud computing. Wireless Pers. Commun. **111**, 541–564 (2020). https://doi.org/10.1007/s11277-019-06872-3
43. Murvay; Pal-Stefan, et al. "Development of an autosar compliant cryptographic library on state-of-the-art automotive grade controllers." 2016 11th International Conference on Availability, Reliability and Security (ARES). IEEE, (2016)
44. Salah; Khaled. "A queueing model to achieve proper elasticity for cloud cluster jobs." 2013 IEEE Sixth International Conference on Cloud Computing. IEEE, (2013)
45. El Kafhali; Said; Khaled Salah. "Stochastic modelling and analysis of cloud computing data center." 2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN). IEEE, (2017)
46. Appel; Michael; et al. "Block ciphers for the IoT–SIMON, SPECK, KATAN, LED, TEA, PRESENT, and SEA compared." (2016)
47. Cazorla, M.; Marquet, K.; Minier, M. Survey and benchmark of lightweight block ciphers for wireless sensor networks. In

2013 International Conference on Security and Cryptography (SECRYPT). pp. 1–6. IEEE, (2013)

48. Lara-Niño C.A.; Morales-Sandoval M.; Díaz-Pérez A. "An evaluation of AES and present ciphers for lightweight cryptography on smartphones," 2016 International Conference on Electronics, Communications and Computers (CONIELECOMP), Cholula, (2016), pp. 87–93
49. Kotel; Sonia; Fatma S.; Medien Z.; Mohsen M.; Adel B.; Rached T. "Performance evaluation and design considerations of lightweight block cipher for low-cost embedded devices." In 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), pp. 1–7. IEEE, (2016)
50. Diehl; William; Farnoud F.; Panasayya Y.; Jens-Peter K.; Kris G. "Comparison of hardware and software implementations of selected lightweight block ciphers." In 2017 27th International Conference on Field Programmable Logic and Applications (FPL), pp. 1–4. IEEE, (2017)
51. Sehrawat; Deepti; Nasib S.G. "Performance evaluation of newly proposed lightweight cipher, BRIGHT.", Int. J. Eng. Adv. Technol. (IJEAT), ISSN: 2249–8958, 8(5), (2019)
52. Doomun; Razvi M.; Soyjaudah K. M. S.; "Analytical Comparison of Cryptographic Techniques for Resource-constrained Wireless Security." IJ network security 9(1) (2009): 82–94
53. Lee, W.K.; Phan, Raphael C.-W.; Goi, B.M.: Fast and energy-efficient block ciphers implementations in ARM Processors and Mali GPU. IETE J. Res. (2020). https://doi.org/10.1080/03772063.2020.1725656
54. Malina; Lukas; Vlastimil C.; Zdenek M.; Jan Hajny; Kimio O.; Vaclav Z. "Evaluation of software-oriented block ciphers on smartphones. In: International Symposium on Foundations and Practice of Security, pp. 353–368. Springer, Cham, (2013)
55. Çakiroglu, M.: Software implementation and performance comparison of popular block ciphers on 8-bit low-cost microcontroller. Int. J. Phys. Sci **5**(9), 1338–1343 (2010)
56. Bayilmis C; Kucuk K; "Internet of things: theory and applications", Daisyscience international publishing house, (2019)
57. Barrett, S.F.; Pack, D.J.: Microchip AVR\txtreg microcontroller primer: programming and interfacing, third edition. Synth. Lect. Digit. Circ. Syst. **14**(2), 1–383 (2019)