An Insight in to Network Traffic Analysis using Packet Sniffer

Jhilam Biswas
8th semester, Department of Electronics and Communication, Manipal Institute of Technology, Manipal, Karnataka, India

Ashutosh
8th semester, Department of Electronics and Communication, Manipal Institute of Technology, Manipal, Karnataka, India

ABSTRACT
Slowdown in the network performance can cause serious concern to network analysts, leading to loss in resources. Such cases are not easy to deal with, due to the lack of time and resources available. Lack of awareness about appropriate tools which detect the attacks or not knowing exactly why a loss in network performance is occurring are some other factors. Connectivity loss or shutting down of terminals within the network for unknown reasons are among the other problems. Mostly, the cause of these problems cannot be detected accurately and is concluded due to poor network architecture, such as inefficiently configured broadcast storms, spanning-tree, usage of unsuitable routing protocols within the network domain, redundant links etc. However, sometimes the cause could be due to attacks by unknown third parties that try to put the web server out-of-service through means of a DoS (Denial of Service) attack, sending traffic with a poisoned ARP in an attempt to discover hosts to infect, or by simply infecting ports with malware to form part of an alien network or botnet. In all these cases, knowing the source of the attack is the first step towards taking appropriate action and achieving correct protection. That is when packet sniffers can be extremely useful to detect, analyze and map traffic. Such packet sniffers identify threats to the network and limit their harmful consequences.

General Terms
Packet sniffers, Wireshark, Data capturing techniques, LAN attacks, graphical usage of Wireshark

Keywords
Packet sniffing tools, Wireshark, LAN attacks

1. INTRODUCTION
Many sophisticated systems such as the MARS (*Monitoring, Analysis and Response System*) by Cisco or IDS/IPS (Intrusion Detection System/Internet Protocol System) help in identifying potential threats to a network. However, these solutions are not cost effective to any organization/company. An alternative solution is to use a packet sniffing software which gives a detailed examination of the network. Examples of such packet sniffers are Wireshark, Capsa Network Analyzer, SkyGrabber, Xplico, Microsoft Network Monitor etc.

The aim of this paper is to make network administrators and technicians aware of the advantages of monitoring the network with a packet sniffer using the free and open source tool Wireshark. All packet sniffing tools work similar to Wireshark, hence Wireshark was chosen for experimental purposes in this paper. Other packet sniffing tools can be used too. The paper also offers practical examples of common attacks to local area networks (LANs) and how Wireshark can be used to detect these attacks. Further, this paper is divided into sections that demonstrate different real attacks to local networks, such as *ARP Spoof, DHCP Flooding, DNS Spoof, DDoS Attacks*, *Port Monitoring*, etc. Wireshark is used as the main support tool to help detect and analyze the problems generated by these attacks. At the same time, different solutions to resolve each of these attacks are proposed.

2. AN OVERVIEW OF WIRESHARK
Wireshark is a free and open-source protocol/ packet tracer. It runs on both Windows and Unix platforms. Formerly known as Ethereal, its prime objective is network troubleshooting, analysis, and networking research. Wireshark facilitates a wide range of filters that supports over 1200 protocols (version 1.10.7), all with a simple front-end that enables one to break down the captured packets on the basis of different layers of the OSI (Open Systems Interconnection) model. The Wireshark engine can decipher the structure of different networking protocols. This feature proves extremely beneficial for users to view the fields of each one of the headers and layers of the packets being analyzed [1]. Thus Wireshark provides a wide range of options to network engineers when performing certain traffic auditing tasks. Many tools, such as Snort, OSSIM and a number of IDS/IPS serve to warn users of some of the network related problems and attacks. However, when one needs to analyze traffic in depth or monitor a network, when time is of prime importance, these tools lack the flexibility that a protocol analyzer such as Wireshark easily offers.

