### **Course Number :-CS F303**

### **Course Title :- Computer Networks**

### **Group Number :- G5**

**PROTOCOL ANALYST**



**BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI**

### Group Members:-

### AKHIL KUMAR AGRAWAL 2014A7PS024P GAURANG BANSAL 2014A7PS128P SUBHAM KUMAR 2014A7PS121P

### 7th April 2017

# **Problem Statement:**

A packet analyzer is an application which is generally used to intercept and log traffic in a network. It captures packets from the socket and decodes the data, extracting any useful information which would help in analyzing network problems, provide network statistics report and network history. There already exist many protocol analysers such as Wireshark, but they are quite bulky and built for a wide range of services leading to increase in complexity. A user may not require the additional functionalities for analyzing the network. Its required to design a simple application which passively captures the packets and extract useful information even in a highly constrained environment.

# **Status of Project:**

We are developing a Protocol Analyst, an application which is capable of capturing and analyzing the packets and extracting the necessary information required by user.

|  |  |  |  |
| --- | --- | --- | --- |
| **Topic** | **Date** | **Mentor** | **Status** |
| Literature Search and Review | 18th February - 3rd March | Akhil Gaurang Subham | **Completed** |
| Analysis and Modeling | 13th March - 16th March | Subham | **Completed** |
| Pseudo Code Generation & Socket Networking | 17th March - 20th March | Gaurang | **Completed** |
| Designing a Simple Prototype | 20th March - 24th March | Subham & Gaurang | **Completed** |
| Network monitor & Protocol Capturing | 25th March - 29th March | Akhil | **Completed** |
| Packet Filtering | 27th March - 1st April | Gaurang | **Completed** |
| Network Utilities & Code Optimization | 1st April - 5th April | Subham | **Almost Completed** |
| Packet Log File & User Display | 7th April - 10th April | Akhil | **On Going** |

**DESIGN DETAILS**

We are implementing a simple Packet-Capturing and Analysis Tool (“Protocol Analyst”) that will be able to capture application, transport, network and data link level structured PDUs.We have provided a brief overview of Design aspects of our model, its execution and implementation details.

**Implementation Aspects and Code Details**

The Protocol Analyst is designed to capture all the packets coming to and going via the ethernet interface. The program for “Protocol Analyst” is written in C Language and is designed for Linux.

We will try to evolve the program for other interfaces as a possible extension of this project. Libraries have been used to the minimum. For capturing the necessary information we have created our own headers of type “my\_protocol” header structure in “packet\_capture.h” file.

**User Interaction**

User can run the “Protocol Analyst” using various command line options.Then the user can open any network application like a browser along with the “Protocol Analyst” running in a terminal. He can analyse data based on:

1. Number of Packets :- User can specify the number of packets he wishes to capture.
2. Maximum Time :- User can specify the maximum time limit for capturing packets.
3. Key stop :- Press a particular key to stop the capturing of packets whenever required.

One of the possible extensions will be to provide the user a choice to select the interface as well.

**Packet Capturing**

The packet analyzer aims to capture packets coming to and going from the host on which it has been set up. It passively captures the packets and then examines the packets to get the relevant information. The packet is reassembled and the headers are taken out,decoded and the information presented as desired by the user. We are using raw sockets to capture packets. Raw sockets are used for direct communication without any protocol specific formatting.

1. ETH\_P\_ALL : It is used to capture all the packets from ethernet interface. All incoming packets of that protocol type will be passed to the packet socket before they are passed to the protocols implemented in the kernel.
2. We have used SOCK\_RAW as the socket type so that the packets pass without any changes in the packet data having the link layer headers.
3. AF\_PACKET denotes Address Family used in the socket which lets us access all packets.

We have allocated a large buffer space that will store the packets entering the socket and will be used during the analysis phase of the packet headers and print useful data in corresponding log files. A raw socket allows an application to directly access lower level protocols, which means a raw socket receives un-extracted packets. After receiving the packet we store it in buffer using saddr structure. After the data is stored in buffer we will use the pointer to access the buffer to get relevant information.

**JOURNEY OF PACKETS THROUGH VARIOUS LAYERS**The packet received goes through different layers from Data Link Layer to Application Layer. At each layer we extract the header and examine it for useful information.

**Data Link Layer**The Ethernet header contains the physical address of the source and destination, also known as MAC address and the protocol of the receiving packet.We will be focusing on information regarding link layer protocols such as ARP (Address Resolution Protocol). Using the ethernet header we can extract useful information such as:  
● Destination Address ● Source Address ● Protocol Number as given by IANA  **Network Layer**The payload of the DLDU (Data Link Data Unit) goes above to the network layer.To get the information regarding network layer header, we will start from end of Ethernet header. After getting the ip header we can use this information to know which protocol is being used by checking the protocol number as defined by Internet Assigned Numbers Authority (IANA).For example:  
● 1 for [Internet Control Message Protocol](https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol) ● 6 for Transmission Control Protocol

