Internet Traffic Classification Tool

Computer Networks Term Project

Fatima Hasan
Department of Computer Science
National University of Computer and Emerging Sciences
Lahore, Pakistan
l174020@lhr.nu.edu.pk

Samra Fakhar
Department of Computer Science
National University of Computer and Emerging Sciences
Lahore, Pakistan
l174031@lhr.nu.edu.pk

Nuzha Khaled
Department of Computer Science
National University of Computer and Emerging Sciences
Lahore, Pakistan
l174162@lhr.nu.edu.pk

Shanzay Gauhar
Department of Computer Science
National University of Computer and Emerging Sciences
Lahore, Pakistan
l174236@lhr.nu.edu.pk

***Abstract*— The Internet is an ever-evolving technology which has made it complex to comprehend the structure of the network. To understand the structure of networks and how protocols work in the real world, internet classification tools were the need of the time. These tools assist in breaking up the traffic into categories by various parameters such as port number, protocol etc. As the internet is widely used and fast growing it poses a challenge for us to understand or characterize the traffic without the assistance of such classification tools. In this project, we identify and quantize different types of traffic/packets such as DNS, VoIP, HTTP/S, RTP, FTP, etc. from live packet captures obtained through windump and then used the pcap file created as a result to perform classification steps. The report presents the Network Traffic Classification based on Port Numbers identification. The application is developed on a Pycharm application using python language. Inbuilt libraries are used to develop graphical user interfaces that are used to show percentage breakdown of traffic for each application over a given time. There has already been a lot of work done in this field and most of them involve machine learning concepts. However, in this project we are using Pyshark - a wrapper for Tshark to analyze a capture file - pcap. Basically, Tshark is a command line interface for the Wireshark, hence we are using its functionality to drop out unnecessary fields while analyzing a certain packet. The report concludes as many protocols use fixed port numbers it is an efficient and effective way to classify traffic on the network. However, the results for applications that do not use well-known ports are not as accurate as the former.**

Keywords— Internet Traffic, Packet Classification, PyShark, WinDump, Live Packet Capture, Data Visualization

I. INTRODUCTION

Network traffic classification is a process which categorizes computer network traffic according to either port number or protocol into a number of traffic classes [3]. It is an important problem in computer networks. The performance of a network highly depends on efficient resource management which in turn relies on analyzing network trends and planning and designing a network based on that analysis. Classification of network traffic packets is very important for Internet Service Providers (ISPs) for determining which type of network applications flow in the network. Moreover, it is also used in a multitude of applications including firewalls, intrusion detection systems and status reports [1].

The internet has been growing exponentially over the years and it has become very important for us to understand and even predict the type of traffic that is flowing, to increase our utilization and performance. There are many different types of traffic including, but not limited to, DNS, HTTP/S, VoIP, RTP, FTP, SMTP, TELNET, SNMP and much more.

There are different methods for the identification of the type of traffic. One of these is port number based, and this is the approach that we will be adopting. The formatter will need to create these components, incorporating the applicable criteria that follow.

A port number is used to identify the specific process that a network message is to be sent to. Both TCP and UDP use port numbers as an endpoint to a logical connection. When a client message is sent to a server, the port number identifies the specific server program that it is to be forwarded to. The range of port numbers is from 0 to 65536, but the port numbers from 0 to 1023 are reserved for privileged services. For example, port 80 is reserved for HTTP. The Internet Assigned Numbers Authority (IANA) is responsible for maintaining the assignments of these port numbers. Some other reserved port numbers are shown in the table below:

Table 1. Some Common Port Numbers

Port Number	Description
20	FTP – Data
21	FTP – Control
23	Telnet
25	Simple Mail Transfer Protocol (SMTP)
53	Domain Name System (DNS)
109	POP2
110	POP3
115	Simple File Transfer Protocol (SFTP)

118	SQL Services
139	NetBIOS
161	SNMP
465	SMTP over SSL
443	HTTPS
546	DHCP Client
547	DHCP Server
5060	VoIP
5353	mDNS
8080	HTTP Proxy

II. Background

According to many reports, there are many methods of successfully classifying network traffic. These include: port-based, payload-based, flow-statistics based and behavior based. We will be implementing port-based classification, but it is useful to look at the other methods as well.

