Penetration Testing of Intrusion Detection and Prevention System in Low-Performance Embedded IoT Device

Tomas Zitta
Czech Technical University in Prague
Faculty of Electrical Engineering
Prague, Czech Republic
zittatom@fel.cvut.cz

Marek Neruda
Czech Technical University in Prague
Faculty of Electrical Engineering
Prague, Czech Republic
marek.neruda@fel.cvut.cz

Lukas Vojtech
Czech Technical University in Prague
Faculty of Electrical Engineering
Prague, Czech Republic
lukas.vojtech@fel.cvut.cz

Martina Matejkova
Czech Technical University in Prague
Faculty of Electrical Engineering
Prague, Czech Republic
matejm24@fel.cvut.cz

Matej Jehlicka
Czech Technical University in Prague
Faculty of Electrical Engineering
Prague, Czech Republic
jehlimat@fel.cvut.cz

Lukas Hach
Czech Technical University in Prague
Faculty of Electrical Engineering
Prague, Czech Republic
hachluka@fel.cvut.cz

Jan Moravec
Czech Technical University in Prague
Faculty of Electrical Engineering
Prague, Czech Republic
moravj27@fel.cvut.cz

***Abstract*—The penetration testing method of provided security protection by IDS/IPS (Intrusion Detection System / Intrusion Prevention System) tool Suricata in the low-performance embedded IoT device is presented. Penetration testing tools are presented with emphasis on software tools, such as NMAP and Metasploit. The used testing method is described in deep. IDS/IPS tool Suricata is implemented in embedded platform Raspberry Pi 3. Main contribution is the implemented testing method for Suricata IPS that can be used in other applications of security rules of embedded IoT devices.**

Keywords— Internet of Things, Intrusion detection, Intrusion prevention, Software testing

I. Introduction

Devices for the Internet of Things (IoT) are deployed in households, factories and cities. Devices such as fridges, thermometers and various sensors and actuators usually do not contain protection software. These devices can be protected by network Intrusion Detection System / Intrusion Prevention System (IDS/IPS) which is a defense system whose task is to monitor network traffic. IDS tools only write to the log file when the attack is detected. IPS can block the threat by packet jams or blocking the connection. IPS provides protection of devices not only in the IoT against takeover attempts, but also against malicious use [1].

IDS/IPS tools can be divided to anomaly-based and signature-based. Anomaly-based tools compare the data deviations and behavior of network traffic against the statistically normal network state or profile [1]. The advantage of this approach is the ability to detect previously unknown attacks. Signature-based tools detect and distinguish between common system states and databases of known traces and patterns suggesting a penetration attempt. The advantage of signature-based detection is a low error rate, but it cannot detect an undefined type of attack. Additionally, systems can be classified as Host Based IDS HIDS (Host Based IDS) and Network Based IDS (NIDS) [2]. HIDS is running on one device and provides protection for a single machine. It monitors not only inbound and outbound traffic, but also the whole system (kernel and system behavior). NIDS monitors network traffic, most often operated on a perimeter network [3]. Each IDS/IPS system generally consists of a sensor, a configuration database, a known attack database, and a module that performs an action. The most common network attacks are port scanning, specific packet response testing, DNS spoofing, eavesdropping, unauthorized connections, or communication with non-standard IP addresses. Additionally, bugs in specific application protocols, various DOS attacks, identity falsifications, and more [2]. It is important to test IPS against network attacks and thereby verify its protection. The types of network attacks are described in [4]. It also presents description of IPS types and their location and new approaches and techniques used in IPS tools. Authors in [5] deals with web-based interfaces testing, system testing, it describes cryptographic issues due to low message entropy, which are sent between the devices. An overview of different intrusion detection systems and comparison of functions of Snort, Suricata and Bro IDS is presented in [1]. The performance of the three IDS under different traffic rates and attacks is presented in [6]. Authors in [7] describe implementation with tool Basic Analysis and Security Engine (BASE). BASE is a web interface written in PHP which provides alert management and analysis engine for detected incidents. It supports IDS/IPS as well as network components such as routers and firewalls.