3. FIRST STEP OF NETWORK ANALYSIS: DISCUSSION ON WHERE TO CAPTURE THE DATA
The very first step in auditing networks is to define where to analyze the traffic. Taking a common scenario for analysis, the following assumptions were made. There is a switched network made up of a number of switches, several terminals and a file server. Network performance has dropped, however the cause is unknown. There is no IDS (Intrusion Detection System) that can alarm or inform about attacks or network malfunction. Also, it is known that there are no problems with the transfer rate of the file server to LAN (*Local Area Network*) terminals [3]. Furthermore, network equipment does not have Netflow protocols to analyze traffic remotely. Wireshark was chosen to analyze the above scenario. The first doubt which arises is where to install Wireshark. It would seem logical to install Wireshark on the file server itself to analyze the traffic that flows through this network segment. However, there could be situations in which there is no access to the server physically or quite simply for security reasons. Thus, Wireshark cannot be installed there. Some alternatives are provided in the following paragraphs that enable to

capture traffic without having to install Wireshark on the server.

3.1 Using a Hub

If a user connects a node where Wireshark is installed to one of the switch ports, he will only see the packets that occur between the switch and his terminal, however this is not desired for traffic analysis. The switch divides the network into segments creating separate collision domains for each port. Unlike a collision domain, in a broadcast domain, the packets are sent to all ports (belonging to the same Virtual LAN -VLAN). This objective is met using a hub, as illustrated in Fig 1, connecting the hub- a broadcast device to the same network segment on the user's server. Now all traffic between the switch and the server can be analyzed on the user's terminal, where Wireshark is installed.

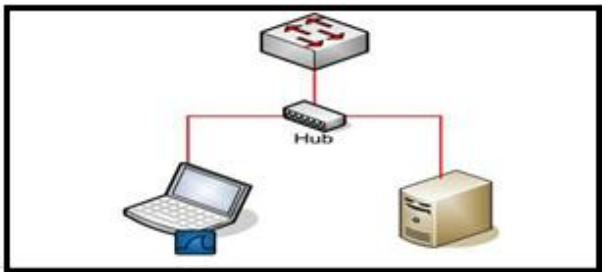

Fig 1: Capture Mode 1- A hub connectivity between the server and the user's terminal where Wireshark is installed.

3.2 Port Mirroring or VACL (VLAN-Based ACLS)

As long as the user has access to the switch, this is the most convenient method to capture network traffic. This way of working is known as Services and Protocols for Advanced Networks (SPAN). It enables the user to duplicate the traffic between one or more switch ports and mirror it to the port that he wants, as shown in Fig 2. In this method, the port configured as mirroring has to be as fast as the port(s) to be monitored to avoid packet loss, while data capturing. This method is used by many administrators to install IDS or other analysis tools [5]. The advantage which Port Mirroring has, that it allows better filtering algorithms when specifying the traffic that he wants to analyze. When configuring Port Mirroring, it is possible to redirect traffic from one port or VLAN to another. Also, with VACL it is possible to specify ACLs to select the type of traffic that the user is interested in.

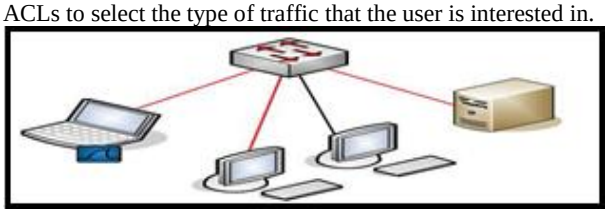

Fig 2: Capture Mode 2- Port Mirroring connection setup, enables to duplicate traffic between various switch ports

3.3 Bridge Mode

If the user is not able to access the switch, he can use a machine with two network cards to position himself between the switch and the server, as illustrated in Fig 3. This is a MitM (*Man in the Middle*), at the physical level, where he has a passive access to all traffic throughput. There are several ways in which the user can configure his PC in this mode. More so, it is easy to install and configure *bridge-utils* (bridge packet utilities for Linux). This is necessary to create a bridge-type interface and thereafter add the physical interfaces that form part of this bridge. Lastly, users can activate the interface and execute Wireshark. The disadvantage of this capture method is the loss of data streams during installation.

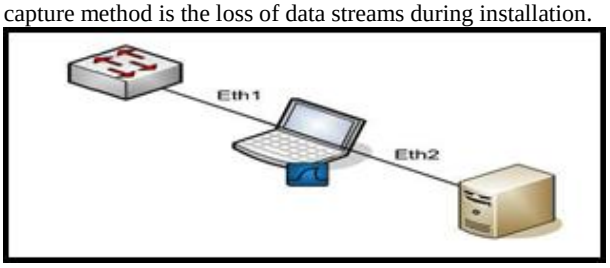

Fig 3: Capture Mode 3- A Bridge setup (Man in the middle) where he has access to the traffic throughput.

