A NOVEL ARCHITECTURE FOR LIGHTWEIGHT BLOCK CIPHER, PICCOLO

Rahiyanath T. Y.

M.Tech Student, Communication Engineering, Kmea Engineering College, Kerala, India

Abstract

Security and privacy are going to be the key factors for the deployment of new applications, since people will only accept these deployments if these are based on secure, trustworthy and privacy-preserving infra-structures. Cryptography plays a major role in the security of data transmission and the development of computing technology imposes stronger requirements on the cryptography schemes. Lightweight cryptography is a cryptographic algorithm or protocol tailored for implementation in constrained environments including RFID tags, sensors, contactless smart cards, health-care devices and so on. These devices are leading to an ever increasing need to provide security. In order to satisfy these needs, secure and efficient encryption and authentication schemes have to be developed. Symmetric-key algorithms, especially lightweight block ciphers, play an important role to provide security in these applications. Piccolo is a new lightweight block cipher which is optimized for extremely constrained devices. Piccolo supports 64-bit block with 80 or 128-bit keys, and has an iterative structure which is a variant of a generalized Feistel network. Piccolo achieves both high security and extremely compact implementation unlike the other Feistel-type structure based lightweight block ciphers. The proposed system deals with an efficient implementation of Piccolo block cipher architecture that will fulfill mandatory requirements such as throughput and speed for low-resource devices like RFID tags and wireless sensors. The new architecture is designed such that it shares the key scheduling block for two plain text blocks concurrently.

Key Words: *Lightweight Cryptography, RFID, Piccolo, and Fiestel Networks.*

----------***----------

1. INTRODUCTION

Security of data is of great importance in most of the applications during communication, especially for those concerning financial and personal issues. It has been a topic of great interest for a long time, mostly in today's societies living in a world of ubiquitous computing. Computers are all over, millions and millions of data items are transferred each and every second using a number of different applications. The confidentiality related with a large number of these applications required to be protected. Cryptography is a tool that has been widely used several years, although at its origin this was limited to diplomatic, military and intelligence services, nowadays cryptography can be used by everyone to protect the confidentiality of their information transmitted over insecure channels as well as to provide privacy, authenticity and data integrity. As the field of cryptography has advanced; today it is assumed as the study of techniques and applications that provide security regarding the integrity, privacy, authenticity and confidentiality of data transfer under complicated circumstances.

With the rapid advancement in wireless networks and peripherals, low-end devices, such as RFID (Radio Frequency Identification) tags, contactless smart cards, wireless sensor nodes etc are deployed in increasing numbers day by day. These devices are widely used in many environments and applications, which increasingly lead the requirement to provide security [1]. While selecting security algorithms for such resource-limited constrained devices, the implementation cost and complexity should be taken into account. In order to satisfy these needs, highly efficient and secure encryption schemes have to be developed. Symmetric-key algorithms, particularly lightweight block ciphers, play a significant role to provide privacy and confidentiality in these applications. Traditional cryptography focus on the solutions to provide high levels of security, ignoring the requirements of constrained devices. Lightweight cryptography (LWC) is a research field that has developed in recent years and focuses in designing schemes for devices with constrained capabilities in power supply, connectivity, hardware and software. Block ciphers are crucial primitives for cryptographic applications such as confidentiality, data integrity, and protection of privacy. At the same time, with the huge deployment of low resource devices such as sensor nodes and RFID tags and increasing need to provide security among in them, the lightweight cryptography has become a great topic. Hence, research on analyzing and designing lightweight block ciphers has received a lot of interest recently. There have been several block ciphers designed for a lightweight hardware implementation such as mCrypton, HIGHT, Piccolo, DESL/DESXL, PRESENT, KATAN/KTANTAN and PRINT cipher. The structures of these ciphers are generally classified into two structures: Feistel-type structures and Substitution Permutation Networks (SPNs).

