Experiment-10

Aim: Study of Wireshark and understand its functionality

Prerequisite: Nil

Outcome: To impart knowledge of Application Layer Protocol

Theory:

- Wireshark stands as a robust network protocol analyzer, empowering users to capture, analyze, and scrutinize data packets circulating across a computer network. This tool offers an intricate view of the traffic within a network, aiding in the comprehension, troubleshooting, identification of security flaws, and enhancement of network performance. Supporting a wide array of network protocols, Wireshark serves as a versatile utility suitable for novices and seasoned network professionals alike.
- Its functionality involves intercepting and logging data packets in transit through a network interface. Once recorded, Wireshark organizes this information in an easy-to-navigate graphical interface, enabling users to filter and dissect packets based on multiple criteria such as source, destination, protocol type, and more. This level of detail allows users to analyze individual packet content, facilitating problem diagnosis, anomaly detection, and a comprehensive understanding of network behavior. Wireshark remains an indispensable tool for network administrators, security professionals, and developers engaged in network-related applications.

Procedure:

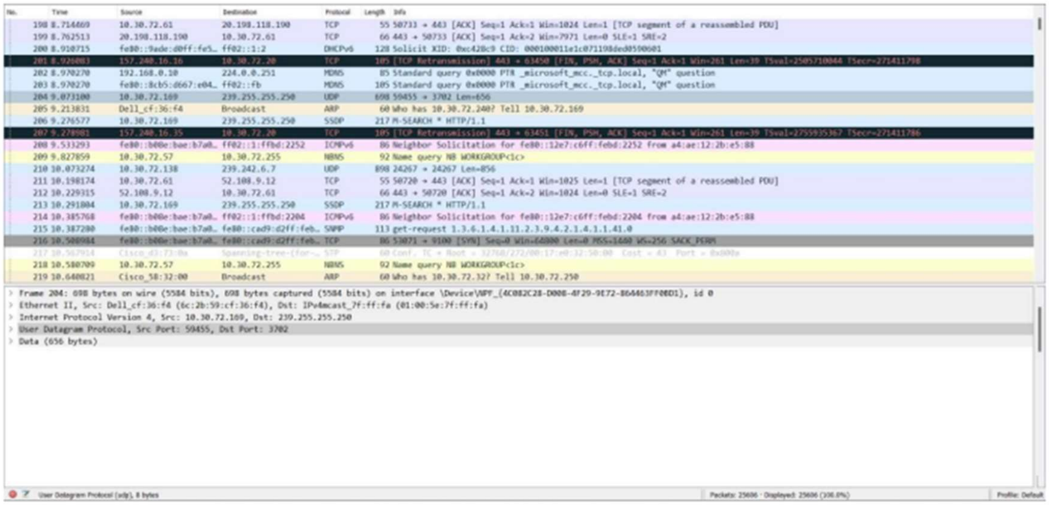
1. Install the Wireshark and integrate it with network simulator.
2. Analyze the traffic using Wireshark.

Steps:

1. Install the Wireshark and look at the different network interfaces on your computer.
2. Identify and select the network interface that you want to monitor. This could be a wireless connection, Ethernet, or any other network interface your computer uses to send and receive data. Click the “Start Capturing Packets” button (the shark fin icon) next to the chosen interface to begin capturing live traffic.
3. Allow Wireshark to capture traffic for a sufficient period to gather the data you need for analysis. This might be a few seconds, minutes, or more, depending on your requirements. Click the red “Stop” button to halt the packet capture when you feel you have enough data.
4. Use the “Display Filter” bar to enter specific protocols, IPs, or criteria to isolate specific types of traffic. For example, typing “http” will show only HTTP traffic. Click on a packet to view detailed information. The middle pane shows the packet details, while the bottom pane shows the raw data.

5. Use the color coding as a guide to quickly identify the types of traffic - each color represents a different protocol or traffic type. Right-click a packet and select “Follow” -> “TCP Stream” (or UDP, etc.) to see the entire conversation between endpoints.
6. Examine the packet details pane for more information on each layer of network protocols within a selected packet. If needed, you can save your packet capture by going to “File” -> “Save As” to analyze it later or use it as documentation.
7. Explore the “Statistics” menu for graphical views, endpoint summaries, protocol hierarchy, and other advanced analyses. You can export specific packets or conversations by selecting them and then choosing the appropriate option under the “File” menu.