**1. ICMP**

Internet Control Message Protocol (ICMP) is a supporting protocol to Internet Protocol in the TCP/IP Architecture. It is commonly used in network devices like routers, to send control information such as error messages. The ICMP header starts after the [IP header](https://en.wikipedia.org/wiki/IPv4#Header) so we will use the ip header information and use it to extract the ICMP data. We have extracted information such as:

● Checksum ● Code

* Type which includes Control Information like

○ Echo Reply ○ Echo Request

**2. IGMP**

The [Internet Grou](https://en.wikipedia.org/wiki/IGMP)p [Management Protocol](https://en.wikipedia.org/wiki/IGMP) is used by hosts to establish multicast group memberships. It is also used for one-to-many networking applications like streaming videos online. They are also extracted using the IP Header information.The extracted data contains:

● Checksum ● Type ● Code

**3. IP**

The Internet Protocol (IP) is the most popular Network layer protocol and is used for relaying [datagrams](https://en.wikipedia.org/wiki/Datagram) across a network. The [IP header](https://en.wikipedia.org/wiki/IP_header) has information about metadata needed to route and deliver the datagram. IP packets have been captured and analysed to extract useful information, some of them being:

● Source IP ● Destination IP ● IP Version

● Checksum ● IP total length ● IP header length

● Type of service ● Time to live ● Protocol

**Transport Layer**

The payload of Network Layer Data Unit (NPDU) goes to the transport layer. The transport layer is responsible for delivering data to the appropriate application process on the host computers.The information coming from this layer is important for determining the services being run on the specified port numbers.

**1. TCP**

Transmission Control Protocol (TCP) is one of the most widely used transport protocols. To capture the TCP information which provides [reliable](https://en.wikipedia.org/wiki/Reliability_(computer_networking)), ordered, and [error-check](https://en.wikipedia.org/wiki/Error_detection_and_correction)ing delivery of [octets](https://en.wikipedia.org/wiki/Octet_(computing)) we use tcphdr which we calculate after removing the header information of Ethernet and Network Layer. “My\_tcphdr” TCP Header structure is used to extract some of features of TCP such as:

● Source Port ● Destination Port ● Sequence Number ● Checksum

● Acknowledgement Number ● Window ● Flag Information

**2. UDP**

User Datagram Protocol (UDP) is extracted in a similar way to TCP, the difference is in the header used is udphdr instead of tcphdr. UDP is generally unreliable and is lightweight protocol.UDP header contains less information as compared to its counterpart TCP header which is more complex. The information extracted from UDP header are the following:

● Source Port ● Destination Port

● UDP Length ● UDP Checksum

**Application Layer**

The transport layer payload contains the Application Protocol data Unit (APDU).Various applications use the information present at this stage.The payload of the APDU is the actual message. The information extracted from the header is:

**1. HTTP**

Hypertext Transfer Protocol (HTTP) is one of most popular application protocols as it is the basis of the World Wide Web. HTTP packets are of two types one is response and other is request. Some of the fields extracted from them are listed as follows:

● Accept (Request) ● Connection ● Content-Length

● If-Modified-Since (Request) ● Last-Modified (Response) ● Allow (Response)

**2. DNS**

Domain Name System (DNS) is an application layer name space translation protocol which associates information with domain names. It uses port 53. We captured the DNS packet from payload of UDP and tried to extract some features listed below:

● Identification ● Flag Bits ● Total Questions

● Total Answer RRs ● Total Authority RR[s](http://www.networksorcery.com/enp/protocol/dns.htm#Total%20Authority%20RRs) ● Total Additional RRs

**Output Display**

The console window will display:

* Count of each Protocol and Total Number of packets
* A graph of Protocols vs Number of packets
* Total Time of Execution

The Extracted Header Details of protocols are displayed in a log file.

**APPENDIX**

Research Paper, Books & References Reviewed

1. Network Traffic Analysis and Intrusion Detection using Packet Sniffer  
   <http://ieeexplore.ieee.org/document/5437681/>
2. Review of Sockets Used for Communication Purpose  
   International Journal of Advanced Research in Computer Science and Software Engineering (Volume 3, Issue 3, March 2013) ISSN: 2277 128X  
   <https://www.ijarcsse.com/docs/papers/Volume_3/3_March2013/V3I3-0155.pdf>
3. James F. Kurose, Keith W. Ross, “Computer Networking: A Top-Down Approach”
4. Unix-Socket-FrequentlyAskedQuestions <http://www.softlab.ntua.gr/facilities/documentation/unix/unix-socket-faq/unix-socket-faq.html>
5. What Are Packet Sniffers And How Do They Work?  
   <https://www.lifewire.com/what-is-a-packet-sniffer-2487312>
6. Packet Sniffing Basics  
   <http://www.linuxjournal.com/content/packet-sniffing-basics>
7. Packet Sniffers - The Tech-FAQ  
   <http://www.tech-faq.com/packet-sniffer.html>
8. Sniffing (network wiretap, sniffer)  
   [http://cs.baylor.edu/~donahoo/tools/sniffer/sniffingFAQ.htm](http://www.tech-faq.com/packet-sniffer.html)