# AUTOMATION OF NETWORK PROTOCOL ANALYSIS

**keerthi Manchikanti**
pursuing M.Tech(CSE)
cvsr college of engineering

**J Shiva Prashanth**
Asst. Professor
cvsr college of engineering

**Vishnu Murthy G**
HOD,CSE
cvsr college of engineering

**ABSTRACT - This paper "Automation of Network Protocol Analysis" is mainly aimed to automate the entire process. Starting from sniffing the network packets till the validation of it has been taken care. Here we have automated the logging part through a C program. Whenever packets will be transmitted from a system, Ethereal/Wireshark will be automatically invoked and start capturing the network packets. That will be stored in .pcap format automatically. To validate the contents the logic has been implemented to check particular pattern of packets or any specific string. The .pcap format will be converted into a text format so that the validation can be accomplished through parsing the entire Ethereal/Wireshark log. Based upon the parsing logic, pass/fail verdict will be indicated to user. The logic can always be extended depending upon the project requirements. The performance of the network is also calculated.**

## INTRODUCTION

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Originally named Ethereal, in May 2006.

In our approach when packets are transmitted out of a system in streams or frames, we generally use tools like Ethereal/Wireshark to sniff the packets and analyze its contents to check the accuracy of it. These open source tools (i.e. Ethereal/Wireshark) are known as network protocol analyzers and they are very useful during development of software projects that are into networking domain.

Wireshark is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible. You could think of a network packet analyzer as a measuring device used to examine what's going on inside a network cable, just like a voltmeter is used by an electrician to examine what's going on inside an electric cable (but at a higher level, of course). In the past, such tools were either very expensive, proprietary, or both. However, with the advent of Wireshark, all that has changed.

Wireshark is perhaps one of the best open source packet analyzers available today. It allows the user to see all traffic being passed over the network (usually an Ethernet network but support is being added for others) by putting the network interface into promiscuous mode.

Wireshark can be helpful in many other situations too. Wireshark is software that understands the structure of different networking protocols. Thus, it is able to display the encapsulation and the fields along with their meanings of different packets specified by different networking protocols.

Wireshark uses pcap to capture packets, so it can only capture the packets on the networks supported by pcap. Data can be captured from the wire from a live network connection or read from a file that records the already-captured packets.

Live data can be read from a number of types of network, including Ethernet, IEEE 802.11, PPP, and loopback. . Wireshark runs on Unix and Unix-like systems, including Linux, Solaris, HP-UX, FreeBSD, NetBSD, OpenBSD and Mac OS X, and on Microsoft Windows. Wireshark is invoked manually for analyzing the packets.

Since wireshark should be invoked manually and even network packet analysis is manual user finds it difficult for analysis. So our approach is to mainly eliminate the manual effort where the developers or test engineers analyze the network packets manually. Here we are automating the entire process right from capturing the network packets till the analysis of it.

This module can save project cost as well as the duration to a major extent when integrated to the software development life cycle (SDLC). The logic can always be extended depending upon the project requirements. Along with analyzing the packets, we are also analyzing the performance of the network.

## RELATED WORK

**Recent work [1] has proposed protocol reverse engineering by using clustering on network traces**. This kind of approach is limited by the lack of semantic information on network traces..

**A novel approach to automatic protocol reverse engineering[2]** works by dynamically monitoring the execution of the application, analyzing how the program is processing the protocol messages that it receives. This is motivated by the insight that an application encodes the complete protocol and represents the authoritative specification of the inputs that it can Accept. In this approach

they explained about information about the fields of individual messages. Then, they aggregate this information to determine a more general specification of the message format, which can include optional or alternative fields, and repetitions.

**DICAP: Distributed Packet Capturing Architecture for High-Speed Network** Links in this approach[3] IP traffic measurements form the basis of several network management tasks, such as accounting ,planning, intrusion detection, and charging. High-speed network links ch4llenge traditional IP traffic analysis tools with their high amount of carried data that needs to be processed within a small amount of time. Centralized traffic measurements for high-speed links typically require high performance capturing hardware that usually comes with a high cost. Software-based capturing solutions, such as *lib pcap* or *PFRING*, cannot cope with those high data rates and experience high packet losses .Thus, in this approach they proposed about a scalable architecture and its implementation for Distributed Packet Capturing (DiCAP)based on inexpensive off-the-shelf hardware running Linux operating system. The prototype designed has been tested as an implementation and was evaluated against other Linux capturing tools. The evaluation shows that DiCAP can perform loss-less IP packet header capture at high speed packet rates when used alone and that it can highly improve the performance of *lib pcap* of *PFRING* when used in combination with those.