Payload-based classification uses Deep Packet Inspection (DPI) to identify the type of traffic. Well-known patterns exist inside packets of the same application. This approach identifies and stores these patterns inside a database and then accesses the DB to detect the type of packet by looking at the contents. This method performs well for many types of applications like HTTP, FTP and SMTP. It is used in many software products as well as open source projects, one of which can be viewed here (http://tstat.tlc.polito.it/). Moreover, it can be used to detect network anomalies and is used in intrusion detection systems [11].

But it is not without its problems. One issue is that it does not work well with the applications that do not have well known patterns such as P2P applications and multimedia [1]. Moreover, it has an additional cost of maintaining the database with up-to-date patterns. Additional cost of string and regular expression matching cannot be ignored either. It is also getting increasingly difficult to access the raw payload since most of the transmitted data is being encrypted [2]. A disadvantage of this method is that it invades the privacy policies of the users and this data may be used against them.

The other method of identifying the internet traffic is by using flow based statistics. It uses the information that is available in packet headers for example length of the packet, TCP window size, packet interarrival time etc. This information is then used by different machine learning techniques to predict the type of traffic. The machine learning methods include naive bayes, K-nearest neighbours, artificial neural networks, clustering, regression and support vector machines [1]. These methods can either be supervised or unsupervised. The supervised methods classify the input according to pre-defined outputs whereas the unsupervised methods such as clustering only separate the samples according to their similarities. This method is better than the payload-based method, since it does not rely on the complete contents of the packet and hence, is not affected by encryption to that extent.

The flow-based classification methods have been increasing in popularity since the rise of machine learning and artificial intelligence. There have been many works that apply neural networks but are all different from our approach here.

In [4], they have used a multi-layer perceptron (MLP) with zero/one hidden layer and they use this to apply a Bayesian analysis. The accuracy that they have received for 10 labels is above 99%.

The use of MLP has been quite popular among researchers as it is the primary classification method in both [5] and [6] as well. The accuracy of these have been above 90% and they have suggested further improvements too. In [5], error correcting output codes have been used with MLP classifiers and the accuracy they achieved was 93.8 for 5 labels. As opposed to this, a particle swarm optimization algorithm is used in [6] to classify 6 labels with an accuracy of about 96%.

The most accurate out of all these researches has been [2]. They have employed a combination of Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN). It provides the best detection results as the accuracy is around 99.59% for 15 labels. By only using the high-level header data, not including the IP addresses since they are confidential, great accuracy has been achieved in the machine learning area.

Moving on, the behavioral classification method looks at the entire traffic received by an endpoint in the network. The traffic patterns are then examined to identify the application running on the target host. Different applications have different patterns for example P2P applications contact a number of peers using the same port for each host, whereas a Web server is contacted by clients with multiple parallel connections.

[7,8] and other related works employ different heuristics like number of contacted ports or transport layer protocols used to characterize the traffic patterns at different detail levels such as functional, application and social. [9,10] have shown in their study that P2P and client-server applications have extremely different behavioral patterns which can be used to classify these classes even in the network core.

A research paper [12] developed an application of internet traffic classification based on fixed IP-ports. They proposed a traffic classification system that accomplished the classification of internet packets by comparing and matching the packet header with the fixed IP-port information. An algorithm for the extraction of the IP-port information was designed. They claimed this method to be a fast and accurate alternative for other classification methods.

This system is not completely based on the information of port numbers, rather it uses the combination of a fixed IP-port triple i.e., IP, protocol, and port number to ensure the accurate classification method. The combination of fixed IP-ports is dedicated for each application and is recorded as the server socket information that is public and is used for a service delivery. The algorithm of extracting fixed IP-ports works by selecting the fixed triple among all the IP-port triples that

appear in the traffic flow records of an application. For this purpose, they differentiated the triples of the application into 3 categories: dynamic IP-port, temporal IP-port, and permanent IP-port. The algorithm uses flexible threshold values and functions by eliminating the dynamically and temporally selected IP-ports and then picking the permanently selecting the IP-port as a fixed IP-port [12]. This method being efficient, however, needs to overview various traffic flow records to ensure accurate determination of fixed IP-ports. This increases the processing overhead.