3.4 Arp Spoof

On certain occasions, if network administrators cannot use the previous methods, they can use the ARP Spoofing technique. This is rather an offensive method and is only useful in non-critical environments where there is a need to capture traffic between various machines. What is achieved from this method, is that the machine which the user wants to monitor sends all segments via his PC where he has Wireshark executing. The process is performed by infecting the cache of the machine with a false IP/MAC association [7]. Some switches have functions available that enable to detect this process (*Dynamic ARP Inspection* and *DHCP Snooping*3), so it is important to deactivate this function in the network devices so that the port does not go into shutdown mode.

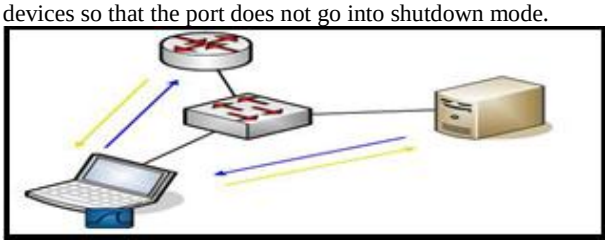

Fig 4: Capture Mode 4- ARP Spoofing connection setup and flow of data within the connectivity.

3.5 Remote Packet Capture

Besides the above methods, there are several options for capturing data remotely. One of them is by means of a RPCAP (*Remote Packet Capture System*).). In this technique, in addition to a client program from which the data will be recovered and viewed; in this case, Wireshark, it is necessary to execute a server program (rpcapd) along with the required libraries on the machine. As like ARP Spoofing, this method is appropriate for non-critical environments where the user can install the software in the machine whose traffic he wishes to analyze, with the associated stability and performance risks. Users can specify the listening port and other options such as authentication, authorized client lists to connect to the server.

