Applied Information Technology And Computer Science Vol. 2 No. 2 (2021) 115-126

AITCS

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/aitcs
e-ISSN :2773-5141

Personal Home Network Intrusion Checking Tool (PacAN)

Nicholas Lim Jun Jie, Khairul Amin Mohamad Sukri*

Faculty of Computer Science and Information Technology,
Universiti Tun Hussein Onn Malaysia, 86400 Parit Raja, Johor, MALAYSIA

DOI: https://doi.org/10.30880/aitcs.2021.02.02.008
Received 13 June 2021; Accepted 09 September 2021; Available online 30 November 2021

Abstract: Home networks have a potential risk of intrusion or disturbances, and packet analysers could be too complex for home users to use by either displaying too much or unsorted information in the analysis result or by requiring users to use a collection of tools and techniques to analyse a network packet capture. This project proposed a tool that allows home users to determine if there are suspicious hosts in their network through packet capture analysis that produces results which are easily read and understood as well as not requiring users to use multiple tools or techniques for the analysis. Object-Oriented Software Development (OOSD) is used as the methodology for flexibility to the developer in order to complete the PacAN packet analysis software tool in the Python 3 programming language with Visual Studio Code. With this tool, users were able to input packet capture file and analysis options, produce result data in table and graph form, and save the graph in the desired location. This tool is able to reduce the risk of intrusion and attacks of home networks as well as providing a compact option that eliminates the users' need to use multiple tools.

Keywords: Home Network, Packet Capture, Packet Capture Analysis

1. Introduction

Computer technology has come a long way, and with the number of computers being used daily increasing over the years, network security is important for private and government organisations. The growth of home and small enterprise networks brings with it a large number of devices and networks that are either managed poorly or not at all. Hosts on these networks may become compromised and become sources of spam, denial-of-service traffic, or the site of a scam or phishing attack site [1]. Community Emergency Response Team (CERT) statistics reports that the number of intrusions has excessively increased year by year, and any malicious intrusion or attack on the network vulnerabilities, computers or information systems may give rise to serious disasters, and violate the computer security policies, i.e., Confidentiality, Integrity and Availability (CIA) [2].

Network traffic analysis could be defined as: "the inference of information from observation of the network traffic data flow" [3]. Through network traffic analysis, suspicious hosts and activities in a network could be determined. There are network packet capture analysers like tcpdump and ngrep where they are able to read packets in a packet capture file generated from network packet capture

*Corresponding author: khairulm@uthm.edu.my

software like Wireshark. However, the output of the result analysis could be overwhelming and disoriented to users new to packet capture analysis. Users would also need to use other tools in tandem with the packet analysis software in order to generate desired results.

The PacAN personal home network intrusion tool is designed to answer to the problems stated by providing a means to reduce the risk of intrusion and attacks of home networks while being a compact solution that encompasses the needed functions in one package and generating easily understood results. The objectives of the project are to design a network packet capture analyser for detecting intrusions in home networks, to develop the personal home network intrusion checking tool and to test the developed tool with set network packet capture samples. The target for the project software tool is the home network users. The result for this project is that the users are able to perform the functions of the PacAN software tool, the presence of suspicious hosts and activity in the network could be determined by the user after PacAN analyses the packet captures and displaying result data in easy-to-understand presentation form, where the decision could be made after comparing the result data for obvious differences in network behaviour.

The rest of this paper is organized as follows: Section 2 reviews all works related to packet analysers. Section 3 presents the methodology used to perform the tool development tasks. Section 4 presents the results and discussion, while conclusions and future works are presented in Section 5.

2. Related Work

This section presents relevant study background about related terms of the proposed tool, such as home networks, packet analysers, packet analysis, network security attacks and a comparative study of the existing tools in relation to the proposed tool.

2.1 Home Networks

A home network is a type of computer network that facilitates communication among devices within the close vicinity of a home. The communication between devices in a home network allows an increase of the quality of life inside the home in various ways, such as wireless printing via network, an increase of personal productivity and an easier access to entertainment. It is safe to say an increasing number of network devices are connected to the Internet through small networks in the home. However, home networks pose a particular challenge to network security because the they are often either managed poorly or not at all [1]. The increased connectivity of devices in a home network allows for more vectors of attack and an increased risk of intrusion of the network.

2.2 Packet Analysers

A packet analyser or packet sniffer is a computer program or computer hardware such as a packet capture appliance, that can intercept and log traffic that passes over a computer network or part of a network [4]. It analyses network traffic and generates a customized report to assist organizations in managing their networks. There are notable packet analysers such as tcpdump, ngrep, snoop, Wireshark and many more.

