NETWORK TRAFFIC ANALYZER

A COURSE PROJECT REPORT

By

AARIKATLA THARUN KUMAR (RA2011030010033)
POTHINA PRAVEEN (RA2011030010036)
VELURU MANOJ (RA2011030010021)

Under the guidance of

<Dr.P.Visalakshi>

In partial fulfilment for the Course

of

18CSC302J - COMPUTER NETWORKS

in <Department of Network and Communications>

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Chenpalpattu District

NOVEMBER 2022

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this mini project report "NETWORK TRAFFIC ANALYZER" is the bonafide work of **AARIKATLA THARUN KUMAR (RA2011030010033), POTHINA PRAVEEN (RA2011030010036)** and **VELURU MANOJ (RA2011030010021)**who carried out the project work under my supervision.

SIGNATURE

<**Dr.P.Visalakshi**>
<Assistant Professor>
<Department of Network and Communications>
SRM Institute of Science and Technology

1. ABSTRACT

The rapid growth of Internet Traffic has emerged as a major issue due to the rapid development of various network applications and Internet services. One of the challenges facing Internet Service Providers (ISPs) is to optimize the performance of their networks in the face of continuously increasing amounts of IP traffic while guaranteeing some specific Quality of Services (QoS). Therefore it is necessary for ISPs to study the traffic patterns and user behaviors in different localities, to estimate the application usage trends, and thereby to come up with solutions that can effectively, efficiently, and economically support their users traffic.

The main objective of this thesis is to analyze and characterize traffic in a local multi-service residential IP network in Sweden (referred to in this report as “Network North”). The data about the amount of traffic was measured using a real-time traffic-monitoring tool from Packet Logic. Traffic from the monitored network to various destinations was captured and classified into 5 ring-wise locality levels in accordance with the traffic's geographic destinations: traffic within Network North and traffic to the remainder of the North of Sweden, Sweden, Europe, and World. Parameters such as traffic patterns (e.g., traffic volume distribution, application usage, and application popularity) and user behavior (e.g., usage habits, user interests, etc.) at different geographic localities were studied in this project. As a result of a
systematic and in-depth measurement and the fact that the number of content servers at the World, Europe, and Sweden levels are quite large, we recommend that an intelligent content distribution system be positioned at Level 1 localities in order to reduce the amount of duplicate traffic in the network and thereby removing this traffic load from the core network.

The results of these measurements provide a temporal reference for ISPs of their present traffic and should allow them to better manage their network. However, due to certain circumstances the analysis was limited due to the set of available daily traffic traces. To provide a more a trustworthy solution, a relatively longer-term, periodic, and seasonal traffic analysis could be done in the future based on the established measurement framework.

ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable **Vice Chancellor Dr. C. MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors.

We would like to express my warmth of gratitude to our **Registrar Dr. S. Ponnusamy,** for his encouragement

We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. T. V.Gopal,** for bringing out novelty in all executions.

We would like to express my heartfelt thanks to Chairperson, School of Computing **Dr. Revathi Venkataraman,** for imparting confidence to complete my course project

We wish to express my sincere thanks to **Course Audit Professor Dr.Annapurani Panaiyappan, Professor and Head, Department of Networking and Communications** and **Course Coordinators** for their constant encouragement and support.

We are highly thankful to our my Course project Faculty **<Dr.P.Visalakshi> , <Assistant Professor> , <Department of Network and Communications>,** for his/her assistance, timely suggestion and guidance throughout the duration of this course project.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project.

TABLE OF CONTENTS

2. INTRODUCTION

Internet traffic has been constantly increasing with the revolutionary developments in communication networks and applications. Global IP traffic is predicted to increase threefold over the next 5 years in Cisco's report on global IP traffic forecast for 2011–2016[1] . The diversified development of communication methods has not only increased demand for Internet access, but also brought heavier network traffic loads. As revealed in [1], most IP traffic originating with PC devices has a tendency to continue to generate increasing traffic loads, meanwhile the traffic generating by non-PC devices would will double in the next few years.

The greatly increased user demands have caused the Internet to successfully evolve into a mainstream market from an esoteric niche. The Internet service providers (ISPs), on one hand, have realized the business opportunities and rapidly developed a wide variety of network applications and Internet services, which in turn brought in considerable revenue while generating increasing traffic loads. On the other hand, ISPs are obsessed with the traffic stress associated with offering various services. Therefore, there is a need to consider potential network management solutions.