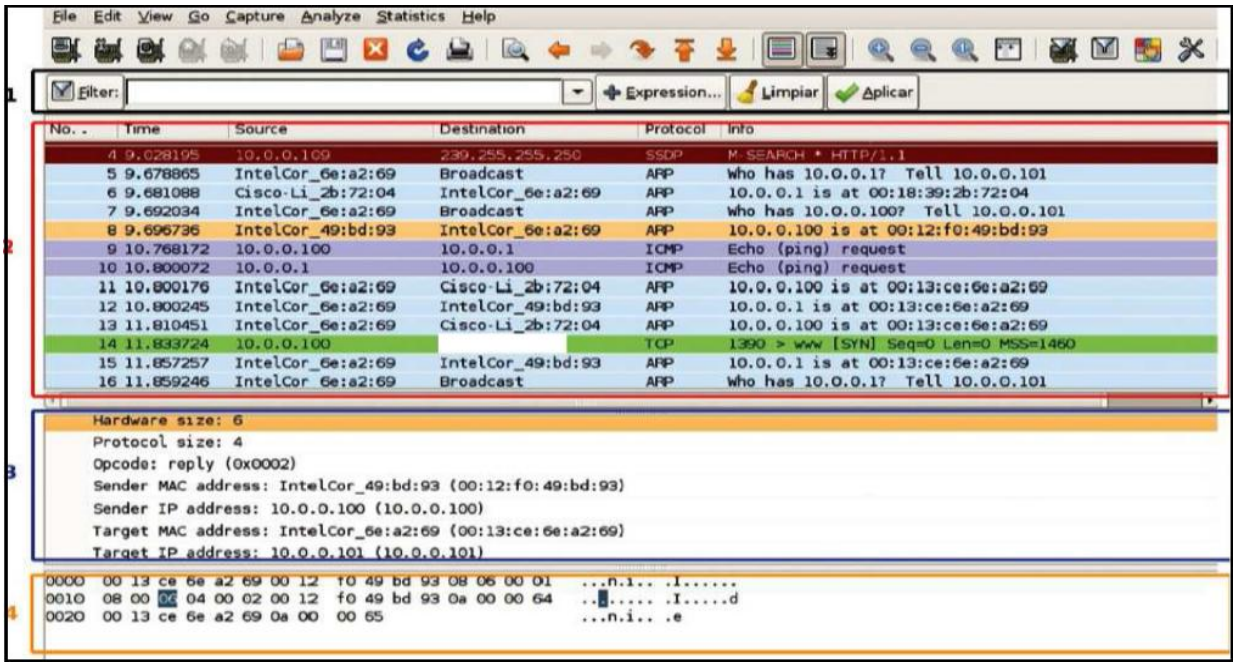

Fig 5: Wireshark interface, data capture

4. EXPERIMENTAL METHODOLOGY

Wireshark was first installed on the system depending on the operating system. Next going to the settings of the web browser, the homepage was set to blank. Wireshark was opened on the system and data capturing was started. A snapshot of the captured data using Wireshark is shown in Fig 5.The following offers a brief description of the various areas that are of interest, which Wireshark displays once data capture starts (Figure 5- Wireshark interface, data capture): *Section 1* is the area where filters are defined. These filters enable to view only those data packets or protocols that are of interest to the user. *Section 2* corresponds to a list to view all packets being captured in real time. This data (protocol type, number sequence, flags, time stamps, ports, etc.) can be interpreted to identify the problem without having to perform a detailed monitoring. *Section 3* enables to classify and navigate through the various layers, each header of the packets selected in *section 2.* Lastly, *Section 4* represents, the packet which was captured by the user's network card, in hexadecimal format.

5. LOCAL AREA NETWORK (LAN) ATTACKS

A client system in a LAN interacts not only with the other client systems within the same LAN but also with the clients machines of other LANs. In this process of communication, the client system is prone to various attacks/ threats. In order to effectively safeguard a LAN from various threats, IT managers need to understand the origins of these attacks, the methods by which they are detected and the potential risk they present to network resources. The most common types of attack along with their mitigation techniques are discussed as follows:

5.1 ARP Spoofing With a Practical Example

Besides being a method of capturing network traffic in specific circumstances, Arp Spoofing is normally used by attackers to intervene between one or more machines with the aim of intercepting and modifying stream of data packets. This is a rather intrusive method of data capturing and is reflected in Fig 5.As depicted in the figure, some abnormality is occurring due to the large quantity of ARP traffic that is being received. A closer look at the behavior of the protocol shows that the server is being attacked. From Fig 5, it is seen that in packet number 5, the machine with IP 10.0.0.101, and a MAC of IntelCor_6e:a2:69, has sent an ARP request to the broadcast address asking for the MAC of the IP 10.0.0.1 (user's network gateway).The router immediately responds with an ARP reply indicating the MAC address. Then the same IP repeats the process and requests the MAC of the IP 10.0.0.100 (file server's IP) using another broadcast diffusion. The server responds with its MAC address (IntelCor_49: bd: 9). Normal functioning occurs up to this point. The two devices - a machine on the LAN (10.0.0.101), that has the MAC server and a router, can now share Ethernet traffic. The problem arises with packet 11, when this machine repeatedly sends to the user's server and the router false ARP reply packets, associating the IP of both with its own MAC (IntelCor_6e:a2:69). This way, all traffic transmitted between the LAN gateway and the server goes through the attacking machine [13]. The basic explanation for the above scenario is as follows: The ARP protocol is a layer 3 protocol used to translate IP addresses to physical network card addresses or MAC addresses. When a device tries to access a network resource, it first sends requests to other devices asking for the MAC address associated with the IP it wants to reach. The caller will keep the IP - MAC association in its ARP cache, to speed up new connections to the same IP address. The attack comes into picture when a particular machine asks the other machines in the LAN to find the MAC address associated with an IP address. The attacker machine will answer to the

caller with fake packets saying that the IP address is associated to its own MAC address and in this way, will build a real IP - MAC association with the attacked host. This attack is referred as ARP poisoning or ARP spoofing. This attack is possible only if the pirate and the victims are within the same broadcast domain.