Recently, the security of data addresses the ubiquitous mobile network environment. The USN (ubiquitous sensor network) emphasizes the availability of network access without limitation of time and place. However, the portable

environment of USN raises critical problems of limited energy sources along with protection of personal data. Some wireless systems comprise the RFID and sensors as major components in ubiquitous sensor networks [1]. However, it suffers from a security risk in its data communication. Block cipher algorithms were evaluated for these applications and they yield a size small enough to be used in such applications. The needs for high throughput of block ciphers become a more critical design concern, because of the ever-increasing communication speed and the fluent access environment. This motivates the researchers to design and implement a new architecture for block ciphers that can enhance its throughput without significant hardware complexity. Piccolo is a new lightweight block cipher which is optimized for extremely constrained devices. Piccolo supports 64-bit block with 80-bit or 128-bit keys, and has an iterative structure which is a variant of a generalized Feistel network. Piccolo achieves both high security and extremely compact implementation unlike the other Feistel-type structure based lightweight block ciphers. A new parallel architecture for Piccolo is designed that can meet the fulfillment of mandatory requirements for low-resource devices such as RFID tags and sensors. The new architecture is implemented such that it shares the key scheduling block for two plain text blocks concurrently. Matlab simulation software is used for the implementation. The new architecture of Piccolo block cipher aims to enhance the throughput and speed of execution than the conventional Piccolo block cipher architecture.

1.1 Lightweight Cryptography

Lightweight cryptography is a cryptographic algorithm or protocol used for implementation in resource-limited constrained environments including RFID tags, sensors, contactless smart cards, health-care devices etc. ISO (International Organization for Standardization)/ IEC (International Electrotechnical Commission) 29192 is a novel standardization project of lightweight cryptography. According to ISO/IEC 29192, the lightweight properties are described on the basis of target platforms. In hardware implementations, energy consumption and size of the chip are the significant way to evaluate the lightweight properties. In software implementations, RAM size and smaller code are the important measures preferable for the lightweight applications. In the implementation properties, from the point of view, lightweight cryptographic methods are superior to conventional cryptographic techniques. The main reasons for why lightweight cryptography is required are the following:

- End-to-end communication

In order to achieve protection during the end to end communication, the end nodes implement a symmetric key algorithm. Less amount of energy consumption is very essential in most of the cryptographic operations, especially in battery-powered devices. The energy consumption for low-resource devices can be reduced by the application of the lightweight symmetric key algorithm.

- Applicability to low resource devices

The path of the lightweight cryptographic primitives is small as compared to the conventional cryptographic techniques. The lightweight cryptographic primitives open a variety of possibilities in more network connections with low resource devices.

Lightweight cryptography is currently used in the Internet security protocols and it also provides sufficient security. Internet of things (IoT) forms a vast array of devices that are made accessible for computing a huge amount of information. The IoT includes embedded wireless devices and will include some familiar devices such as smart phones. Smart phones are well-resourced and can be recharged and powered readily. The IoT involves the inter-networking of many objects, which makes the data more accessible for Internet [2]. The future of IoT depends on its ability to protect hard-to-secure resource-limited devices. Researchers found that lightweight cryptography can be implemented for securing these devices also. Embedded devices have their limitations in terms of processing power, memory, energy and storage.

Traditional cryptography focus on the solutions in providing high levels of security, ignoring the requirements of constrained devices. Lightweight cryptography is a research field that has developed in recent years and focuses in designing schemes for devices with constrained capabilities in power supply, connectivity, hardware and software. Schemes proposed include hardware designs, which are typically considered more suitable for ultra constrained devices, as well as software and hybrid implementations for lightweight devices. Each and every designer of lightweight cryptography has to deal with the costs, security and performance trade-offs. For block ciphers, security-cost trade-off is provided by the key length, while the security-performance trade-off is achieved using the number of rounds and the hardware architecture gives cost-performance trade-off [3]. The lightweight cryptographic algorithm can be designed in two ways. Either it can be developed from remembering that target objects are resource constrained or the functionalities such as block size, no. of rounds etc. of the available traditional cryptographic algorithms such as DES, RSA can be optimized. For the first case, new logic needs to be designed making complexity and cost of the technique low. For the second case, there must be confidence for the existing algorithm but optimization of certain parameters in the logic may enhance complexity or may compromise security [2]. For example, the key length is reduced from 256 bits to 56 bits, the mode of architecture shifts from parallel to serialized, and the number of rounds in the encryption logic is reduced from 48 to 16. In addition, the processing speed comes down from GHz to KHz and the memory requirement is reduced from giga bytes to kilo bytes.

