Journal of Theoretical and Applied Information Technology
31st August 2021. Vol.99. No 16

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

PERFORMANCE ANALYSIS OF IoT-ENABLED DDoS BOTNETS IN WEARABLE DEVICES

[1] SHWETARANI, [2] NAWAB MUHAMMAD FASEEH QURESHI, [3] DONG RYEOL SHIN

[1] Department of Electrical & Computer Engineering, Sungkyunkwan University, South Korea

[2] Department of Computer Education, Sungkyunkwan University, South Korea

[3] Department of Electrical & Computer Engineering, Sungkyunkwan University, South Korea

E-mail: [1]shwetamora@skku.edu, [2] faseeh@skku.edu, [3]drshin@skku.edu

Corresponding Author : Nawab Muhammad Faseeh Qureshi

ABSTRACT

Wearable devices (WD) such as smartwatches, fitness trackers, Medical wearables, smart headphones, smart glasses, and smart clothing, etc. are gaining popularity in recent years as the number of users is increasing. With sensing, computing, and communication capability wearable devices are forming a new segment called Wearable Internet-of-Things (WIoT). There is a high chance for these WIoT devices to be new sources of attack for malicious activities such as botnet attacks. The goal of our research is to present a performance analysis of DDoS capable IoT botnets in wearable devices. In the first part, we will show the possibilities of deploying these WIoT devices in DDoS attacks. To demonstrate this, we conducted repeated UDP, TCP, HTTP, and ICMP flooding attacks using BoNeSi DDoS botnet simulator, targeting a wearable device. We also proposed a mitigation technique using Cuckoo Filter (CF) which is designed based on benign source information.

Keywords: *Wearable Device, Botnets, IoT Botnets, DDoS Attacks, BoNeSi DDoS Botnet Simulator, Cuckoo Filter.*

1. INTRODUCTION

A botnet has become one of the most serious security issues for many internet-related security issues. For understanding Botnets, it is important to understand what a bot is, botmaster, and command and control system (c & c) [1]. A bot is a type of malware also known as Zombie, is a short form of Robot. Botmaster installs this malware into compromised devices and controls using a command and control system. So, we can say botnet is an intelligent network formed using bots, these bots are remotely handled by the botmaster. Botmaster controls botnet using the c & c system. Based on the communication methods used by botmaster botnets are classified as Direct, Centralized, Decentralized, or Peer-to-Peer, and Hybrid [2]. Botnets are being used for data stealing, spam sending, DDoS attacks, which also makes easy to collect personal information of users by accessing the device. The growing advances in IoT and IoT's increasing popularity have made IoT devices an easy and alluring target for botmasters. As we know IoT is interconnected computing, digital and mechanical-devices group that communicates via the internet also sends and receives data via the internet. Sensors used in these and their processing power made these adaptable in many environments [3]. IoT is used for many applications like home automation, health care appliances, smart city, smart grids, smart retails, autonomous cars, industrial automation, and many other fields. IoT devices are capable of creating and analyzing an individual's information and can take actions according to those analyzed information [4]. But most of the IoT devices use default passwords which makes bots get access easily. As IoT devices constantly connect to the internet, use an ever-growing array of networked systems, and as many use default passwords allows for easy access. Botmaster can build and use spam mailing, bitcoins and DDoS attacks, etc. within minutes using IoT botnets. Physical attack,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

network attack, software attack, and encryption attack are the classes of IoT botnet attacks [5]. DDoS attacks using IoT devices are the biggest threat to network security issues. DDoS attacks are the extended versions of denial-of-service (DoS) attacks. Usually, in DDoS attacks, the botmaster intends to increase their botnet size as much as possible by adding more and more bots. When we see DDoS attacks history Mirai botnet has become the biggest IoT based DDoS attack in recent years. Advances in technology, components miniaturization, power sources efficiency, and alternatives for network solution, newly introduced sensors contributed to the wearable devices development. Wearable devices are made to be used for several kinds of purposes. Wearable devices have high potential and many known benefits, but their growing usage gives rise to many privacy suggestions. Wearable devices handle personal information of users by collecting, transmitting, and storing the data continuously. This information can be publicly available, where it can be shared with a network of known or unknown or untrusted parties. WDs collects data about the users along with their surrounding may lead to serious privacy issues, risks, and threats. When this data is misused it affects not only the individual users but also the involved surroundings such as society and the enterprises. Wearable devices are being applied in several application fields such as fitness tracking, health-care appliances, Security, home automation, entertainment. Their growing popularity, autonomy, and small size increase the potential to be used in different activities and scenarios. So, it is important to understand the possibilities of malicious activities to be performed using these wearable devices. For fulfilling this purpose here in our work, we will present performance analysis of DDoS capable IoT botnets in wearable devices. Firstly, we will show the possibilities of deploying these WIoT devices in DDoS attacks by conducting repeated DDoS attacks using the DDoS botnet simulator BoNeSi, targeting a wearable device. The outcomes include wearable device sending RST (reset) packet response for confirming the denial of service, resetting the device IP address, and losing the connection with the tracking system. We also proposed a mitigation technique using Cuckoo Filter (CF) designed based on benign source information and tested using some benign and malicious data sets. Further, our paper will be arranged as in section 2 we present the related work of this research work which includes Mirai IoT botnet analysis and wearable device working. Section 3 presents the proposed attack methodology followed by the proposed attack method's evaluation and results in section 4. Section 5 presents the proposed mitigation technique followed by the mitigation technique's evaluation and results in section 6. At last, section 7 concludes our paper.

2. RELATED WORK

Internet-of-Things (IoT) devices have become a target for various malicious activities, one of such serious malicious activities is DDoS attacks. DDoS attack history shows Mirai botnet has become the biggest and predominant IoT based DDoS attack in recent years.

2.1 Mirai Botnet

Malware Must Die research group detected Mirai malware in 2016, August. Mirai is a Japanese word meaning "the future" [6]. Mirai malware has the ability to control and infect DVRs, home routers and CCTV camera etc. In [6], the authors explained the main components of the Mirai botnet, step by step operation and communication methods, Mirai variants along with the impact of Mirai botnet. In [7], the authors explained the botnets and DDoS attacks along with the Mirai DDoS working and family of Mirai malware. A detailed classification of DDoS attacks along with real-world attack examples are given in [8], authors have also given the analysis of Mirai's framework and operating principles. In [9] authors showed details about IoT botnets and how these botnets are becoming successful with the example of Mirai and a few other such malware along with the strategies for protecting from IoT malware. In [10], authors have presented the growth of Mirai malware upto 600k infections over a seven months period. The authors also showed which types of devices got affected and how the various forms of Mirai malware being developed. In [11], authors presented an analysis of issues of mirai malware and methods for predicting it in IoT. In [12], the authors showed how to use an online available Free and Open-Source Software (FOSS) for detecting, classifying, and removing malware from infected systems along with Mirai's in-depth security analysis. In [13], the authors presented a strategy to show the security issues in IoT devices by deploying the same compromise vector as Mirai malware. Mirai,s attack is based on a 62

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

entry dictionary. Once compromised devices are reported to a control server to use as a part of a large botnet[14]. After being added into a large-scale-botnet it can be exploited in various DDoS attacks. Mirai Malware's main components and its method of operation and communication are explained in detail in [6]. Bot, c & c server, loader, and report-server are the major parts of the Mirai Botnet. The malware which compromises the devices and tries to extend its botnet by targeting other vulnerable devices is known as Bot. The bot can attack the target server immediately after getting the attack- command from its botmaster. The c & c server center of the management of the botnet. It checks the condition of the botnet and plans the new DDoS attacks accordingly. Communication in a botnet is usually conducted by the Tor network. The loader targets different platforms such as ARM, MIPS, and x86, etc. in total 18, by spreading of executables. The database of all the devices in the botnet is maintained by the report server. Newly compromised devices directly communicate with the report server. The c & c can work as a database server of MYSQL. User accounts are created in such database for those wish to hire DDoS service. Mirai has SYN flood, UDP flood, ACK flood, UDP plain flood, TCP stomp flood, DNS server flood, GRE Ethernet flood, valve source engine specification flood, and HTTP flooding attack options. Mirai's basic structure and its source code analysis are explained in detail in [8] for a better understating of its working. The authors have given a detailed explanation of how the Mirai malware attack history, its overview, how it operates along with each folder of its source code available on GitHub. In [15] the authors presented an analysis of Mirai malware with the help of a virtual environment. They have also explained the settings needed to install on run Mirai on the proposed virtual environment for the easy understanding of Mirai malicious software. They have provided both the static analysis and dynamic analysis of Mirai. In [9] the authors presented the high-level details of Mirai malware and its variant malware. The bot code of Mirai is in C programming language which searches for IP addresses of new vulnerable devices and reasons for causing DDoS attacks. Most of Linux based IoT devices have been a target for Mirai malware. c & c server of Mirai is in GO programming language which is the main part of the Mirai botnet for communication. Till now Mirai malware has targeted 18 different platforms few are Intel x86, ARM, MIPS, and SPARC. Mirai malware targeted Telnet 23 port and TCP port 2323. For finding login credentials attacker used brute force dictionary-based techniques, which contains 62 pairs of username and passwords. The authors also presented the Mirai botnet's variant Hajime botnet overview along with the comparison of both botnet's basic functions.