The question of how to avoid traffic bottlenecks is obsessing ISPs all of the time. An efficient method to address network traffic issue is to monitor the network performance based upon real time continuous data collection, and by understanding the network traffic patterns to propose effective and economical solutions to support the expected traffic.

ISPs connect end users to the Internet. Additionally, these ISPs exchange traffic with other ISPs so that the users connected to different ISPs can communicate with each other. This is called interconnection [2]. The growing amount of network traffic transiting the Internet has required tremendous expenditures by ISPs. However, the ISPs want to minimize the cost of operating their business.

A common way to reduce the network traffic and cost for ISPs is to use peering between two or among several ISPs[3]. Figure 1-1 shows the basic topology of these network interconnections, in which transiting and peering are the two main functions.

Transiting is a simple service that forward packets from one user to the upstream ISP, and the upstream ISP decides where these packets should be forwarded based upon entries in its routing table. ISPs need to defray certain expenses to obtain access to the upstream ISP's routing[4]. When two service providers have nearly same network scale, cost, and traffic volumes, it is unnecessary for each of them to pay a transit fee in both directions, as they would be paying each other equal amounts of money. In this case the service providers will implement a peering solution.

3. LITERATURE SURVEY

This project presents a survey on various such network analysis and traffic prediction techniques. Moreover, various accomplished areas of analysis and prediction of network traffic have been summed.
The present study is a survey of various works carried out in the field of network traffic analysis and prediction.

4. REQUIREMENTS

Since, We are asked to analyze the traffic in the network. We are also asked to filter the packets using different criterias like protocol,Source address etc. These can be easily achieved if we can

- Capture all the packets that are owing through the network interface and if we can
- Capture these packets as a whole i.e before stripping of the packet begins. (Our packet should contain the data along with the other headers)

Technologies and tools:

- Analyzer developed using Python 2.7
- MySQL – stores all the info in encrypted format.
- GeoLite City database – provides visual location of destination using Google Earth.
- PyGeoLite – correlates IP address to physical location which retrieves the latitude, longitude and region of the user.
- Wireshark – used to capture packets (pcap files).
- Ettercap – network security tool used in computers for security auditing and network protocol analysis. It is used for intercepting traffic on network segment, conducting active eavesdropping against a number of protocols and capturing passwords.
- Google Earth API – displays geographical location of the destination of the packet.