2.3 Packet Analysis

Packet analysis is a primary traceback technique in network forensics, which, providing that the packet details captured are sufficiently detailed, can play back even the entire network traffic for a particular point in time [5]. Through packet analysers, users are able perform many capabilities such as analysing network problems, detect network intrusion attempts, gain information for effecting a network intrusion, identify suspect content in network traffic and many more.

There are a lot of notable packet analysers, to name a few would be dSniff, Microsoft Network Monitor, tcpdump, Xplico, ngrep, and Wireshark. These utilities have differences in their software

design such as platform or user interfaces. However, they serve the same fundamental function which is to perform packet analysis or serve as packet sniffers

Packet analysis provides numerous benefits to network security. It is able to lower the number of security breaches by getting an accurate copy of network data and giving security teams a chance to prevent and mitigate security breaches from happening. Another benefit is higher network performance by preventing dropped packets

2.4 Network Security Attacks

There are many methods and vectors for a malicious entity to attack a network. To name a few examples, there are attacks such as Denial-of-Service (DOS), unauthorized access, privilege escalation, insider threats. Before these attacks, the entity would do reconnaissance on the network by performing network scans and vulnerability scans to determine the weakest point of attack and infiltrating the network through that vector.

Network scanning is the process of discovering active hosts on the network and information about the hosts, such as operating system, active ports, services, and applications [6]. There are two types of scanning: active scanning and passive scanning. Active scanning is when the tool sends a ping to each device on the network and awaits a response. The scanner then looks at the responses it gets to see if there are inconsistencies or vulnerabilities. For IP networks, this is often done by sending a ping to each possible IP address and getting a response to determine its status.

Passive vulnerability scanning is the process of monitoring network traffic at the packet layer to determine topology, services and vulnerabilities [7]. This approach looks at network information as soon as a device or system appears and starts sending messages to the network.

DOS attacks work by flooding systems, servers, and/or networks with traffic to overload resources and bandwidth. In addition to denial-of-service (DoS) attacks, there are also distributed denial-of-service (DDoS) attacks. DoS attacks saturate a system's resources with the goal of impeding response to service requests. On the other hand, a DDoS attack is launched from several infected host machines with the goal of achieving service denial and taking a system offline, thus paving the way for another attack to enter the network/environment.

There are DDoS attacks such as volume-based attack and protocol attack. Volume-based attack will flood the bandwidth of the attacked site and it is measured in bits per second. This kind of attack includes UDP flood, ICMP flood and other spoofed-packet flood. The protocol attack on the other hand will interrupt actual server resources and it is measured in packets per second. This includes TCP SYN flood, fragmented packet attack, Ping of Death and Smurf attack [8].

2.5 Study of Existing Packet Analysers

Network packet analysers has been widely developed by many teams and companies as they are useful when it comes to analysing network packet captures. tcpdump and ngrep are the two examples of the existing packet analysers.

tcpdump is an open-source data-network packet analyser computer program that runs under a command line interface. It allows the user to display TCP/IP and other packets being transmitted or received over a network to which the computer is attached. tcpdump was originally written in 1988 by Van Jacobson, Sally Floyd, Vern Paxson and Steven McCanne who were, at the time, working in the Lawrence Berkeley Laboratory Network Research Group [9]. It is written in C programming language, and although it is native to Linux, it is compatible with Windows (WinDump) and MacOS operating systems (OS) as well. tcpdump is able to read network packet captures in addition to performing packet captures. The user interface for tcpdump is command-line interface (CLI) and only generates output in the terminal by default unless tcptrace is used as well where in that case, a graph

could be produced for further analysis. tcptrace is a free and open-source tool for analyzing TCP dump files [10].

ngrep (network grep) is a network packet analyser written by Jordan Ritter. It has a command-line interface, and relies upon the pcap library and the GNU regex library. Same with tcpdump, it is an open-source application that is available to be downloaded from its ngrep site. It can be compiled and ported to multiple platforms such as Linux, Windows and MacOS. ngrep's functionality is similar to tcpdump with the ability to look for a regular expression in the payload of the packet and show matching packets on the console. A regular expression is a sequence of characters that specifies a search pattern. It also can be used to capture traffic and store pcap dump files, or read files generated by other applications like Wireshark. The user interface for ngrep is similar to tcpdump, which is a command-line interface, and generates output on the console.