2.2 Wearable Technology

For wearable devices, it is important to maintain privacy and transmit data securely over the network as these devices transmit vital personal information. Wearable device's system with wireless communication is a new frontier technique which adds another layer to human and machine interaction [16] [17]. An example framework of a wearable technology is shown in figure 1 [18]. Many previous works have shown how wearable devices can be involved in malicious activities. In [19] authors showed how wrist wearables are used by the attacker to know the mechanical lock's unlock combination by presenting deterministic and probabilistic attack frameworks. In [20] authors showed a security issue exploiting a pebble-smartwatch. They developed a appalication as a malware which helped students to cheat in multiple choice exam. In [21] authors presented a work to collect PINs entered on smartphones when smartphone users are wearing a smartwatch being compromised by the botmaster. The authors used a random forest classifier for collecting the PINs entered by users. In [22], presents a attack on numeric touch screen of smartphones based on keystroke inference when the user is wearing smartwatch. In [23] the authors developed a method called MoLe to reveal what a user typing on their keyboard

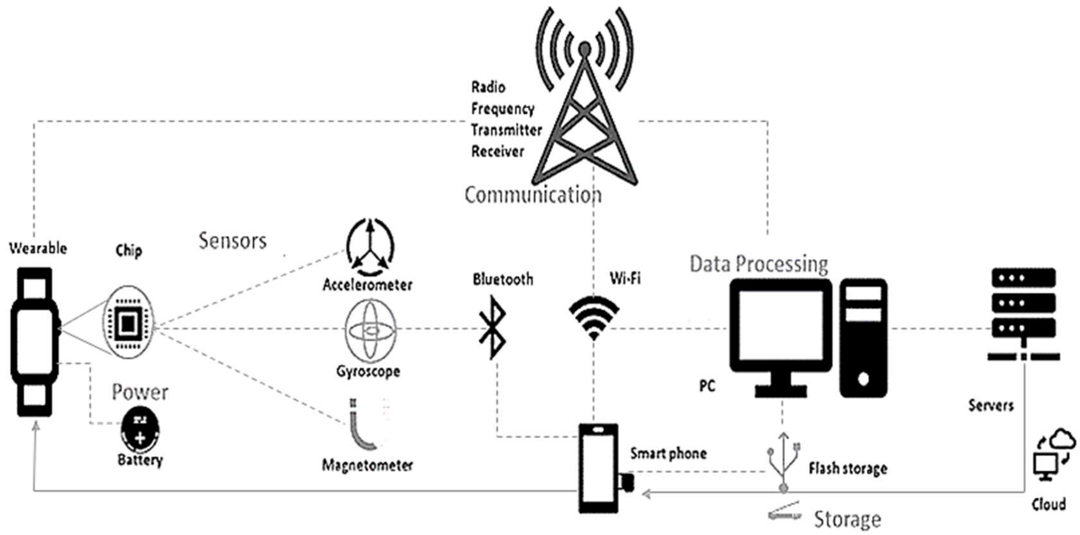


Figure 1. Wearable technology framework

when wearing smartwatch. Specially designed for samasung gear live smartwatch. In [24] the authors proposed a method to get the inputs on keyboard using smartwatch sensors developed through a side-channel attack. Many research work has been done to present the privacy-related issues in collecting the personal data of individual users of wearable devices along with some other security issues but as of our best knowledge, our work is the first to present the threats related to DDoS attacks on wearable devices. The wearable device used in this research work is Tizen based Samsung Galaxy Watch, as a target device.

3. PROPOSED ATTACK METHOD

In our proposed attack methodology we are using Tizen based Samsung Galaxy Watch (SGW) [25] as a victim or target wearable device (VWD). Exploiting BoNeSi DDoS botnet simulator for performing DDoS attack experiments on SGW.

3.1 Closed Testbed Environment

The closed testbed environment for the proposed attack method is illustrated in figure 2, which includes a laptop with Ubuntu OS as an attacker device. Tizen based SGW as a VWD. The attacker device and the VWD are connected to the same WiFi.

3.2 Technical Flowcharts

The technical flowchart is shown in figure 3 explains the details of the DDoS attack,

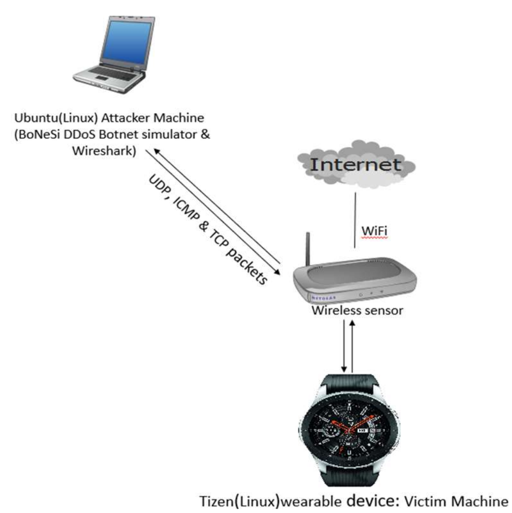


Figure 2. Closed Testbed Environment

a process performed using the BoNeSi simulator. BoNeSi usage options are given in figure 5. After the installation of the DDoS simulator tool on the attacker device, start the Wireshark and then start attacking the VWD with the available UDP, TCP, and ICMP flooding packets. Once the attack begins, we can analyze the network packets using Wireshark and record the details of attack packets.

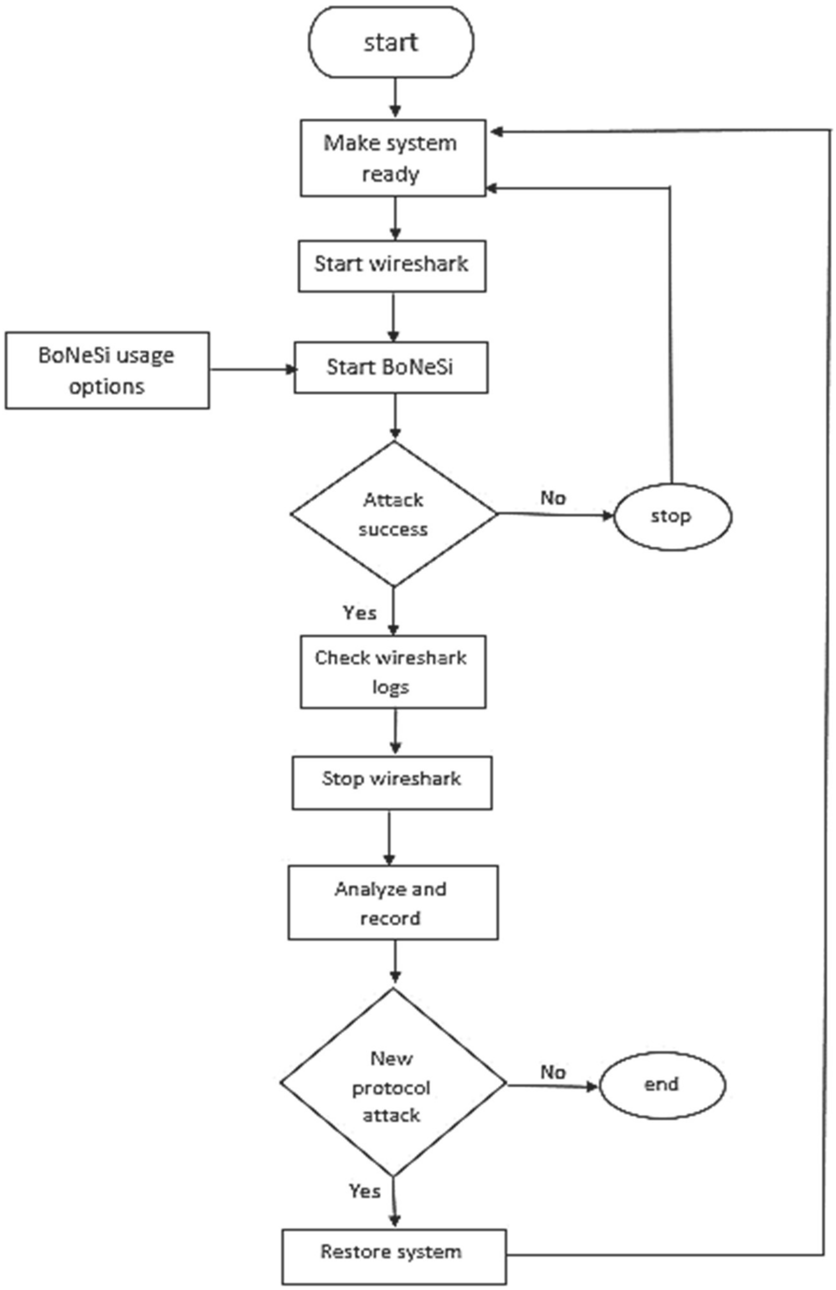


Figure 3. Flowchart explaining the attack process performed on victim wearable device using BoNeSi DDoS botnet simulator

4. EVALUATION AND RESULTS OF ATTACK METHOD

4.1 Tools Used in Attack Method

4.1.1 Bonesi DDoS botnet simulator [26], used to generate network-traffic for several types of protocols. Using this simulator one can produce flooding attacks such as ICMP, UDP, and TCP/HTTP attacks of specified size of botnet having various IP addresses. This simulator can simulate traffics generated by botnet on wire based testbed environment. BoNeSi can spoof IP addresses of sources even while TCP traffic is being generated which made BoNeSi simulator to have simple stack of TCP to handle TCP connections. It is important to send response-packets to host where BoNeSi is running to get best results. Many tools are available for spoofing IP addresses using UDP and ICMP. But not many tools are available for TCP. For HTTP-GET flood simulation BoNeSi is first used tool.