There are specialized network monitoring cards,**called Dag cards [4]** that can be installed in standardPCs in order to capture packets. Such commercial grade products are mainly intended for broadband networks and could cost many thousands of dollars. Onthe other hand it is possible to build an inexpensivesoftware-based sniffer on a high-end PC to capture packets on Fast Ethernet, or potentially on a Gigabit Ethernet network.

There are a number of enhancements in Winpcapwhich are not present in libpcap. One of these is thedump to disk capability which allows Winpcap to writepackets to disk directly from the kernel buffer without going through the user level application [5].

TCP/IP (Transmission Control Protocol/Internet Protocol) is the basic communication language or protocol of the Internet, and it is a set of protocols developed to allow cooperating computers to share resources across anetwork. The TCP/IP protocol suite is made of five layers:
physical, data link, network, transport, and application.
The layers contain relatively independent protocols that can be mixed and matched depending on the needs of the system [6].

Apart from limiting the number of exported packet records, we can also reduce the size of each record to decrease the overall data volume. This can be achieved by exporting only selected header fields and payload sections. Thus, fields which are not required for the analysis can be omitted. More over ,reduced size encoding as specified in [7] can be applied to encode small integer and float values with fewer octets thanthe original field length. For example, low port numbers (0 to255) can be encoded within a single octet instead of two.

The increased use and interconnection of electronic components in automobiles has made communication behavior in automotive networks drastically more complex. Both communication designs at application level and complex communication scenarios are often under-specified or out of scope of existing analysis techniques.

In the [8]traditional protocol analyzers in order to capture communication at the level of abstraction that reflects application design and show that the same technique can be used to specify, monitor and test complex scenarios. We present CFR (Channel Filter Rule) models, a novel approach for the specification of analyzers and a domain-specific language that implements this approach. From CFR models, we can fully generate powerful analyzers that extract design intentions, abstract protocol layers and even complex scenarios from low level communication data. We show that three basic concepts (channels,filters and rules) are sufficient to build such powerful analyzers and identify possible areas of application.

The amount of electronics and software in automobileshas been increasing rapidly over the last two decades. Modernvehicles contain a growing amount of Electronic ControlUnits (ECUs) that are in charge of different subsystems,ranging from motor control to entertainment [9]. Bus systemsconnect these distributed ECUs into communication networks and thus allow previously autonomous subsystems to exchange information in order to provide more advanced functionality. Coping with the system complexity that resultsfrom increasingly sophisticated and more and more inter connected subsystems poses one of the great challenges for the automotive industry today. Problems caused byfaulty electronics and/or software are quickly becoming thenumber one reason for car defects. Electronics and software related product recalls cost car manufacturers heavily in money and reputation. In addition to that, crucial subsystemssuch as breaks, steering and airbags require utmost reliability from software and electronics [10]. Exported packet records are received by the real-time network analysis frame workTOPAS[11]and examined by the open-source network analyzer Wireshark. Monitoring devices are configured with a Monitor Manager in order to export only data needed to achieve the analysis goal. Apart from an architectural description, this concept contains the results of experimental performance evaluations and a discussion on the advantages and limitations of our approach. Network monitoring is an important means for network administrators for supervision and fault diagnosis. In somecases, simple traffic statistics are insufficient, and deep packetinspection is necessary to trace and understand a certainoccurrence or behavior. Network monitoring and analysis onpacket level is also deployed by protocol and system engineers in order to test and debug new protocol implementations.

NetworkBehavior Analysis (NBA)[12] is a method which passivelyobserves the incoming and outgoing traffic in a network for acertain period and forms a benchmark for normal

traffic.Future behavior is compared to this benchmark to find anyusual activity in a network. Any unusual and new patterns areindicated as a threat or intrusions. IDS based on NBA are veryefficient. However there a variety of NBA Tools in market tohelp network admin in many ways. But this tool helps networkadmin to identifY any user in a network. Unlike other tools, itdoesn't look for malicious code or attacks in network traffic.