Lightweight cryptographic techniques can be classified as hardware-oriented and software oriented. Hardware-oriented techniques are applicable in places where main concern is about the chip size and number of clock cycles required for their execution. Such types of algorithms are more useful in smart card type objects. Estimation of power consumption for the hardware implementations are not accurate Software oriented techniques need to be considered when main focus

is on memory (RAM and ROM) requirements, required number of clock cycles, and power consumption. There are several contradictions between hardware and software implementation. The bit permutation operation will significantly slow down the software implementation, whereas in hardware the bit permutation operations are virtually free. Use of large substitution tables makes the hardware realization costly whereas in case of software, it makes the realization simpler. Another classification type is symmetric versus asymmetric. Asymmetric cryptographic techniques are more secure than symmetric one, but in cases where authentication and integrity are of primary importance, symmetric technique will work better. This lowers the power consumption and can save additional computational cost. Asymmetric ciphers are computationally challenging and more demanding, in the case of both hardware and software. However, it is the designer's choice to select the appropriate techniques, based on the requirement of the application along with the constraints carried by them [2].

1.2 Piccolo Block Cipher

Piccolo is a lightweight block cipher that has an iterative structure based on generalized Feistel network and has a block size of 64 bits. There are two versions of Piccolo: Piccolo-80 and Piccolo-128. Piccolo-80 supports 64-bit block with 80-bit key and Piccolo-128 supports 64-bit block with 128-bit key. In case of hardware, Piccolo achieves extremely compact implementation. The area required for the implementation of Piccolo-80 and Piccolo-128 are only 683 and 758 gate equivalents (GE) with 432 and 528 cycles per block, respectively. The Piccolo block cipher is based on generalized Fiestel structure with four 16-bit branches which performs a complicated permutation instead of a simple shift as well as whitening [4]. The permutation operates on 8 bit words, even though the branches of the Fiestel structure are made of 16 bits. Piccolo block cipher achieves good security and very compact implementation unlike the other Feistel-type structure based lightweight block ciphers. Both the cipher versions of Piccolo consist of a data processing part and a key scheduling part. The main difference between Piccolo-80 and Piccolo-128 lie in the number of Piccolo-80 uses 64-bit block with 80-bit keys and 25 rounds, and Piccolo-128 uses 64-bit block with 128-bit keys and 31 rounds.

Figure 1 shows the encryption function of Piccolo. In one round, the operations of Piccolo encryption include two F functions ($F : \{0, 1\}^{16} \rightarrow \{0, 1\}^{16}$, as shown in figure 2), an AddRoundKey function and a RoundPermutation function ($RP : \{0, 1\}^{64} \rightarrow \{0, 1\}^{64}$). The F function is composed of three operations: SubNibble, MixColumn, and SubNibble operation [5]. The AddRoundKey function XORs the output of F function with the round key generated from the key scheduling part. The RoundPermutation function groups the 64 bits of a block into eight bytes and permutes the bytes as shown in figure 3. The F function has 16-bit input, which are grouped as four nibbles and produces a 16-bit output. In the SubNibble operation, each nibble is passed through an S-box. The MixColumn views the four nibbles as a 4×1 vector and then multiplies it with a 4 × 4 matrix M. The structure of M is as shown below.

$$M = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix}$$

The four nibbles generated by the MixColumn is now passed through the SubNibble operation again and this become the output of the F function.

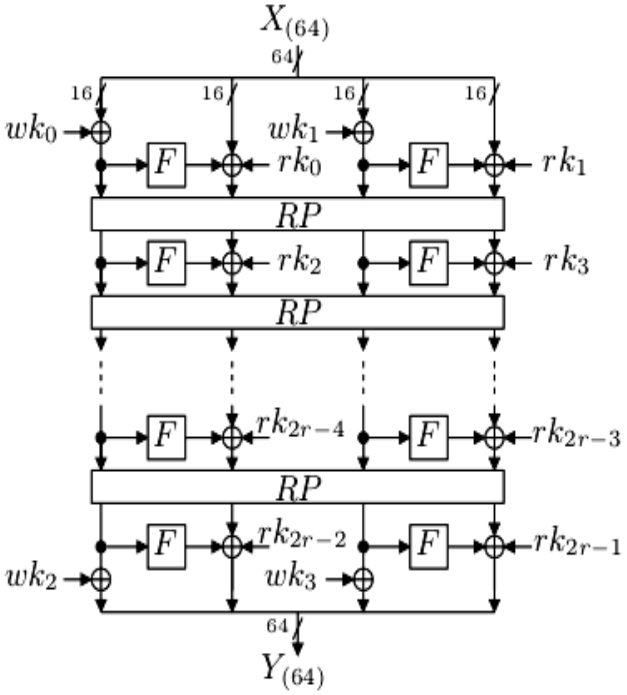

Fig-1: Encryption function of Piccolo [4]