4.1.2 Wireshark [27], is a network packet analyzer that analyzes every packet getting

through a network interface. Wireshark has a graphical front-end along with integrated sorting and filtering options. Here we used Wireshark to record and analyze the malformed packet sent to the victim's machine.

4.1.3 Samsung galaxy watch [25]**,** in this study we used the Tizen-based Samsung Galaxy Watch as a target or victim wearable device. For getting the IP address of the Samsung Galaxy Watch followed the steps explained in [28]. Hacking wearable device is not a part of our work, we have assumed the wearable device is already hacked or compromised.

4.2 System Configurations, System configurations of these attacker and victim devices are provided in Table 1 for attack experiments performed using the BoNeSi simulator.

Table 1. System Configurations used for performing DDoS attacks on VWD using BoNeSi DDoS botnet Simulator.

Systems used	OS	RAM
Laptop (attacker)	Ubuntu 20.04.1 LTS	8 GB
Samsung Galaxy Watch(victim)	Tizen 4	1.5 GB

4.3 Attack Using BoNeSi

Ubuntu Laptop is used with below network information, as shown in below figure 4

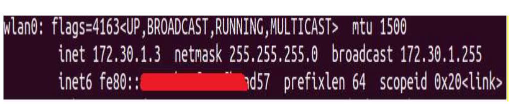


Figure 4. Used Network Details

4.3.1 Installing BoNeSi on attacker device

Step 1: Login into the attacker device that is a laptop with Ubuntu OS.
Step 2: Open the "Terminal" to install BoNeSi by using "shell" commands.
Step 3: Firstly, Install BoNeSi dependency libraries by running the below commands, to compile the BoNeSi tool successfully.

```
sudo apt get update
sudo apt install build-essential
sudo apt install make
sudo apt install autoconf
sudo apt install libpcap-dev
sudo apt install automake
sudo apt install gcc
sudo apt install git
sudo apt install libnet1-dev
```

Step 4: Install BoNeSi by running the below commands

```
git                                    clone
https://github.com/MarkusGo/bonesi.git
autoreconf -f -i./configure
make
make install
```

Now BoNeSi is ready to be run.
Step 5: The installation of Wireshark on Ubuntu is done by following steps in [29].

Step 6: Run Wireshark in the terminal.

```
$wireshark
```

Step 7: Run the BoNeSi in the terminal.

```
$bonesi
```

We get the output as shown in figure 5 with BoNeSi usage options.
BoNeSi needs a wearable device's IP address

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

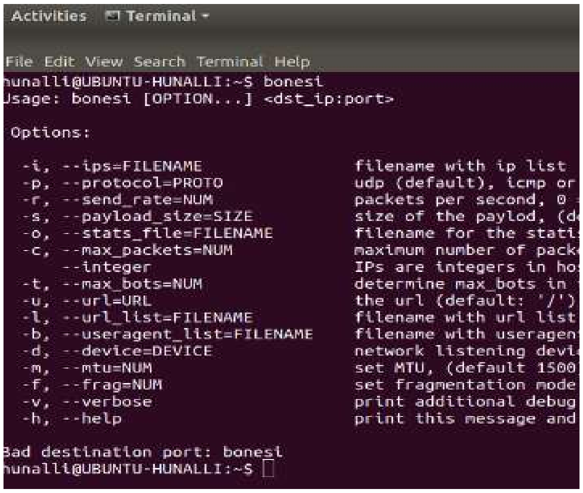


Figure 5.BoNeSi usage options

[28] to start the flooding attack by using the different methods shown in Figure 6(a),(b)and (c),7(a),(b) and (c) and 8(a),(b) and (c). 50k-bot is an additional option given by the BoNeSi tool which is generated randomly by exploiting 50,000 IP addresses, This file is part of the BoNeSi package. We used the 50k-bot file as a parameter to attack the wearable devices with the "TCP" protocol. Figure 6,7 and 8 gives the BoNeSi attack methods,Wireshark logs and related graphs for UDP,ICMP and TCP respectively on Ubuntu.

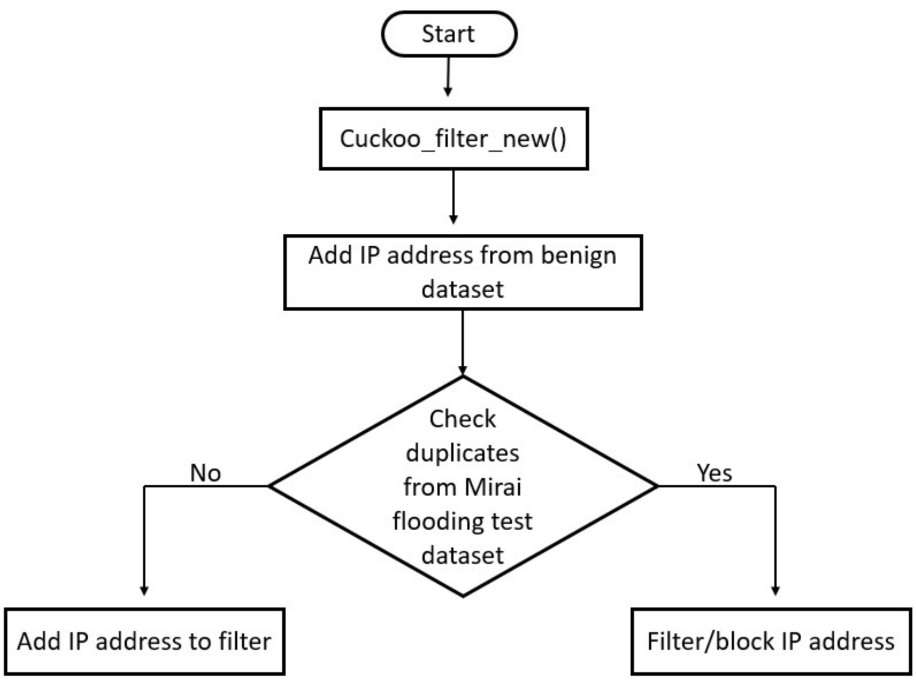


Figure 9. Flow chart for proposed Cuckoo filter-based mitigation technique

5. METHODOLOGY OF PROPOSED DDoS ATTACK MITIGATION TECHNIQUE

Cuckoo Filter (CF), is used for checking the presence of an element in a set which is probabilistic data structure and space

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

efficient[30]. Cuckoo Filter gives results "possibly a member of the set" or "definitely not a member of the set". In other words CF can give false positive results but not false negative results. CF has the advantages of less space, deleting existing items in the filter, and low false-positive rates. The disadvantage of the Cuckoo Filter is the complexity in adding the items to the filter. CF has a hash-table called cuckoo which stores the item's frinerprints needs to store in CF using cuckoo hash function [31]. Cuckoo Filter uses a cuckoo hashing [32] based n-way set associative hash-table to store the fingerprints of all items. Fingerprints uniquely identify each item by a bit string. A cuckoo hashtable maps an item of tye set into two possible bukets of an array of bukets using two hash-functions as shown below equations,