Table 1: Comparison of Existing Packet Analysers

Attributes	tcpdump [10]	ngrep [12]	PacAN
Platform	Linux, Windows, MacOS	Linux, Windows, MacOS	Linux, Windows, MacOS
User Interface	CLI	CLI	CLI
Technology Used	C	C	Python
Read network packet capture	Yes	Yes	Yes
Capture Network Packets	Yes	Yes	No
Result Output	Terminal (graph with tcptrace)	Terminal	Terminal and generated graph

Based on the observation of the comparative study in Table 1 with the features that each application offers, the PacAN application is able to provide most of the basic functions compared to the other systems.

Furthermore, PacAN is available to produce data in presentable graph form through its own function where ngrep is unable to do so and tcpdump requires using another tool (tcptrace) to produce the graph data. However, PacAN is not able to capture network packets unlike tcpdump and ngrep. All three of the compared packet analysers are able to be used in multiple platforms such as Linux, Windows and MacOS with a similarity on the user interface which is CLI. PacAN is written in Python 3 whereas tcpdump and ngrep are both written in C.

3. Methodology

The methodology in software development serves as a systematic framework that determines the path or the structure in developing the software system. The detailed processes of project analysis, project design, and project implementation is included in the framework, which serves as an indicator of success. This project utilizes the Object-Oriented Software Development model to develop the proposed software tool using the Python 3 programming language.

3.1 Object-Oriented Software Development

Object-Oriented Software Development (OOSD) is a method where the software is developed by constructing objects or visual models in developing a software tool. The visuals help to apprehend the problems and create the proper pathway to produce well-designed programs. It creates a picture of the system that is modifiable, reusable, reliable and understandable [11]. The purpose of choosing the OOSD model is because of its flexibility that can move back and forth between design phase,

programming and testing phase, which enables the developer to easily identify errors in the early stages of the software development process.

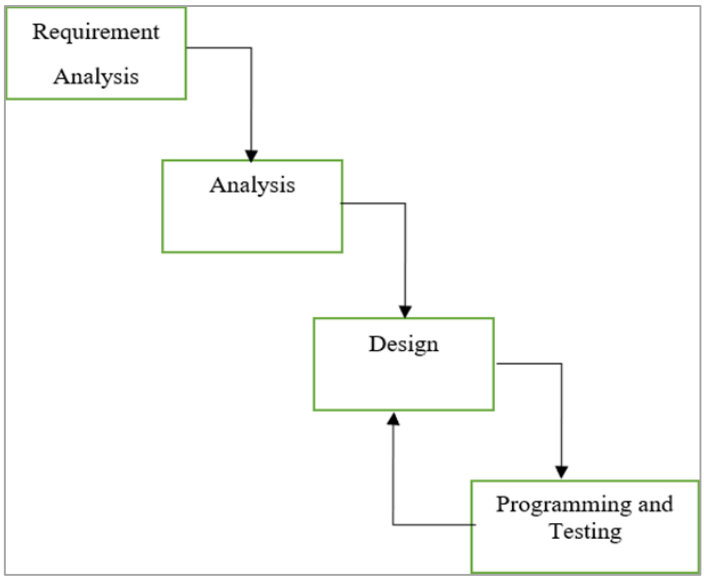

Figure 1: OOSD Model

Figure 1 depicts the OOSD model processes, where the first phase is requirement analysis, which moves on to analysis, then design and testing. The activities and outcomes of each phase from the methodology are outlined in Table 2.

Table 2: System Development Activity and Outcome

Phase	Activity	Outcome
Requirement Analysis	1. Proposed project objectives, problem statements and expected result. 2. Determined project schedule. 3. Study existing applications. 4. Search for resources online.	1. Proposal. i. Scope ii. Objectives iii. Problem statement iv. Project milestone 2. Literature review.
Analysis	1. Determine and analyse features according to the comparison of existing application. 2. Identify software and hardware requirements.	1. User requirements 2. Functional and non-functional requirements.
Design	1. Design CLI options and output graph.	1. CLI parse design. 2. Output graph design.
Programming and Testing	1. Develop code. 2. Functional testing.	1. PacAN tool. 2. Test tool.

Figure 2 presents the interaction between the user and the application while Figure 3 shows the activity diagram. The user would be able to perform the tasks through the application such as calculating the flow count between source and destination IP, finding the most used ports, calculating the byte count as well as finding out the data over time in the network packet capture file, after which the application would be able to produce the result graph for the selected functions.

