**Bulletin of Electrical Engineering and Informatics**
Vol. 10, No. 2, April 2021, pp. 1035~1045
ISSN: 2302-9285, DOI: 10.11591/eei.v10i2.2741 

# Rasefiberry: Secure and efficient Raspberry-Pi based gateway for smarthome IoT architecture

**Vincent Simadiputra, Nico Surantha**
Department Computer Science, Binus Graduate Program-Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia

**Article Info**

***Article history:***

Received Aug 11, 2020
Revised Feb 6, 2021
Accepted Feb 25, 2021

***Keywords:***

Encryption
Intrusion detection system
IoT gateway
Raspberry pi

**ABSTRACT**

Internet-of-Things or IoT technology becomes essential in everyday lives. The risk of security and privacy towards IoT devices, especially smarthomes IoT gateway device, becoming apparent as IoT technology progressed. The need for affordable, secure smarthome gateway device or router that smarthome user prefer. The problem of low-performance smarthome gateways was running security programs on top of smarthome gateway programs. This problem motivates the researcher designing a secure and efficient smarthome gateway using Raspberry Pi hardware as an affordable smarthome gateway device and able to run both smarthome gateways and security programs. In this research, researchers implemented snort as intrusion detection system (IDS), openHab as IoT gateway applications, and well-known encryption algorithms for file encryption in Raspberry PI 3B+ model. The researcher evaluated Snort capability on network attacks and compared each of the well-known encryption algorithm efficiency. From the result, we found Rasefiberry customized snort configuration for Raspberry pi 60 percent of the simulated network attacks. Twofish encryption algorithms were found to have best encryption time, throughput, and power consumption compared to other encryption algorithms in the research. Rasefiberry architecture successfully processes both lightweight security programs and Openhab smarthome gateway programs with a low-performance computing device such as Raspberry Pi.



***Corresponding Author:***

Nico Surantha
Department Computer Science
Binus Graduate Program-Master of Computer Science
Bina Nusantara University, Jakarta, Indonesia
Email: nico.surantha@binus.ac.id

## 1. INTRODUCTION

Technology and the internet are becoming an integral part of everyone's lives; they support a broad age range from children to the elderly. As both technology and the Internet growing, different kinds of applications that combine both technology and the Internet will become an essential part of improving people's life. IoT or Internet of things is a concept of devices connected through a network. Also, those IoT devices can be accessed and controlled remotely through the Internet. It provides user ease of access to devices integrated into a connected network. There are different kinds of IoT architectures model can be build based on the needs and targeted users; one of that architecture is smarthomes. Smarthome is a combination of multiple home IoT devices that automate basic homes function and apply new security vector that can be monitored through the Internet. Smarthomes were mainly targeted for older people to monitor their health using IoT sensors that can record medical data such as blood pressure, blood sugar level, heart

rate, and other health parameters. Now smarthomes goal is to increase the quality of life of its residents and reduce the operating cost of the home. Based on the statistic website, there were more than 15 billion IoT devices connected through the network in 2015 [1]. In the year 2020, it was expected to be double the amount of numbers exceeding 30 billion IoT devices. The overall IoT devices market will worth more than 1 billion US dollars per year.

As IoT technology evolves and progress, the threat of malicious network attack or criminal act also becoming more frequent. Since IoT devices are connected through a network that is the Internet, important data or information will be sent through the Internet. The possibility of a cracker (criminal hacker) trying to steal that information transmitted from both IoT gateway to IoT devices and IoT gateway to the Internet by exploiting gateway device vulnerability, DDOS (distributed denial of service) attack, phishing, and newer method of attack as IoT technology develops over time. Other than network attacks, there also a risk of IoT devices left at home will be stolen physically by thieves since IoT device can cost monetary loss. Privacy and security in the real world and the Internet have become an important aspect of securing and improving quality of life. There are several requirements for secure IoT from malicious attacks such as resilience to attack, data authentication, access control, and client privacy. Resilience to attack is a requirement for IoT devices should be automatically recovered from crashes during data transfer. Data authentication is the requirement for data and information must be authenticated and allow data only can be accessed by authenticated IoT devices. Access control is a requirement when the authorized user must access an IoT device. Client privacy is a requirement where private data of client or IoT devices cannot be access other than the owner or authorized user [2].

These issues arise mostly caused by smarthome service providers underestimated the need for security in smarthome devices. Service providers reasoned few attackers are motivated to perform an attack on smarthome owners, and smart homeowners preferred cheaper devices rather than more expensive secure devices [3]. Also, smarthome service provider tends to provide short support maintenance and end of life policy on their devices which lead to those devices vulnerable to attacks since it is already no supported or there is no security or maintenance patch. Not only that, the smart homeowners lacking awareness of how their private data could be stolen, and smarthome service provider could not keep its consumer data privacy in smart homes.

There are multiple approaches and methods to secure private data in IoT, such as intrusion detection system or IDS (low-cost flow-based, zero-knowledge proof, and snort), data encryption (DTLS, STI, AES, and ECDH), network traffic monitor, and vulnerability patch manager. All these methods and approaches from existing IoT architectures are installed in a custom smarthome hub or router provided by a smarthome service provider or hardware supplier. Security approaches to be installed to a smarthome hub or route would be expensive. There is a preference of smarthome users to purchase affordable smarthome gateway devices or router but not applied any security applications. To provide a secure also affordable and easily accessible smart gateway, this thesis will develop a Raspberry Pi-based smarthome gateway with secure applications and an energy-efficient encryption algorithm.