5. ARCHITECTURE AND DESIGN

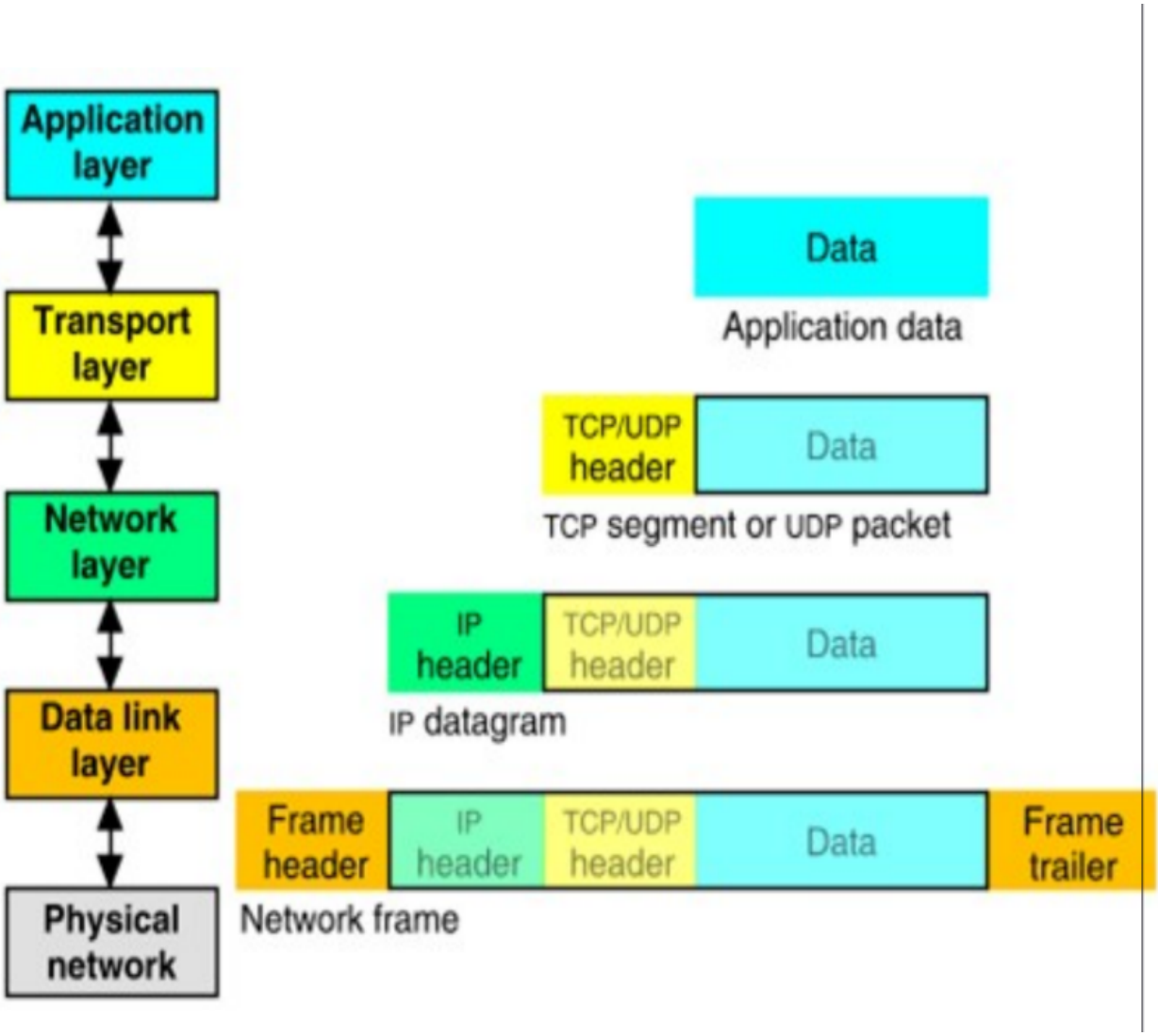
5.1 Network Architecture

The network architecture is as follows:

The diagram below exhibits a general view of the network traffic analyzer (Image source - Google)

The architecture consists of three major networks:

- Company Network(s)
- Public Internet
- Network maintained by the Internet Service Provider



The picture given above explains the stripping of the packet clearly.

6. IMPLEMENTATION

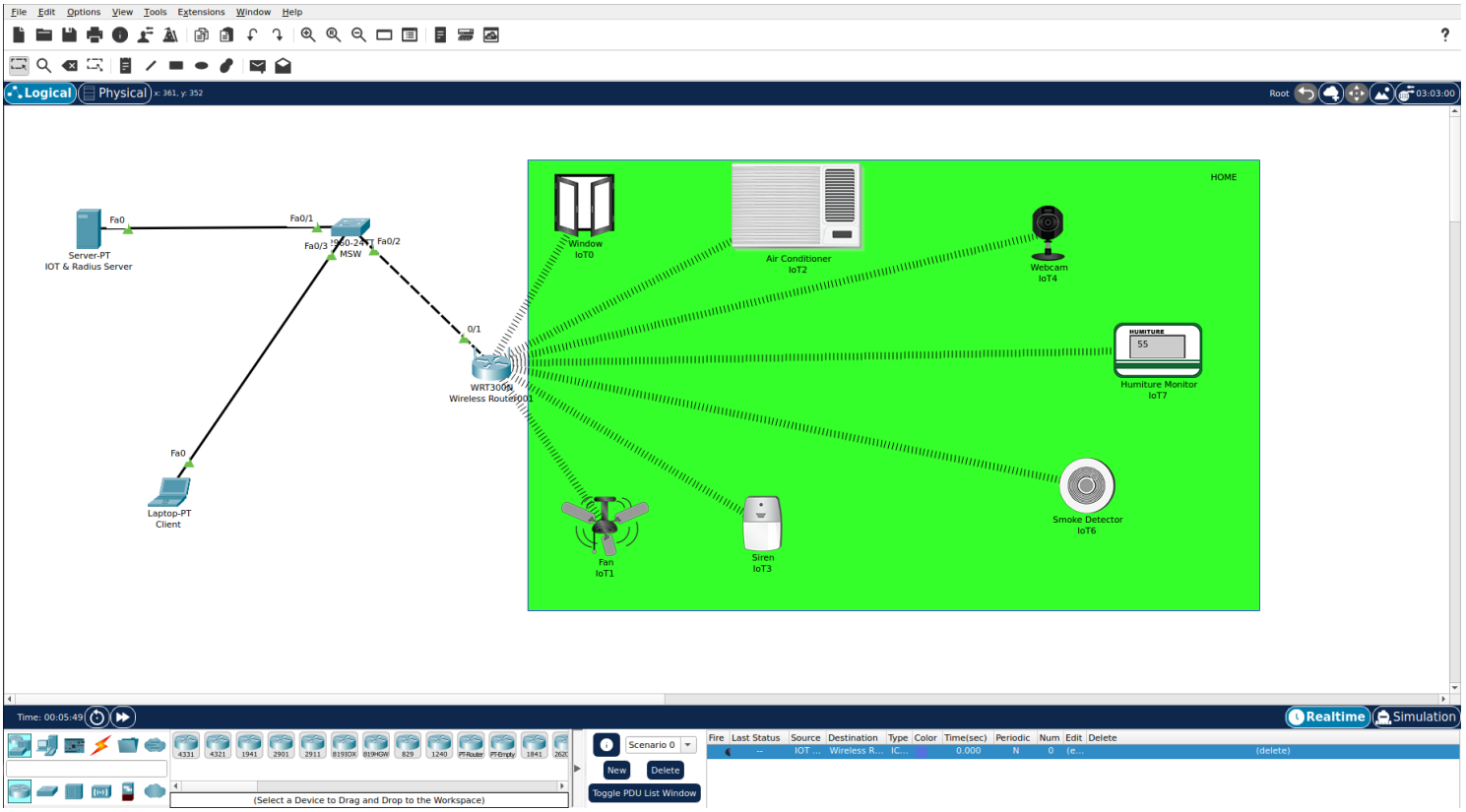
- Initially, packets are captured using wireshark and live packet are captured using Ettercap.
- Dpkt – a python module used as an analyzing tool for parsing packets. It analyzes each individual packet and analyzes the protocol layer. It provides IP address of the user downloading the application from an illegal or a blacklisted site.
- The geographical location of the packet can be found by using PyGeoIP which queries the GeoLiteCity database.
- Google maps API gives a geographical display of the destination of the packet that is being analyzed in KML format. All the information about the packet i.e. source and destination IP, timestamp, URL, geographical latitude, longitude and location plotted gets stored in the database.
- All the information stored in the database is encrypted for security purposes.