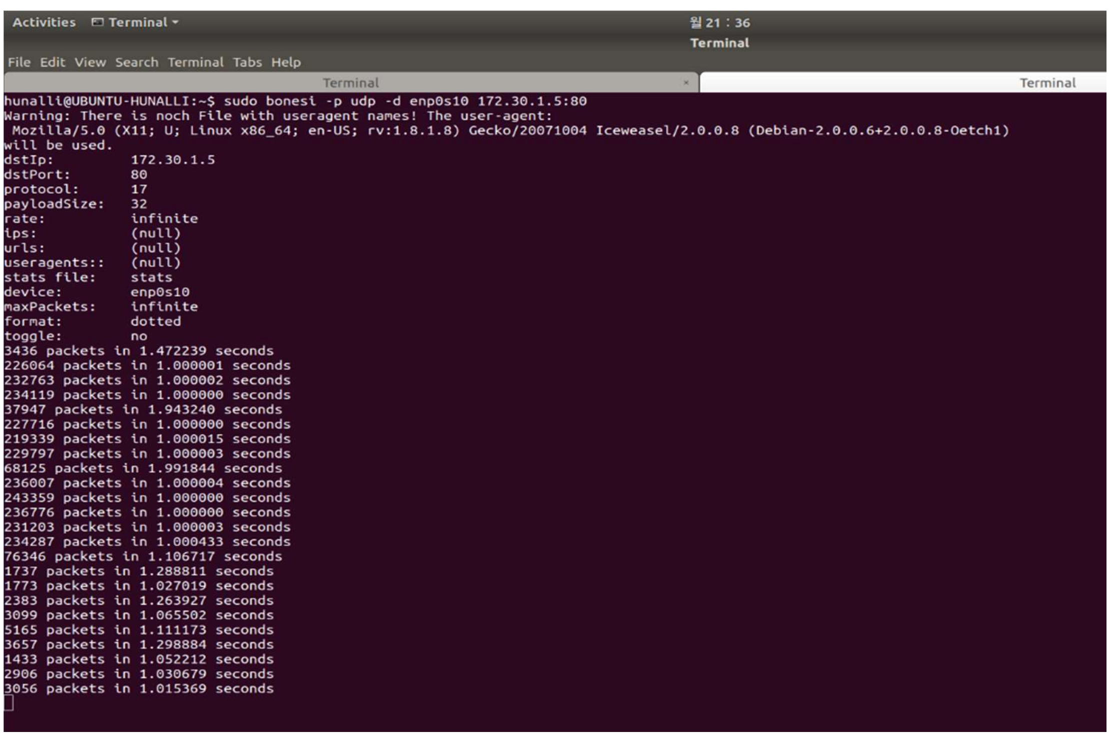
$$i = h_1(x) = hash(x)$$

$$j = h_2(x) = h_1(x) \oplus hash(fingerprint\ (x))$$

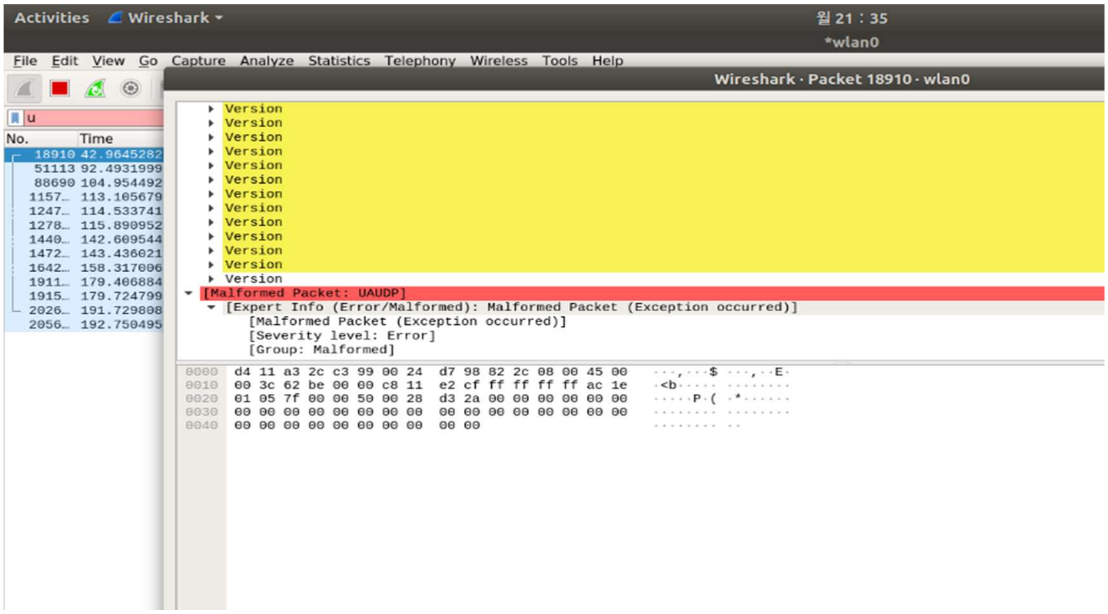
As it stores only the fingerprints of items and uses partial key cuckoo hashing, the cuckoo filter can achieve both high utilization and compactness. The cuckoo filter requires bits of space per inserted key is,

$$(\log_2(1 / \epsilon) + 2) / \alpha$$

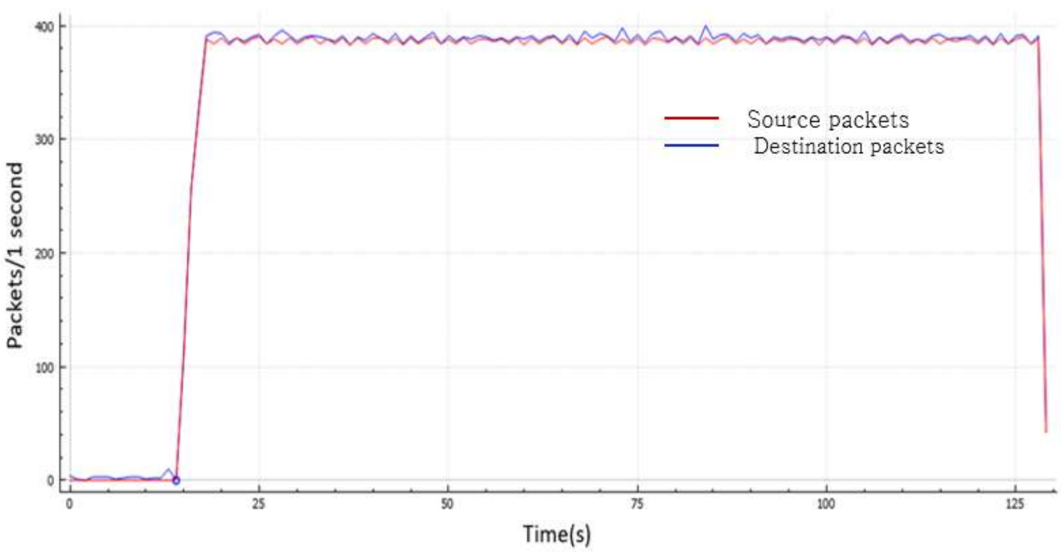
where α is the hash table load factor. Delete and lookup operations are easy, as there are a maximum of two locations to check by $h_1(x)$ and $h_2(x)$. Figure 9 illustrates the working flow of the proposed cuckoo filter-based mitigation technique. Constructed the Cuckoo filter using the cuckoo hash function by adding the collected benign source IP address data. And then for test purposes, we used Mirai UDP and Mirai HTTP flooding data. If the test IP address is already present in the cuckoo filter no need to add it to the cuckoo filter and we can just block from further communication. If the IP address is not present then add it to the cuckoo filter in its first access request, and block if the repeated access requests are received.



6(a)

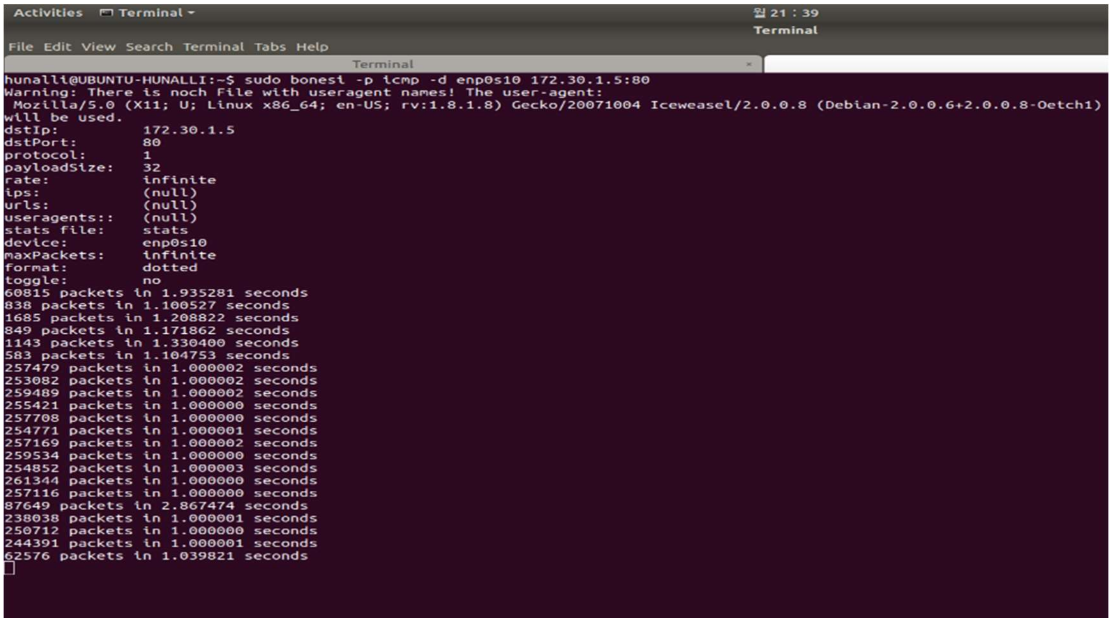


6(b)