IoT or Internet of things is a network consisting of physical devices integrated through the Internet and identified over IP (internet protocol) in an application. It allows the user to be connected with IoT devices anytime, anyplace, anything, and anyone through any network and services [4]. IoT concept was firstly introduced in 1999 by Kevin Astthon when wireless devices are popular, and RFID (radio frequency identification) [2], wireless sensor network, and cloud computing are also introduced that allow IoT to develop until now [5]. IoT creates an opportunity for industrial systems and applications development, especially in the IoT integrated transportation system. IoT integrated transportation system allows the user to monitor vehicle location, movement, road traffic, and predicting its destination [6]. IoT intelligent transportation system is advanced applications that goal is to provide innovative services related to different modes of transport and traffic management, also enable its user to be informed of safer, coordinated, and smarter use of transport networks. The combination of smarthome devices and medical diagnostics sensors creates a new type of IoT called IoMT (internet of medical things). IoMT is defined as medical devices connectivity to a healthcare system through an online network such as the cloud. IoMT is mostly used as remote chronic illnesses patient monitor, medication orders trackers, and wearable mobile health devices [7].

Smart Home was a residence or houses that contain a network of interconnected devices, enabled them to be controlled, monitored, accessed from a specific network. Smarthomes, as an automated living residence, can provide their users or residents with proactive services. There are some unique characteristics of smarthomes environments such as ubiquity, invisibility, and sensing [4]. Ubiquity shows infrastructures of smarthome devices will be available everywhere and will be important in our life. Invisibility means smarthome devices are not invisible physically, but those devices will be smaller as technology progress, and smart home users will be unaware of their smart home devices. Sensing means smart home sensors can cover a wide range of activities with modern mini and multi-purpose sensor devices. Smart homes security threats

are not only limited to network attack but also physical attack/tampering [8]. A smart door can be cracked by an unauthorized party or thieves with malicious code to open them without breaking the door physically. It can cause smart homeowner loss of property. Another scenario in which trespasser can tamper with IoT devices or add unknown devices to eavesdrop information from the IoT devices.

Smarthomes sensors are used to monitor the security condition of the residence, there are important or private information captured by those sensors. If those sensors were to be compromised by a malicious user, they could access important information from those sensors or use it to gain important information. Malicious user's goal is to stop or malfunction smart home devices by sending bulk messages filled with send/request to send through the target smart home network. This type of attack can also disrupt IoT devices functionalities. Different IoT disruption methods such as sending group messages to smart devices, clear to send, request to send to disrupt the target home network, Bluetooth type denial of service, DOS (denial of service) by continuously requesting pairing to smarthome devices. DOS was also an attack that can reduce the IoT device's battery.

DOS and DDOS attack categories were divided according to OSI model layers: The application layer, presentation layer, session layer, transport layer, network layer, data link layer, and physical layer. The application layer DOS attack was complex and had more interference than other layers of the DOS attack. An example of this layer DOS is HTTP (hypertext transfer protocol) GET, HTTP POST, and slowloris. DOS attack in the presentation layer send deformed SSL request to the target since most transactions generally used SSL. Examples of presentation DOS attacks are protocol misuse attack and SSL traffic floods. Session layer DOS attack used synchronization and termination of connections over the network by taking advantage of login and log off protocols or telnets. Examples of these attacks consisted of telnets such as telnet Brute force, telnet communication sniffing, and telnet DOS. Transport layer DOS attacks were based on transmission and generating an enormous volume of traffic to deactivate or entirely block the availability of services or resources in the network for the target host. Examples of transport layer attacks are TCP synflood attack and UDP flood attack. DOS network layer attack operates by injecting the target host network with a large amount of traffic so the target host cannot handle them. Examples of DOS attacks are the smurf attack and Ping flood attack. Datalink layer DOS attacks are based on attacking data frame detection, medium access control, multiplexing of data-streams, error control, and communication between device nodes. Examples of this attack are unfairness attack, collision attack, and exhaustion attack. Lastly, DOS physical layer DOS attack is a type of attack that targets a network device that uses wireless to interrupt the connection. Examples of physical layer attacks are jamming attacks and tampering attacks [9].

When smart home devices initialized connection to the application server through its network, hackers in that application network as a man in the middle attack can collect those packets by changing the routing table in the smart home gateway router. Even though SSL encrypts the data, the hacker can use a forged certificate to access encrypted data. Sinkhole attack was an attack where a malicious gateway/router receives a network packet from its target by broadcasting false routing information to its target devices to change network routing. A malicious user can lure network traffic to its network to sniff for important information. Sinkhole attack can affect IoT network devices to be interrupted also consume more power and time. A successful sinkhole attack can also lead to a DOS attack. A wormhole attack can manipulate the original bit of the communication channel, which has low latency [10]. Then attacker relocates the original bits in the communication channel with false bits and gets access to IoT systems. Wormhole attack did not need to depend on decrypting encrypted packet. Sybil attack targets IoT sensors and sends it a large amount of identification attempt so the sensors node can accept false information due to overload and also disturb the IoT network.

Key-based encryption algorithms were divided into asymmetric/public key and symmetric key/secret key-based. Symmetric key encryption was implemented by the authorized sender and recipient, keeping secret of both secret key and public key for their communication. The allocation of different keys to different users increases the overall transmission security. The strength of symmetric key encryption is affected by the secrecy of both encryption and decryption keys. The symmetric key can be divided into block and stream cipher based on the grouping of message bits [11]. Block cipher group of messages fixed-sized characters is encrypted all at once and transmitted to the receiver. In contrast, stream cipher block size is one character and is not appropriate anymore for software processing due to key length. Examples of symmetric key encryption algorithms are DES, 3DES, AES, Blowfish, and RC4.

IDS [12] are divided based on the detection approach for website-based IDS, the signature approach, and behavioral approaches. Most signatures approach IDS are host IDS in the application level, HIDS learning techniques, or defining the signatures of a known attack. Once the signatures defined, regular expression or pattern matcher is used to recognize those attacks. Behavioral approaches IDS did not use any internal information of the program. The reference model of behavior in these approaches can be defined by the specifications of the application. Most of the behavioral approaches are network intrusion detection at the

network level to monitor the network. It analyzes passing traffic on the specified network and matches the traffic passed on those networks to the database of known attacks. If any network attack or abnormal behavior is detected, an alert can be sent to the administrator. An example of this behavior approaches IDS is snort. There were several related research and approaches for secure and efficient smarthome IoT architecture. This part will discuss various architecture and approaches to reach a secure and energy-efficient IoT smarthome gateway. One of IoT's smarthome architecture that focuses on encryption in user data and authentication is Peter. Peter is a smarthome cloud-based security hub place within the Smarthome network to create a decentralized smarthome environment unliked centralized cloud entity [13]. This architecture proposed multiple methods to maintain the security and privacy of smarthomes IoT devices such as ECC cryptography for encrypting both smarthome communication and user private on smarthome devices, and user authentication implemented using Deffie Hellmen key algorithm, zero-knowledge proof, and physical unclonable function.

