Performance Evaluation of Trivium on Raspberry Pi

Ari Kusyanti[1], Yazid Samanhudi[4]
Department of Information Technology
Universitas Brawijaya
Malang, Indonesia

Syahifudin Shahid[2]
Department of Computer System
Universitas Brawijaya
Malang, Indonesia

Harin Puspa Ayu Catherina[3]
Department of Information System
Universitas Brawijaya
Malang, Indonesia

***Abstract*—High connectivity of billions of IoT devices lead to many security issues. Trivium is designed for IoT to overcome the security challenges of IoT. The objective of this study is to implement a security service to provide confidentiality for the communication of IoT devices. Furthermore, this study aims to analyze Trivium performance in terms of keystream generation time and memory utilization on Raspberry Pi Zero, Raspberry Pi 2B, and Raspberry Pi 3B. The result showed that there was a statistically significant difference between the keystream generation time and memory utilization on Raspberry Pi Zero, Raspberry Pi 2B, and Raspberry Pi 3B based on Kruskal-Wallis H test. Further test of Jonckheere-Terpstra indicates that the fastest keystream generation time was on Raspberry Pi 3B, and the smallest memory utilization was on Raspberry Pi 2B. The implemantation of Trivium on three versions of Raspberry Pi shows promising results with less than 27 MB of memory utilization for cryptography leaves more resources available to applications.**

Keywords—Trivium; Raspberry; Kruskal-Wallis

I. Introduction

The Internet of Things (IoT) being a promising technology that enables people and objects in the physical world also numerous devices and sensors will be communicating with each other. The increased volume of communication with a huge number of nodes has the potential leading to serious security challenges. One node of IoT is an important sources of information and also a sensitive information that need to be protected, while IoT is vulnerable to attacks [1], [2], such as message modification and/or alteration, traffic analysis, Denial of Service (DoS), Distributed DoS, eavesdropping, Sybil attacks, etc. According to a study by HP states that 70% of the devices in IoT are vulnerable to attacks [3].

Various cryptographic algorithms have been researched to overcome the problem. Trivium is a stream cipher designed by Christophe De Canniere and Bart Preneel who participated in the eSTREAM competition and has been selected as part of a portfolio for hardware-oriented ciphers [4]. Based on eSTREAM, NIST recommends Trivium as one of the standard for lightweight cryptography.

Some studies evaluated Trivium and other lightweight cryptographic algorithms performance as in [5], [6] and [7]. The hardware implementation of eSTREAM ciphers have been implemented on FPGA devices [8] or 8-bit AVR microcontrollers [9] and on NodeMCU [10].

The objective of this study is to implement Trivium on three versions of Raspberry Pi as an embedded device concept of IoT to provide security services and compare the results among them. To evaluate the performance, the memory utilization and keystream generation time of Trivium algorithm is observed. To analyze the data, The Kruskal-Wallis H test is used to examine the difference implementation of Trivium on three Raspberry Pi versions.

This paper is structured as follows. Section 2 explains Trivium algorithm and the target devices, i.e Raspberry Pi Zero, Raspberry Pi 2B and Raspberry Pi 3B. Section 3 describes an explanation related to the implementation of Trivium on Raspberry Pi and the result analysis. Finally, conclusions are drawn in Section 4.

II. Trivium Architecture and Raspberry Pi

This section will disscuss the overview of Trivium and the target devices i.e. Raspberry Pi Zero, Raspberry Pi 2B and Raspberry Pi 3B, also the test to analyze the data namely Kruskall-Walis H test.

A. Trivium Architecture

In 2005 Christophe De Canni`ere and Bart Preneel developed a stream cipher called Trivium [11]. Trivium is a synchronous stream cipher that takes a secret key and an Initial Value (IV) as inputs and produces keystream that is used for encryption process. It is designed for constrained devices to generate up to bits of keystream from an 80-bit secret key and an 80-bit IV. The state consists of 288 bits which are denoted as s_0, s_2,....,s_{287} and depicted in Figure 1. There are two phases in Trivium process i.e. phase internal initialization state by using the key and IV.

