TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI

VICTORIA
UNIVERSITY OF WELLINGTON

NWEN 302

Lab 1

Wael Aldroubi
300456658

Introduction:

Computers talk between each other by sending information as packets through the network, computers have their own language as we human have our own, their language consist of only (0,1).
Combination of 0s and 1s mean data in real life, in 1970s, the international organization for standards (ISO) came with a model explain the process of exchange information between two computers and summarized as 7 layers.
Where Network layer (Layer 3) can be understood by routers and Datalink layer (Layer 2) can understood by switches and physical layer transmit data using cables of Wi-Fi.

This assignment was set to understand how data transmit between end users and how different kind of packets are made, how routers and switches understand them and convert them from computer language (Machine language) to Human readable language.

The outcome is two applications to take traffic flow, analysis it, understand the details of each packet header and return all of its details.

What does packet sniffer do?

Our application is coded by python and C programming languages.
The concept of both is almost the same.
I will explain how it is work and then how the code is divided.

1. How the application works:
   * Usually the application listen to an open communication on the network using communication port, but in our case it will read a pcapy file, which is a file stored by another listen application like (Wireshark or TCP dump).
   * Our application will read the file line by line defining headers type (what kind of information is being transmitted), and will change all 0s and 1s to human readable format depending on the header length and protocol number.
2. How the code is divided:
   * The main function will read the file.
   * And for all headers a header handler will process this header and decide which type it is (IPV4/6-TCP-UDP-ICMP-Unknown) depending on the header information and protocol number in python.
   * Whenever the header handler defined a packet will call a function specific in that type and transfer all data into human readable format.

Python code explanation (Pseudo-code):

1) Function to read header from the data given in the pcapy file called packet header base.
2) Ethernet: function to read MAC addresses which consist of type, source and destination port and the rest of the data, the function will process the header then change it to string and print the source and destination addresses and then print the user data.
3) IPV4: function to process IP version 4 packets, it has a lot of information like (version/DSCP/Tll/length/identification/flags + offset/checksum) and then source IP and destination IP, then will print then with user data.
4) IPV6: function to process IP version 6 packets, it has various information like (version/payload/next header/ hop limit) and source IP and destination IP.
5) ARP: is used to get the MAC address of a destination device in case we only have the IP address. It has a lot of information as well but it has MAC and IP address for both Source and destination.
6) TCP: When sending information is important and when packets lost, resending it is important, what make it unique is the ACK (acknowledgment), sequence number and of course source and destination.
7) UDP: When sending information is not important to deliver complete, especially when streaming.
8) ICMP: Internet control message protocol, the most frequently used protocol, handle errors messages and provide troubleshooting, at network layer. It has type, code, checksum and rest of the header, the difference between it and TCP, UDP, is it does not send data, it just handle error messages.
9) Unknown: when the header is not defined, it will print a message to user.
10) Process packet: is a function to process the file line by line get the header type and call the desired function from above.
11) Main function: will read the pcapy file and call the process packet function to translate the data into human readable format.

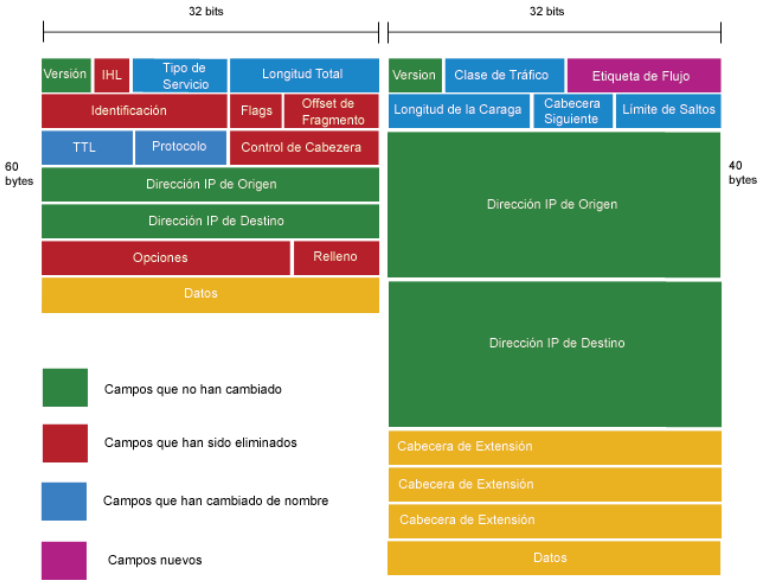
C code explanation (Pseudo-code):

C programming language has libraries to handle all kind of packets, so no need to define how many bytes for each header section, need to define global variables and call them within functions.

1) Main function: to read pcapy file and traverse the document capturing the headers and process them.
2) Handle packet: is the function responsible of process each packet after main function sort their type, will process IPV4 types and if it is IPV6 will send it to handle IPV6 handle function.
3) Handle IPV6: function to process packets when their header recognized as any IPV6 types.
4) Print IPV4: to print IP version 4 packets.
5) Print IPV6: to print IP version 6 packets.
6) Print TCP: function to process and print tcp packets.
7) Print UDP: function to process and print udp packets.
8) Print ICMPv6: function to process and print ICMPv6 packets.
9) Print payload: to print user data.