Rather than encrypting data authentication and IoT private data, this low-cost flow-based security solution is an IDS approach to securing IoT architecture [14]. Low-cost flow-based smarthome security solution is a secure smarthome architecture solution that implements a low-cost flow method to detect an intrusion or network attack rather than the standard DPI (deep packet inspection) or VNF (virtual network functions) for detecting threats. Low-cost flow detection requires dynamically characterize the traffic at the flow level because of the different vulnerabilities detected. This architecture used cloud-based software to manage and process traffic flow. The main security features of this architecture are the analysis engine and network rules.

IDS based IoT smarthome architecture designed by Simpson *et al.* [15] with in Hub security manager. This architecture applied security manager in hub device that able to monitor communication or network traffic between IoT devices to Hub and Hub to external entities (third-party IoT service and manufacturer cloud). In Hub security manager logs all communication and device fingerprinting and save them in Hub [15]. Also, provide security/vulnerability patch and updates to each different smarthome IoT devices, whether it is the latest or out of date IoT device.

A secure authentication based smart home gateway architecture, SEA was a smarthome gateway device architecture proposed to be used in healthcare smarthome [16]. This architecture provides security in authentication and authorization between devices and its user handled by those smart gateways. Smarthome gateway performed DTLS (datagram transport layer security) in the authentication and authorization process. The secure management system architecture was introduced after researching attacks toward different security layers in IoT architecture, such as the presentation layer, network layer, and application layer. The system aims to create lightweight and efficient power consumption using a symmetric-key for encryption algorithm and access token as a user authentication algorithm [10]. SIT (secure IoT), secure and energy-efficient approach to smarthome, unlike Peter encryption algorithm. SIT is a hybrid lightweight encryption algorithm combined from Feistel architecture used in encryption block cipher such as AES and substitution-permutation network used in encryption block cipher such as DES [17]. Feistel architecture, both encryption and decryption, has a similar process and substitution-permutation network uses substitution and transpositions in cipher text, so it has changed in a pseudo-random manner. SIT is a symmetric key block cipher with a 64-bit key and plain text. SIT Algorithm to improve energy efficiency, does five rounds of encryption rounds rather than 10 or 20 rounds of encryption rounds. Each round, SIT included mathematical operations that operate 4 bits of data. To further secure the encryption process, SIT applies Feistel architecture to add complexity to the encryption.

Another flow-based IDS smarthome architecture, ProvThing, is a flow-based analysis IoT architecture that used data provenance technique to collect information from data in IoT apps and API. ProvThings provide graphs from provenance data collected, containing a history of interactions between architectures objects [18]. Lastly, ProvThing has a policy monitor to enforce policy based on the provenance of system events. LegIoT or lightweight edge IoT gateway architecture used a SBC (single board computer) such as Raspberry Pi or Ordroid to make a flexible and energy-efficient IoT gateway [19]. LegIoT used Docker container technology to make instances that more efficient than hypervisor-based virtualization to run services available. Using Docker container technology, it is possible for moving or back up core instances to another SBC assuming SBC specification can handle instances from the previous SBC. LegIoT divides two modules to Northbound, which manage connection through the Internet, and southbound, which manage the connection with IoT sensors or devices.

A different IoT gateway architecture can self-configure whenever new IoT devices connected to this gateway or Old IoT devices were not used or idle for a particular time. The gateway will automatically configure the new IoT devices and remove old unused IoT devices without user command. This architecture uses IoTivity to collect IoT devices attribute and also configured when registering new IoT devices over HTTP [20]. Lastly, an encryption approach IoT smarthome architecture, Fog computing is an IoT

architecture based on cloud computing where the cloud provides computational and storing resources to the IoT sensor nodes. To reduce energy consumption in fog computing, researcher traditionally focused on the nodes which run on a limited power source, also start to research on IoT gateway power efficiency [21]. Security in IoT fog computing is mostly overlooked since low computation IoT gateways security mechanism would increase the power consumption of IoT fog computing architecture. This research focused on TLS encryption in IoT gateway communication, specifically RSA and ECC encryption algorithm.

According to the studies that have been conducted concerning secure IoT gateways, researchers are motivated to develop a secure and power-efficient smarthome IoT gateway named Rasefiberry. We evaluate which well-known encryption algorithm is efficient enough for Rasefiberry and can default snort community rule or custom Rasefiberry snort rule effectively detect network attacks towards raspberry-based IoT gateway. Researchers add mean over time as new parameters to measure the efficiency encryption algorithm. This paper is organized as follows. The proposed method explained in section 2, the research method explained in section 3, the evaluation results and discussion are explained in section 4, and the result of the research is concluded in section 5.

## 2. PROPOSED METHOD

The goal of the proposed method to provide its IoT smarthome owner user with network secure and efficient power usage smarthome gateway. The proposed method also uses mostly open-source programs available in most Linux application repository. The proposed model has its accessibility and lower-cost alternative to commercially available IoT gateway with the architecture needed raspberry pi or low-cost board computer. The research design used Rasefiberry Pi as the hardware with a specification of SoC: Broadcom BCM2837, CPU: 4×ARM Cortex-A53, 1.2GHz, GPU: Broadcom Video Core IV, RAM: 1GB LPDDR2 (900 MHz), Networking: 10/100 Ethernet, 2.4GHz 802.11n wireless, Bluetooth: Bluetooth 4.1 Classic, Bluetooth low energy and storage: microSD [22]. The Rasefiberry hardware utilized Raspberry Pi 3B+ and malicious PC, a laptop with the specification above. Raspberry Pi was connected to Mini Wi-Fi USB adapter, HDMI cable, and micro USB power supply. Mini Wi-Fi USB adapter is used for creating a new network for connection between Raspberry and IoT devices. HDMI cable is connected to the monitor display for configuring initial Raspbian settings.