Another research [13] presents the TCP/IP Traffic Classification that is based on Port Numbers. To identify packets on a specific port number or range of port numbers, they introduced a classification model consisting of two stages. The fist stage determines the application for a given port number and the next stage identifies the type of traffic. They used five ranges of port number space that covers the well-known ports as well as new servers and clients. A port number can be easily mapped to its corresponding application, but the reverse of this only ensures reliable identification of applications for the well-known port numbers. They also defined a traffic class to most applications however, it becomes impossible for unfamiliar sources. Therefore, this classification method functions with few limitations [13].

Coming towards the port-based method, according to some articles, the port-based method might not provide very reliable results in the near future since the services of the internet are growing and the new applications may not have a well-known port number assigned by IANA. It would become difficult to classify such traffic using this method. However, for the applications that do have reserved port numbers, this method would provide 100% accurate results, and in some cases it may be important to only classify some applications, and not all. In such circumstances, this method would provide very good results.

III. Design and Methodology

We have developed this system for windows. It is developed in python. The Integrated Development Environment (IDE) used for development is PyCharm. Moreover, it uses windump as a supplementary program for data capture. A .pcap file is generated upon running the tool. This contains the captured data packets.

Our tool uses port numbers of the packets to classify them on the basis of their application layer protocol. A port number is used to identify the specific process that a network message is to be sent to. Both TCP and UDP use port numbers as an endpoint to a logical connection. When a client message is sent to a server, the port number identifies the specific server program that it is to be forwarded to. The range of port numbers is from 0 to 65536, but the port numbers from 0 to 1023 are reserved for privileged services. For example, port 80 is reserved for HTTP. Almost all application layer protocols use fixed port numbers to exchange data.

We will be focusing on the identification and quantization of different protocols from live captures obtained through windump. Using windump live data is captured and kept in a file. Quantization of different protocols from live captures is obtained through windump. The method we will be following to detect the type of packets is identifying the port number of each packet, since each protocol uses a fixed, pre-defined port number. By analyzing the packets and identifying the source and destination port numbers, it will be easy to classify which protocol is being used. The file containing captured packets is then read and used to visualize the data in form of a table depicting all the packets captured as well as a bar chart and pie chart.

We have developed a user-friendly GUI. Using this user can start capturing the data packets at a given point in time on pressing the “Start Capture” button. Windump captures as long as the user does not specify it to stop. This is achieved through the user pressing the “stop capture” button. Once stopped, the table is viewed. On this very screen depicting table, two buttons provide the option to display the data graphically.

A. Windump

Windump is the same as tcpdump, except that tcpdump is for UNIX and windump is for Windows. Tcpdump is a data-network packet analyzer. It is a command line packet sniffer which captures TCP/IP packets sent or received on a specific interface over a network. This is what we will be using in our project to capture live packets for classification.

Command “Windump -i 2 -w file.pcap” captures the data and stores them to a file “file.pcap”. if the file exists already, it overwrites that. Else, it creates the file and stores packets data to it.

B. Python Libraries

1) Pyshark

We have used PyShark for the analysis of data packets captured. PyShark is a Python wrapper for TShark, allowing python packet parsing using Wireshark dissectors. TShark is a network protocol analyzer. It allows capturing packet data from a live network, or reading packets from a previously saved file. TShark's native capture file format is pcap format. The file generated by windump is in the same format.

In our project, the pcap file is parsed using PyShark, after which packet data is available through the different layers. For example, if one wants to get the ip address of the destination, it is simply available through the following command:

packet['ip']. dst

PyShark can also be used to capture live packets and then analyze them, but for the packet capture, we will be using the option of windump described above.

PyShark also has many other functionalities. It has a collection of mechanisms which are employed to classify the protocols of different packets. It not only determines the application layer protocols, but can also determine the underlying transport as well as network layer protocols.

ii) PyQt5

We have employed PyQt5 for the development of GUI. Traditional UI development as well as many other features including multimedia, Bluetooth connectivity, location services and NFC are implemented in Qt which is a set of C++ libraries. PyQt5 is a set of python bindings for Qt. It is sometimes also used in C++ applications to enhance their functionalities.

In our project, QtWidgets and QtGui are imported from PyQt5. QtWidgets used include QApplication, and QMainWindow. Through this very easy to understand GUI, users can begin capturing packets by the simple click of a button which says “Start Capturing”.