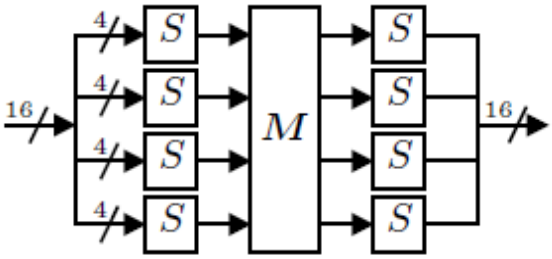

Fig-2: F-function [4]

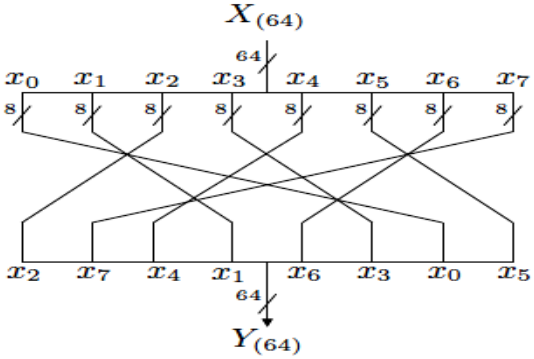

Fig-3: Round Permutation [4]

- Key Scheduling Part

The key scheduling part of Piccolo block cipher generates 16-bit whitening keys $wk_{i(16)}$ $(0 \le i < 4)$ and round keys $rk_{j(16)}$ $(0 \le j < 2r)$ which are used for data processing. The key scheduling functions for the two variants of Piccolo, Piccolo-80 and Piccolo-128 are referred to as KS_r^{80} and KS_r^{128}, respectively. In order to describe each schedule, the 16-bit constants con_i^{80} and con_i^{128}, have to be defined first. The new Piccolo architecture implemented is Piccolo-128 version and hence only the key schedule for 128-bit key mode (KS_r^{128}) is required here. The key scheduling function for the 128-bit key mode, KS_r^{128}, divides a 128-bit key $K_{(128)}$ into eight 16-bit sub-keys $k_{i(16)}$ $(0 \le i < 8)$ and provides $wk_{i(16)}$ $(0 \le i < 4)$ and $rk_{j(16)}$ $(0 \le j < 2r)$. In Piccolo block cipher, pre-whitening and post-whitening are done before the first round and after the last round, respectively. In the whitening key generation, the 64-bit block is split into two halves and the left 16 bits of each half are XORed with the pre-whitening key or post-whitening key. The key scheduling of Piccolo block cipher produces four 16-bit whitening keys wk_i $(0 \le i < 4)$ and two 16-bit round keys for each round. These keys are generated by XORing the master key with 16-bit constants.

- Data Processing Part

The data processing part of Piccolo consists of r rounds and it takes a 64-bit data $X \in \{0, 1\}^{64}$, four 16-bit whitening keys $wk_i \in \{0, 1\}^{16} (0 \le i < 4)$ and 2r 16-bit round keys $rk_i \in \{0, 1\}^{16} (0 \le i < 2r)$ as the inputs, and outputs a 64-bit data $Y \in \{0, 1\}^{64}$. The operations of F-function and round permutations are performed under the data processing part. F function is called the non-linear part of Piccolo block cipher [6]. The F-function in Piccolo block cipher is defined as $F : \{0, 1\}^{16} \rightarrow \{0, 1\}^{16}$. It consists of two S-box layers which are separated by a diffusion matrix 'M'. The round permutation in Piccolo block cipher is denoted as $RP : \{0, 1\}^{64} \rightarrow \{0, 1\}^{64}$. It divides a 64-bit input $X_{(64)}$ into eight 8-bit data as $X_{(64)} = x_{0(8)}|x_{1(8)}|...|x_{7(8)}$. It then permutes and concatenates $(x_{0(8)}, x_{1(8)}, ... x_{7(8)})$ into a 64-bit data $X_{(64)}$, which is as shown in figure 3.