5.2 PORT FLOODING

This attack is similar to the previous one, but easier to detect. In this attack, multiple false segments are sent to a switch port in order to saturate the switch assignment table. A switch has an internal memory space called the CAM (*Content-Addressable Memory*), where ports are assigned corresponding to MAC addresses. When a segment arrives at a port, the CAM adds an entry to the table specifying the MAC of the machine that sent the segment along with the port in which it is located. Thus, whenever a switch receives a segment directed to a machine it knows from what port it must send it. If the destination of the segment is unknown, because the associated entry to this machine has expired, the switch copies the segment and sends it to all ports of the same VLAN except to the port that received it. This way, all machines connected to the switch receive this segment and only the corresponding machine with a MAC that corresponds with the segment destination MAC replies. The switch then adds this entry in the CAM table with the new MAC/port association [9]. With this done, the switch need not to flood all ports with future packets destined to this machine. However, if hundreds of segments are sent falsifying the source MAC of the destination machine to fill up the CAM table, the behavior of the switch depends on the manufacturer. Low-end switches do not contain sophisticated CAM tables; as a result of which if a machine fills the table with the maximum number of entries (MAC/ port associations), all VLANs are infected.

5.2.1 Proposed Solutions

Detecting this attack using a protocol analyzer is easy. *Macof*, a member used in the Dsniff suit toolset, sends TCP segments without considering the protocol specifications (and hence it shows *'Malformed Packet'* as in Fig-6). *Macof* is used to flood the switch on a LAN with MAC addresses. Since the switch regulates the flow of data between its ports, it actively monitors (cache), the MAC address on each port and passes data only to the targeted machine. Wireshark can be helpful for the detection of these attacks. As already mentioned above, this attack takes place when packet flooding occurs at all ports and the CAM table is full, it is also possible to let Wireshark eavesdrop on any switch ports and monitor it for malformed packets that are being received. High-end switches can be configured with specific parameters to reduce this type of attack. Some of the parameters that can be configured are: he number of MAC by port (*port security*), the flooding level of packets allowed by VLAN and MAC (*Unicast Flooding Protection*), and the expiry time of the MAC in the CAM table (*ageing time)*.

346	13.300620	39.39.218.123	67.129.128.67	TCP	[Malformed Packet]
347	13.301344	65.30.29.120	192.164.170.9	TCP	[Malformed Packet]
348	13.302264	82.8.242.103	225.173.109.6	TCP	[Malformed Packet]
349	13.303184	88.125.244.10	81.219.96.39	TCP	[Malformed Packet]
350	13.305176	92.236.234.36	103.223.24.56	TCP	[Malformed Packet]
351	13.306176	40.255.13.13	57.31.185.74	TCP	[Malformed Packet]

Fig 6: Packets captured, generated by *Macof*

5.3 DDoS ATTACKS

In this attack, a large number of remotely controlled systems attack a single target, causing denial of service for users of the targeted system. The flood of incoming messages to the target system forces it to shut down, thereby denying service to the system to licit users. Sometimes the traffic is enough to shut the site down completely. This type of attack is called distributed denial-of-service (*DDoS*) attack. Referring to Fig 7, it represents an example of distributed denial-of-service (*DDoS*) attack as soon as Wireshark starts the capture process. As depicted, an Apache is installed on machine 10.0.0.101. A large number of TCP segments with the SYN flag activated from the same IP that do not receive a response (ACK) from the web service are generated. In Wireshark, the packet sequence can be graphically seen by selecting from the menu *Statistics ->Flow Graph*. This tool enables to track the behavior of TCP connections as shown in Fig 7. It illustrates, using arrows, the source and target of each packet, highlighting the active flags that interfere in the connection flow [4]. It can be easily noticed that there is a short period of time when a number of connection attempts are made by the IP 10.0.0.200 to port 80 of machine 10.0.0.101. This is a rather suspicious scenario. The server has tried to resolve the MAC of the client many times (for example in packet 7852), but when no acknowledgement is received, it cannot send an ACK-SYN to the same machine to continue the three-step (handshake) connection. Thus the TCP/IP stack of the server has to wait for a set time for each connection. During this idle time more packets keep arriving that trigger new connections. For each new connection, a structure in memory called the TCB (*Transmission Control Block*) is created and used by the TCP/IP stack of the operating system to identify each connection.