The initialization process is performed by loading 80 bits of key and 80 bits of IV into the initial state of 288-bit and the rest are padded with 0, except for bit S_{286}, S_{287}, S_{288}. Then it is rotated four times, a process that is done similar to the key generation phase, but without generating a bit key stream. A complete explanation is given by the following pseudo-code:

$$(s_1, s_2, \ldots, s_{93}) \leftarrow (K_1, \ldots, K_{80}, 0, \ldots, 0)$$
$$(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (IV_1, \ldots, IV_{80}, 0, \ldots, 0)$$
$$(s_{178}, s_{279}, \ldots, s_{288}) \leftarrow (0, \ldots, 0, 1, 1, 1)$$

for i = 1 to 4 · 288 do

$$t_1 \leftarrow s_{66} + s_{91} \cdot s_{92} + s_{93} + s_{171}$$

$t_2 \leftarrow s_{162} + s_{175} \cdot s_{176} + s_{177} + s_{264}$
$t_3 \leftarrow s_{243} + s_{286} \cdot s_{287} + s_{288} + s_{69}$
$(s_1, s_2, \ldots, s_{93}) \leftarrow (t_3, s_1, \ldots, s_{92})$
$(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (t_1, s_{94}, \ldots, s_{176})$
$(s_{178}, s_{279}, \ldots, s_{288}) \leftarrow (t_2, s_{178}, \ldots, s_{287})$
end for

Keystream generation consists of an iteration process by extracting values from 15 specific bits and using them to update 3 bits andproducing keystream z_i. Full description according to pseudocode below:

for i = 1 to N do
$t_1 \leftarrow s_{66} + s_{93}$
$t_2 \leftarrow s_{162} + s_{177}$
$t_3 \leftarrow s_{243} + s_{288}$
$z_i \leftarrow t_1 + t_2 + t_3$
$t_1 \leftarrow t_1 + s_{91} \cdot s_{92} + s_{171}$
$t_2 \leftarrow t_2 + s_{175} \cdot s_{176} + s_{264}$
$t_3 \leftarrow t_3 + s_{286} \cdot s_{287} + s_{69}$
$(s_1, s_2, \ldots, s_{93}) \leftarrow (t_3, s_1, \ldots, s_{92})$
$(s_{94}, s_{95}, \ldots, s_{177}) \leftarrow (t_1, s_{94}, \ldots, s_{176})$
$(s_{178}, s_{279}, \ldots, s_{288}) \leftarrow (t_2, s_{178}, \ldots, s_{287})$
end for

B. Raspberry Pi

Raspberry Pi is a single-board, low-cost and high-performance developed in the UK by the Raspberry Pi Foundation that can also be used for embedded systems [10]. There are a variety of Raspberry Pi models with different features and improvements in their latest versions that add more components. In this paper, three types of Raspberry are used, i.e. Raspberry Pi Zero, Raspberry 2B, and Raspberry 3B. Table I details the comparison among Raspberry Pi.

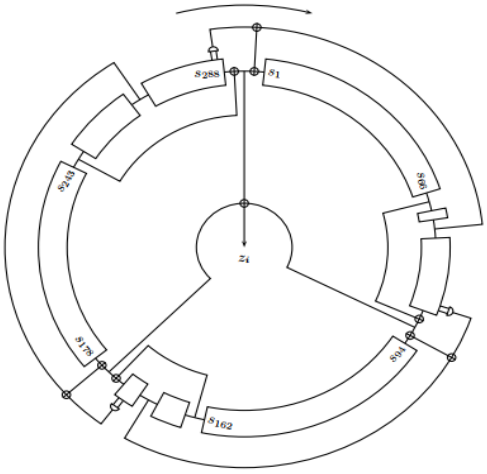

Fig. 1. Trivium Architecture.

TABLE I. RASPBERRY PI SPESIFICATION

	Rasp Pi Zero [12]	Rasp Pi 2B [13]	Rasp Pi 3B[14]
CPU	1GHz ARMI1 Core	An ARM Cortex-A7 Quad-Core 900MHz CPU	An ARM Cortex-A7 Quad-Core 900MHz CPU
RAM	512 MB	1 GB	1 GB

C. The Kruskal-Wallis H Test

The Kruskal-Wallis H test is a test that can be utilized to analyze if there are statistically significant differences between two or more categories of an independent variable [15]. Kruskal-Wallis is used to determine the significance (p-value) also the medians or mean ranks of three or more categories. This test is a non-parametric test in which the data within each of the multiple categories do not need to follow a normal distribution.

