Implementation and Evaluation of Network Intrusion Detection System on Raspberry Pi Device

Dimitrije Krstic, Nadja Gavrilovic, and Vladimir Ciric

Abstract – In recent years, with the growing development of IoT, devices with limited resources are increasingly present. Having a simple architecture and short time-to-market makes cyber security defence of these devices challenging. One possible approach in securing a system is to place a dedicated Intrusion Detection System (IDS) as a common network gateway of all devices, while the other is to embed the IDS into the device itself. Until recently, the first approach was dominant in IoT because of the limited device resources. The goal of this paper is to explore possibilities of robust mainstream IDS implementation on an IoT device itself. The implementation of the Snort IDS on Raspberry Pi 4 is given, and the performances are analysed under the simulated attack. It is shown that the device can process up to 30% of malicious packets from incoming traffic, which is sufficient not only to protect itself, but others as well.

Keywords – Intrusion Detection Systems, Raspberry Pi, Snort IDS.

I. Introduction

Technology is constantly evolving, which makes it easier for people to function on a daily basis. New technology wave in recent years introduced a whole new set of small devices and systems embedded in common appliances, bringing new qualities and expanding its applications. Nowadays it is not uncommon to have a dishwasher connected to the Internet [1].

As a consequence, the number of devices connected to the Internet exceeded the number of people using it [1]. Those systems are developed to make tasks easier for everyone. However, their simple architecture and insufficiently tested software because of short time-to-market make the task of an intruder easier too [2].

An analysis of common attack techniques led to the following conclusion. Attacks that have the greatest consequences use common packets but with content modified by the attacker. Therefore, protection systems are implemented to analyze the content of the packet against the known malicious signatures in order to determine whether the packet was sent by the attacker [3]. As the database that holds malicious signatures can have a large number of entries, analysis can be a resource-demanding task. This task was hard to reach until recently for almost all IoT devices. Because of that, the security was usually implemented as a common and centralized IDS system on the network gateway.

Dimitrije Krstic, Nadja Gavrilovic, and Vladimir Ciric are with the University of Nis, Faculty of Electronic Engineering, Aleksandra Medvedeva 14, E-mail: dimitrije.krstic@elfak.rs, nadja.gavrilovic@elfak.ni.ac.rs and vladimir.ciric@elfak.ni.ac.rs.

The goal of this paper is to explore the possibilities of robust mainstream IDS implementation on an IoT device itself. The implementation of the Snort IDS on Raspberry Pi 4 will be given, and the performances will be analysed under simulated attack. In the simulation, we will use different quantities of malicious packets and evaluate the utilization of system resources. It will be shown that the device can process up to 30% of malicious packets from incoming traffic, which is sufficient not only to protect itself, but others as well in most cases.

The paper is organized as follows. Section 2 gives a brief introduction to Raspberry Pi device. Section 3 gives a quick overview of IDS. Section 4 explains how Snort is implemented on a Raspberry Pi, and where device is set in topology, as a basis for the proposed implementation. Also, a detail view of malicious traffic that is used in simulation is given. Section 5 is the main section and presents evaluation results of proposed system, while concluding remarks are given in Section 6.

II Raspberry Pi

In 2006, Eben Upton have seen that students and entry level engineers are not familiar enough with practical use of personal computers. He then realized that the main issue is the market price of computers. So, he came up with an idea to invent one affordable computing platform for wide area of application [4].

In 2012, Eben presented the first two models of new Raspberry Pi device (model A and model B) that represented the beginning of a low price personal computers era [4].

In further years this device was enhanced even more. This produced several versions of Raspberry Pi: Raspberry Pi 2 (February, 2015); Raspberry Pi 0 (November, 2015); Raspberry Pi 0W (February, 2017); Raspberry Pi 3B (February, 2016); Raspberry Pi 3B+ (2018); Raspberry Pi 4B (2019) [4].

In Fig. 1 Raspberry Pi 4B which is used for the implementation proposed in this paper is shown [4].

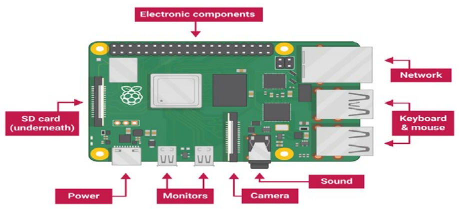

Figure 1. Raspberry Pi device [5]

Regardless the fact that Raspberry Pi is labeled as low performance device, this small device has very powerful hardware, which is given in the Table I bellow.