Main contribution of the paper is the presented testing method for Suricata IPS and the comparison of its response implemented on the embedded platform to various tools and types of network attacks. The rest of the paper is organized as follows: The section II deals with testing of IDS/IPS, section III describes the penetration testing of Suricata IPS. Results of penetration testing are presented in chapter IV. Conclusions are presented in section V.

This paper is supported by the grants no. SGS18/183/OHK3/3T/13 and by the grant in TACR ZETA program TJ01000182.

II. Methods of Testing of IDS/IPS

Over the years, various attempts have been made to test IDS/IPS objectively [8]. Individual requirements and metrics are not standardized. IDS/IPS itself is broken down by location (NIDS and HIDS), detection mechanism (anomaly-based and signature-based detection) and data processing (real-time and offline). Therefore, IDS performance evaluation can be seen as a complex issue. It is possible to select several approaches based on speed, quantitative parameters (TP – true positive – warning of a real attack, FP – false positive – warning of non-intrusive behavior, TN – true negative – no intrusion attempt or warning, FN – false negative – no warning will be triggered when trying to intrude), scope and possibilities of the rules of IDS/IPS [8]. Another problem is generating background traffic [8]. The traffic can be real or artificial. Real traffic ensures the best possible evaluation for a network, but it cannot be repeated because of privacy issues. Data of real traffic are publicly non-disclosable [8]. Real traffic cannot be considered as a standard because it cannot be assured that an attack or attack attempt occur. Artificially induced traffic in random packets generation does not yield relevant results in IDS evaluations [8] because intrusion detection algorithms depend on content of the packets.

The combination of these two methods is chosen by DARPA (Defense Advanced Research Project Agency) that used the method of capturing real traffic on the "victim" network and then artificially adding attacks [9]. Subsequently, other similar traffics were created, such as the KDD Cup '99 data set [10]. DARPA Intrusion Detection Evaluation data sets are considered as a standard in objective IDS/IPS testing [9]. Their use is focused on substantially larger networks than IoT networks, not for low-performance embedded IoT devices.

A. *Testing Network Security*

Testing of IDS/IPS corresponds to the process of assessing the quality of the system and its security rules. There are two ways to test the network [11]:

- Whitebox – Information about the target of the attack is shared with testers. This test is used to verify the effectiveness of securing known vulnerabilities of the system in which the testers will try to penetrate using known vulnerabilities.
- Blackbox - Information about the target of the attack is not communicated. The intent is to detect security against the attack from outside. These tests are closer to real conditions and aim to uncover the unknown vulnerabilities that could be used to penetrate the system.

There are two types of security tools for system security verification, i.e., the the vulnerability scanning tools and the penetration testing tools. The vulnerability scanning tools are focused on detection of vulnerabilities of systems and applications and subsequently generate reports of potential risks. The penetration testing tools exploit these vulnerabilities to penetrate the network. It is appropriate that the methods complement each other for the best possible verification of IDS/IPS functionality [12].

B. *Vulnerability Scanning Tools*

Vulnerability scanning tools identify potential vulnerabilities of devices in network such as firewalls, routers, switches, servers and applications. Tools compare the detected information with the database of known vulnerabilities [13]. This method is automated and vulnerability scans can be performed on any number of devices. There are many tools that can be used such as OpenVAS and Nmap.

Nmap (Network Manager) is open source tool that detects all devices and services in the network [14,15]. It is designed for fast scanning of large networks, but it also works for testing individual devices. Nmap supports operating systems Linux, Windows and Mac OS. It supports terminal, GUI (Graphical User Interface) called Zenmap and debugging tool Ncat [16].

Hping3 is a tool that utilizes terminal [17]. It supports TCP, UDP, ICMP, RAW-IP protocols and traceroute mode. It is used for port scanning, firewall and network testing, OS fingerprinting.