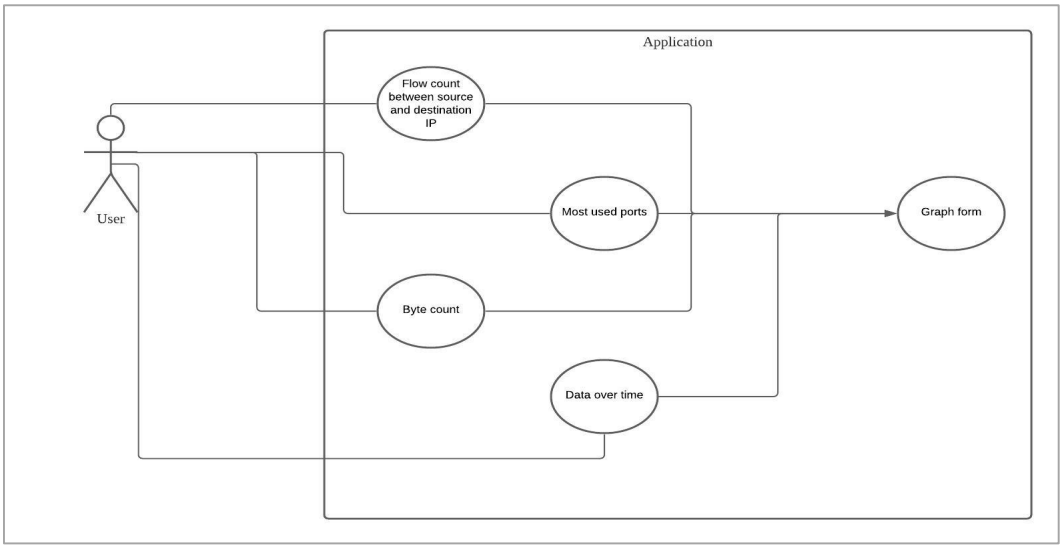

Figure 2: Use Case Diagram

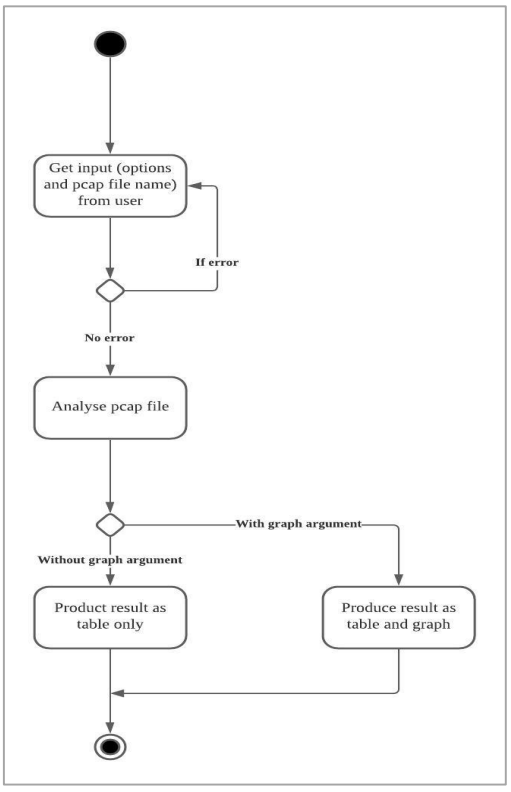

Figure 3: Activity Diagram

4. Results and Discussion

This section presents the implementation and testing that was conducted for the PacAN network packet capture analyser tool.

4.1 Implementation

The coding for the PacAN tool was done using Python 3 programming language through the use of Visual Studio Code code editor. To start using the tool, the user would need to install Python 3 as well

as the required libraries used by Python. A user manual for steps to install the requirements and use the tool was provided for users to follow.

After installing the requirements, the user would need to open their operating system terminal. As majority users use the Windows operating system, users would just have to go to the Windows search and type 'cmd' and enter. The Windows terminal would open and users would just have to navigate to the folder containing the tool and enter the command-line to initialize the tool. After inputting the required command options and file input, the tool would analyse the network packet capture file and produce the result in table and graph form.

```
def simpleCount(ipList, limit, headerOne, headerTwo, title):
    table= PrettyTable([headerOne, headerTwo])
    cnt = Counter()
    yData=[]
    xData=[]
    for ip in ipList:
        cnt[ip] += 1
    a=0
    for item, count in cnt.most_common():
        item=str(item)
        table.add_row([formatCell(item),count])
        #for the graph
        yData.append(count)
        xData.append(item)

        if limit:
            if a >= limit:
                break
        a+=1
    print(title)
    print(table)
    if args.graphs:
        createGraph(xData, yData, headerOne, headerTwo, title)
```

Figure 4: Function Source Code

Figure 4 presents a section of source code for one of the functions of the PacAN packet analyser. A function is created with parameters that will be used, and the PrettyTable class is used to formulate a table for the data. The Counter class is called to store the elements as dictionary keys (in this scenario it would be the IP address) and their counts as dictionary values. The most_common method is used to list the most common element in descending order. The object in the item variable is converted to string and added into the xData list for the purpose of being a part of data in producing the graph result.