2. PROPOSED ARCHITECTURE

The architecture of Piccolo block cipher is a variant of generalized fiestel network which has light round functions and can easily support decryption function without much cost of implementation. By adopting a permutation based key scheduling method, the required number of gates can significantly reduced. Using the effective permutation based key scheduling method and the half-word based round permutation, Piccolo efficiently does the key expanding and avoids several known attacks.

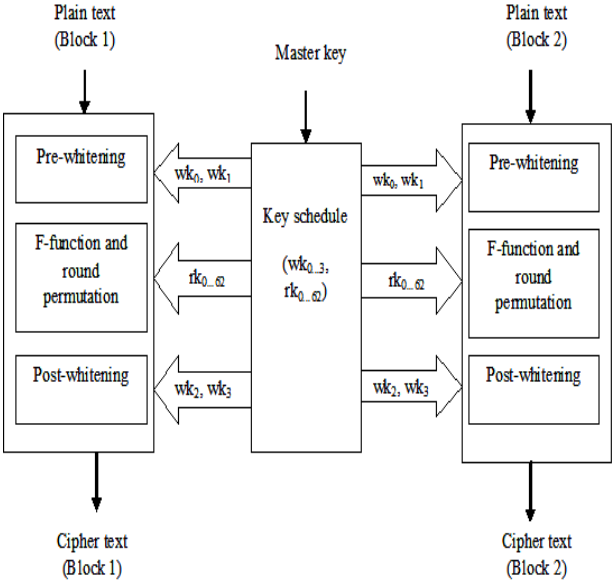

Fig-4: Novel architecture for Piccolo

Parallelism can provide high speed computation and can be employed to accelerate the encryption and decryption process. Such architectures offer a large amount of concurrency to enhance the throughput [7]. An efficient implementation for Piccolo block cipher architecture can meet the fulfilment of mandatory requirements for low-resource devices such as RFID tags and sensors. The new architecture is designed such that it shares the key scheduling block for two different blocks of the plain text concurrently. The design for the novel architecture of Piccolo block cipher is shown in figure 4. The main advantage with this proposed system is that by encrypting two blocks of the plain text concurrently, the speed of execution is increased and it enhances the system throughput. In the proposed system, the encryption time, decryption time and throughput are the performance metrics measured for the comparison of conventional Piccolo and New Piccolo. In this work, the throughput is calculated separately for encryption and decryption. The throughput of the encryption is calculated as the total plaintext in bytes encrypted divided by the encryption time whereas the throughput of the decryption is calculated as the total cipher text in bytes decrypted divided by the decryption time.

3. RESULTS AND DISCUSSIONS

The designed novel architecture for Piccolo is implemented and compared with the conventional Piccolo block cipher architecture. In the proposed system, the performance metrics measured are encryption time, decryption time and throughput. The encryption time is defined as the time the encryption algorithm takes to produce a cipher text from a plain text. Encryption time is used to calculate the throughput of an encryption scheme that indicates the speed of encryption. The throughput of the encryption scheme is calculated as the total plaintext in bytes encrypted divided by the encryption time. The decryption time is defined as the time the decryption algorithm takes to produce plain text

from the cipher text. Decryption time is used to calculate the throughput of a decryption scheme that indicates the speed of decryption. The throughput of the decryption scheme is calculated as the total cipher text in bytes decrypted divided by the decryption time [8, 9]. The throughput of the encryption scheme is calculated using the following formula [9].

$$Throughput = \frac{Tp}{Et}$$

where *Tp* is the Total Plain text and *Et* is the Encryption time. Similarly, the throughput of the decryption scheme is calculated using the following formula.

$$Throughput = \frac{Tc}{Dt}$$

where *Tc* is the Total Cipher text and *Dt* is the Decryption time.

Along with the encryption and decryption output of conventional Piccolo and that of new architecture of Piccolo, the execution time and throughput of encryption and decryption of both the conventional Piccolo and new architecture of Piccolo are plotted. The plotted output is shown in the following figures. Figure 5 and 6 represents the encryption time and decryption time respectively, for the conventional Piccolo and new Piccolo block cipher.