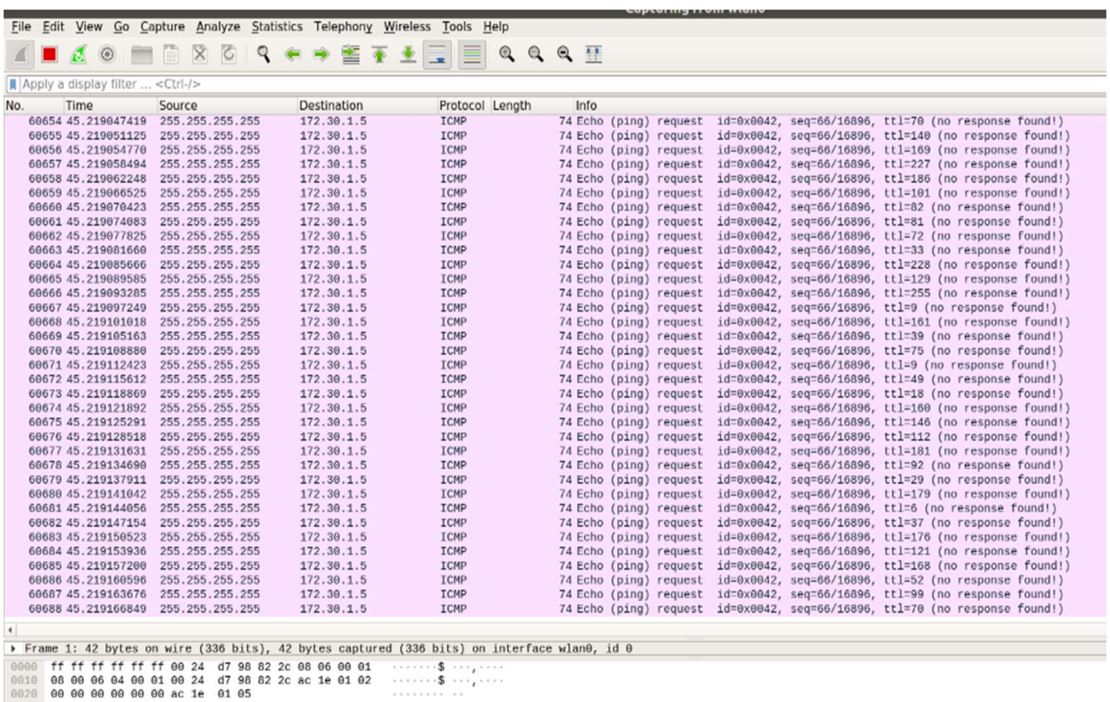


6(c)

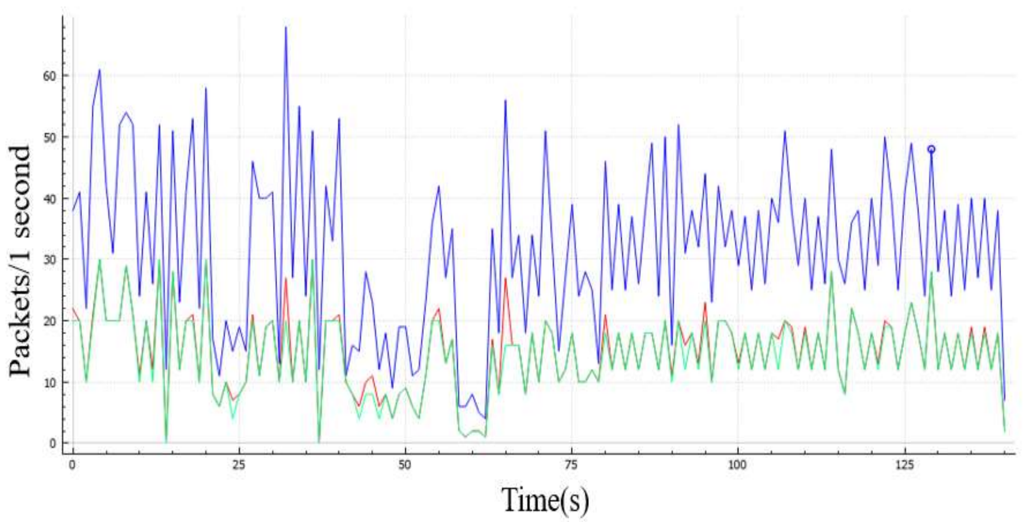
Figure 6. BoNeSi attack method, Wireshark logs, and Related Graph using UDP Packets


Activities Terminal
월 21 : 39
Terminal
File Edit View Search Terminal Tabs Help
Terminal
hunalli@UBUNTU-HUNALLI:~$ sudo bonesi -p icmp -d enp0s10 172.30.1.5:80
Warning: There is noch File with useragent names! The user-agent:
Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.8.1.8) Gecko/20071004 Iceweasel/2.0.0.8 (Debian-2.0.0.6+2.0.0.8-0etch1)
will be used.
dstIp: 172.30.1.5
dstPort: 80
protocol: 1
payloadSize: 32
rate: infinite
ips: (null)
urls: (null)
useragents:: (null)
stats file: stats
device: enp0s10
maxPackets: infinite
format: dotted
toggle: no
60815 packets in 1.935281 seconds
838 packets in 1.100527 seconds
1685 packets in 1.208822 seconds
849 packets in 1.171862 seconds
1143 packets in 1.330400 seconds
583 packets in 1.104753 seconds
257479 packets in 1.000002 seconds
253082 packets in 1.000002 seconds
259489 packets in 1.000002 seconds
255421 packets in 1.000000 seconds
257708 packets in 1.000000 seconds
254771 packets in 1.000001 seconds
257169 packets in 1.000002 seconds
259534 packets in 1.000000 seconds
254852 packets in 1.000003 seconds
261344 packets in 1.000000 seconds
257116 packets in 1.000000 seconds
87649 packets in 2.867474 seconds
238038 packets in 1.000001 seconds
250712 packets in 1.000000 seconds
244391 packets in 1.000001 seconds
62576 packets in 1.039821 seconds

7(a)