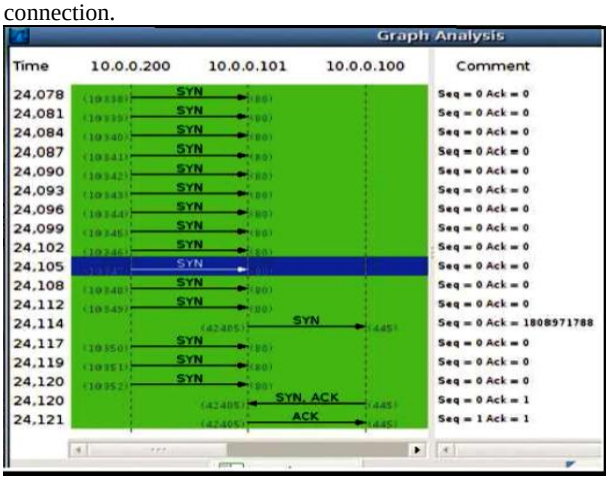

Fig 7: Data Flow Capture

5.3.1 Proposed Solutions:

Besides the DDoS attacks mentioned above, there are various other types of DDoS attacks like *IP unreachable attack*,

ICMP transit attacks, *Direct Attacks*, *TTL expiry attack*, *Reflection Attacks* etc. They are very difficult to analyze due to the high volume of traffic involved. Many devices that enable to stop these attacks are expensive. Contacting the ISP seems the most appropriate action to monitor such attacks. However, when the magnitude of attack is not very excessive, an appropriate configuration of the operating system and kernel services could help to counteract the attack. For example, there are various Linux kernel parameters that enable to modify the traffic behavior which are very useful to protect the server from these attacks.

5.4 DHCP AND DNS SPOOF

An interesting feature of the DHCP protocol is that it does not have authentication mechanisms which enable to verify the source of the packets during the exchange of configuration parameters. In this kind of attack, the hacker can access a network traffic by falsifying responses that is sent by an authentic DHCP server. It allows a hacker to monitor and sniff all the traffic. All the data packets in the LAN cross the attacker's interface which makes it vulnerable to packet capturing. Giving a brief description how DHCP Spoofing occurs, when a client sends a DHCP request on the network, this request is broadcasted and all hosts on the LAN receive it. Only the DHCP server is aware of the actual meaning of this request. In the normal functioning, the real DHCP server replies to the client with its IP address, Subnet mask and Default Gateway. The attacker in the network simulates a DHCP server on its host PC. With this, the attacker can reply to DHCP request before the actual DHCP server. It configures the client host with not only the IP address of that subnet but in addition, it also gives the host a false Default Gateway address and sometimes even a false DNS server address. In this manner, the attacker redirects all the data packets of the client host to himself [7]. The hacker later on forwards the frames from client host to real destinations in order to make the client communication possible. The essence of this attack is that the client never knows that his communication is always going through the attacker PC and that the attacker is actually sniffing all the frames.

5.4.1 Proposed Solutions:

Wireshark warns the user of any abnormal use of DHCP protocol. Yet another symptom of this attack is the generation of errors on the machines due to duplicated IPs. Tools such as Yersinia (layer 2 hacking tool), Ettercap can be used for analyzing and monitoring networks. Ettercap is the mnemonic for Ethernet Capture. It supports excellent features like analyzing a network using protocol dissection, sniffing of live data connections etc. Other alternative is to configure a DHCP server in the attacking machine, such as dhcpd3, to create MitM (*Man in the Middle*) using false DHCP responses [8]. Tools like those mentioned above are an aid to detect DHCP and DNS attacks on a local network.

6. GRAPHICAL USAGE OF WIRESHARK

Wireshark like previously seen, is a versatile tool that offers a wide range of options to examine the performance of a network graphically based on a multiple parameters. Graphical representation of statistics enable to study the network in a lucid manner. Two graphs in Wireshark prove to be extremely useful for traffic analysis. One of them is the *Stevens* graph and the other one is *I/O* graph.

A TCP session can be tracked graphically to study the relationship between time and sequence number in a data stream. This graph is called a *Time Sequence Graph* (*Steven*) and can be found under the tab *Statistics -> TCP Stream Graph* [12]. Under ideal conditions, the graph represents a line growing over time indicating efficient performance the TCP connection [13]. However, in some scenarios there will be gaps and bumps that intervene the continuity of the line. This occurs due to a resend of data as result of lost ACK duplications, segments retransmissions due to packet loss, expired timeouts etc. This graph can be extremely beneficial to detect irregularities in the behavior of TCP data flow. Another graph that gives valuable information about network traffic is the one on input/output. It can be found in *Statistics -> I/O Graph*. Users can select various filters based on which they want to filter the data. It gives the graphical representation of various filters in different colors. A concise data on the percentage of use of each protocol captured can be obtained under *Statistics -> Protocol Hierarchy*, where the hierarchy and precedence of each protocol, sent/received packets and their size are shown.