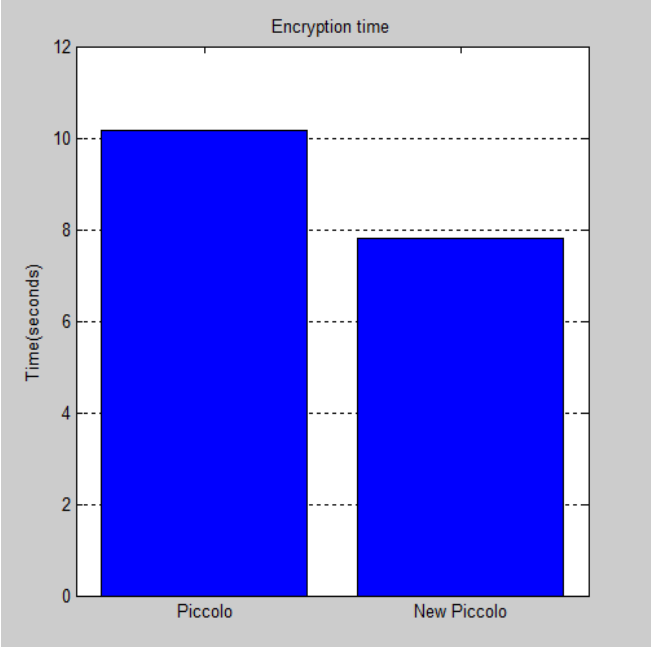

Fig-5: Encryption time

Figure 7 and 8 represents the throughput for encryption and decryption respectively, for the conventional Piccolo and new Piccolo block cipher architecture. The results are analyzed for inputs varying in different sizes and the encryption throughput is calculated for conventional Piccolo and new Piccolo. Similarly, the decryption throughput is also calculated for the conventional Piccolo and new Piccolo.

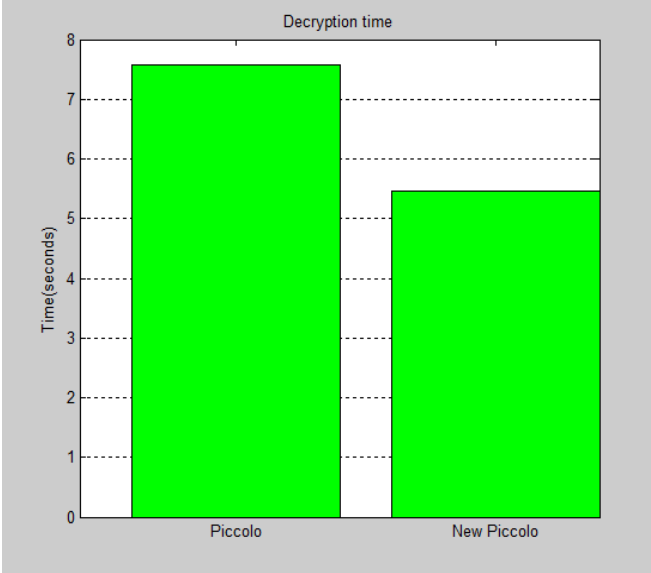

Fig-6: Decryption time

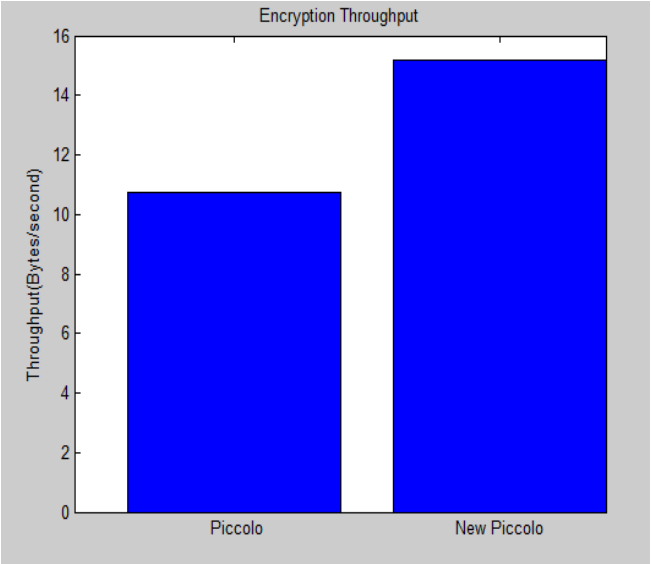

Fig-7: Throughput (Encryption)