For Rasefiberry software, Raspberry PI was installed with Raspbian as the default operating system. Raseffiberry used Openhab open-source IoT gateway application as the gateway to connect to IoT or smart devices. Openhab was used to customize items/things to a scriptable command to the host or to IoT devices. Rasefiberry installed OpenSSL version 1.1.01, GPG version 2.1.1.8, and seccure version 0.50 as the encryption engine software. Rasefiberry installed snort with version 3.9.7.0, which is the latest snort version in the Raspbian repository.

As shown in Figure 1, three main features will be focused on Rasefiberry smarthome IoT architecture, encryption engine, authentication, and IDS (intrusion detection system). The encryption engine feature in Rasefiberry will request IP and MAC addresses of IoT devices to match with registered IoT devices on Rasefiberry. If the IP and MAC address did not match the registered IP and MAC address, it would deny the connection request from that IoT devices, and the process ends. If the IP and MAC addresses were matched, the IoT device could establish a secure encrypted connection with Rasefiberry. IoT device will send private data through the network toward a secure hub. The private data will be encrypted and saved within Rasefiberry storage. The encryption engine provides its user with multiple encryption algorithms in Rasefiberry with the Twofish encryption algorithm as the default encryption algorithm. The Twofish encryption algorithm is chosen as the primary encryption algorithm of Rasefiberry as both secure and efficient compared to a well-known encryption algorithm available.

For the authentication function, the user from the Internet or smarthome network must first input their user credentials (user password and IDs) to access the gateway. If the user credential verification failed, the login failed error would prompt, and the user has to try to input the correct password to log in. If the user credentials verification succeeded, the Rasefiberry would encrypt the authentication connection, and the user allowed access to Rasefiberry gateway management. The Rasefiberry authentication integrates openHAB for its IoT gateway authentication and SSH login to Rasefiberry.

The last function, Rasefiberry IDS, will monitor network traffic from both Internet through Rasefiberry and smart home network to Rasefiberry. Rasefiberry will first detect if whether or not any data transmission or network activity in which data can exist contained login data, patch, and device identification as data flow. Rasefiberry analyzes the data traffic then compares it to the data flow signature saved in the database. If the IDS detect any malicious network attack signature, then IDS will drop malicious network traffic. If the data transmission network does not match any malicious network attack, then the network transmission will be forwarded to the destination and safe the data flow within the Rasefiberry database. Snort was implemented in Raspberry PI as an IDS to reduce the payload of processing speed and RAM usage.

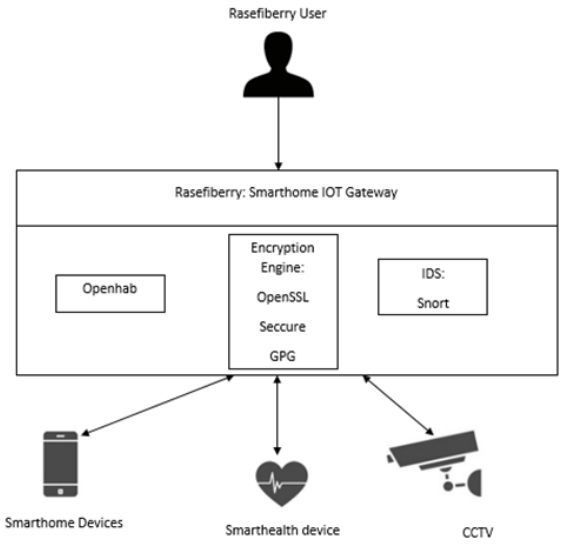


Figure 1. Rasefiberry architecture design

## 3. RESEARCH METHOD

The Rasefiberry gateway device design in this paper was comprised of two parts. The first part is the architecture of Rasefiberry as well as hardware and software specifications. The second part is the research evaluation methods to test the security of Rasefiberry IDS with Kali Linux tools and power efficiency of Rasefiberry available encryption algorithm based on [23, 24] research. Figure 2 showed the evaluation scenario to evaluate research is implemented as designed. A malicious user is successfully accessed Rasefiberry through the Wi-Fi router. The malicious user has no information on authentication through the Rasefiberry and found the Rasefiberry open application port of 22 (Secure Shell Port), 8080 (HTTP port), and 8443 (HTTPS port). From the open port known by the malicious user, a possible attack that disrupts Rasefiberry confidentiality and availability of CIA (confidentiality, integrity, and availability) network security triad. The research has chosen multiple DOS types to test Rasefiberry availability and SSH password brute force with a sniffing attacks to test Rasefiberry confidentiality.