## SYSTEM DESIGN

In our proposed system it consists of client and server module. Client interacts with server by entering an option. Based upon the option, the server calls a corresponding function. The functions implemented are based on three logics. One for UDP, SIP and QoS parameters each. Another module is implemented to convert raw data packets to .pcap format by appending the pcap header.

UDP and SIP modules implement the parsing logic, the result (success/failure) of which is sent back to the client. The QoS module displays the network performance parameters like delay and speed.

## SYSTEM DESIGN DESCRIPTION

The purpose of the design is to plan the solution of a problem specified by the system requirements. This phase is the first step in moving from problem to the solution domain. In other words, starting with what is needed design takes us to work how to satisfy the needs. The design of the system is perhaps the most critical factor affecting the quality of the software and has a major impact on the later phases, particularly testing and maintenance.

System design aims to identify the modules that should be in the system, the specifications of these modules and to interact with each other to produce the desired results. At the end of the system design all the major data structures, file formats, output formats as well as major modules in the system and their specifications are decided.

Client module creates two sockets. Through one socket data is sent and through the other command. User options are displayed to the user. The user option is sent to the server processing. Corresponding to the user options, the data packets are sent through data socket to the server. It receives the result from the server.

Depending on the user option Socket, Sip or Performance handlers will be called. For socket or sip handler the server receives the raw data packets. The pcap header is appended to the raw packets so that the Tshark can recognize it. The pcap file is then converted to text file using tshark commands. The text file is used for parsing. If the user option is performance, performance function will be called in which speed and delay of the network are calculated.

## LAYOUT OF PROTOCOL ANALYSIS SYSTEM

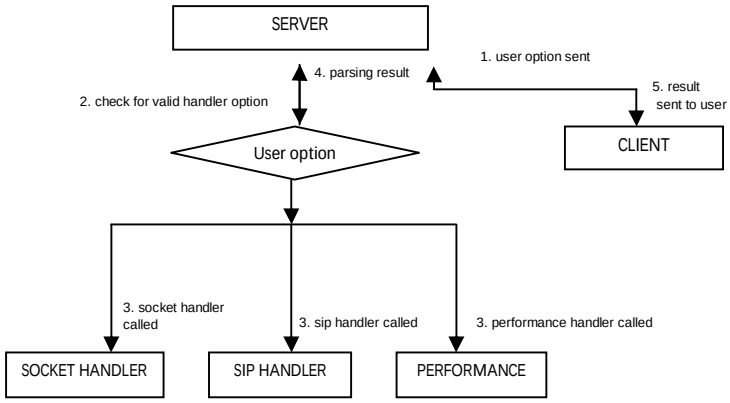


Fig 1: Layout of protocol analysis system

## GOALS

- Capture live packet data from a network interface.
- Display packets with very detailed protocol information.
- Open and Save packet data captured.
- Captured network data can be browsed via a GUI, or via the terminal (command line) version of the utility, tshark.
- Filter packets on many criteria.
- Search for packets on many criteria.
- Hundreds of protocols can be dissected.

## ARCHITECTURAL STRATEGIES

Here we will discuss about the architecture of Network Protocol Analysis System which includes the following The modules of the Network Protocol Analysis system, Interfaces, Context level Data flow diagram.

The Network Protocol Analysis System Module 1: Client Module 2: Server

Client module creates two sockets. Through one socket data is sent and through the other command. User options are displayed to the user. The user option is sent to the server processing. Corresponding to the user options, the data packets are sent through data socket to the server. It receives the result from the server.

Depending on the user option Socket, Sip or Performance handlers will be called. For socket or sip handler the server receives the raw data packets. The pcap header is appended to the raw packets so that the Tshark can recognize it. The pcap file is then converted to text file using tshark commands. The text file is used for parsing. If the user option is performance, performance function will be called in which speed and delay of the network are calculated

## DATA FLOW DIAGRAMS

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. A data

flow diagram can also be used for the visualization of data processing (structured design). It is common practice for a designer to draw a context-level DFD first which shows the interaction between the system and outside entities. DFD's show the flow of data from external entities into the system, how the data moves from one process to another, as well as its logical storage. This context-level DFD is then "exploded" to show more detail of the system being modeled. There are only four symbols:

- Squares representing external entities, which are sources or destinations of data.
- Rounded rectangles representing processes, which take data as input, do something to it, and output it.
- Arrows representing the data flows, which can either, be electronic data or physical items.