4.2 Testing

Once the tool has been developed, a testing phase is commenced to examine the functionality of the application. Testing was conducted to identify any sorts of error that could possibly surface when using the PacAN packet capture analyser tool. Another purpose for testing is to find out whether the tool is able to achieve their objective and scope specified. Table 3 shows the summary of the functional testing results.

Table 3: Functional Test

Function Tested	Function Testing	Expected Result	Tested Result
Input Function	1. Users leave command options empty.	1. Error message and user guidance message shown.	Success
	2. Users input wrong packet capture filename/file extension.	2. Error message shown.	Success
	3. Users did not install required libraries before running the tool.	3. Error message and user guidance message shown.	Success
	4. Users entered help option.	4. User guidance and detailed description of options shown.	Success
	5. Users entered the options and file input correctly and press enter.	5. Tool will proceed with analysis based on command options.	Success
Analysis Function	1. Users entered '--limit' as part of command option.	1. Result produced will be limited to the number the user input.	Success
	2. Users entered '--all' as command option.	2. Result produced will involve every option.	Success
	3. Users entered '--src' as command option.	3. Result produced will involve source IP count.	Success
	4. Users entered '--dst' as command option.	4. Result produced will involve destination IP count.	Success
	5. Users entered '--sport' as command option.	5. Result produced will involve source port count.	Success
	6. Users entered '--dport' as command option.	6. Result produced will involve destination port count.	Success
	7. Users entered '--ports' as command option.	7. Result produced will involve total port count.	Success
	8. Users entered '--bytes' as command option.	8. Result produced will involve source and destination byte count.	Success
	9. Users entered '--portbytes' as command option.	9. Result produced will involve ports by bytes.	Success
	10. Users entered '--flows' as command option.	10. Result produced will involve summary of flow between IP addresses.	Success

Table 3: (cont.)

Function Tested	Function Testing	Expected Result	Tested Result
Analysis Function	11. Users entered '--graphs' as command option. 12. Users entered '--timeseries' as command option.	11. Graph result will be produced along with table result. 12. Result produced will involve the data over time.	Success Success

After the functional testing of the PacAN network packet capture analyser, another test was conducted to affirm its capability of determining suspicious hosts and activities in the network. Sets of network packet captures of the home network and its activity were captured through Wireshark, and PacAN was used to take the sets as input for packet analysis. Among the sets of two, one packet capture is a packet capture of a normal home network activity and the other is a packet capture that has a host committing network scanning and packet flooding attacks in the network. The duration for the packet captures were kept the same for each set for consistency. The following figures present a comparison between the two graphs generated from the set of packet captures, where the left graph is the compromised network and the right graph shows the normal network.