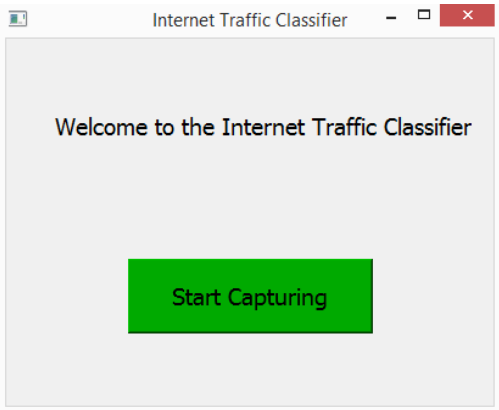


Figure 1. Start Capturing Dialog Box

When they want to stop the capture, they again just have to click one button that says "Stop Capturing".

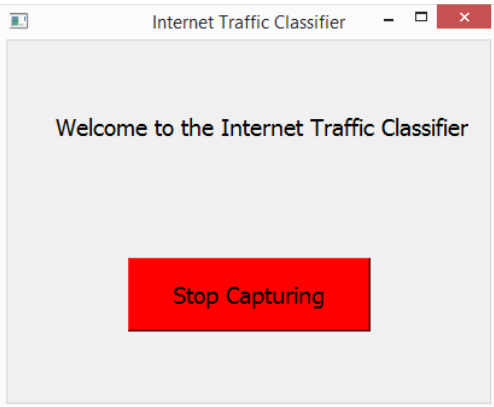


Figure 2 Stop Capturing Dialog Box

Then the user is given multiple options to visualize the results. This includes a table, which shows the information of port numbers, length of the packet, and the protocol which we have classified. Another visualization method is a pie chart which shows the relative percentages of the different types of protocols of the packets captured. Yet two more types of graphs can be displayed, which we have shown below.

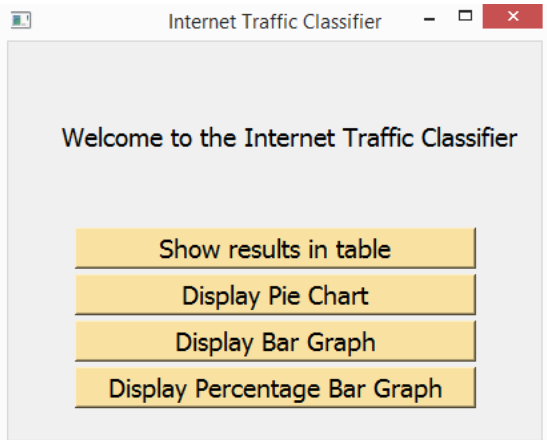


Figure 3. View Results Dialog Box

iii) Matplotlib and Numpy

Matplotlib and Numpy are used for data visualization. Matplotlib is a very comprehensive library which allows the creation of interactive visualizations in Python. These visualizations can easily be customized according to the requirements. We have used matplotlib for generating the bar graph, percentage bar graph, pie chart as well as the table in our tool. The mathematical calculations associated with these visualizations are done using Numpy which is a python library for the manipulation of multi-dimensional arrays and matrices.

Libraries like subprocess, time, signal, sys, os are imported to write a script which uses command prompt to run windump.

C. Packet Classification

Our tool uses port numbers of the packets to classify them on the basis of their application layer protocol. A port number is used to identify the specific process that a network message is to be sent to.

When a client message is sent to a server, the port number identifies the specific server program that it is to be forwarded to. The range of port numbers is from 0 to 65536, but the port numbers from 0 to 1023 are reserved for privileged services. For example, port 80 is reserved for HTTP. Almost all application layer protocols use fixed port numbers to exchange data. We have created a file called ports.txt, which contains a list of the port numbers that our tool is capable of classifying. These can easily be extended to include any protocol which is necessary to be classified. We have kept the list short, to only the popular protocols which we desire to classify. Our program reads this file and saves the port numbers and the corresponding protocols in a list. Then, packets are captured using windump and parsed using PyShark. Once we get the source and destination port numbers of the packets, we use these to search the list of protocols and determine which protocol is being used in that particular packet..

D. Representation

The captured data packets can be visualized in the following manner:

1) Table

The table contains source IP address of packet, destination IP address of packet, source port number of the packet, destination port number of the packet, length of the packet and name of the application layer protocol of the packet.