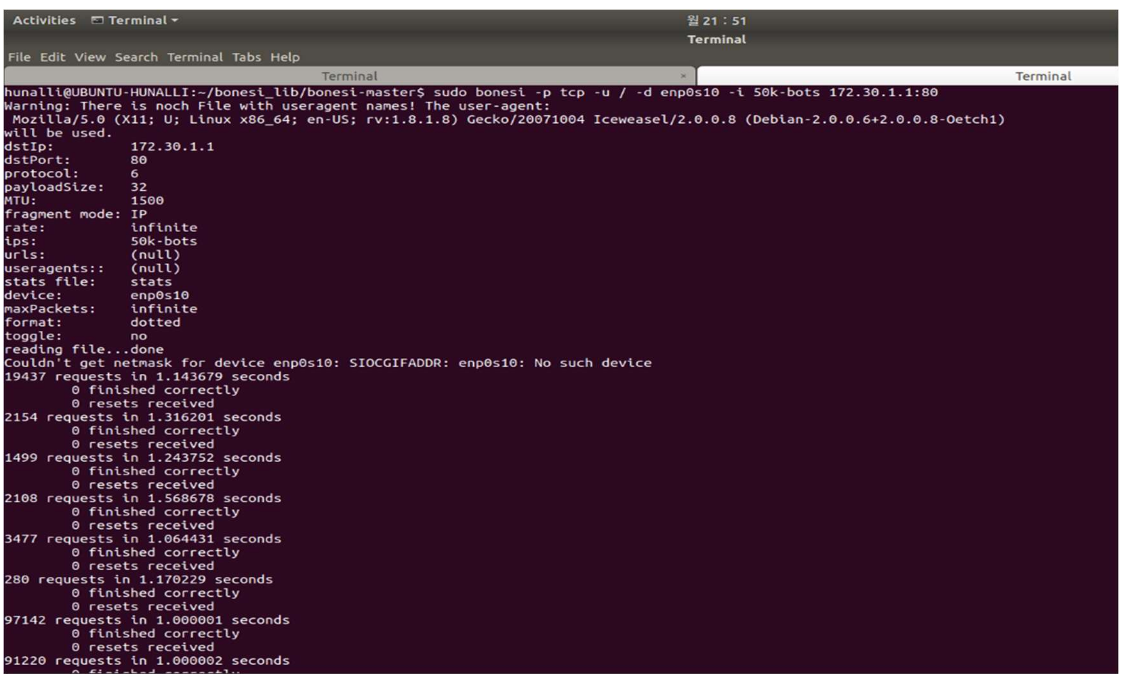

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
60654 45.219047419 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=70 (no response found!)
60655 45.219051125 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=140 (no response found!)
60656 45.219054770 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=169 (no response found!)
60657 45.219058494 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=227 (no response found!)
60658 45.219062248 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=186 (no response found!)
60659 45.219066525 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=101 (no response found!)
60660 45.219070423 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=82 (no response found!)
60661 45.219074083 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=81 (no response found!)
60662 45.219077825 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=72 (no response found!)
60663 45.219081660 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=33 (no response found!)
60664 45.219085666 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=228 (no response found!)
60665 45.219089585 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=129 (no response found!)
60666 45.219093285 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=255 (no response found!)
60667 45.219097249 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=9 (no response found!)
60668 45.219101018 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=161 (no response found!)
60669 45.219105163 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=39 (no response found!)
60670 45.219108880 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=75 (no response found!)
60671 45.219112423 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=9 (no response found!)
60672 45.219115612 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=49 (no response found!)
60673 45.219118869 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=18 (no response found!)
60674 45.219121892 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=160 (no response found!)
60675 45.219125291 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=146 (no response found!)
60676 45.219128518 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=112 (no response found!)
60677 45.219131631 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=181 (no response found!)
60678 45.219134690 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=92 (no response found!)
60679 45.219137911 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=29 (no response found!)
60680 45.219141042 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=179 (no response found!)
60681 45.219144056 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=6 (no response found!)
60682 45.219147154 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=37 (no response found!)
60683 45.219150523 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=176 (no response found!)
60684 45.219153936 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=121 (no response found!)
60685 45.219157200 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=168 (no response found!)
60686 45.219160596 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=52 (no response found!)
60687 45.219163676 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=99 (no response found!)
60688 45.219166849 255.255.255.255 172.30.1.5 ICMP 74 Echo (ping) request id=0x0042, seq=66/16896, ttl=70 (no response found!)
Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface wlan0, id 0
0000 ff ff ff ff ff ff 00 24 d7 98 82 2c 08 06 00 01
0010 08 00 06 04 00 01 00 24 d7 98 82 2c ac 1e 01 02
0020 00 00 00 00 00 00 ac 1e 01 05

7(b)

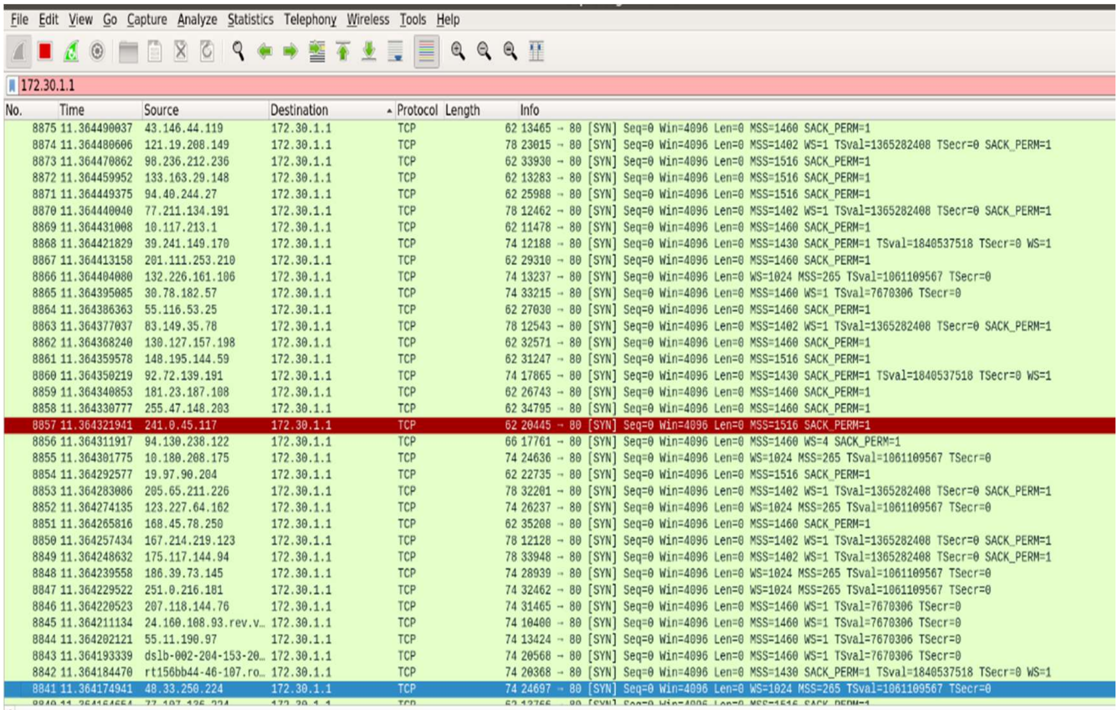


7(c)

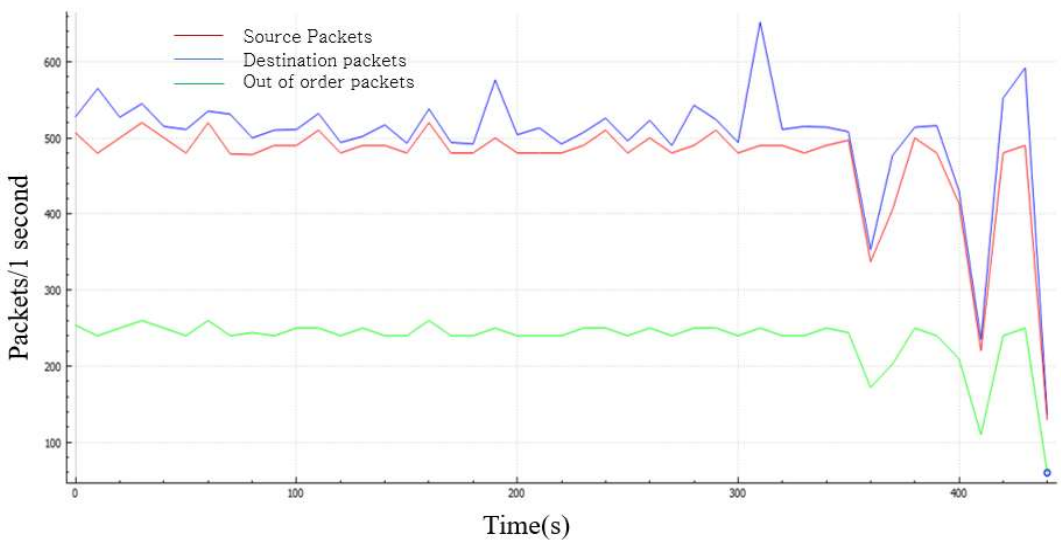
Figure 7. BoNeSi attack method, Wireshark logs, and Related Graph using ICMP Packets



8(a)



8(b)



8(c)

Figure 8. BoNeSi attack method, Wireshark logs, and Related Graph using TCP Packets

6. EVALUATION AND RESULTS OF PROPOSED DDoS ATTACK MITIGATION TECHNIQUE

In the first step of designing the CF, we created a dataset based on the signature of IP address history from the daily traffic. The signature of the IP address's frequency, size of the packet, and port number are created. A complete Three-Way handshake is not possible by a spoofed IP address, so those TCP traffic IP address which have a successful TCP handshake are considered as the benign IP addresses. To create a more reliable signature-based IP address history, for UDP traffic we ignored those incoming UDP packets that just send a single

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

packet in a day. In the second step created the CF using this dataset. CF is initialized and the hash function is applied to the element that needs to be added to the bit array. To evaluate the proposed Cuckoo Filter based mitigation technique, we used two different data sets, IoT network intrusion dataset [33] and DDoS evaluation dataset (CICDDoS2019) [34]. Using Wireshark we converted all .pcap files to a .csv file from the mentioned two data sets. This CSV
file contains multiple columns of data with different header types, such as number, time, source, destination, protocol, length, etc. We used a CPP program [35] to convert this .csvfile to a .txt file which helps to select the required data such as source IP addresses, port numbers, etc.

6.1 Datasets

6.1.1 IoT network intrusion dataset [33]**,** is created by authors in an IoT environment for academic purposes. Captured exploiting a wireless network-adapter in monitor mode at various times. The data collected includes 42 raw network-packets in pcap form. All the packets are captured using tools such as Nmap except Mirai malware packets, while simulating attacks. A laptop is used to generate Mirai malware-packets and later manipulated as if originated by IoT devices.

6.1.2 DDoS evaluation dataset (CIC-DDoS2019) [34]**,** the data collected is in pcap form which consists of genuien and also some major upto date DDoS attacks. This dataset collects the raw-data and network-traffic each day in pcap form and organizes each day data. Event logs per machine for Ubuntu is also collected. The algorithm of the proposed Cuckoo Filter is explained below.

Creates a filter , cuckooFilterNew(filter, maxKeyCount, maxKickAttempts, seed);

add IP Address to the filter ,cuckooFilterAdd(filter, IPAddress, IPAddLengthInBytes);

Check if filter contains the IP address already, If (cuckooFilterContains(filter, IPAddress, IPAddLengthInBytes)) then block else add the IP address to filter.

Finally free the filter cuckooFilterFree(filter);

6.2 Results

The three basic operations supported by CF are Add, Delete, and Lookup. We created CF by adding the benign IP addresses collected from the IP history and the benign data from the used datasets. For testing purposes, we used the malicious data packets, such as Mirai-UDP flooding & Mirai-HTTP flooding from the mentioned datasets as listed in Table 2. Figure 10, 11 , 12 and 13 shows the output of the proposed CF mitigation technique with false-positive ε *value* 0.001.