MONITOR NETWORK OF A NETWORK MODEL USING CISCO PACKET TRACER

Actually network monitor is a must, because if you build a network and if you don't monitor your network, then it can be harmful. So network monitor of this network model is a must. If you monitor your network on a daily basis, you can be able to identify unauthorized access into your network. So it is recommended to monitor the network regularly. We'll analyze network virtually using cisco packet tracer.

NETWORK MODEL

First we build a small network as a network model and after that we'll see how our networks works by monitoring network of this network model. We'll also monitor our network more and more deeply. We'll do all this works using a network simulator software.



Source Code:

```
import sys
import pygeoip
import dpkt
import socket
import time
import webbrowser



# These dictonaries are used for the key value pairs. Keys being the IP addresses and the values being the lat-long position
abc={}
dest_abc={}
black_listed_ip=['217.168.1.2','192.37.115.0','212.242.33.35','147.137.21.94']
# This dictionary is used for keeping the record of authorized users.
auth_users={"root":"root","soumil":"soumil"}
"""
This function is used for generating information from the IP addresses. It uses the GeoLiteCity data base for doing this .
'gic.record_by_addr()' return a dictonary that has all the parameters of the corresponding IP address. We can print any of them.
"""
def geoip_city(string):
                        if string in black_listed_ip:
                          path='/home/soumil/build/geoip/GeoLiteCity.dat'
```

```
                        gic=pygeoip.GeoIP(path)
                        #print gic
                        try:
                                a=gic.record_by_addr(string)
                                #print a
                                pcity=a['city']
                                pregion=a['region_code']
                                pcountry=a['country_name']
                                plong=a['latitude']
                                plat=a['longitude']
                                print  "\n"
                                print '[*] Target : ' + string + ' Geo Located .'
                                print " \n"
                                print '[+] City: '+ str(pcity) + ', Region : '+
str(pregion) + ', Country: ' + str(pcountry)
                                print "\n"
                                print '[+] Latitude : ' +  str(plat) + ',
Longitude : ' + str(plong)
                                print "\n"
                        except:
                                print " \n ********** IP Unregistered
**********"
                    else:
                       pass
"""
```

We are using this function to generate latitudes and longitudes an IP

address from the GeoLiteCity database. We will be using these Latitudes and longitudes fro plotting placemarks on google maps (my maps). This function is used for Source IP addresses.