2) Bar Graph

The bar graph depicts the exact count of packets captured. Each bar corresponds to a protocol name along the x-axis and the length of bar depicting the number of packets.

3) Pie Chart

Pie chart represents their percentage of each individual protocol in complete capture. Each segment in a unique color shows the percentage and the name of protocol. Legend also provides information pertaining to color and the protocol represented by it

4) Percentage Bar Graph

This graph shows the percentage breakdown of the captured data. Unique colors depict the different protocols.

The name of the protocol as well as the percentage is displayed.

IV. Testing and Evaluation

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

As we mentioned before, we have displayed results in 4 different formats, namely, a table, a pie chart, a bar graph and a percentage bar graph. All of these were used in the testing of our tool. We captured packets for around 10-15 seconds and the percentage breakdown has been displayed in the pie chart, bar graph and percentage bar graph as shown below:

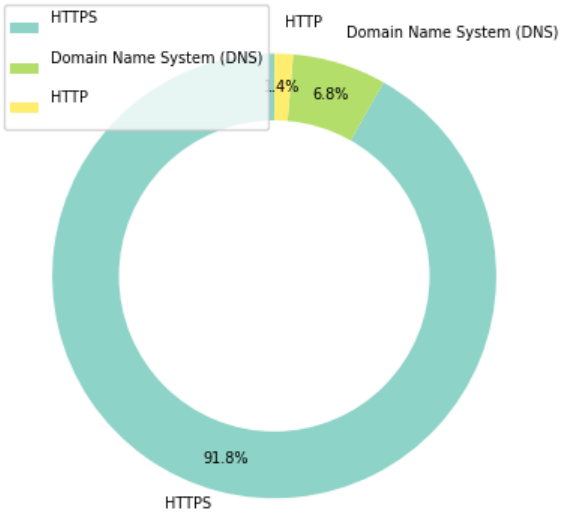


Figure 4. Piechart Representation of Result

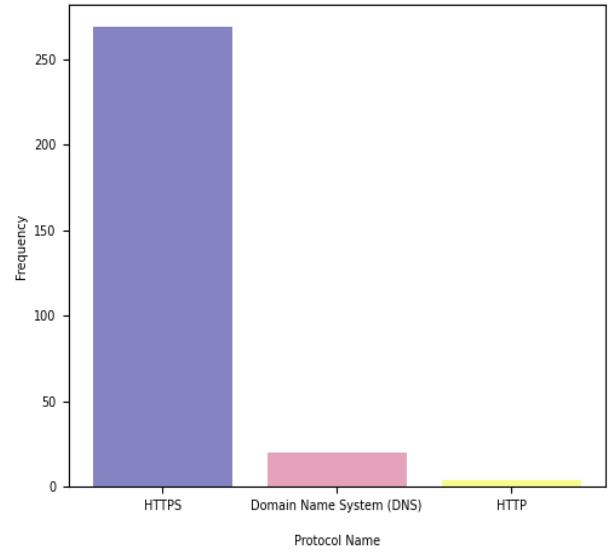


Figure 5. BarGraph Representation of Result

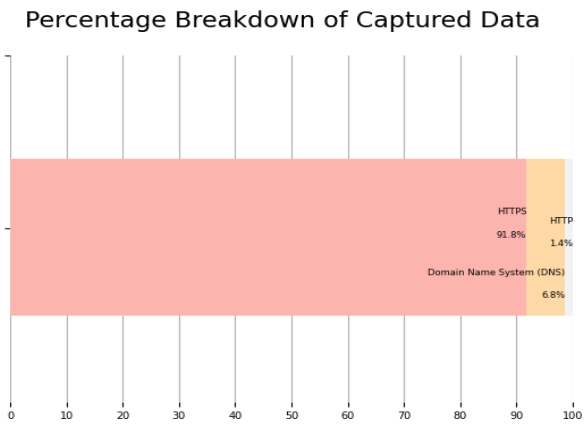
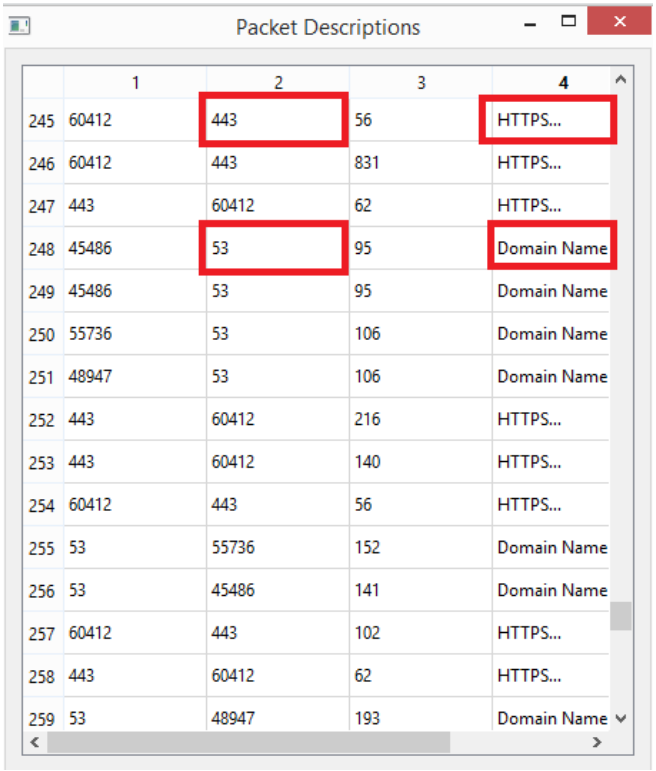


Figure 6. Percentage Bargraph Representation of Result