Output:



```
No.     Time       Source                 Destination            Protocol  Length Info
    198 8.714469   10.30.72.61            20.198.118.190         TCP         55 50733 → 443 [ACK] Seq=1 Ack=1 Win=1024 Len=1 [TCP segment of a reassembled PDU]
    199 8.762513   20.198.118.190         10.30.72.61            TCP         66 443 → 50733 [ACK] Seq=1 Ack=2 Win=7971 Len=0 SLE=1 SRE=2
    200 8.910715   fe80::9ade:d0ff:fe5…   ff02::1:2              DHCPv6     128 Solicit XID: 0xc428c9 CID: 000100011e1c07119 8ded0590601
    201 8.926083   157.240.16.16          10.30.72.20            TCP        105 [TCP Retransmission] 443 → 63450 [FIN, PSH, ACK] Seq=1 Ack=1 Win=261 Len=39 TSval=2505710044 TSecr=271411798
    202 8.970270   192.168.0.10           224.0.0.251            MDNS        85 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QM" question
    203 8.970270   fe80::8cb5:d667:e04…   ff02::fb               MDNS       105 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QM" question
    204 9.073100   10.30.72.169           239.255.255.250        UDP        698 59455 → 3702 Len=656
    205 9.213831   Dell_cf:36:f4          Broadcast              ARP         60 Who has 10.30.72.240? Tell 10.30.72.169
    206 9.276577   10.30.72.169           239.255.255.250        SSDP       217 M-SEARCH * HTTP/1.1
    207 9.278981   157.240.16.35          10.30.72.20            TCP        105 [TCP Retransmission] 443 → 63451 [FIN, PSH, ACK] Seq=1 Ack=1 Win=261 Len=39 TSval=2755935367 TSecr=271411786
    208 9.533293   fe80::b00e:bae:b7a0…   ff02::1:ffbd:2252      ICMPv6      86 Neighbor Solicitation for fe80::12e7:c6ff:febd:2252 from a4:ae:12:2b:e5:88
    209 9.827859   10.30.72.57            10.30.72.255           NBNS        92 Name query NB WORKGROUP<1c>
    210 10.073274  10.30.72.138           239.242.6.7            UDP        898 24267 → 24267 Len=856
    211 10.198174  10.30.72.61            52.108.9.12            TCP         55 50720 → 443 [ACK] Seq=1 Ack=1 Win=1025 Len=1 [TCP segment of a reassembled PDU]
    212 10.229315  52.108.9.12            10.30.72.61            TCP         66 443 → 50720 [ACK] Seq=1 Ack=2 Win=1024 Len=0 SLE=1 SRE=2
    213 10.291804  10.30.72.169           239.255.255.250        SSDP       217 M-SEARCH * HTTP/1.1
    214 10.385768  fe80::b00e:bae:b7a0…   ff02::1:ffbd:2204      ICMPv6      86 Neighbor Solicitation for fe80::12e7:c6ff:febd:2204 from a4:ae:12:2b:e5:88
    215 10.387280  fe80::b00e:bae:b7a0…   fe80::cad9:d2ff:feb…   SNMP       113 get-request 1.3.6.1.4.1.11.2.3.9.4.2.1.4.1.1.41.0
    216 10.500984  fe80::b00e:bae:b7a0…   fe80::cad9:d2ff:feb…   TCP         86 53071 → 9100 [SYN] Seq=0 Win=64800 Len=0 MSS=1440 WS=256 SACK_PERM
    217 10.567914  Cisco_d3:73:0a         Spanning-tree-(for-…   STP         60 Conf. TC + Root = 32768/272/00:17:e0:32:50:00  Cost = 43  Port = 0x800a
    218 10.580709  10.30.72.57            10.30.72.255           NBNS        92 Name query NB WORKGROUP<1c>
    219 10.640821  Cisco_58:32:00         Broadcast              ARP         60 Who has 10.30.72.32? Tell 10.30.72.250
```

```
> Frame 204: 698 bytes on wire (5584 bits), 698 bytes captured (5584 bits) on interface \Device\NPF_{4C082C28-D008-4F29-9E72-864463FF08D1}, id 0
> Ethernet II, Src: Dell_cf:36:f4 (6c:2b:59:cf:36:f4), Dst: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
> Internet Protocol Version 4, Src: 10.30.72.169, Dst: 239.255.255.250
> User Datagram Protocol, Src Port: 59455, Dst Port: 3702
> Data (656 bytes)
```

User Datagram Protocol (udp), 8 bytes | Packets: 25606 · Displayed: 25606 (100.0%) | Profile: Default

Observation & Learning

- In Wireshark presents a comprehensive array of functionalities designed to capture and analyze network traffic originating from various sources. Its real-time analysis capabilities empower users to swiftly identify and resolve network issues, ultimately streamlining the troubleshooting process. One of Wireshark's key strengths lies in its broad support for numerous protocols, enabling a thorough examination of network traffic across various communication standards.
- Moreover, Wireshark excels in packet decoding, transforming captured data packets into a human-readable format, offering valuable insights into the underlying protocols and their respective fields. This feature is essential for obtaining a clear understanding of the complexities within network communication. Additionally, the tool supports packet filtering, enabling users to refine their analysis by applying specific criteria. This functionality significantly enhances the precision of investigations by focusing on relevant traffic. Overall, Wireshark serves as a robust and indispensable tool for network professionals aiming to gain comprehensive visibility into their network traffic.

Conclusion

Wireshark is a crucial tool valuable for network administrators, security experts, and developers. Its primary function involves capturing, analyzing, and examining network traffic, delivering a multitude of advantages. This encompasses swift troubleshooting via real-time analysis and packet filtering. Wireshark's extensive protocol support provides a deep understanding of network behaviors. Moreover, it serves as a valuable resource for monitoring network performance and investigating security incidents. Its adaptability through scripting and dissectors further amplifies its usefulness for specialized tasks, while its availability across multiple platforms ensures accessibility for users operating various systems. In summary, Wireshark is an indispensable asset for those aiming to comprehensively comprehend and efficiently manage network traffic.