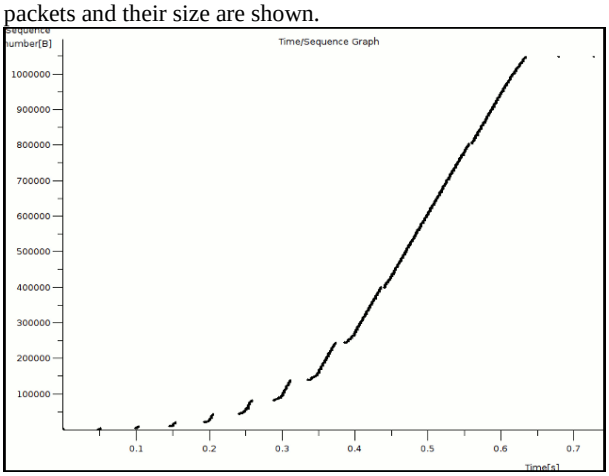

Fig 8: Stevens (Time/ Sequence)Graph in Wireshark

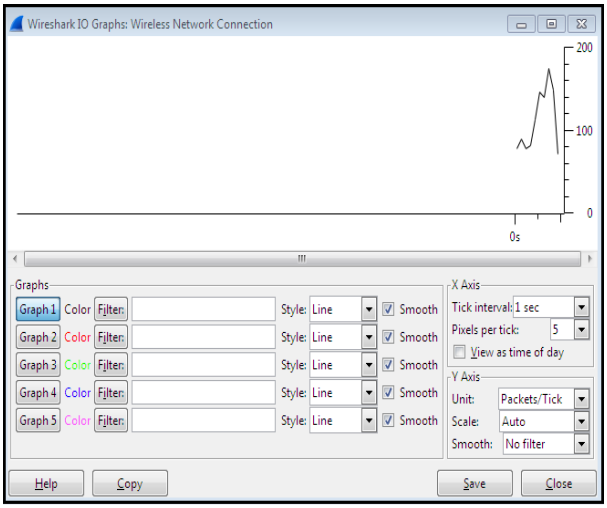

Fig 9: Wire shark IO (Input Output Graph) after data capturing.

7. FUTURE WORK

Wireshark as described above proves to be extremely advantageous for analyzing and monitoring a network and also to detect various threats that can slow down a network. However, Wireshark cannot foresee the future trends of packet streams in a network. It cannot warn the users well in advance that an attack is likely to take place. It will be a boon to the IT industry and a massive prevention to the loss of resources, if it is known well in advance that a bug is likely to

affect the network. Network analysts must look into methods so that Wireshark can predict the flow of data streams.

8. CONCLUSION

Wireshark is one of the many packet sniffing software that are available today. It was chosen as the tool to discuss detailed analysis of network traffic due to its advantages mentioned under the paragraph 'AN OVERVIEW OF WIRESHARK'. Just to summarize, Wireshark is a useful tool that offers countless functions that help to analyze multiple network problems; those caused by poor network configuration or device failures and also a variety of external and internal LAN attacks. The first step in resolving network problems consists of analyzing where the problem has occurred in a network and what has led to performance loss. Network administrators should be aware of the importance of using this type of tool, as it is a key utility to detect the source of network problems that would otherwise take a great deal of resources and time to discover.

Packet sniffers like Wireshark can be a boon for network monitoring, but sometimes it is necessary to prevent packet sniffers from collecting sensitive information like passwords, in order to maintain security in the IT industry. Two important actions that can protect users from packet sniffers and other eavesdropping attacks are as follows: First one is data encryption (hiding). Data encryption helps protect sensitive data and passwords while in transit, packet sniffers merely prove useless, if data is encrypted. Encryption can be implemented in a number of ways: SSL-Secure socket layer (HTTPS- where S stands for SSL) connections to Web servers, encrypted SSL or TLS (Transport Layer Security) connections to mail servers, or other application-specific techniques. Alternatively, a virtual private network (VPN) can be used to encrypt entire communications links, regardless of protocol. Second one being use of a switched network. In this case, a packet sniffer will only be able to capture data stream packets arriving on its own local switch port. If each system is assigned to an individual switch port, there simply won't be any packets for the packet sniffer to intercept.