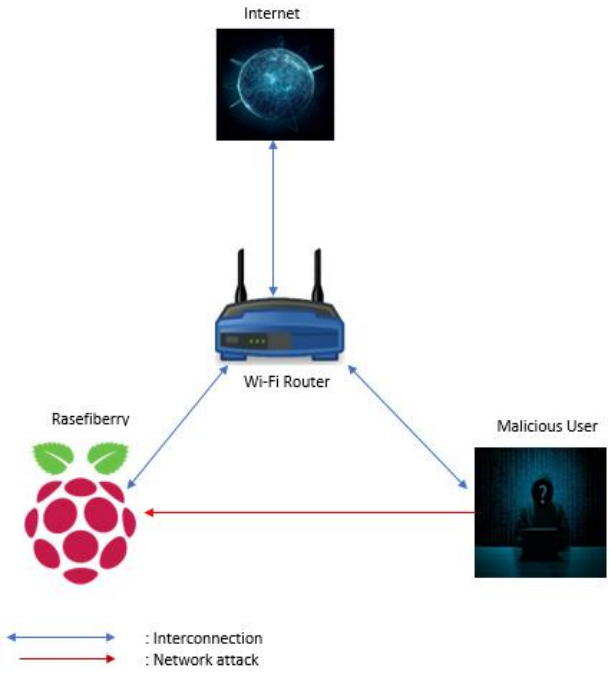


Figure 2. Rasefiberry evaluation scenario

Evaluation test for IDS and Encryption engine function consists of Malicious PC is a virtual host in a virtual box installed with Kali Linux VM. Kali Linux already pre-installed with most security applications for penetration testing, for this research application used was Metasploit and Wireshark. The malicious PC also uses HULK (HTTP unbearable load king) script for running flood type of DOS. The researcher implemented snort with default community rules and custom Rasefiberry rules shown in Figure 3 to help detect network attacks.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 22 (msg:"SSH incoming";\
    flow:stateless; flags:S+; sid:100006927; rev:1;)
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"Possible TCP DoS"; flags:S;\
    flow:stateless; detection_filter:track by_dst, count 70, seconds 10; sid:199203)
alert tcp $EXTERNAL_NET any -> $HOME_NET 8080 (msg:"SlowLoris DoS attempt"; \
    flow:established,to_server,no_stream; content:"X-a|3a20|b|0d0a|"; dsize:<15; \
    detection_filter:track by_dst, count 3, seconds 30; \
    classtype:denial-of-service; sid:1; rev:1; )
```

Figure 3. Rasefiberry custom snort rules

Wireshark utilized to run sniffing attack, Metasploit to run auxiliary contained slowloris and synflood attack for DOS type attack also password brute-force attack, and HULK python script to run flood-based DOS attack. The network attack will be run based on a basic penetration test [25-27]. Efficiency parameter evaluation will be tested in Rasefiberry generally used encryption algorithms such as DES, AES, Blowfish, ECDH, and Twofish. The Rasefiberry will test encryption speed by encrypting and decrypting a text file of 1 MB size with multiple different encryption algorithms until ten times of iteration. The research encryption algorithm comparison will be compared with a recent encryption algorithm comparison [28-30] with different hardware devices. The comparison with other existing encryption algorithms is made to show whether if the Twofish algorithm as the proposed default encryption algorithm is the most efficient. The application used for encrypting will be using OpenSSL, seccure, and GPG (GNU Privacy Guard). To measure time Rasefiberry, Rasefiberry use time Linux application to measure Linux system time used when encrypting target files. It also shows CPU usage of the single Linux command. Based on research conducted by Minaam *et al.* [23] and Haii *et al.* [24], the parameter to measure cryptography efficiency, there are:

– Computational time: Time needed by Rasefiberry to encrypt and decrypt data affected by processing power and algorithm complexity

$$Encryption\ Time = \frac{Total\ time\ of \times iteration\ encryption\ (second)}{x} \quad (1)$$

$$Decryption\ Time = \frac{Total\ time\ of \times iteration\ decryption\ (second)}{x} \quad (2)$$

– Power consumption: Power measured from average encryption time multiplied by Raspberry power consumption and 1 hour or 3600 seconds.

$$Power\ Usage\ (Joule) = \begin{array}{c} Encryption\ time\ (seconds) \times Voltage\ (Volt) \\ \times power\ consumption\ (milliAmpere) \times 3600 \end{array} \quad (3)$$

– Mean encryption time: The difference between the encryption process and the decryption process.

$$Mean\ over\ time = \Delta(Encryption\ time - Decryption\ Time)(second) \quad (4)$$

– Throughput: The rate encryption of byte size per second.

$$Throughput(kilobyte/second) = \frac{1000(Kilobyte)}{Encryption\ Time(second)} \quad (5)$$

## 4. RESULTS AND DISCUSSIONS

The results for Rasefiberry security and encryption algorithm efficiency evaluation are discussed. Two parameters are evaluated, which were Efficiency and Security. For efficiency, the raspberry 3B+ researcher decides 800 mA as default power usage [31]. Raspberry power usage with Rasefiberry hardware is connected to USB, CPU usage over 90 percent, and running openHAB as the IoT. The security parameter is

measured how snort can detect selected network attacks with default community rules compared to Rasefiberry custom rules added.

### 4.1. Efficiency

This part provides the result and analysis of comparing multiple IoT encryption algorithms. The result is by Figure 4. Figure 4(a) showed how much time used each encryption algorithm for encrypting files. Two fish has the fastest encryption speed, with an average of five milliseconds compared with another encryption algorithm. Both 3DES and blowfish cipher has the same encryption time with an average of 10 milliseconds. AES and ECDH have slowest encryption time with 20 milliseconds of encryption time. Figure 4(b) showed how much Rasefiberry power consumed to encrypt the file of each encryption algorithm. Rasefiberry runs both openHAB and snort, connected to WIFI and USB devices, and the limitation of 1 GB RAM of the same Raspberry model. From the graph, Twofish has the lowest power usage of 72 joules, followed by 3DES and Blowfish, both with 144 joules. The highest power consumption was AES and ECDH cipher with 288 joules.