To check that our results were correct, we analyzed the table which was created by our tool, and checked the protocol against the port number. All the packets were correctly identified. Some screenshots are shared below:



	1	2	3	4
245	60412	443	56	HTTPS...
246	60412	443	831	HTTPS...
247	443	60412	62	HTTPS...
248	45486	53	95	Domain Name
249	45486	53	95	Domain Name
250	55736	53	106	Domain Name
251	48947	53	106	Domain Name
252	443	60412	216	HTTPS...
253	443	60412	140	HTTPS...
254	60412	443	56	HTTPS...
255	53	55736	152	Domain Name
256	53	45486	141	Domain Name
257	60412	443	102	HTTPS...
258	443	60412	62	HTTPS...
259	53	48947	193	Domain Name

Figure 7. Description of Captured Packets

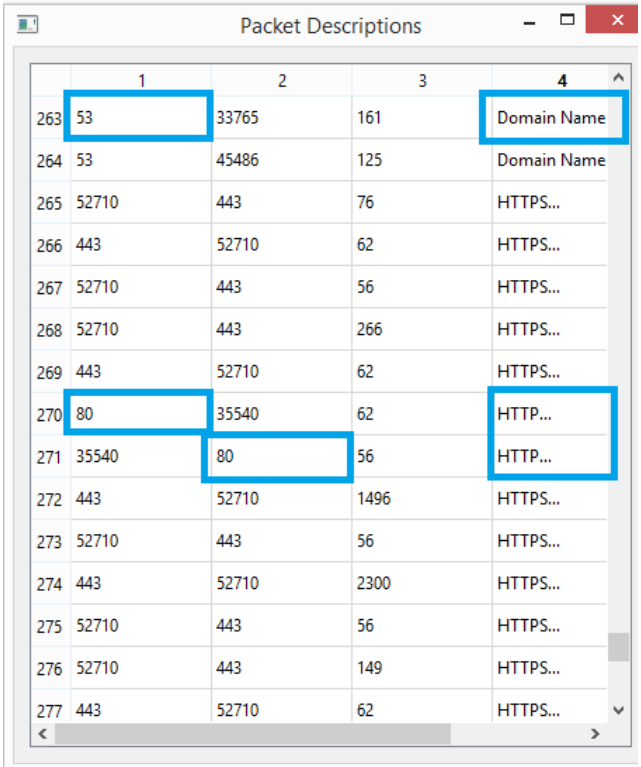


Figure 8. Description of Captured Packets

One small issue that we faced is that when there are a lot of categories of packets, and some of them have very small percentages, the labels of the pie chart tend to overlap and it is difficult to determine the percentage of each. To overcome this, we created a key which depicts the colour of each protocol so that it is easier to identify.

We tested our tool on 4 different PCs, with different time intervals for capturing packets each time and evaluated the results. The protocols were identified correctly each time. However, due to the limited time and resources, we were not able to test it as extensively as we would have liked. There may be some packets which remain unclassified, but overall, the accuracy of the application would remain greater than 95% at least.