III. IMPLEMENTATION

Trivium algorithm is tailored for IoT to address the security challenges. The vital process in Trivium, as a symmetric key algorithm, is the generation of keystream. The keystream generation process involves complex mathematical operations. To examine the performance, Trivium is evaluated on the basis of memory utilization and keystream generation time. To measure the memory utilization and the keysteram generation time of the Trivium, each of Raspberry Pi is configured in single mode. The memory utilization and keystream generation time of Trivium algorithm are observed and compared among three types of Raspberry Pi.

The experiment run 40 times for each Raspberry Pi with various combination of Key and IV as inputs which generates keystream. The average of keystream generation time was 3391.275 microsecond for Raspberry Pi Zero, 1426.8 microsecond for Raspberry Pi 2B, and 1051.775 microsecond for Raspberry Pi 3B. In the meantime, memory utilization was 26.75 MB for Raspberry Pi Zero, 20.45 MBfor Raspberry Pi Zero and 24.00 MB for Raspberry Pi Zero. With less than 27 MB of memory utilization indicated that with fewer resources for cryptography leaves more resources available to applications.

The collected data were then analyzed by using Kruskal-Wallis H test to determine whether there is statistically significant differences of memory utilization and keystream generation time among Raspberry Pi Zero, Raspeberry Pi 2B, and Raspberry Pi 3B. Kruskall-Wallis H test was used since the data were not normally distributed. In Kruskall-Wallis H test, the p-value for the chi-square approximation test is reasonably accurate when the number of experiments is greater than or equal to 30 [16].

The result of Kruskal-Wallis H test is presented in Table II to V.

TABLE II. KRUSKAL-WALLIS H TEST RESULT OF KEY GENERATION TIME

Ranks	Raspberry Pi	N	Mean Rank
Keystream Generation Time	Raspberry Pi Zero	40	99.40
	Raspberry Pi 2 B	40	58.38
	Raspberry Pi 3 B	40	23.73
	Total	120	

TABLE III. TEST STATISTICS RESULT OF KEYSTREAM GENERATION TIME

Test Statistics	Keystream Generation Time
Chi-Square ($\chi 2$)	94.886
Asymp.Sig.	.000

TABLE IV. KRUSKAL-WALLIS H TEST RESULT OF KEY MEMORY UTILIZATION

Ranks	Raspberry Pi	N	Mean Rank
Memory Utilization	Raspberry Pi Zero	40	100.50
	Raspberry Pi 2 B	40	20.50
	Raspberry Pi 3 B	40	60.50
	Total	120	

TABLE V. TEST STATISTICS RESULT OF MEMORY UTILIZATION

Test Statistics	Memory Utilization
Chi-Square ($\chi 2$)	112.848
Asymp.Sig.	.000

Based on Table II and Table III, there was a statistically significant difference between the keystream generation time ($\chi 2$= 94.886, p = 0.000), with a mean rank of 99.40 for Raspberry Pi Zero, 58.38 for Raspberry Pi 2B and 23.73 for Raspberry Pi 3B.Whilst for the memory utilization, according to Table IV and Table V, there was a statistically significant difference between memory utilization ($\chi 2$= = 112.848, p = 0.000), with a mean rank of 100.50 for Raspberry Pi Zero, 20.50 for Raspberry Pi 2B and 60.50 for Raspberry Pi 3B. For further analysis, if the compared groups are ordered in a certain way, Jonckheere-Terpstra Test is conducted.

Table VI and VII presents the result of the Jonckeere-Terpstra test.

Based on Table VI, Jonckheere-Terpstra test for ordered alternatives showed that there was a statistically significant trend of keystream generation time of Raspberry Pi Zero, Raspberry Pi 2B, and Raspberry Pi 3B T_{JT} = 133.00, p < 0.0005. Or by way of the fastest keystream generation time was on Raspberry Pi 3B. As for the memory utilization, Jonckheere-Terpstra indicated that there was a statistically significant trend of memory utilization of Raspberry Pi Zero, Raspberry Pi 2B, and Raspberry Pi 3B T_{JT} = 1600.00, p < 0.0005. Or in other word, the largest memory utilization was on Raspberry Pi Zero.