Figure 4(c) showed the time difference between the encryption process and the decryption process of these encryption algorithms. Blowfish encryption algorithm has the lowest mean over time with no difference between decryption and encryption, followed by Twofish with four milliseconds difference. Both AES and DES got the same mean over time with 10 milliseconds and ECDH with 11 milliseconds of differences. Figure 4(d) showed the throughput rate of encryption for each encryption algorithms. The graph showed that the Twofish encryption algorithm has the highest throughput with 200 mb/s followed by Blowfish and 3DES with 10 mb/s throughput. The lowest throughputs were ECDH and AES cipher with 5 mb/s.

The result has shown better power usage, and throughput of each encryption algorithm was affected by the computational time parameter, which is the time for encryption and decryption. Although Twofish overall was the most efficient than other encryption algorithms, the Blowfish algorithm was found to have no time difference between the decryption and encryption process. This finding shows that Blowfish can be used for customized file encryption and decryption process. Twofish algorithms were the most efficient compared to the well-known algorithm used in this research.

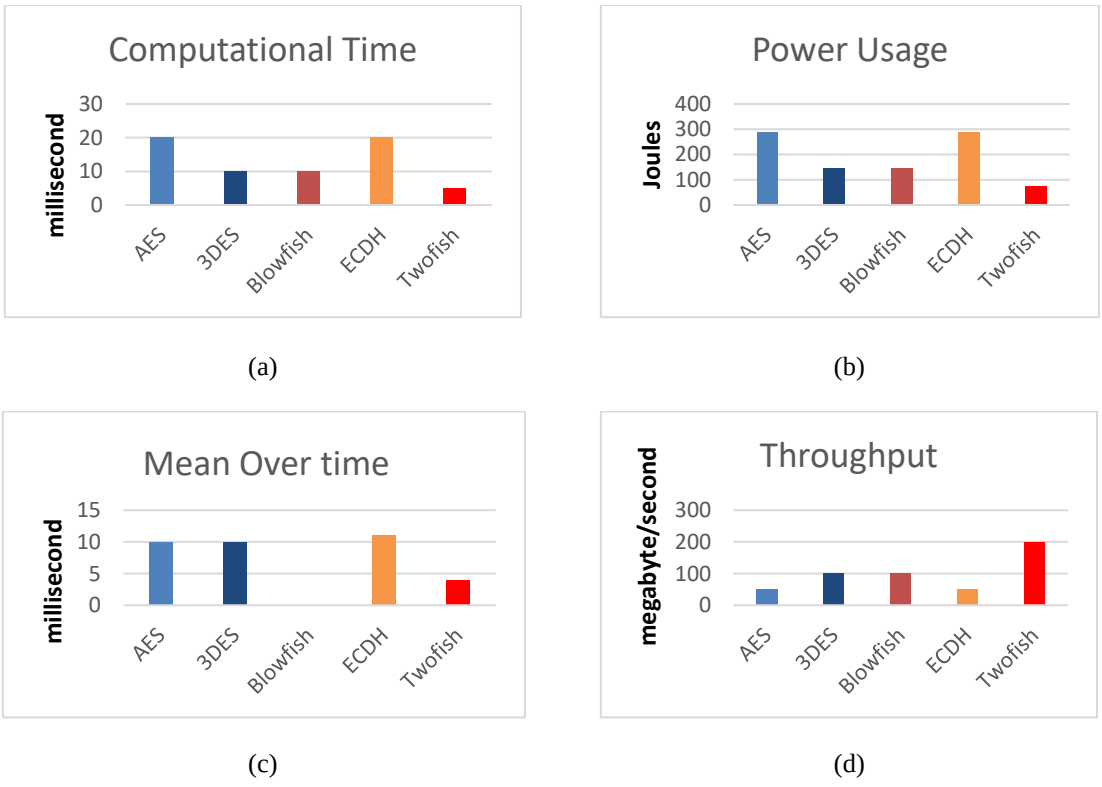


Figure 4. Encryption algorithm comparison graph, (a) Computational time, (b) Power usage, (c) Mean over time, (d) Throughput

Table 1 displayed encryption computational time comparison between Rasefiberry and another journal with a similar chosen encryption algorithm. The research in [28] was using Samsung mobile phone with specification arm 7 Samsung processor and 1 GB RAM to compare between Twofish and AES encryption algorithm. Research in [29] implemented the comparison in Python 3.6.9 on a laptop with the specification of 3.8 GHz Intel i5-9300H processor with 8GB RAM on Ubuntu 18.04, Linux kernel 5.3.0-53 comparing AES, Blowfish, and Twofish. Research in [30] used the java encryption library to compare the encryption speed of AES, 3DES, and Blowfish.

The Twofish encryption algorithm is faster than most encryption compared the research with different experiment environment also the most of the computational time are significantly affected by device hardware RAM and the library or application they used for encrypting the file. The interesting finding was the 1.9 millisecond difference between Rasefiberry and experiment using 8 GB RAM pc. This show that Rasefiberry Pi based device can use GPG as the selected encryption algorithm for its efficiency.

Table 1. Rasefiberry computational time comparison with existing available research

| No. | Encryption cipher | Research respective average computational time (milliseconds) | | | |
|---|---|---|---|---|---|
| | | Rasefiberry | [28] | [29] | [30] |
| 1 | AES | 20 | 90 | 8.9 | 800 |
| 2 | 3 DES | 10 | - | - | 800 |
| 3 | Blowfish | 10 | - | 4.2 | 500 |
| 4 | ECDH | 20 | - | - | - |
| 5 | Twofish | 5 | 60 | 3.1 | - |

### 4.2. Security

Details for the experiment result shown whether Rasefiberry's IDS effectively detect network attacks simulated by malicious PC. The result of the experiment is shown in Table 2. The results show Rasefiberry IDS could detect SYN flood attack with snort community rule and Rasefiberry custom rule. Snort detected SNMP trap alert SNMP trap from Rasefiberry to spoofed IP created by SYN flood attack. This project SYN flood script also uses other network protocols to execute SYN flood. Rasefiberry hardware performance is not affected by a SYN flood attack over 5 minutes of testing. The delay of each execution is about five milliseconds to another attack. For future work, Rasefiberry can deny connection by configuring the snort rule to drop network packet from the source of the IP attack.

Table 2. Rasefiberry snort detection capability

| No. | Network attacks | Network attack detection capability | |
|---|---|---|---|
| | | Default snort community rules | Rasefiberry custom rules |
| 1 | SYN flood DOS | Detected | Detected |
| 2 | Slowloris | Not detected | Not detected |
| 3 | Hulk DOS | Not detected | Detected |
| 4 | SSH Bruteforce password attack | Not detected | Detected |
| 5 | Wireshark sniffing | Not detected | Not detected |

Rasefiberry IDS detected slowloris attack with Rasefiberry custom snort rule, but the basic community snort rules did not detect slowloris attack to Rasefiberry. Slowloris sending HTTP requests delay is 1 millisecond for other requests. The Rasefiberry CPU performance shows an increase in CPU used to handle those HTTP requests by slowloris. Slowloris was executed for 5 minutes until the researcher stopped the script. For future work, it would be creating a more effective snort configuration to detect the slowloris network signature.

IDS captured the Hulk DOS attack with snort Rasefiberry custom snort rule but not with snort default community rule. Hulk DOS attack overloaded Rasefiberry performance 1 minute after the attack was executed. The researcher abruptly ends the Hulk script attack before the Rasefiberry shuts down or hanged because of overflowing requests. The delay of each Hulk DOS attack is about 0.05 milliseconds. The result shows Hulk DOS as a volume base DOS attack. As this research expects Rasefiberry snort to detect DOS attack rather than handling the DOS, it is not feasible to block the source attack IP. Each Bruteforce attack attempt came from multiple-port sources to Rasefiberry SSH port. The delay time between each Bruteforce attack is about 2 or 3 seconds to execute. Compared to default snort community rules, snort did not detect SSH Bruteforce attacks. Most Linux has their SSH protection handled by other applications, excluding snort.