TABLE I
Raspberry Pi device specification [6]

Procesor	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
RAM	4GB LPDDR4-3200 SDRAM
Wi-Fi adapter	2.4 GHz and 5.0 GHz IEEE 802.11ac wireless
Bluetooth adapter	Bluetooth 5.0, BLE
Ethernet adapter	Gigabit Ethernet
USB ports	2 USB 3.0 ports; 2 USB 2.0 ports.
Display ports	2 × micro-HDMI ports (do 4kp60) 2-lane MIPI DSI display port 2-lane MIPI CSI camera port
GPU	OpenGL ES 3.0 graphics
SD slot	Micro-SD slot
Power	5V DC via USB-C connector
Work temperature	0 – 50 degrees C ambient

This hardware can not only perform common IoT tasks, but it is also suitable for more demanding computations.

III Overview of IDS

IDSs are divided in two large groups based on the technique used to determine whether network packet is regular or malicious. There are pattern (signature) matching and anomaly-based IDSs [7].

Anomaly-based IDSs monitor network traffic and compare packets and events occurred on network against the definitions of the activity that is assumed to be normal, to identify significant deviation. These systems are capable of detecting zero-day attacks, because the detection does not depend on previously detected attacks. Unfortunately, sometimes they can generate high amount of false positive alerts. In newer IDS systems, artificial intelligence is often used in order to further improve system detection [7].

IDSs that use detected properties of previous attacks for conclusion are called Signature based systems. Pattern of previous attacks and threats which are identified during attack are stored in a database of signatures. Pattern matching IDSs recognize possible intrusions by comparing network traffic to malicious attempt patterns. However, it can't detect threats that haven't been seen before [7].

One of the most used signature based IDSs is Snort. This open-source IDS is portable and very customisable. Great advantage is that the operation of Snort does not take much memory and processor time to be efficient. Snort can be deployed on various network hosts and platforms. For detection of attacks and malicious activities, Snort uses a set of patterns. Those patterns define what kind of network traffic is labeled as a threat. Snort's pattern sets are called Snort rules. Formal definition of a rule, and a typical example of a Snort rule, are given bellow, respectively [8]:

<rule action><protocol><source ip><source port><direction><dest ip><dest port><rule options>

alert tcp any any -> any any 21 (content:"user root";)

In the given example, *alert* option is set as rule action, which defines that the alert data is stored for later gathering and further analysis. The rest of the fields, except the rule options field, describe packet attributes (source and destination ip address, source and destination port, protocol). In the example, ip addresses are defined as *any*. The last part of the rule gives key-value pairs with further rule description. In the given example, the *content* is the name of the field that should be matched against the value of *user root* [8].

The main components of Snort architecture are given in Fig. 2. Sniffer component monitors network traffic and sends data to the preprocessor, which checks packets against available plug-ins for a certain type of behaviour from the packets. Detection of the intrusion is done in detection engine component. This component verifies data against set of rules. As soon as the first rule matches the data, the action specified by the rule is triggered. Examples of an action can be sending the alert to the log file through network connection, storing the alerts in an SQL database, sending the event via e-mail to notify system administrator, etc. [9].

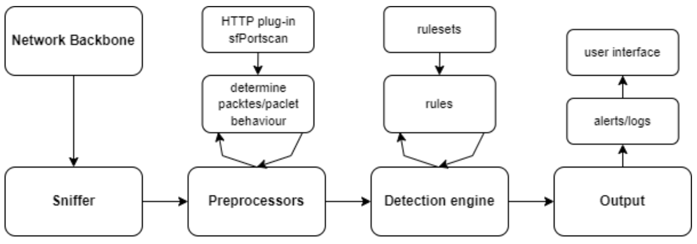

Figure 2. Snort architecture

The Snort implementation used in this paper utilizes the Boyer-Moore string matching algorithm, which is widely regarded to provide the best average-case performance of any known algorithm [10].

IV. Implementation of IDS system on Raspberry Pi device

One possible approach in IoT security is to place a dedicated IDS system as a common network gateway of all

IoT devices [11]. This approach is cost-effective if the number of devices is large. In case there is a single or a small number of IoT devices, the better solution is to embed the security mechanisms into it. The goal of the system implemented in this paper is to mix those two approaches.