C. *Penetration Testing Tools*

Penetration testing is a method of evaluation of security of devices, systems and applications. Different tools are used for analysis, bypass the security and penetration [11]. The whole process is not automated, and it is very expensive. It is usually performed once per year and it is focused on vulnerabilities with the highest priority in vulnerability scanning. One of the penetration testing tool is an open source penetration framework called Metaspoilt, which is designed for exploitation and security fixes development [17]. It is a modular based framework which allows development of custom modules [18].

D. *Network Attacks*

The network attack can be either active or passive in accordance with the method of data acquisition. Network attack can be considered as a passive if the attacker intercepts transmitted data over the network. Network attack can be considered as an active if the attacker attempts to disrupt the normal operation of the network [19]. The network attacks used in the penetration testing of Suricata, described in following chapter, are as follows [20]:

- Denial of Service (DoS) – attacks when a system is overwhelmed to disrupt the running of Internet-connected servers. An attack can be considered DDoS (Distributed DoS) when a large number of scattered computers are used. As an example of DoS attacks can be mentioned Smurf, Neptune, Ping of death, ICMP flood, SYN flood and UDP flood.
- Probing / Reconnaissance Attack – attacks when the system is scanned to detect vulnerabilities. As an example of these attacks can be mentioned ipsweep, mscan, nmap and portsweep.

III. Penetration Testing of Suricata

The experimental setup of penetration testing is shown in Fig. 1. The attacking side is represented by a computer, which is connected to an external network. Another computer is used as the server, which is the target of attacks. Raspberry

Pi 3 with installed Suricata is connected between them in order to monitor network traffic.

The Linux distribution for penetration testing and ethical hacking, called Kali Linux, is used on the computers. During penetration testing, the server runs FTP, Apache, MySQL and email server.

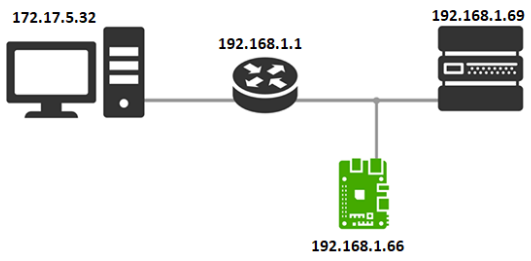


Fig. 1. Experimental setup for penetration testing.

A. Setting of Raspberry Pi 3

There are a number of platforms on the market, such as Arduino, NanoPi, Banana Pi, Parallella and Raspberry Pi 3. Raspberry Pi 3 is chosen because of various operating systems support and availability [21].

It has 64bit SoC with 4 cores and 1 GB of RAM. The platform provides limited performance, which is not suitable for filtering large numbers of packets in large networks, but performance is sufficient for experimental purposes. Installation of Suricata on Raspberry Pi 3 is executed on the command line with following commands:

- *sudo apt-get -y install libpcre3-dbg libpcre3-dev build-essential autoconf automake libtool libpcap-dev libnet1-dev libyaml-0-2 libyaml-dev zlibig zlibig-dev libcap-ng0 make libmagic-dev libjansson-dev libjansson4 pkg-config # installation of dependencies*
- *wget https://www.suricata-ids.org/download/ suricata-4.0.4.tar.gz # download of Suricata*
- *tar –xzvf suricata-4.0.4.tar.gz # extraction of file in tar format*
- *cd suricata-4.0.4 # move to the directory*
- *./configure -prefix=/usr -sysconfdir=/etc -localstatedir=/var*
- *make # compilation*
- *sudo make install-conf # automatic configuration of default configuration files*
- *sudo make install-rules # automatic download of rules*
- *sudo nano /etc/suricata/suricata.yaml # editation of file suricata.yaml using editor nano*
- *HOME_NET <ip.addr> # setting the "home" IP address*
  host-os-policy: linux [ip.addr.network, ip.addr.PC] # setting the IP address of the network and the PC
  detect-thread-ratio: 1.5 # change the value to 1.5
- *sudo ethtool –K eth0 gro off lro off # turning off the elements GRO (Generic Receive Offload) a LRO (Large Send Offload)*