"""

```
def kml_geoip_city(string):
                        if string in black_listed_ip:
                          path='/home/soumil/build/geoip/GeoLiteCity.dat'
                          gic=pygeoip.GeoIP(path)


                          try:
                                a=gic.record_by_addr(string)
                                pcity=a['city']
                                plong=a['latitude']
                                plat=a['longitude']
                                abc[str(string)]=str(plat)+","+str(plong)
                                # Here we are inputing the IP:lat,long to the source Dictionary defined earlier.
                          except:
                                pass
                        else:
                          pass
```

"""

We are using this function to generate latitudes and longitudes an IP address from the GeoLiteCity database. We will be using these Latitudes and longitudes fro plotting placemarks on google maps (my maps). This function is used for Destination IP addresses.

```
"""
def kml_dest_geoip_city(string):
                        if string in black_listed_ip:
                          path='/home/soumil/build/geoip/GeoLiteCity.dat'
                          gic=pygeoip.GeoIP(path)
                          try:
                                a=gic.record_by_addr(string)
                                pcity=a['city']
                                plong=a['latitude']
                                plat=a['longitude']
                                dest_abc[str(string)]=str(plat)+","+str(plong)
                                # Here we are inputing the IP:lat,long to the
destination dictionary defined earlier.
                          except:
                                pass
                        else:
                          pass
"""
The printcap function is used for printing information related to a
particular IP address. This function takes in a pcap.
It uses the dptk module in python to parse the pcap to find out all the
source IP and destination IP of all the packets.
It now prints all the information on the screen in text format.
"""
def printpcap(pcap):
                        for (ts,buf) in pcap:
```

```
                        try:
                                eth=dpkt.ethernet.Ethernet(buf)
                                ip=eth.data
                                src=socket.inet_ntoa(ip.src)
                                dst=socket.inet_ntoa(ip.dst)
                                if src in black_listed_ip:
                                        print
"--------------------------------------------------------------------------------
----------"
                                        print '[+] Source IP: '+str(src)+ '------->
Destination IP: '+ str(dst)
                                        print "Source IP Information:"
                                        print geoip_city(str(src))
                                elif dst in black_listed_ip:
                                        print
"=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=--=-"
                                        print "Destination IP Information:"
                                        print geoip_city(str(dst))
                                        print
"--------------------------------------------------------------------------------
----------"
                                else:
                                        pass
                        except:
                                pass
"""
```

The view_google function is some what similar to the printpcap function that is written above.
Even this function takes in a pcap, parses it using 'dpkt' and then sends the source and dest ip addresses
to the fnction that generates kml format for the data so that it can be used to represet on a graph.
"""

```
def view_google(pcap):
                        for (ts,buf) in pcap:
                          try:
                                eth=dpkt.ethernet.Ethernet(buf)
                                ip=eth.data
                                src=socket.inet_ntoa(ip.src)
                                dst=socket.inet_ntoa(ip.dst)
                                kml_geoip_city(str(src))
                                kml_dest_geoip_city(str(dst))
                          except:
                                pass
```

"""
The print_placemarks_in_kml(abc) funtion is used for generating the kml format of the data for the representation of the
google maps. It takes in a dictionary that contains key-value pairs (IP:lat,long). And inturn generate the kml format

of the data. This function is used for generating kml for Source IP addresses.

```
"""
def print_placemarks_in_kml(abc):
                        for i in abc.keys():
                          print """
  <Placemark>
    <name> SOURCE IP Address: %s</name>
    <styleUrl>#exampleStyleDocument</styleUrl>
    <Point>
      <coordinates>%s</coordinates>
    </Point>
  </Placemark>
"""%(i,abc[i])
"""
```

The print_dest_placemarks_in_kml(dest_abc) funtion is used for generating the kml format of the data for the representation of the google maps. It takes in a dictionary that contains key-value pairs (IP:lat,long). And inturn generate the kml format
of the data. This function is used for generating kml for destication IP addresses.

```
"""
def print_dest_placemarks_in_kml(dest_abc):
                        for i in dest_abc.keys():
                          print """
  <Placemark>
```

```
    <name> DESTINATION IP Address: %s</name>
    <styleUrl>#exampleStyleDocument</styleUrl>
    <Point>
      <coordinates>%s</coordinates>
    </Point>
  </Placemark>
"""%(i,dest_abc[i])



"""
Now the main function starts :
The first try-catch block is used to validate if the user is authorized to use
this tool. If the username entered by the user is present in
our record (i.e. the dictionary"auth_users"), he/she gets to use the tool.
Else One cannot access the tool. This can be considered as one of
the security feature of our tool.
"""
if len(sys.argv)<2:
                        print "\n ---------- Please enter the required
arguments---------- "
                        print "\n Correct syntax is : <FileName>.py
username password cli/kml\n"
                        print "\ncli- stands for Command Line Output"
                        print "\nkml- stands fro a KML output which is
required for visualization using a Google Map"
else:
```

```
                    try:
                      if str(sys.argv[1]) in auth_users:
"""
```