A. Network topology

The Raspberry Pi with Snort IDS system is implemented as a common gateway for the local network, as shown in Fig. 3. In this setting, the Snort IDS protects at the same time both Raspberry Pi on which it is installed and the other devices on the local network. It is assumed that the attacker is positioned outside the network.

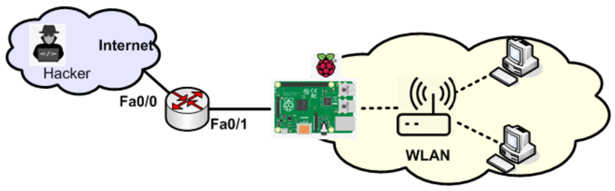

Figure 3. Network topology used for simulation

As shown in Fig. 3, the inbound and outbound traffic for the local network passes through the proposed system. In order to detect malicious traffic, we used the Snort IDS 2.9. This software comes with the rules within the malicious signatures database for packet analysis, which consists of 58219 signatures.

B. Preparation of malicious traffic

For the malicious attack simulation, we used previously prepared .pcap files that contain regular as well as malicious packets. The .pcap is the common file format for storing network traffic packets [9].

The test files are prepared by merging files that recorded normal traffic with files that recorded malicious traffic. The malicious traffic includes packets that are commonly found in known attacks. The prepared files contain the network packets which include following attacks:

DDoS (distributed denial-of-service) is a malicious attempt to disrupt the normal traffic of a targeted server, service or network by overwhelming the target or its surrounding infrastructure with a flood of Internet traffic. Flood of traffic is generated when large number of packets is sent to target IP address in order to flood the network and disrupt regular traffic [12].

DNS spoofing is a term that describes an attack where a DNS server accepts and responds to incorrect information from a host that is not authorized to get that information. DNS spoofing is malicious cache poisoning where fabricated information is stored in the name servers cache memory. On the internet 33% of all DNS servers are not immune to these types of attacks. Spoofing attacks can be root of significant security issues for devices susceptible to DNS attacks, for example users can be directed to wrong IP address of internet page, or an e-mail could be routed through mail servers which are not reliable [13].

The ZeroAccess botnet is an incredibly huge set of attacked machines, which are joined by a custom peer-to-peer protocol. They can be instructed to carry out click fraud and Bitcoin mining by the creator of botnet. Also, consequences of this type of an attack can be even more malicious activities. It can be made up of approximately 1 million attacked devices that have the potential to generate large monthly income for their creators [14].

For this simulation, files containing 10%, 20%, and 30% of malicious traffic were used. In order to highlight the difference in resource requirements, the malicious traffic is concentrated in the middle of the file (Table II). This distribution relates to the time sequence of the packets as well. File names, sizes, and malicious traffic percentages are shown in Table II.

TABLE II
Files used for the simulation

File name	File size[Gb]	Malicious packets [%]
test10.pcap	2.38	10
test20.pcap	2.33	20
test30.pcap	2.41	30

V. Evaluation Results

The proposed system is evaluated using Raspberry Pi 4 with 4GB LPDDR4-3200 SDRAM RAM, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz processor, Gigabit Ethernet adapter and 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless Wi-Fi adapter.

During the simulation of attacks, we monitored RAM and CPU to determine resources utilization of the device to perform intrusion detection for the scenario that we prepared.

Fig. 4 displays RAM usage for the test files test10.pcap, test20.pcap, and test30.pcap, with solid, dashed, and dotted lines, respectively.

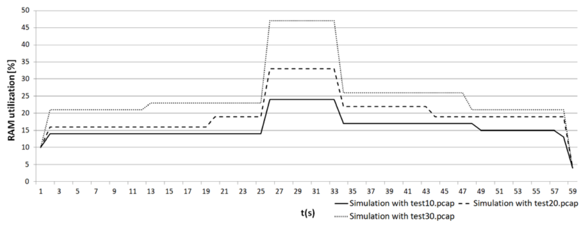

Figure 4. Utilization of RAM during simulated attack

In Fig. 4 vertical axis shows RAM utilization in percents, and the horizontal axis represents time in seconds since the beginning of the simulation. As we can see in Fig.

4, RAM utilization is the greatest around the middle of the simulation. During the simulation, the memory consumption didn't exceed 50% even in the case of 30% of malicious packets.