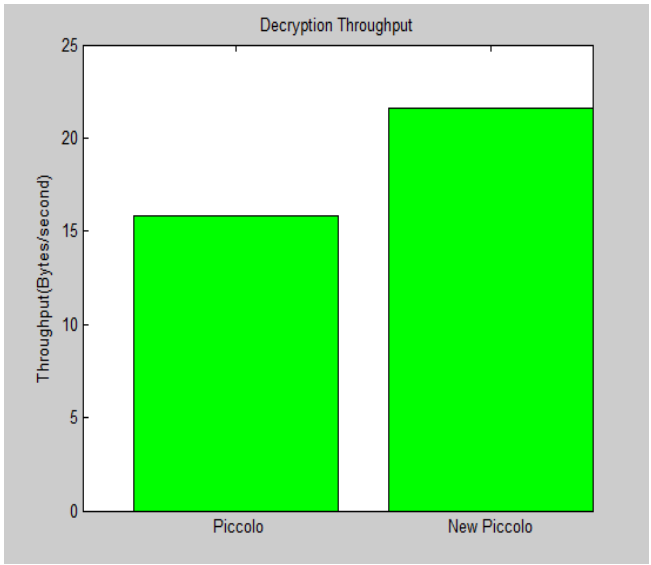

Fig-8: Throughput (Decryption)

Table-1: Results based on encryption throughput for different data sizes

Data size (Bytes)	Encryption Throughput (Bytes/second)	
	Piccolo	New piccolo
50	8.9	9.3
100	12.91	13.78
200	14.75	18.24
300	14.89	19.6
400	15.32	20.45
500	15.28	20.50
600	15.59	21.35
700	15.63	21.17
800	15.67	21.37
900	15.62	21.49
1000	15.85	21.54

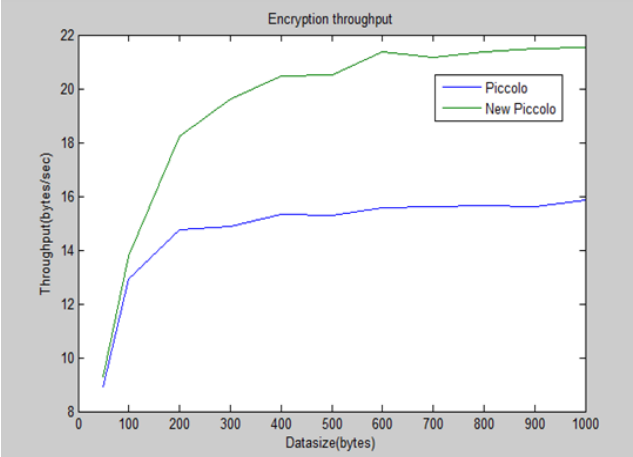

Fig-9: Encryption throughput

Table-2: Results based on decryption throughput for different data sizes

Data size (Bytes)	Decryption Throughput (Bytes/second)	
	Piccolo	New piccolo
50	15.01	18.17
100	15.38	19.08
200	15.60	20.48
300	15.96	21.43
400	15.98	22.26
500	16.00	22.05
600	15.83	22.52
700	15.99	22.64
800	15.96	22.72
900	16.04	22.76
1000	16.06	22.81

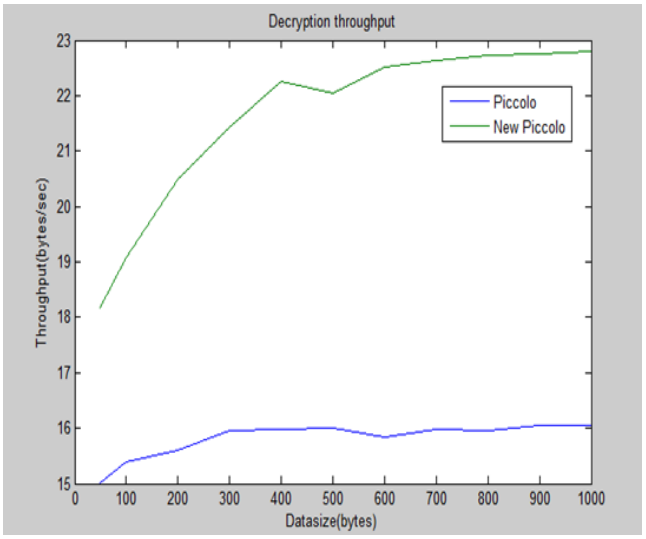

Fig-10: Decryption throughput

The table 1 shows the results of encryption throughput for conventional Piccolo and new Piccolo based on different input data sizes and table 2 shows the results of decryption throughput for conventional Piccolo and new Piccolo based on different input data sizes.