The second try-catch block is used to validate the password entered by the user to the corresponding entered username. Once the username is validated, we must validate the password as well. This is used fro securit hardening of the tool.

```
"""


                        try:
                    if str(sys.argv[2])==auth_users[str(sys.argv[1])]:
"""
```

The user must specify what type of output does he/she desire. If he/she desires an output on the command line or a textual outputhe should use "cli" option. If he/she desires to use maps for visualizaing the contents of the analysis. There is a need for a KML file that must be uploaded to the google maps webiste that opens an the end. For Visualizing it output on the maps. He/She must redirect the output of the program using pipes to a ".kml" file. This .kml file is then uploded on the mymaps website.

```
"""
                                        try:
                                            if str(sys.argv[3])=="cli":
f=open('/home/soumil/Downloads/fuzz-2006-06-26-2594.pcap')
                                        pcap=dpkt.pcap.Reader(f)
                                            printpcap(pcap)
                                            f.close()
```

```
                sys.exit(1)
            elif str(sys.argv[3])=="kml":
                print """
            <?xml version="1.0"
encoding="UTF-8"?>
            <kml
xmlns="http://www.opengis.net/kml/2.2">
            <Document>

<name>sourceip.kml</name>
              <open>1</open>
              <Style
id="exampleStyleDocument">
                <LabelStyle>
            <color>ff0000cc</color>
                </LabelStyle>
              </Style>\n"""
f=open('/home/soumil/Downloads/fuzz-2006-06-26-2594.pcap')
            pcap=dpkt.pcap.Reader(f)
                view_google(pcap)
                f.close()
        print_dest_placemarks_in_kml(dest_abc)
            print_placemarks_in_kml(abc)
                #print abc
                print """\n
            </Document>
```