ICMPv4 is handled in handle packet function to print it and to print TTL (Time to live) or to return the packet.

Some headers:

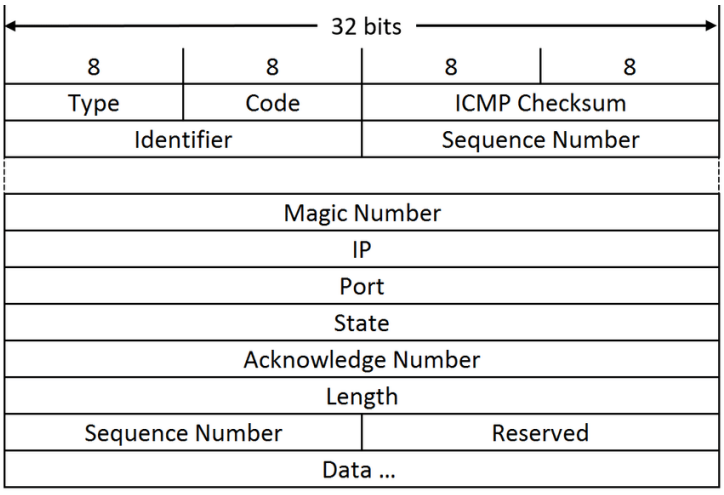


TCP Segment Header Format

Bit #	0 – 7	8 – 15	16 – 23	24 – 31
0	Source Port		Destination Port	
32	Sequence Number			
64	Acknowledgment Number			
96	Data Offset	Res	Flags	Window Size
128	Header and Data Checksum		Urgent Pointer	
160...	Options			

UDP Datagram Header Format

Bit #	0 – 7	8 – 15	16 – 23	24 – 31
0	Source Port		Destination Port	
32	Length		Header and Data Checksum	



Conclusion and results:

There was a lot of tutorials and explained codes online most of them applications listening to port 80 (Internet port), the hard part was to read from file, to change the source of info from port to file.

The code is organized and explained, comments added in every section to explain the functionality and when needed.

This application is similar to Wireshark and Tcp dump applications but both of them has unfriendly interface, so I do recommend to implement this application to be connected to a nice friendly interface, so families can watch internet traffic for their children to protect them from bad sites like dark web or porn sites, will help computer people to watch when hackers listen to their traffic.

References

buckyroberts. (2015, Dec 30). *github*. Retrieved from github: https://github.com/buckyroberts/Python-Packet-Sniffer/blob/master/sniffer.py

Essa, M. (2015, Dec 19). *youtube*. Retrieved from youtube: https://www.youtube.com/watch?v=4KbWu6yqDCY&index=92&list=PLMYF6NkLrdN9wzmjRlcO1UsqgO6KUTODC

Kendzierski, L. (2015, Feb 23). *scribd*. Retrieved from scribd: https://www.scribd.com/document/259948478/Packet-Sniffer-Code-in-C-Using-Sockets-Linux

Mompeán, J. (2010, Nov 7). *youtube*. Retrieved from youtube: https://www.youtube.com/watch?v=hVdNck7gi8A

moon, S. (2009, Apr 28). *binarytides*. Retrieved from binarytides: https://www.binarytides.com/packet-sniffer-code-c-libpcap-linux-sockets

tcpdump. (n.d.). *tcpdump*. Retrieved from tcpdump: www.tcpdump.org/sniffex.c

thenewboston. (2015, Dec 29). *youtube*. Retrieved from youtube: https://www.youtube.com/watch?v=dM9grWOdTBI&list=PL6gx4Cwl9DGDdduy0IPDDHYnUx66Vc4ed&index=3

TheSecurityTube. (2012, Mar 9). *youtube*. Retrieved from youtube: https://www.youtube.com/watch?v=O-tp0EYYMWg&t=505s

V's, C. (2017, Oct 10). *youtube*. Retrieved from youtube: https://www.youtube.com/watch?v=Js2_0955n3o

Acknowledgment:

Some of my code took from the following resources:
-https://www.binarytides.com/packet-sniffer-code-c-libpcap-linux-sockets/
-http://www.tcpdump.org/sniffex.c
-https://www.scribd.com/document/259948478/Packet-Sniffer-Code-in-C-Using-Sockets-Linux
-https://www.youtube.com/watch?v=O-tp0EYYMWg&t=505s
-https://www.youtube.com/watch?v=Js2_0955n3o
https://www.youtube.com/watch?v=WGJC5vT5YJo&list=PL6gx4Cwl9DGDdduy0IPDDHYnUx66Vc4ed (thenewboston)(English)
-https://www.youtube.com/watch?v=4KbWu6yqDCY&t=442s (Muhammed Essa)(Arabic)
-https://www.youtube.com/watch?v=ghokDuCDcMY&t=27s(Ana Balica)(English)
-https://www.youtube.com/watch?v=vVNqNeXninE&t=69s(Hacktilizer)(English)
-https://www.youtube.com/watch?v=Q9sqfPVadDY&t=76s(Learn Just What Needed - Python)(No sound just video)

https://en.wikipedia.org/wiki/List_of_IP_protocol_numbers