IDS was unable to show any logs concerning kali Linux sniffing even with the Rasefiberry custom snort rule. Although the snort configuration was configured to detect sniffer ICMP packet, snort does not

detect any data flow or network signature from the kali Linux sniffing process. The sniffing process itself is passive and did not leave any network traces. Wireshark sniffing did not affect Rasefiberry performance.

## 5. CONCLUSION

The motivation for this research was to create a secure and efficient smarthome gateway to handle network attacks on Smarthome IoT architecture, also efficient enough for limited performance devices such as raspberry pi. In this research, the researcher experimented on Rasefiberry security and power efficiency. The security parameter of Rasefiberry tested by simulating the most commonly executes network attack towards the Rasefiberry network to show whether Rasefiberry IDS can detect them. Rasefiberry IDS could detect and alert most of the brute force attacks that include all DOS attacks (SYN flood, slowloris, Hulk) and password attack except sniffing from another device in the same network. The researcher found default snort community rules did not detect or alert general attacks except by add custom rules to be included in the snort config. For the encryption efficiency parameter, the overall efficient encryption cipher was Twofish. Twofish encryption cipher is shown to be 30% faster overall speed and throughput than another encryption algorithm used in the research. The researcher found that blowfish has the same encryption and decryption speed. Twofish has proven to be power efficient and fast encryption. Twofish was also available in openGPG encryption software that can be easily applied for general file encryption. For future works, we would implement more lightweight IoT gateway application specialized for a low-performance device such as raspberry pi to improve further the performance of the device implement secure connection between Rasefiberry device with other IoT devices since IoT device is one of the points of network attack other than the Rasefiberry device, customize IDS snort default rules specialized for Raspbian OS and consideration of setting up alert or rules to drop specific network attack pattern.

## ACKNOWLEDGEMENTS

This work is supported by the Directorate General of Strengthening for Research and Development, Ministry of Research, Technology, and Higher Education, Republic of Indonesia as a part of Penelitian Terapan Unggulan Perguruan Tinggi Research Grant to Binus University entitled "Portable Sleep Quality Monitoring Prototype and Application based on Cloud Computing Technology and Machine Learning" with contract number: 039/VR.RTT/IV/2019 and contract date: 29 April 2019.

## REFERENCES

[1] IHS, "IoT: number of connected devices worldwide 2012-2025," Statista, 27-Nov-2016. [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. [Accessed: 20-March-2019].

[2] M. A.Razzaq, S. H. Gill, M. A. Qureshi, and S.Ullah, "Security Issues in the Internet of Things (IoT): A Comprehensive Study*", International Journal of Advanced Computer Science and Applications*, vol.8, no. 6, 2017.

[3] C. L. É. V. Y. Bencheton, E. Darra, G. Tétu, G. Dufay, and M. Alattar, "Security and Resilience of Smart Home Environments," ENISA, 20-Apr-2018. [Online]. Available: https://www.enisa.europa.eu/publications/security-resilience-good-practices. [Accessed: 07-Oct-2019].

[4] D. Desai and H. Upadhyay, "Security and Privacy Consideration for Internet of Things in Smart Home Environments," *International Journal of Engineering Research and Development*. vol. 10, no. 11, pp. 2278–67, 2014.

[5] A. Riahi Sfar, E. Natalizio, Y. Challal, and Z. Chtourou, "A roadmap for security challenges in the Internet of Things," *Digital Communication and Networks*, vol. 4, no. 2, pp. 118–137, 2018.

[6] Asghari, A. M. Rahmani, and H. H. S. Javadi, "Internet of Things applications: A systematic review," *Computer Networks*, vol. 148, no. December, pp. 241–261, 2019.

[7] S. Singh, P. K. Sharma, and J. H. Park, "SH-SecNet: An enhanced secure network architecture for the diagnosis of security threats in a smart home," *Sustainability*, vol. 9, no. 4, 2017.

[8] J. Bugeja et al., "Securing wireless communications of the internet of things from the physical layer, an overview," *ICT Express*, vol. 9, no. 1, pp. 1–16, 2017.

[9] H. S. Obaid and E. H. Abeed, "DoS and DDoS Attacks at OSI Layers*," International Journal Multidisciplinary Research and Publication*, vol. 2, no. 8, pp. 1–9, 2020.

[10] S. O. M. Kamel and N. H. Hegazi, "A Proposed Model of IoT Security Management System Based on A study of Internet of Things (IoT) Security," *International Journal of Science and Engineering Research*, vol. 9, no. 9, pp. 1227–1244, 2018.