Table 2. Number of total packets used for test and number number of packets not found in Cuckoo filter

Attack types	Total packets	Blocked packets
Mirai UDP flooding	417864	2292
Mirai HTTP flooding	367880	4641

Journal of Theoretical and Applied Information Technology
31st August 2021. Vol.99. No 16

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

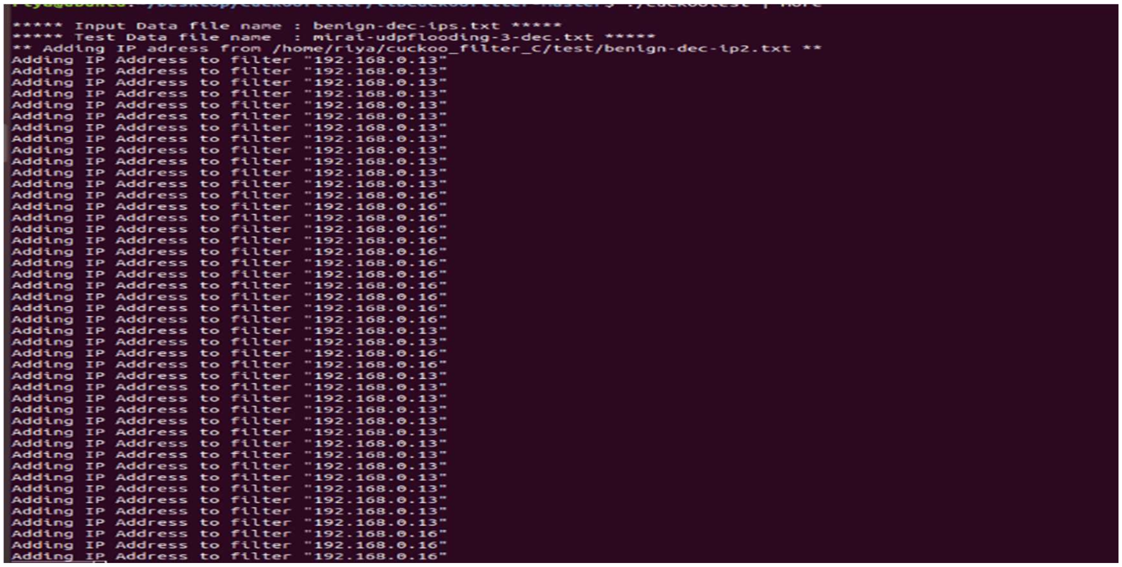


Figure 10. Adding IP Address to Cuckoo Filter

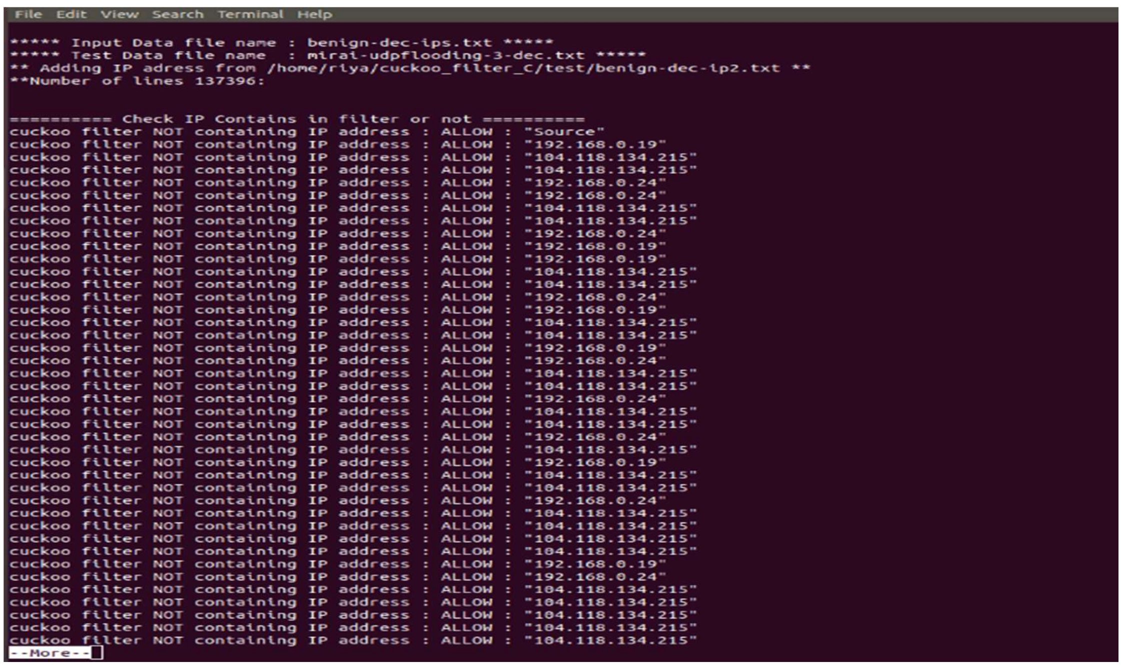


Figure 11. Cuckoo Filter Allowing Benign IPs

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

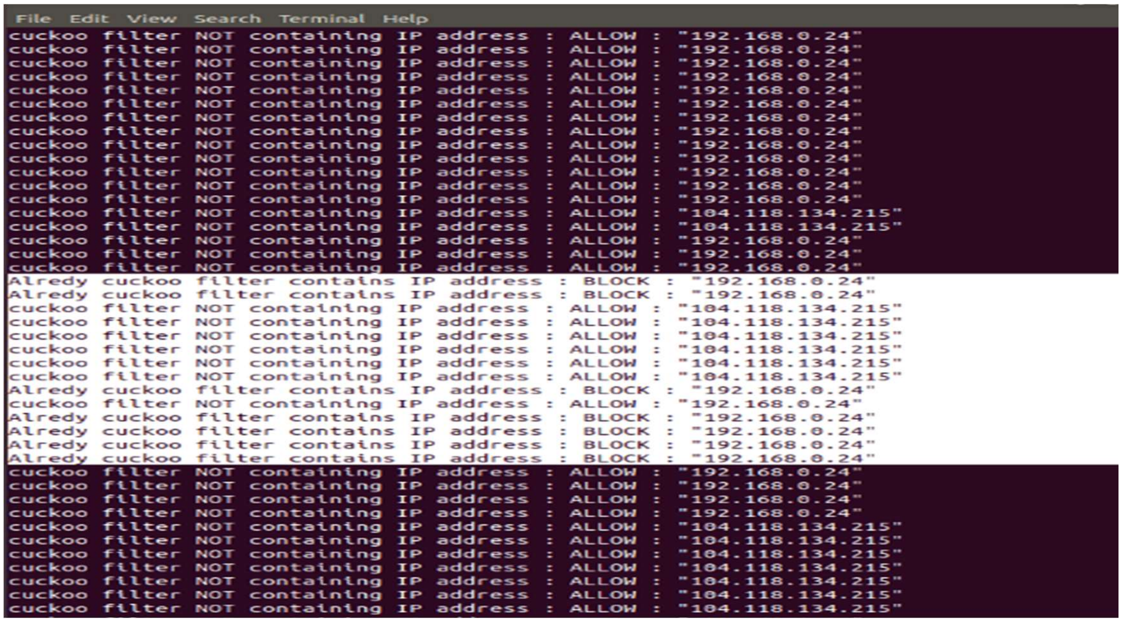


Figure 12.Cuckoo Filter Blocking Malicious IPs

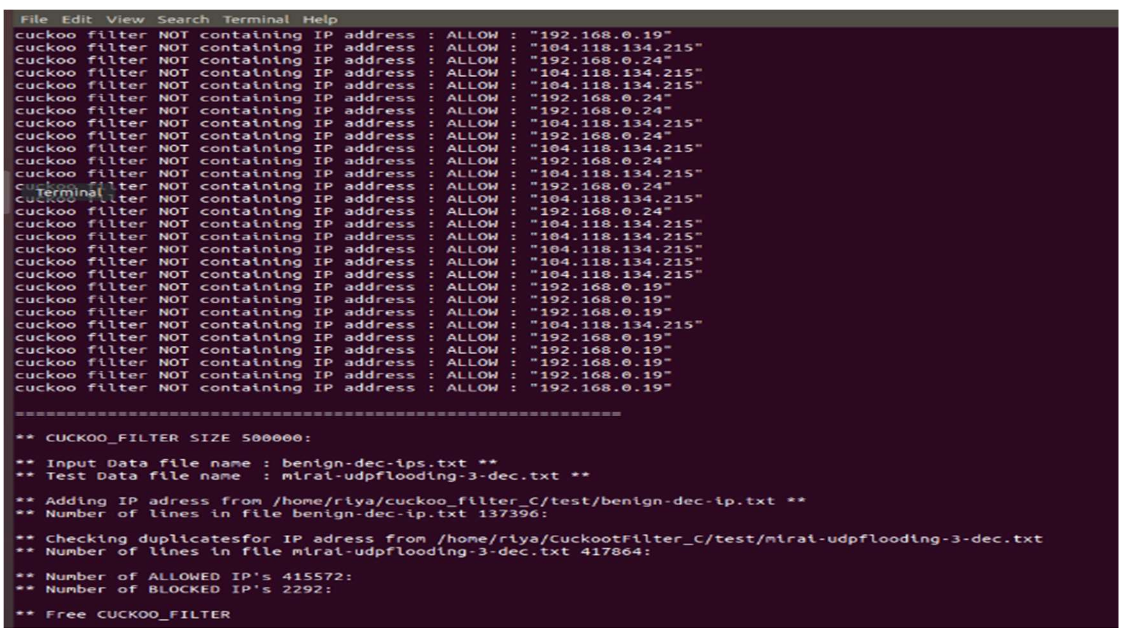


Figure 13. Total Number of Packets Allowed and Blocked/Filtered by Cuckoo Filter

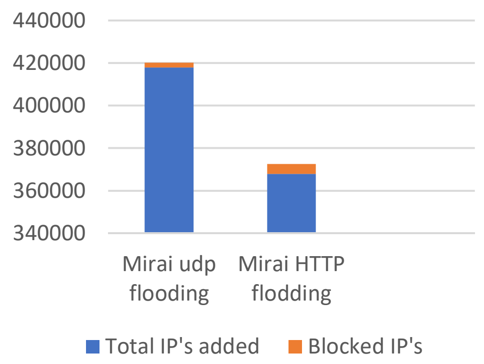


Figure 14. Total Number of Added and Blocked IPs

Figure 14 shows the number of allowed data packets and the number of filtered or blocked data packets by Cuckoo Filter. If repeated access requests are received from the added data then that data is blocked or deleted from further communication with the requested device.

7. CONCLUSION

Most of the IoT-devices connected to the internet continuously are poorly secured and use default username and password for login this attracts many attackers to perform malicious activities. Wearable devices are being used by many individual users and are gaining popularity day by day. Wearable-devices are being used in various applications as health care appliances, smart-homes, security, gaming, and entertainment, etc. These wearable devices are creating a new platform for malicious activities. Our research work showed the possibilities of deploying wearable devices in DDoS attacks, using the BoNeSi DDoS botnet simulator. We used the IP based flood attacks and our Cuckoo Filter-based DDoS mitigation technique. There is huge scope to expand the same experiment by using Bluetooth or NFC based connections.

ACKNOWLEDGMENT

This paper was supported by Samsung Research Fund, Sungkyunkwan University, 2018.

REFERENCES

[1] H. R. Zeidanloo and A. A. Manaf, “Botnet Command and Control Mechanisms,” 2009 Second International Conference on Computer and Electrical Engineering, Dubai, pp. 564-568. 2009.

[2] Emmanuel C. Ogu, Olusegun A. Ojesanmi, Oludele Awodele and Shade Kuyoro, “A Botnets Circumspection: The Current Threat Landscape, and What We Know So Far,” in Information 2019, 10(11), 337, October 30, 2019. https://doi.org/10.3390/info10110337 89ugbnmno[ij[o

[3] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, “A Survey on Security and Privacy Issues in Internet-of-Things, ” in IEEE Internet of Things Journal, vol. 4, no. 5, pp. 1250-1258, Oct. 2017.

[4] Irfan Saif and Sean Peasley and Arun Perinkolam, “Safeguarding the internet of things: Being secure, vigilant, and resilient in the connected age,” https://dupress.deloitte.com/dup-us-en/deloittereview/issue-17/internet-of-things-data-security-and-privacy.html, 2015.

[5] I. Andrea, C. Chrysostomou, and G. Hadjichristofi, "Internet of Things: Security vulnerabilities and challenges," 2015 IEEE Symposium on Computers and Communication (ISCC), Larnaca, pp. 180-187, 2015.

[6] Kolias, Constantinos, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. "DDoS in the IoT: Mirai and other botnets." Computer 50, no. 7 (2017): 80-84.

[7] Hallman, Roger, Josiah Bryan, Geancarlo Palavicini, Joseph Divita, and Jose Romero-Mariona. "IoDDoS-the internet of distributed denial of service attacks." In *2nd international conference on internet of things, big data and security. SCITEPRESS*, pp. 47-58. 2017.

[8] De Donno M, Dragoni N, Giaretta A, Spognardi A. DDoS-capable IoT malwares: Comparative analysis and Mirai investigation. Security and Communication Networks. 2018 Feb 18;2018.

[9] G. Kambourakis, C. Kolias and A. Stavrou, "The Mirai botnet and the IoT Zombie Armies," MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM), Baltimore, MD, 2017, pp. 267-272, doi: 10.1109/MILCOM.2017.8170867.

[10] Antonakakis, Manos, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric et al.