V. Conclusion

The report presents the Network Traffic Classification based on Port Numbers identification. Different techniques and advanced classification models have been introduced in the field of networks to classify the real-time, interactive, and bulk traffic, essential for the management of Internet Service Providers (ISPs). These techniques have their benefits and limitations as discussed. We proposed a detailed analysis on the designed methodology for detection of the protocols being used by the packets. This detection was based on their port numbers and therefore, the traffic classification was achieved on the basis of the application layer protocol. The approach used is one of the simplest ways of analyzing packets and proves to be not only effective but efficient as well for many network traffic due to the fixed port numbers associated with various protocols. Accurate traffic classification is possible for well-known port numbers in the range between 1 and 1023 port numbers. However, identifying the packets associated with the applications that do not use well-known port numbers can be challenging in this regard. An example of this are P2P Applications. These do not use well-known port numbers, and if the purpose is to identify P2P applications, then using machine learning to classify protocols would be a better solution.

The methodology that we presented followed the capturing and analyzing of live data that was done using windump. The packets were analyzed easily by the identification of their source and destination numbers. We developed our tool with effective GUI techniques that uses the input data capture file to classify different packets according to their port numbers and display the results. The evaluation of our tool showed 100% accurate classification of the traffic captured. The promising results suggest the reliable packet classification for applications using well known port numbers. Applications, then using machine learning to classify protocols would be a better solution.

VI. Refrences

[1] M Dashevskiy, Z Luo. *Conformal Prediction for Reliable Machine Learning*, Morgan Kaufmann Publishers Inc, 2004.

[2] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, J. Lloret, "*Network traffic classifier with convolutional and recurrent neural networks for Internet of Things*", *IEEE Access*, vol. 5, pp. 18042-18050, 2017.

[3] "Traffic classification", *En.wikipedia.org*, 2020. Available: https://en.wikipedia.org/wiki/Traffic_classification.

[4] T. Auld, A. W. Moore, S. F. Gull, "Bayesian neural networks for Internet traffic classification", *IEEE Trans. Neural Netw.*, vol. 18, no. 1, pp. 223-239, Jan. 2007

[5] X. Xie, B. Yang, Y. Chen, L. Wang, Z. Chen, "Network traffic classification based on error-correcting output codes and NN ensemble", *Proc. 6th Int. Conf. Fuzzy Syst. Knowl. Discovery*, pp. 475-479, Aug. 2009

[6] Z. Chen et al., "Improving neural network classification using further division of recognition space", *Int. J. Innov. Comput. Inf. Control*, vol. 5, no. 2, pp. 301-310, 2009.

[7] Thomas Karagiannis, Andre Broido, Michalis Faloutsos, and Kc claffy. Transport layer identification of P2P traffic. In 4th ACM SIGCOMM Internet Measurement Conference (IMC'04), Taormina, IT, October 2004.

[8] Kuai Xu, Zhi-Li Zhang, and Supratik Bhattacharyya. Profiling internet backbone traffic: behavior models and applications. ACM SIGCOMM Comput. Commun. Rev., 35(4):169– 180, 2005

[9] Marios Iliofotou, Prashanth Pappu, Michalis Faloutsos, Michael Mitzenmacher, Sumeet Singh, and George Varghese. Network monitoring using traffic dispersion graphs (tdgs). In Proc. of IMC '07, San Diego, California, USA, 2007.

[10] Yu Jin, Nick Duffield, Patrick Haffner, Subhabrata Sen, and Zhi-Li Zhang. Inferring applications at the network layer using collective traffic statistics. SIGMETRICS Perform. Eval. Rev., 38, June 2010

[11] Vern Paxson. Bro: a system for detecting network intruders in real-time. Elsevier Comput. Netw., 31:2435–2463, December 1999

[12] Sung-Ho Yoon, Jin-Wan Park, Jun-Sang Park, Young-Seok Oh, and Myung-Sup Kim. Internet Application Traffic Classification using Fixed IP-Port. In 12th Asia-Pacific Network Operations and Management Symposium, APNOMS 2009.

[13] Patrick Schneider. TCP/IP Traffic Classification Based on Port Numbers, Harvard University, 29 Oxford Street, MA 02138, USA, 2012