TABLE VI. JONCKHEERE-TERPSTRA TEST RESULT OF KEYSTREAM GENERATION

Jonckheere-Terpstra Test	Key GenerationTime
Number of Levels in Raspberry Pi	3
N	120
Observed J-T Statistic	133.000
Asymp. Sig. (2-tailed)	.000

TABLE VII. JONCKHEERE-TERPSTRA TEST RESULT OF MEMORY UTILIZATION

Jonckheere-Terpstra Test	Memory Utilization
Number of Levels in Raspberry Pi	3
N	120
Observed J-T Statistic	1600.000
Asymp. Sig. (2-tailed)	.000

IV. CONCLUSION

Trivium is designed for IoT tohandle the security issues of IoT. The implemantation of Trivium on Raspberry Pi Zero, Raspberry Pi 2B, and Raspberry Pi 3B presents promising results that it is suitable candidate to be adopted in IoT applications as with less than 27 MB of memory utilization for cryptography leaves more resources available to applications. The results of memory utilization and keystream generation time were then compared among three types of Raspberry Pi. The result showed that there was a statistically significant difference between the keystream generation time on Raspberry Pi Zero, Raspberry Pi 2B, and Raspberry Pi 3B, and the fastest keystream generation time was on Raspberry 3B. While for memory utilization, the result presented that there was a statistically significant difference between memory utilization on Raspberry Pi Zero, Raspberry Pi 2B, and Raspberry Pi 3B, and the smallest memory utilization was on Raspberry Pi 2B.

REFERENCES

[1] M. A. Simplicio Jr, M. V. Silva, R. C. Alves, and T. K. Shibata, "Lightweight and escrow-less authenticated key agreement for the internet of things," Computer Communications, 2016.

[2] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," Computer networks, vol. 54, no. 15, pp. 2787–2805, 2010

[3] S. A. Kumar, T. Vealey, and H. Srivastava, "Security in internet of things: Challenges, solutions and future directions," in 2016 49th Hawaii International Conference on System Sciences (HICSS). IEEE, 2016, pp. 5772–5781.

[4] C. De Canni`ere & B. Preneel. TRIVIUM A Stream Cipher Construction Inspired by Block Cipher Design Principles. In Workshop on Stream Ciphers Revisited (SASC2006), 2006.

[5] T. Good and M. Benaissa, "ASIC hardware performance," New Stream Cipher Designs: The eSTREAM Finalists, LNCS 4986, pp. 267–293, 2008.

[6] Robshaw, M., Billet, O. (eds.): New Stream Cipher Designs. The eSTREAM Finalists. Springer (2008)

[7] Mora-Gutiérrez J.M., Jiménez-Fernández C.J., Valencia-Barrero M. (2013) Low Power Implementation of Trivium Stream Cipher. In: Ayala J.L., Shang D., Yakovlev A. (eds) Integrated Circuit and System Design.

Power and Timing Modeling, Optimization and Simulation. PATMOS 2012. Lecture Notes in Computer Science, vol 7606. Springer, Berlin, Heidelberg

[8] Levent Ertaul, Arnold Woodall. "IoT Security: Performance Evaluation of Grain, MICKEY, and Trivium - Lightweight Stream Ciphers" on Int'l Conf. Security and Manaement, SAM 2017.

[9] A. Jafarpour, A. Mahdlo, A. Akbari and K. Kianfar, "Grain and Trivium Ciphers Implementation Algorithm in FPGA Chip andAVR Micro Controller," in ICCAIE, 2011.

[10] G. Meiser, T. Eisenbarth, K. Lemke-Rust and C. Paar, "Efficient Implementation of eSTREAM Ciphers on 8-bit AVR Microcontrollers," Industrial Embedded Systems, 2008.

[11] C. De Canni`ere & B. Preneel. TRIVIUM Specifications. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/001, 2005. http://www.ecrypt.eu.org/stream.

[12] Raspberry. Raspberry Pi 1 B+. raspberry.org.

[13] Raspberry. Raspberry Pi 2 B. raspberry.org.

[14] Raspberry. Raspberry Pi 3 B. raspberry.org

[15] Kathleen F. Weaver Vanessa Morales Sarah L. Dunn Kanya Godde Pablo F. Weaver. An Introduction to Statistical Analysis in Research: With Applications in the Biological and Life Sciences. Wiley Book, 2017.

[16] Green, S. B., & Salkind, N. J. Using SPSS for Window and Macintosh: Analyzing and Understanding data (5th ed.). Upper Saddle River, NJ: Pearson Prentice Hall. 2008.