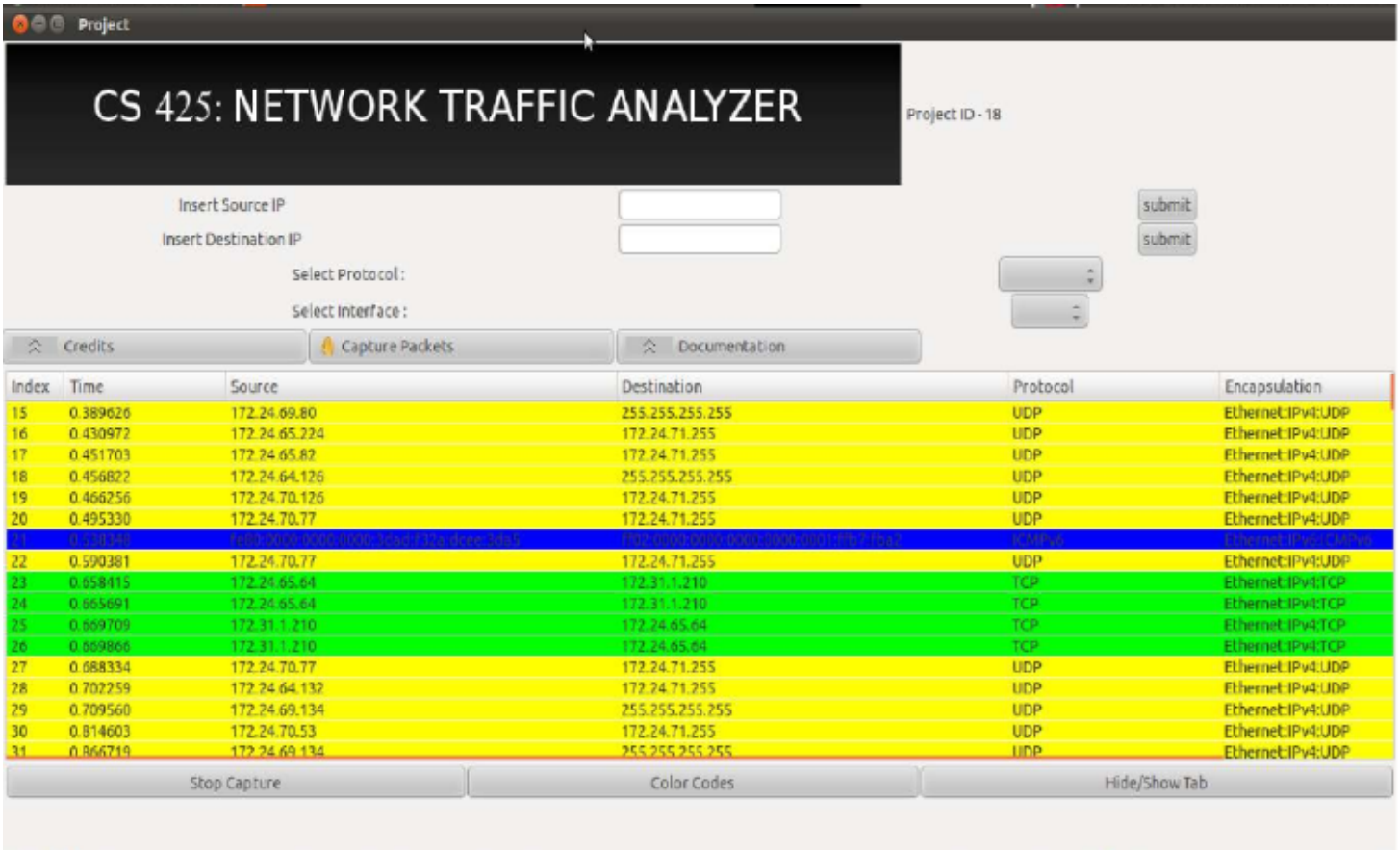
```
                                        </kml>
"""
new=1
url="https://www.google.com/maps/d/splash?app=mp"
webbrowser.open(url,new=new)
                                        else:
                                            raise exception


                                    except:
                                        print "\nYou Entered a
worng option. Or may be your Syntax is wrong"
                                        print "\n Correct syntax is :
<FileName>.py username password cli/kml\n"
                                        print "\ncli- stands for
Command Line Output"
                                        print "\nkml- stands fro a
KML output which is required for visualization using a Google Map"
                                    else:
                                        raise exception
                                except:
                                    print "\n The PASSWORD you entered
is NOT CORRECT !!!!!! "
                                    print "\n Correct syntax is :
<FileName>.py username password cli/kml\n"
                                    print "\ncli- stands for Command Line
```