The following commands are needed to work with Suricata:

- *sudo suricata -c /etc/suricata/suricata.yaml -i eth0 # start the Suricata*
- *tail -f /var/log/suricata/fast.log # log creation*

B. Probing Attacks

Port scanning is performed by using tool Nmap which supports five modes of operation based on speed and bandwidth: insane (the fastest, the widest bandwidth), aggressive, normal, polite, sneaky a paranoid (the slowest) [14].

Port scanning is performed in normal, paranoid and sneaky modes because higher scan speeds burden the target network more. Modes paranoid a sneaky should be able to bypass IPS [14]. The following commands are used for DoS attacks:

- *root@kali: ~# nmap –sS <ip.addr.victim>*
- *root@kali: ~# nmap sneaky <ip.addr.victim>*
- *root@kali: ~# nmap paranoid <ip.addr.victim>*

The attack XMasTree (Christmas Tree) is also carried out. The packets of this attack are used to explore the network and they are evaluated as suspicious. Therefore, these packets should be easily detectable by IDS [22]. Metasploit tool is used for this attack. The following commands are used for XMasTree attack:

- *msf > use auxiliary/scanner/portscan/xmas*
- *msf auxiliary (scanner/portscan/xmas) > set RHOSTS <ip.addr.victim>*
- *msf auxiliary (scanner/portscan/xmas) > set PORTS <no_port>*

C. DoS Attacks

In addition, the SYN flood attack is tested which is the type of DoS attack, when the attacker sends the SYN packets to the destination server, server responds with SYN-ACK packet, but attacker does not respond with ACK packet. SYN flood is one of the most common DoS or DDoS (Distributed DoS) attack. The aim of these attacks is overloading the router and the switch so that their performance is not enough to process generated traffic which can have up to hundreds of Gbps [17]. For testing purposes, it is sufficient to send a SYN flood with Metasploit and track the Suricata response. The following commands are used for SYN flood attack:

- *root@kali: ~# s*
- *ostgresql start*
- *root@kali: ~# msfconsole*
- *msf > use auxiliary/dos/tcp/synflood*
- *msf auxiliary (dos/tcp/synflood) > set RHOST 192.168.1.1*
- *msf auxiliary (dos/tcp/synflood) > set RPORT 80*

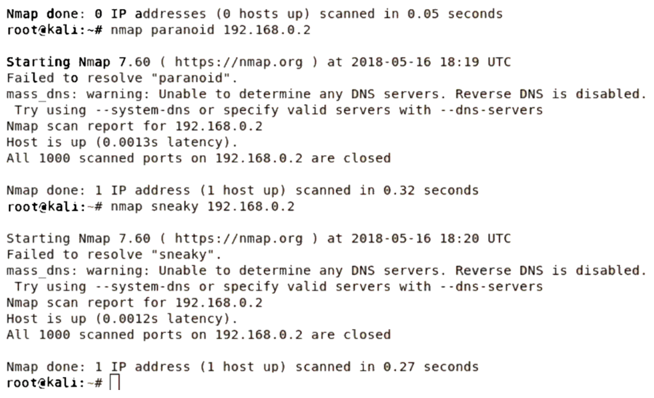
Additionally, an ICMP flood attack is performed using Hping3. An attacker typically sends ICMP echo (ping) which do not comply with the recommended size 548 B by RFC, but packets have size up to 65 kB. The destination server sends back the ICMP Echo response while maintaining the packet size. The line is overwhelming twice, if the victim responds [17]. The following command is used for this attack:

- *hping3 -c <no_packets> -1 --flood –V –i eth0 <ip.addr.victim>*

Where -c is number of packets transmitted per second. Parameter -1 is used to setting ICMP as the default mode. Parameter -V turns on verbosity