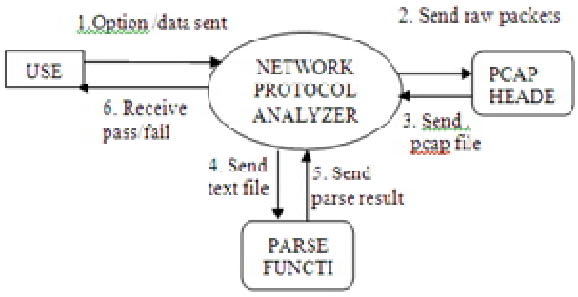


Fig 2: Data flow diagram for proposed architecture

**Context Diagram:** Context diagram of Network Protocol Analysis System contains following information.

- It contains one process called Network Protocol Analyzer which accepts data, performs operations on them and sends back the result to the user.
- It contains one external entity that is user which selects an option and sends the data to the server.
- It contains two processes, one Pcap Header Function and Parse Function (one for each option). Pcap Header Function appends pcap header to raw data packets. Parse function applies parsing operations on the text file sent by the server.

## CONCLUSION

The study is limited only to the wired networks as Wireshark captures packets through the wired medium only. Since the manual work in analysis is reduced here, this work can be used in real time systems where a large number of packets have to be analyzed.Since wireshark should be invoked manually and even network packet analysis is manual user finds it difficult for analysis. So our project is aimed to eliminate the manual effort where the developers or test engineers analyze the network packets manually. Here we are automating the entire process right from capturing the network packets till the analysis of it.In this paper a module can save project cost as well as the duration to a major extent when integrated to the software development life cycle (SDLC). The logic can always be extended depending upon the project requirements. Along with analyzing the packets, we are also analyzing the performance of the network.

**REFERENCES:**

[1] W. Cui, J. Kannan, and H. J.Wang. Discoverer: Automatic Protocol Description Generation from Network Traces. USENIX Security Symposium, Boston, MA,August 2007.

[2] Automatic Network Protocol Analysis Gilbert Wondracekx, Paolo Milani Comparettiz, Christopher Kruegel, and Engin Kirda

[3] DiCAP: Distributed Packet Capturing Architecture for High-Speed Network Links Cristian Morariu, Burkhard Still.

[4] Endace Measurement Systems. Available at monitorin-cards/ (Aug 15, 2007).

[5] F. Risso, L. Degioanni, "An Architecture for High Performance Network Analysis," in Proc. eh IEEE Symposium on Computers and Communications (ISCC 2001), (Hammamet, Tunisia, July 2001).

[ 6] Behrouz A.Forouzan, TCP/IP Protocol Suite [M], Third Edition. Beijing: Tsinghua University, 2006.

[7] B. Claise, S. Bryant, G. Sadasivan, S. Leinen, T. Dietz, and B. H.Trammell, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," RFC 5101 (Proposed Standard), Jan. 2008.

[8] A Language for Advanced Protocol Analysis in Automotive Networks by Tim Reichert, Edmund Klaus, Wolfgang Schoch, Ansgar Meroth, Dominikus Herzberg, ICSE'08, May 10–18, 2008, Leipzig, Germany.

[9] A. Pretschner, M. Broy, I. H. Kruger, and T. Stauner.Software engineering for automotive systems: Aroadmap. In *FOSE '07: 2007 Future of SoftwareEngineering*, pages 55–71, Washington, DC, USA,2007. IEEE Computer Society.

[10] J. Botaschanjan, L. Kof, C. K¨uhnel, and M. Spichkova.Towards verified automotive software. In SEAS '05: Proceedings of the second international workshop onSoftware engineering for automotive systems, pages1–6, New York, NY, USA, 2005. ACM Press.

[11]Distributed Network Analysis Using TOPAS and Wireshark Gerhard M¨unz, Georg Carle Computer Networks and Internet Wilhelm Schickard Institute for Computer Science, University of Tuebingen, Germany

[12]Sindhu Kakuru Electrical Engineering Department San Jose State University San Jose CA, 95112978-1-61284-486-2/111$26.00 ©2011 IEEE