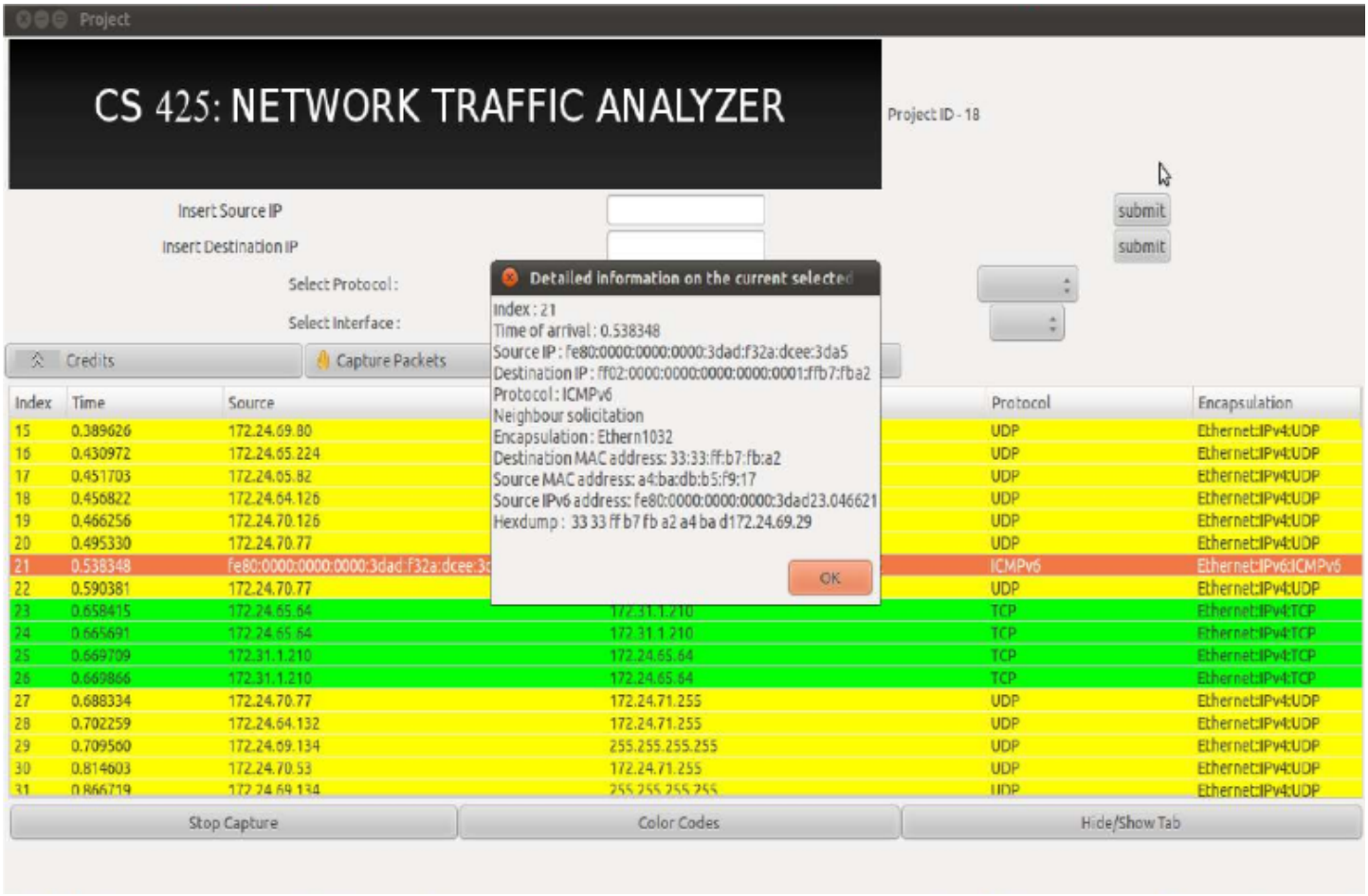
```
Output"
                                        print "\nkml- stands fro a KML output
which is required for visualization using a Google Map"
                                else:
                                        raise exception
                            except:
                                print "\n Sorry %s. You are NOT AUTHORIZED
to use this tool!!!!!!!!!!!"%str(sys.argv[1])
                                print "\n Correct syntax is : <FileName>.py
username password cli/kml\n"
                                print "\ncli- stands for Command Line Output"
                                print "\nkml- stands fro a KML output which is
required for visualization using a Google Map"
```

7. RESULTS AND DISCUSSION

- While capturing packets first choose the interface and other filters you want to apply
- Click on the ‘Capture Packets’ button.



- After capturing packets or during capturing if you click on a particular row you get a pop-up with the detailed information of packet printed on it.



8. CONCLUSION AND FUTURE ENHANCEMENT

This network traffic analyses show there is unwise bandwidth utilization in the campus network, and is a serious and emerging challenge for almost all organisations in the present world information technology. Lack of appropriate bandwidth management is preventing useful Internet access which in turn yielding low quality of academic and research works. Better management of bandwidth makes Internet access wider especially for those who need it in actual. Unfortunately there is relatively little understanding about the importance of managing bandwidth because of low awareness, lack of technical staff, improper implementation of Internet usage policy, non supportive attitude of authorities, etc. Bandwidth is a very valuable and limited resource, so there is a need to enhance awareness level among all stakeholders-students, researchers and staff within the university education system in addition to implement a common acceptable policy.

Network traffic analysis as a method of tracking network activity to spot issues with security and operations, as well as other irregularities. This article elaborates on the working, importance, implementation, and best practices of network traffic analysis.

Hence, it is a necessary condition to implement appropriate bandwidth management so as to reduce the bandwidth starvation in the campus network.

9. REFERENCES

1. https://www.studocu.com/row/document/national-open-university-of-nigeria/introduction-to-data-organization-and-management/network-traffic-analyzer-project-05082013-054125-network-trafffic-analyser/23328598
2. https://github.com/VaideheeBarde/Network-Traffic-Analyzer
3. https://www.researchgate.net/publication/339927785_A_review_of_network_traffic_analysis_and_prediction_techniques
4. https://www.researchgate.net/publication/327530819_Network_Traffic_Analysis_A_Case_Study_of_ABU_Network
5. https://www.spiceworks.com/tech/networking/articles/network-traffic-analysis/