Utilization of CPU for test files test10.pcap, test20.pcap, and test30.pcap from Table II is given in Fig. 5.

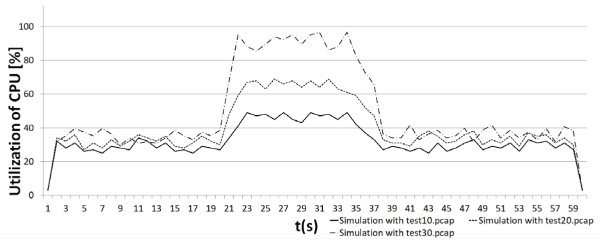

Figure 5. Utilization of CPU during simulated attack

In Fig. 5 y-axis represents CPU utilization in percent, while the x-axis represents time in seconds since the beginning of the simulation. As shown in Fig. 5, CPU utilization is the greatest in the middle of the simulation where the majority of the malicious packets are located, as was the case with the memory utilization. While varying the percentage of malicious packets we determined that the analysed device can handle up to 30% of malicious packets before reaching its CPU limits (Fig. 5).

VI. Conclusion

In this paper, the possibilities of the robust mainstream IDS implementation on an IoT device were explored. The implementation of the Snort IDS on Raspberry Pi 4 was given, and the performances were analysed under the simulated attack. In the simulation, malicious packets were used to evaluate the utilization of system resources. It is shown that the device can process up to 30% of malicious packets from incoming traffic, which is sufficient not only to protect itself, but others as well in most cases.

References

[1] Van Der Spek, P. and Verhoef, C. "Balancing Time-to-Market and Quality in Embedded Systems", Systems Engineering, 17, pp. 166-192, 2014.

[2] Breitenbacher, Dominik & Homoliak, Ivan & Aung, Yan & Tippenhauer, Nils Ole & Elovici, Yuval, "HADES-IoT: A practical host-based anomaly detection system for IoT devices", Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security, pp. 479-484, 2019.

[3] Dhanabal, L., and S. P. Shantharajah, "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms", International journal of advanced research in computer and communication engineering, pp. 446-452, 2015.

[4] Richardson, Matt and Shawn P. Wallace, Getting Started With Raspberry Pi. Sebastopol, CA: O'Reilly Media, 2012.

[5] Raspberry Pi 4 Computer Model B, Product Details https://www.seeedstudio.com/Raspberry-Pi-4-Computer-Model-B-2GB-p-4079.html, visited 28.11.2021.

[6] Raspberry Pi 4, Specifications https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/, visited 28.11.2021.

[7] H. J. Liao, C. R. Lin, Y. C. Lin, K. Y. Tung, "Intrusion detection system: A comprehensive review", Journal of Network and Computer Applications, Vol. 36, Issue 1, pp. 16-24, 2013.

[8] U. Aickelin, J. Twycross, T. Hesketh-Roberts, "Rule Generalisation in Intrusion Detection Systems using Snort", International Journal of Electronic Security and Digital Forensics, Vol. 1, 2008.

[9] Vladimir Ciric, Dušan Cvetkovic, Nadja Gavrilovic, Natalija Stojanovic, Ivan Milentijevic, "Input Splits Design Techniques for Network Intrusion Detection on Hadoop Cluster", Facta Universitatis, Series: Electronics and Energetics, Vol 34, No 2, pp. 239-257, 2021.

[10] S. O. Al-Mamory, A. Hamid, A. Abdul-Razak and Z. Falah, "String matching enhancement for snort IDS", 5th International Conference on Computer Sciences and Convergence Information Technology, pp. 1020-1023, 2010.

[11] Wanjohi, Rose Wambui, and Michael Gitau Mbuguah, "Open Source IDS for a Resoure Constrained Set-Up", 2016.

[12] Douligeris, Christos & Mitrokotsa, Aikaterini, "DDoS attacks and defense mechanisms: classification and state-of-the-art", Computer networks, Volume 44, Issue 5, pp. 643-666, 2004.

[13] Singh, Simar Preet & Maini, A. Raman, "Spoofing attacks of domain name system internet", National Workshop-Cum-Conference on Recent Trends in Mathematics and Computing (RTMC), 2011.

[14] Wyke, James, "The zeroaccess botnet-mining and fraud for massive financial gain" Sophos Technical Paper, 2012.