By observing these results, it is found that as size of the input data increases, the encryption throughput is increasing for both conventional Piccolo and new architecture for Piccolo. Similarly, the decryption throughput is also increasing for both conventional Piccolo and new architecture for Piccolo as the size of input data is increased. Using this data, a graph is plotted for showing the variations in throughput for different input data sizes.

Figure 9 and figure 10 show the graph plotted showing the throughput variations of conventional Piccolo and new Piccolo for encryption and decryption respectively. From these figures, it is clear that even though if the throughput is increased for increasing data sizes, the throughput is greatly enhanced for the new architecture for Piccolo block cipher as compared to conventional Piccolo block cipher. As a result, the execution time as well as the throughput of the novel architecture for Piccolo is enhanced than its conventional Piccolo architecture.

4. CONCLUSION

A high speed algorithm with security is always essential for both wired and wireless environments. Lightweight cryptography is a cryptographic algorithm used for the implementation in resource limited, constrained environments including RFID tags, contactless smart cards, and sensors. Lightweight block ciphers based on symmetric key algorithms play an important role to provide security in such applications. Piccolo is one of the competitive ultra-lightweight block ciphers which are suitable for extremely constrained environments. The designed novel architecture for Piccolo is implemented and compared with the conventional Piccolo block cipher architecture for inputs of different data sizes. From the results, it is obtained that the

execution time of the new architecture for Piccolo is reduced as compared to the conventional Piccolo architecture. It is observed that as the size of the input data increases, the throughput is increased for encryption as well as decryption. It is also found that the throughput of the novel architecture for Piccolo is enhanced by about of 25% to 38%, than its conventional Piccolo architecture. This efficient implementation for Piccolo block cipher architecture fulfils the basic requirements like speed and throughput for the extremely constrained, low-resource devices.

REFERENCES

[1]. Hai Liu, Miodrag Bolic and Amiya Nayak, "Taxonomy and Challenges of the Integration of RFID and Wireless Sensor Networks", *IEEE Networks*, vol. 8, pp. 26-32, Nov/Dec 2008.

[2]. Tapalina Bhattasali, "LICRYPT: Lightweight Cryptography Technique for Securing Smart Objects in Internet of Things Environment", *CSI Communications*, May 2013.

A. Poschmann, "Lightweight Cryptography – Cryptographic Engineering for a Pervasive World." *IACR ePrint archive 2009/516*, 2009.

[3]. K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, and T. Shirai, "Piccolo: An ultra-lightweight block cipher", *Cryptographic Hardware and Embedded Systems-CHES 2011, Lecture Notes in Computer Science 6917*, Springer, Berlin, 2011, pp. 342–357.

[4]. Fan ZHANG, Xinjie ZHAO, et.al, "Improved Algebraic Fault Analysis: A Case Study on Piccolo and with Applications to other Lightweight Block Ciphers", *4th International Workshop, Constructive Side-Channel Analysis and Secure Design*, Volume 7864, pp 62-79, 2013.

[5]. Seyyed Arash Azimi, Zahra Ahmadian, "Impossible Differential Cryptanalysis of Piccolo Lightweight Block Cipher*", IEEE*, vol. 14, pp no. 89-94, 2014.

[6]. A Desai, K. Ankalgi, "Parallelization of AES algorithm for Disk Encryption Using CBC and ICBC modes", *IEEE 4th ICCCNT 2013*, July 2013.

[7]. S. Pavithra, E. Ramadevi, "Throughput Analysis of Symmetric Algorithms", *Proceedings On IJANA*, Vol. 4, No. 2, pp. 1574-1577, 2012.

[8]. M. Anand Kumar and Dr.S.Karthikeyan, "Investigating the Efficiency of Blowfish and Rejindael (AES) Algorithms", *Proceedings on IJCNIS*, Vol. 2, pp. 22-28, 2012.

BIOGRAPHIES

Rahiyanath T. Y. received the B.Tech degree in Electronics and Communication Engineering in 2012 under Cochin University of Science and Technology, is at final stage of M.Tech in Communication Engineering at KMEA Engineering College, Mahatma Gandhi University, Kerala.