[11] M. Faheem, S. Jamel, A. Hassan, Z. A., N. Shafinaz, and M. Mat, "A Survey on the Cryptographic Encryption Algorithms," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 11, pp. 333–344, 2017.

[12] P. Wanda and H. Jie, "A Survey of Intrusion Detection System," *International Journal of Informatics and Computation*, vol. 1, no. 1, Nov. 2018.
[13] R. M. Vrooman, "Enhancing Privacy in Smart Home Ecosystems Using Cryptographic Primitives and a Decentralized Cloud Entity," 2017.
[14] A. Sivanathan, D. Sherratt, H. H. Gharakheili, V. Sivaraman and A. Vishwanath, "Low-cost flow-based security solutions for smart-home IoT devices," 2016 *IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, Bangalore, 2016, pp. 1-6, doi: 10.1109/ANTS.2016.7947781..
[15] A. K. Simpson, S. N. Patel, F. Roesner, and T. Kohno, "Securing Vulnerable Home IoT Devices with an In-Hub Security Manager," *IEEE PerCom*, vol. 30, pp. 17–1, 2017.
[16] S. R. Moosavi, T. N. Gia, A. M. Rahmana, E. Nigussie, S. Virtanen, J. Isoaho, H. Tenhunen, "SEA: A secure and efficient authentication and authorization architecture for IoT-based healthcare using smart gateways," *International Conference on Ambient Systems, Network, and Technologies*, vol. 52, no. 1, pp. 452–459, 2015.
[17] M. Usman, I. Ahmed, M. Imran, S. Khan, and U. Ali, "SIT: A Lightweight Encryption Algorithm for Secure Internet of Things," *Int. Journal of Advanced Computing Science Applications*, vol. 8, no. 1, pp. 1–10, 2017.
[18] Q. Wang, W. Ul Hassan, A. Bates, and C. Gunter, "Fear and Logging in the Internet of Things," *Network and Distributed System Symposium*, 2018.
[19] R. Morabito, R. Petrolo, V. Loscrì, and N. Mitton, "LEGIoT: A Lightweight Edge Gateway for the Internet of Things," *Future Generation Computer Systems*, vol. 81, pp. 1–15, 2018.
[20] B. Kang and H. Choo, "An experimental study of a reliable IoT gateway," *ICT Express*, vol. 4, no. 3, pp. 130–133, 2018.
[21] M. Suárez-Albela, T. M. Fernández-Caramés, P. Fraga-Lamas, and L. Castedo, "A practical evaluation of a high-security energy-efficient gateway for IoT fog computing applications," *Sensors* (Switzerland), vol. 17, no. 9, pp. 1–39, 2017.
[22] Raspberry Pi Foundation, "Raspberry Pi 3 Model B+ Datasheet," *Datasheet*, p. 5, 2016.
[23] D. S. A. Minaam, H. M. Abdual-Kader, and M. M. Hadhoud, "Evaluating the effects of symmetric cryptography algorithms on power consumption for different data types," *International Journal of Network Security*, vol. 11, no. 2, pp. 78–87, 2010.
[24] M. El-Haii, M. Chamoun, A. Fadlallah and A. Serhrouchni, "Analysis of Cryptographic Algorithms on IoT Hardware platforms," 2018 *2nd Cyber Security in Networking Conference (CSNet)*, Paris, 2018, pp. 1-5, doi: 10.1109/CSNET.2018.8602942.
[25] R. L. J. Jesús, A. H. Anastacio, T. M. Cristhian, P. V. Omar, R. G. M. René, and F. M. Heberto, "How to improve the IoT security implementing IDS/IPS tool using Raspberry Pi 3B+," *International Journal of Advanced Computer Science and Application*, vol. 10, no. 9, pp. 399–405, 2019.
[26] T. Zitta et al., "Penetration Testing of Intrusion Detection and Prevention System in Low-Performance Embedded IoT Device," 2018 *18th International Conference on Mechatronics - Mechatronika (ME)*, Brno, Czech Republic, 2018, pp. 1-5..
[27] D. Stiawan, M. Y. Idris, R. F. Malik, S. Nurmaini, N. Alsharif, and R. Budiarto, "Investigating Brute Force Attack Patterns in IoT Network," *Journal of Electrical and Computing Engineering*, vol. 2019, pp. 1-13, 2019.
[28] D. A. F. Saraiva, V. R. Q. Leithardt, D. de Paula, A. S. Mendes, G. V. González, and P. Crocker, "PRISEC: Comparison of symmetric key algorithms for IoT devices," *Sensors* (Switzerland), vol. 19, no. 19, pp. 1–23, 2019.
[29] A. Ghosh, "Comparison of Encryption Algorithms : AES, Blowfish and Twofish for Security of Wireless Networks," *International Research Journal of Engineering Technology*, vol. 7, pp. 4656–4658, 2020.
[30] M. Nazeh Abdul Wahid, A. Ali, B. Esparham, and M. Marwan, "A Comparison of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish for Guessing Attacks Prevention," *Journal Of Computer Science Applications and Information Technology*, vol. 3, no. 2, pp. 1–7, 2018.
[31] J. Corral-García, F. Lemus-Prieto, J. L. González-Sánchez, and M. Á. Pérez-Toledano, "Analysis of energy consumption and optimization techniques for writing energy-efficient code," *Electron*, vol. 8, no. 10, 2019.

## BIOGRAPHIES OF AUTHORS

**Vincent Simadiputra** received the bachelor degree in Computer Science in 2014. From 2018 until now, he is a graduate student of Information Technology at Bina Nusantara University. His research interest is network design and network security.

**Nico Surantha** received his B.Eng (2007) and M.Eng. (2009) from Institut Teknologi Bandung, Indonesia. He received his Ph.D. degree from Kyushu Institute of Technology, Japan, in 2013. Currently, he serves as an assistant professor in Computer Science Department, Binus Graduate Program, Bina Nusantara University. His research interest includes wireless communication, health monitoring, network design, digital signal processing, system on chip design, and machine learning. He is an IEEE member.