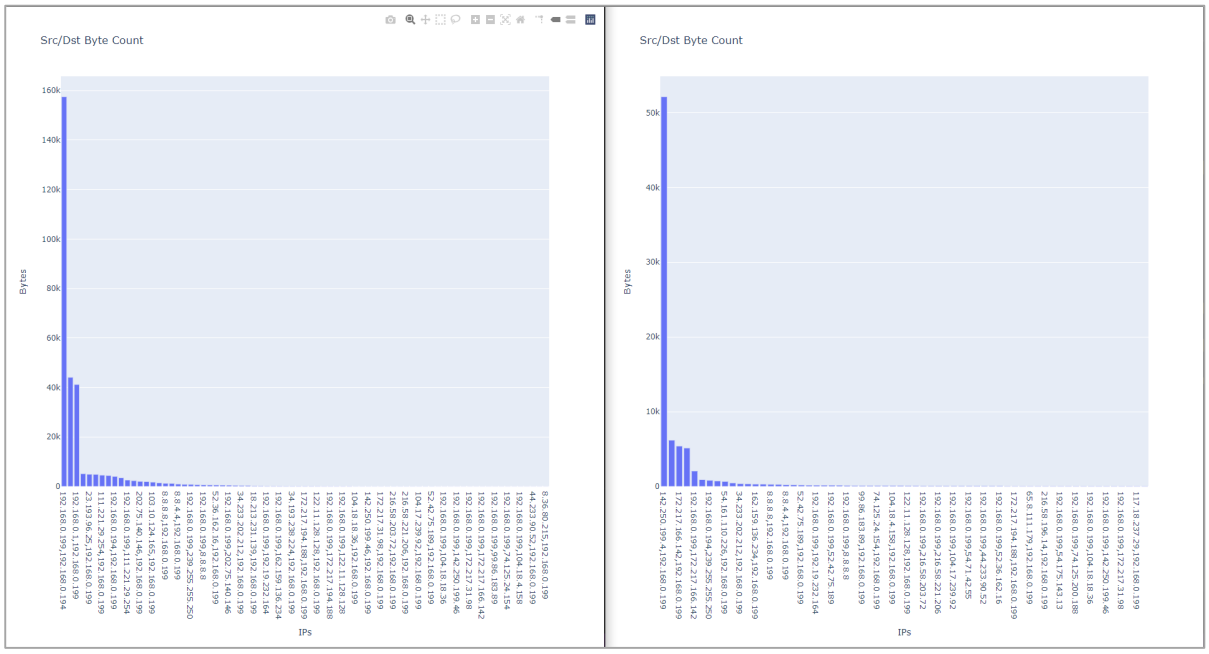

Figure 5: Source and Destination Byte Counts by IP Graphs Generated by PacAN

Figure 5 shows the graphical data for both packet captures based on the source and destination byte counts. The x-axis is the source IP address followed by the destination IP address, and the y-axis is the bytes counts. We can see that there is a big difference between the two packet captures even though the duration is the same. The host that was committing the network scanning and packet flooding during the test was shown to have the highest byte count as shown in the left graph, where its IP address is 192.168.0.199. On the right, we can see that the top largest source and destination byte sizes are spread out over multiple IP addresses as well as not having a spike in the number of byte size.

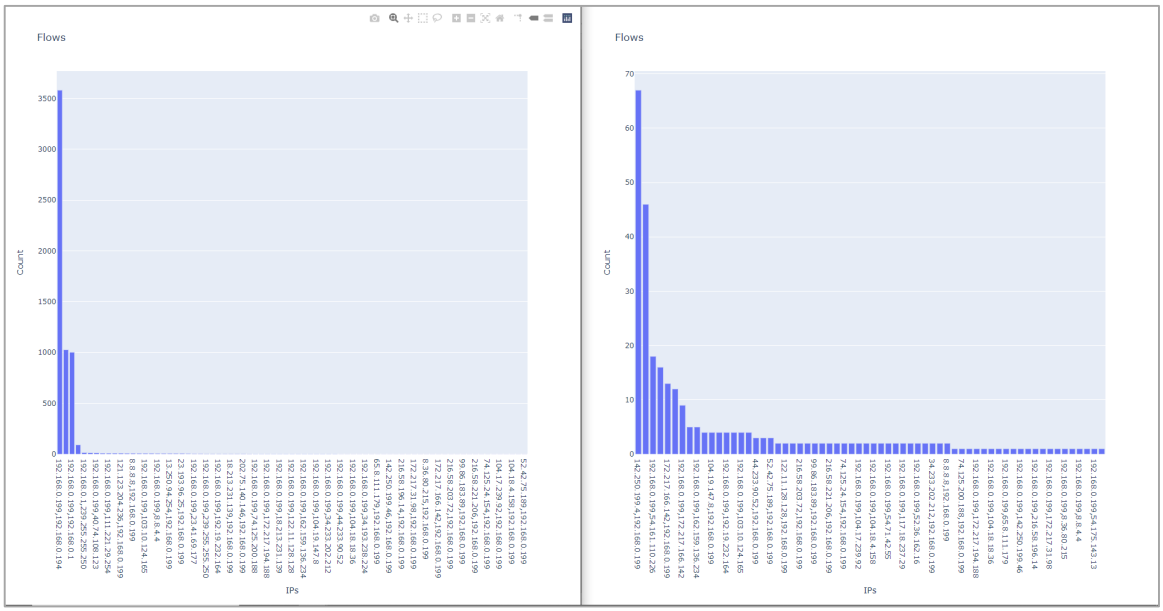

Figure 6: Flow Counts between Hosts by IP Graphs Generated by PacAN

Figure 6 shows the graphical data for both packet captures based flow counts between hosts. The x-axis shows the source IP address followed by the destination IP address, and the y-axis shows the flow counts. The left graph we can see that the maximum number of flows counts between hosts is over 3000 and the top 10 has the same source IP, whereas on the right graph we can see the flows are more average as well as not having high number of counts. We can conclude from this result that the source IP on the left graph is committing suspicious activities such as network scanning and packet flooding.