IV. RESULTS

The testing of probing attack shows that Suricata IPS is capable of detection of probing attacks, performed by the NMAP tool for port scanning in the normal, sneaky and paranoid modes. Results of network testing using Nmap tool are shown in Fig. 2. The Suricata IPS successfully blocked the port scanning.



```
Nmap done: 0 IP addresses (0 hosts up) scanned in 0.05 seconds
root@kali:~# nmap paranoid 192.168.0.2

Starting Nmap 7.60 ( https://nmap.org ) at 2018-05-16 18:19 UTC
Failed to resolve "paranoid".
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
 Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.0.2
Host is up (0.0013s latency).
All 1000 scanned ports on 192.168.0.2 are closed

Nmap done: 1 IP address (1 host up) scanned in 0.32 seconds
root@kali:~# nmap sneaky 192.168.0.2

Starting Nmap 7.60 ( https://nmap.org ) at 2018-05-16 18:20 UTC
Failed to resolve "sneaky".
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
 Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.0.2
Host is up (0.0012s latency).
All 1000 scanned ports on 192.168.0.2 are closed

Nmap done: 1 IP address (1 host up) scanned in 0.27 seconds
root@kali:~# 
```

Fig. 2. Screenshot of NMAP scanning.

Although the XMas Tree attack is considered as easily detectable, Suricata IPS was unable to detect it. That is why a rule was added that Suricata could detect the attack. The rule checks whether incoming traffic contains TCP packets with Fin, PSH & URG flags. The rule is written as follows:

- *alert tcp !$HOME_NET any -> $HOME_NET any (flags: FPU; msg:"Nmap XMAS Tree Scan"; sid:1000006; rev:1;)*

The attack is successfully detected by Suricata with this presented activated rule. The rule for blocking TCP packets with Fin, PSH & URG flags is than written as:

- *drop tcp !$HOME_NET any -> $HOME_NET any (flags: FPU; msg:"Nmap XMAS Tree Scan"; sid:1000006; rev:1;)*

Network traffic was captured using Wireshark tool while performing DoS attacks tests, i.e., the SYN flood attack is one of these DoS attacks. Suricata IPS was unable to detect network traffic with number of packet exceeding 85.000 packets per second. That is why a rule was added that Suricata could detect the attack. The rule checks whether the number of packets arriving at port 80 exceeds 1000. The rule was written as follows:

- *alert tcp !$HOME_NET any -> $HOME_NET 80 (flags: S; msg:"Possible TCP DoS"; flow: stateless; threshold: type both, track by_src, count 1000, seconds 10; sid:10001;rev:1;)*

This activated rule results in successfully detection of network traffic with more than 1000 packets per second. In IPS Suricata mode, the drop rule can be added, i.e., the alert action is replaced by drop action in presented rule, and the network traffic is than blocked.

The ICMP Flood attack was transmitted from the eth0 interface to the victim's IP address. Suricata was unable to detect the attack without setting the following rule.

- *alert icmp !$HOME_NET any -> $HOME_NET any (msg:"Possible ICMP Flood"; icode:0; itype:8; classtype:misc-activity; sid:2100384; rev:6;)*

By replacing the alert action with the drop action, the Suricata IPS was capable of blocking ICMP Flood attack.

Penetration tests of Suricata show that open version of emerging threats rule-set cannot prevent all types of attacks. Future work will deal with tests of pro version of emerging threats rule-set and other rule-sets.

V. CONCLUSIONS

This paper presents penetration testing method of IDS/IPS tool Suricata implemented in low-performance embedded IoT device Raspberry Pi 3. Vulnerability scanning tools, penetration testing tools and network attacks are discussed. Main contribution of our work is comparison of the Suricata IPS response implemented on the Raspberry Pi 3 to various tools and types of network attacks. Suricata is installed with default set of rules. These rules were tested and new rules were added in order to avoid the XMas Tree attack, ICMP Flood attack, and the SYN flood attack. Experimental use of the IDS on an embedded platform is not usually implemented in real networks, because IPS typically analyzes a large amount of data and for these platforms this type of analysis is very demanding. Suricata on Raspberry Pi 3 has proven to be usable in smaller IoT networks where there is not much data traffic.