"Understanding the mirai botnet." In *26th {USENIX} security symposium ({USENIX} Security 17)*, pp. 1093-1110. 2017.

[11] Vengatesan K., Kumar A., Parthibhan M., Singhal A., Rajesh R. (2020) Analysis of Mirai Botnet Malware Issues and Its Prediction Methods in Internet of Things. In: Pandian A., Senjyu T., Islam S., Wang H. (eds) Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI - 2018). ICCBI 2018. Lecture Notes on Data Engineering and Communications Technologies, vol 31. Springer, Cham. https://doi.org/10.1007/978-3-030-24643-3_13

[12] Jaramillo LE. Malware Detection and Mitigation Techniques: Lessons Learned from Mirai DDOS Attack. Journal of Information Systems Engineering & Management. 2018;3(3):19.

[13] J. A. Jerkins, "Motivating a market or regulatory solution to IoT insecurity with the Mirai botnet code," 2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, 2017, pp. 1-5, doi: 10.1109/CCWC.2017.7868464.

[14] Mansfield-Devine, Steve. "DDoS goes mainstream: how headline-grabbing attacks could make this threat an organisation's biggest nightmare." *Network Security* 2016.11 (2016): 7-13.

[15] Sinanović, Hamdija, and Sasa Mrdovic. "Analysis of Mirai malicious software." *2017 25th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE, 2017.

[16] Sang Don Kim, Sam Muk Lee, and Seung Eun Lee. Secure Communication System for Wearable Devices Wireless Intra Body Communication. IEEE International Conference on Consumer Electronics (ICCE), 2015.

[17] A. A. Pyattaev, K. Johnsson, Sergey Andreev, and Y. Koucheryavy. Communication Challenges in High Density Deployments of Wearable Wireless Devices. IEEE Mobile Wearable Communications, 2015.

[18] Aroganam, Gobinath, Nadarajah Manivannan, and David Harrison. "Review on wearable technology sensors used in consumer sport applications." *Sensors* 19.9 (2019): 1983.

[19] Maiti, A. et al. "A Framework for Inferring Combination Lock Codes using Smartwatches." *ArXiv* abs/1710.00217 (2017): n. pag.

[20] Migicovsky, Alex, et al. "Outsmarting proctors with smartwatches: A case study on wearable computing security." *International Conference on Financial Cryptography and Data Security*. Springer, Berlin, Heidelberg, 2014.

[21] Sarkisyan, Allen, Ryan Debbiny, and Ani Nahapetian. "WristSnoop: Smartphone PINs prediction using smartwatch motion sensors." *2015 IEEE international workshop on information forensics and security (WIFS)*. IEEE, 2015.

[22] Maiti, Anindya, et al. "(Smart) watch your taps: side-channel keystroke inference attacks using smartwatches." *Proceedings of the 2015 ACM International Symposium on Wearable Computers*. 2015.

[23] Wang, He, Ted Tsung-Te Lai, and Romit Roy Choudhury. "Mole: Motion leaks through smartwatch sensors." *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. 2015.

[24] Liu, Xiangyu, et al. "When good becomes evil: Keystroke inference with smartwatch." *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. 2015.

[25]https://www.samsung.com/us/mobile/wearables/smartwatches/galaxy-watch--46mm--silver--bluetooth--sm-r800nzsaxar/

[26] https://github.com/Markus-Go/bonesi

[27]https://resources.infosecinstitute.com/topic/loic-dos-attacking-tool/

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[28] https://developer.samsung.com/galaxy-watch-develop/testing-your-app-on-galaxy-watch.html

[29]https://linuxhint.com/install_wireshark_ubuntu/

[30]https://en.wikipedia.org/wiki/Cuckoo_filter

[31]https://blog.fastforwardlabs.com/2016/11/23/probabilistic-data-structure-showdown-cuckoo-filters-vs.-bloom-filters.html

[32]https://en.wikipedia.org/wiki/Cuckoo_hashing

[33] Hyunjae Kang, Dong Hyun Ahn, Gyung Min Lee, Jeong Do Yoo, Kyung Ho Park, Huy Kang Kim, September 27, 2019, "IoT network intrusion dataset", IEEE Dataport, doi: https://dx.doi.org/10.21227/q70p-q449.

[34] Iman Sharafaldin, Arash Habibi Lashkari, Saqib Hakak, and Ali A. Ghorbani, "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy", IEEE 53rd International Carnahan Conference on Security Technology, Chennai, India, 2019.

[35]https://stackoverflow.com/questions/10598872/how-to-read-some-columns-from-a-csv-file-with-c