Network traffic analysis is of prime importance in any organization. In such a highly IT driven generation, if the network is slow/ down even for a few minutes, it can incur huge losses to the company. Hence it is important to analyze networks and be aware of the network threats beforehand. Network analyzers like Wireshark are cost effective and a very useful tool for in depth network analysis. This is a review paper that describes several ways of analyzing network traffic with Wireshark, with the help of practical examples. The paper also discusses some common attacks that can threaten local area networks and appropriate measures to mitigate the impact of these attacks on the network. This paper can be extremely beneficial to a novice who wishes to have a detailed monitoring of his network. Starting with a brief description about Wireshark, then moving on to ways to capture data under different circumstances and finally discussing various attacks that can affect a network (with their proposed solutions), this paper has it all. Lastly, the paper describes how to use graphs in Wireshark to interpret the benefits and efficiency of the network. The paper also talks about some of the security issues related to Wireshark and suggests solutions for the same. Wireshark, apart from being one of the best protocol analyzers today, is an excellent source of knowledge for any IT professional, network analyst or communications enthusiast.

9. ACKNOWLEDGMENTS

Sincere thanks to the professors of the Department of Electronics and Communication and the department of Information and Communication Technology for providing excellent laboratory facilities to carry out the experimental methodology using Wireshark listed under the paragraph 'EXPERIMENTAL METHODOLOGY'

10. REFERENCES

[1] Wireshark Documentation: http://www.wireshark.org/docs/wsug_html_chunked/index.html

[2] Stolze M, Pawlitzek R and Hild S (2009a) Task Support for Network Security Monitoring. In ACM CHI Workshop on System Administrators Are Users, Too: Designing Workspaces for Managing Internet-Scale Systems.

[3] Madsen, P., Koga, Y., Takahashi, K.: Federated identity management for protecting users from ID theft Proceedings of the 2005 workshop on Digital identity management Fairfax, VA, USA (2010) 77-83

[4] Gouda, M.G., Liu, A.X., Leung, L.M., Alam, M.A.: Single Password, Multiple Accounts. Proceedings of 3rd Applied Cryptography and Network Security Conference (industry track), New York City, New York (2008)

[5] Riley, S.: Password Security: What Users Know and What They Actually Do. Usability News, Vol. 2006. Software Usability Research Laboratory, Department of Psychology, Wichita State University, Wichita (2009)

[6] 2010 18th IEEE Symposium on High Performance Interconnects Innovating in Your Network with OpenFlow: A Hands-on Tutorial

[7] 2011 Fourth International Joint Conference on Computational Sciences and Optimization: Application Design of Data Packet Capturing Based on Sharpcap

[8] The 6th International Conference on Computer Science & Education (ICCSE 2011) August 3-5, 2011. SuperStar Virgo, Singapore: Application of Mini Case Study in Computer Networks

[9] Andrew S. Tanenbaum, "Computer Networks." 4th ed. Beijing: Tsinghua University Press, 2004, pp.41.

[10] IEEE 2008 publication: Bottleneck Analysis of Traffic Monitoring using Wireshark

[11] I. Kim, J. Moon, H. Y. Yeom, "Timer-Based Interrupt Mitigation for High Performance Packet Processing," in Proc. 5th International Conference on High-Performance Computing, (Asia-Pacific Region, 2011).

[12] J. Cleary, S. Donnelly, I. Graham, "Design Principles for Accurate Passive Measurement in Networks," in Proc. PAM2000 Passive and Active Measurement Workshop (Apr. 2000).

[13] Traffic Analysis with Wireshark, February 2011, Author: Borja Merino Febrer, The National Communications Technology Institute (Instituto Nacional de Tecnologías de la Comunicación - INTECO) Manuel Belda, from Valencia's Computer Security Incident Response Team (CSIRT-cv) and Eduardo Carozo Blumsztein from the ANTEL CSIRT of Uruguay.

[14] Di Guangqun, Hu Guijang. "Development and implementation of packet sniffer", CNKI: SUN: WJSJ.0.2009-21- 082.pp.1-5, 2009