REFERENCES

[1] D. A. Bhosale and V. M. Mane, "Comparative study and analysis of network intrusion detection tools", in *2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, 2015, pp. 312-315.

[2] P. Dorosz, "Intrusion Detection Systems (IDS) Part 2 - Classification; methods; techniques," 15 6 2004. [Online]. Available: https://www.alienvault.com/solutions/cloud-based-intrusion-detection.

[3] N. Wattanapongsakorn, „A Practical Network-Based Intrusion Detection and Prevention System," IEEE Xplore, 6 9 2012.

[4] S. Latha and S. J. Prakash, "A survey on network attacks and Intrusion detection systems", in 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS), 2017, pp. 1-7.

[5] C. -K. Chen, Z. -K. Zhang, S. -H. Lee, and S. Shieh, "Penetration Testing in the IoT Age", Computer, vol. 51, no. 4, pp. 82-85, 2018.Comparative Study and Analysis of Network Intrusion Detection Tools

[6] K. Thongkanchorn, S. Ngamsuriyaroj, and V. Visoottiviseth, "Evaluation studies of three intrusion detection systems under various

attacks and rule sets”, in 2013 IEEE International Conference of IEEE Region 10 (TENCON 2013), 2013, pp. 1-4.

[7] K. Vinod a O. P. Sangwan, “Signature Based Intrusion Detection System Using SNORT,“ International Journal of Computer Applications & Information Technology, sv. I, n. 3, 2012.Fff

[8] S. Zanero, “Flaws and frauds in the evaluation of IDS/IPS,“ DEI - Politecnico di Milano, Miláno, Itálie, 2013.

[9] L. L. -. M. I. o. Technology, “DARPA Intrusion Detection evaluation,“ [Online]. Available: https://www.ll.mit.edu/ideval/docs/index.html.

[10] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set”, in 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1-6.

[11] N. C. S. Centre, “Penetration Testing Guidance,“ 9 8 2017. [Online]. Available: https://www.ncsc.gov.uk/guidance/penetration-testing.

[12] B. Mahmood, “The Difference between Vulnerability Scanning and Penetration Testing,“ 04 07 2017. [Online]. Available: https://www.tripwire.com/state-of-security/vulnerability-management/difference-vulnerability-scanning-penetration-testing/.

[13] P. Rubens, “Are There Open Source Vulnerability Assessment Options?,“ 26 01 2016. [Online]. Available: https://www.esecurityplanet.com/open-source-security/are-there-open-source-vulnerability-assessment-options.html.

[14] Nmap, “Nmap.org,“ [Online]. Available: https://nmap.org/.

[15] P. Kim, Hacking - praktický průvodce penetračním testováním, Brno: Zoner Press, 2015.

[16] Kali.org, “Kali Tools - Nmap,“ 15 2 2014. [Online]. Available: https://tools.kali.org/information-gathering/nmap.

[17] “Metasploit.com,“ [Online]. Available: https://www.metasploit.com/.

[18] Offensive-security.com, “Buiding A Module,“ [Online]. Available: https://www.offensive-security.com/metasploit-unleashed/building-module/.

[19] V. Mohan a J. Anuradha, “Network Security and Types of Attacks in Network,“ Science Direct, sv. 48, pp. 503-506, 2015.

[20] S. Latha a S. Prakash, “A Survey on Network Attacks and Intrusion Detection Systems,“ IEEE Xplore, 24 8 2017.

[21] “Raspberry Pi,“ [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/.

[22] “Christmas tree packet,“ 15 7 2017. [Online]. Available: https://en.wikipedia.org/wiki/Christmas_tree_packet.