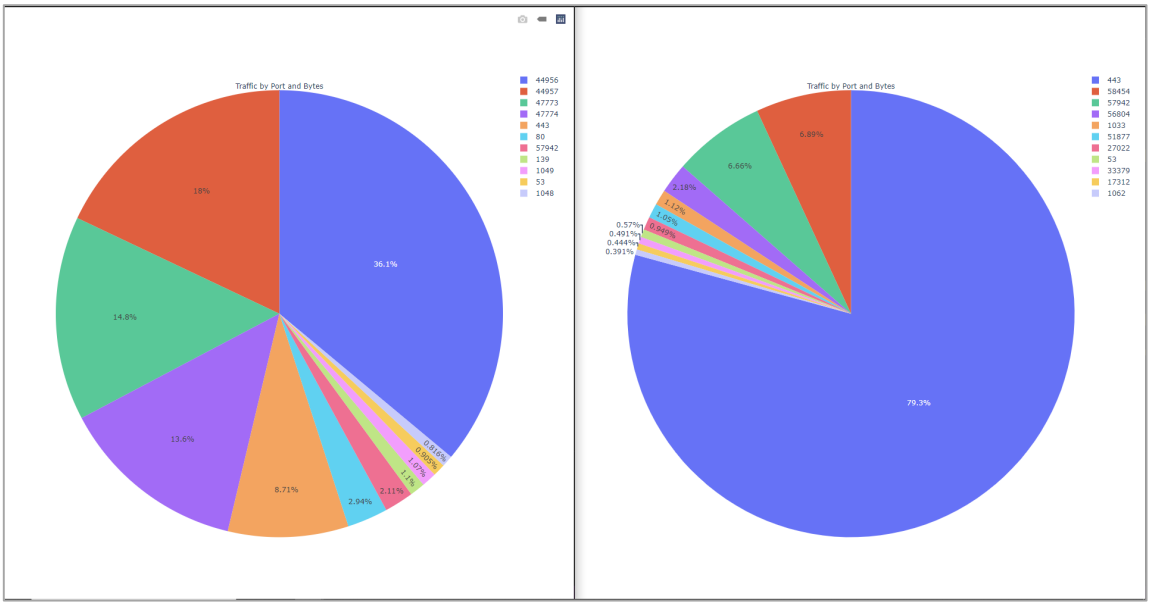

Figure 7: Bytes by Port Graphs Generated by PacAN

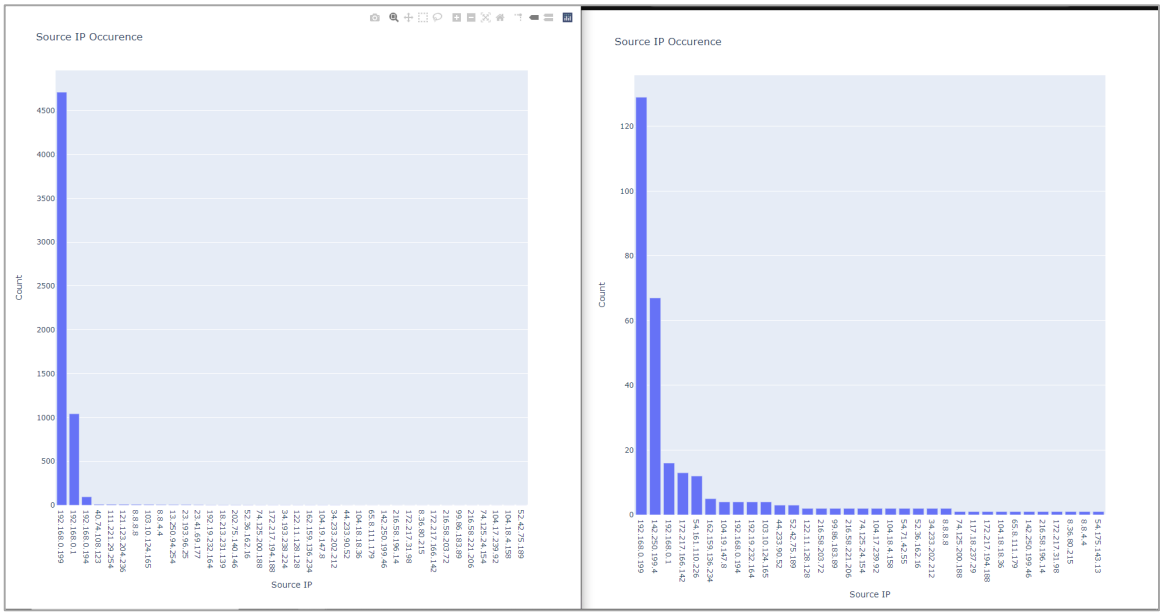

Figure 8: Source IP Occurrences by Source IP Graphs Generated by PacAN

Figure 7 and Figure 8 shows the graphical result data for bytes by port and source IP occurrences. The following section would provide explanation regarding the result data. In Figure 7, the most commonly used port on the left graph is port 44956 and 44957 which are TCP/UDP ports where the most commonly used port on the right graph is port 443 which is a HTTPS port. This suggests that a host in the network is performing network scanning as network scans send TCP and UDP packets to hosts inside the network for responses from the hosts. Figure 8 shows the most common source IP address found in the packet capture files. We can see between the two graphs that the left graph has a significantly larger number of occurrences than the right graph, and the most common IP address is 192.168.0.199, the host used to commit network scans and packet floods.

4.2 Discussions

From the data and result that is generated, it is safe to say that the host 192.168.0.199 is a suspicious host, and the user can take appropriate action by blacklisting the IP address in the router or limiting its network traffic usage. The tool has success in generating the graph result for analysis and achieves its objective of providing users a convenient solution of generating easily comprehended results in presentable form without needing to use multiple tools in tandem.

5. Conclusion

The objectives of the project which were to design a network packet capture analyser for detecting intrusions in home networks, to develop the personal home network intrusion checking tool and to test the developed tool with set network packet capture samples has been achieved by the PacAN network packet capture analysis tool. PacAN is able to determine intrusions through analysing the network for suspicious hosts and activities such as abnormally high IP address occurrence and suspicious number of flows between a source IP address and a destination IP address. The PacAN tool allows users to have a clearer understanding of their home network activities and mitigate security risks such as network attacks.

There are several limitations that can be found in the PacAN tool, which is that the tool is not able to capture network packets by itself. Besides that, the tool is not able to be run in real-time so it is unable to provide real-time analysis of the network.

There are some improvements that can be implemented for future works of the PacAN tool, where a graphical user interface (GUI) could be designed for more convenience, as well as adding a function to capture network packets instead of relying on other software like Wireshark.

Acknowledgement

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support and encouragement throughout the process of conducting this study.

References

[1] N. Feamster, "Outsourcing home network security," in Proceedings of the 2010 ACM SIGCOMM workshop on Home networks, 2010, pp. 37-24, doi: 10.1145/1851307.1851317.

[2] H.J. Liao et al., "Intrusion detection system: A comprehensive review," Journal of Network and Computer Applications, 36(1), pp 16-24, Sep. 2012.

[3] P. Asrodia and H. Patel, "Network traffic analysis using packet sniffer," International Journal of Engineering Research and Applications, vol. 2, no. 3, pp. 854-856, Jun. 2012.

[4] K. J. Conolly, Law of Internet Security and Privacy. Delaware, DE: Aspen Law and Business, 2002.

[5] L. F. Sikos, "Packet analysis for network forensics: A comprehensive survey," Forensic Science International: Digital Investigation, vol. 32, 2020, doi: 10.1016/j.fsidi.2019.200892.

[6] A. Orebaugh and B. Pinkard, Nmap in the enterprise: your guide to network scanning. Burlington, MA: Elsevier, 2011.

[7] R. Deraison et al., "Passive vulnerability scanning: Introduction to NeVO," Revision 9, pp. 1-13, Aug 2003.

[8] M. A. M. Yusof et al., "Detection and defense algorithms of different types of DDoS attacks," International Journal of Engineering and Technology, vol. 9, no. 5, pp. 410, Oct. 2017.

[9] S. McCanne, "libpcap: An architecture and optimization methodology for packet capture," [Online]. Available: https://sharkfestus.wireshark.org/sharkfest.11/presentations/McCanne-Sharkfest'11_Keynote_Address.pdf. [Accessed April 4, 2021].

[10] R. Blum, Network Performance Open Source Toolkit: Using Netperf, Tcptrace, NISTnet, and SSFNet. Indianapolis, IN: Wiley Publishing Inc., 2003.

[11] E. Colbert, "The object-oriented software development method: a practical approach to object-oriented development," in Proceedings of the conference on Tri-Ada'89: Ada technology in context: application, development, and deployment, 1989, pp. 400-415, doi: 10.1145/74261.74291.

[12] R. C. Joshi and E. S. Pilli, "Network forensic tools," in Fundamentals of Network Forensics: A Research Perspective, London: Springer, 2016, pp. 71-93.