**Graduate Program in Information Assurance Engineering**

# IA-500: Introduction to Information Assurance

Lab 9 – Packets – Packet Analysis

**Lab Objective:**

The objective of this lab is to introduce students to Wireshark and packet analysis. The students will be able to use Wireshark to analyze packets on a network. Students will also be introduced to what network packets are.

**Laboratory Deliverable:**

1. Screenshot of filtering packets sent from IP address 192.168.100.201.
2. Screenshot of filtering packets sent to IP address 192.168.100.201.
3. Screenshot of filtering packets sent via the ICMP protocol.
4. Screenshot of filtering packets sent via the TCP protocol.
5. Screenshot of filtering packets sent via TCP port 80.
6. Screenshot of filtering packets sent to TCP port 80.
7. Screenshot of filtering packets sent from TCP port 80.
8. Screenshot of filtering packets sent via TCP ports greater than 32.
9. Screenshot of analyzing HTML form submission on HTTP server.
10. Screenshot of analyzing FTP password request.
11. Screenshot of analyzing SSH response.

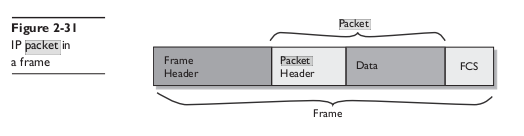
**Materials:**

1. Kali Linux Virtual Machine
2. Metasploitable Virtual Machine

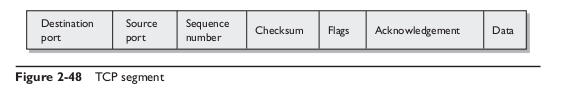
**Background:**

A network packet is a container containing data that is sent between networked computers. Packets can use different protocols and their data will differ. A frame is the container for a packet and allows for the packet to be sent from one device to another.

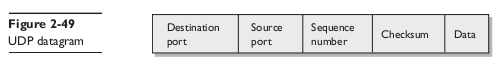
The following is an IP packet and its frame.



The data within the IP packet is called the TCP segment. It looks like the following:



The data within an IP packet could also be a UDP datagram. It looks like the following:



**Instructions:**

1. Open your Kali Linux VM and then open up the Wireshark application.
2. Press File->Open and navigate to the file that you captured in Lab 9 and open it.
3. The Filter toolbar is used very commonly when analyzing packet captures. It allows the user to sort the packets via certain filters. We can quickly organize packets to only see packets sent from a certain host or a packet of a certain protocol.

Let’s try filtering all packets from the IP address 192.168.100.201, in the Filter toolbar enter ‘ip.src == 192.168.100.201’ and press [ENTER].

**1Take a screenshot.**

1. We can find all the packets sent to the IP address 192.168.100.201 as well. In the Filter toolbar enter ‘ip.dst == 192.168.100.201’ and press [ENTER].

**2Take a screenshot.**

1. We can also find all the packets that are sent with the ICMP protocol. These are the packets that are created when we ping other hosts. To filter these in the filter toolbar enter ‘ip.proto == ICMP’ and press [ENTER].

**3Take a screenshot.**

1. Instead of finding all the packets with ICMP, let’s find the ones using TCP. In the filter toolbar enter ‘ip.proto == TCP’ and press [ENTER].

**4Take a screenshot.**

1. We can filter packets by the port that they were sent from too. Let’s find the packets that used the TCP port 80 (HTTP) by entering in the filter toolbar ‘tcp.port == 80’.

**5Take a screenshot.**

1. We can also filter the packets by port if they used the UDP protocol too by just replacing the ‘tcp’ portion of the filter expression with ‘udp’.
2. Just like when we search for the IP destination address, we can filter by TCP destination port. In the filter toolbar, enter ‘tcp.dstport == 80’.

**6Take a screenshot.**

1. Let’s filter by the TCP source port on 80 by entering in the filter toolbar, ‘tcp.srcport == 80’.

**7Take a screenshot.**

1. Let’s filter the TCP ports that are greater than 32. We can do this by using a greater than sign in the filter toolbar, ‘tcp.port > 32’.

**8Take a screenshot.**

1. Clear your filter and press [ENTER]. Click on the first TCP packet that you see. In a TCP packet there are three parts and a frame overview in Wireshark’s packet details window. The first part is the frame overview and contains the frame information. The second part is the Ethernet header information which tells you the MAC address of the two host computers involved in the communication. The third part is the IPv4 header which tell you the IP addresses involved in the communication and many more state values of the packet. The last part is the TCP specific data, although this portion is always the data specific to what kind of packet we are looking at, which contains the ports involved in the communication and other state values.
2. Now we will filter out the packets with ICMP data. We do this just like we did earlier, with the expression ‘ip.proto == ICMP’.

You should see six frames. Click the first.

In the packet details window, press the plus button next to the line that reads ‘Internet Control Message Protocol’ and then look at the sequence number. See how it is 1?

Now press the second packet. See how the sequence number is still 1 but that this packet is a reply to the first frame that was a request?

Now press the third packet. See how the sequence number is now 2? The sequence number is important because it is what allows the host to see if another host has successfully communicated back. If a reply doesn’t come back with the same sequence number, then a packet has been dropped (has not successfully transmitted).

1. Now we will filter out the packets with HTTP content data. We do this by entering ‘http.content\_type’ in the filter toolbar and pressing [ENTER].

Find a packet with the Info description of ‘POST /twiki/bin/register/Main/WebHome HTTP/1.1 (application/x-www-form-urlencoded)’ and click it.

In the packet details window (the middle window) click the plus-sign next to the line that reads ‘Line-Based text data: application/x-www-form-urlencoded’. Do you remember the form you filled out when you registered an account last lab? This is the data submission to the server. The data you submitted is stored right here in plain text because we didn’t use a secure server (HTTPS). Click the line with the ‘[truncated]’ bit in it. You will see the packet bytes (bottom window) have a portion of its data highlighted. The highlighted part is the data we have selected from the packet.

**9Adjust the bottom window so the entire highlighted portion can be seen and take a screenshot.**

1. Now we will filter out the packets with FTP content. We do this by filtering the packets to the TCP protocol on port 21. The expression is ‘tcp.port == 21’.

Click a packet with the info ‘Request: USER msfadmin’. Then select the plus-sign next to the part that reads ‘File Transfer Protocol (FTP)’ and open ‘USER msfadmin\r\n’.

See how there is a request command and argument? That is the payload of this FTP packet.

1. Now click a packet with the info ‘Request: PASS msfadmin’. Then select the part that reads ‘File Transfer Protocol (FTP)’ and open up ‘PASS msfadmin\r\n’.

See how there is a request command and argument again? This is another payload for the FTP packet. It also happens to be in plaintext because we are using secure FTP.

**10Take a screenshot.**

1. Now we will filter all the packets that are using the SSH protocol. We filter all packets using TCP on the port 22. The expression is ‘tcp.port == 22’.

Click on a packet with the info ‘Encrypted response packet’. Open the ‘SSH Protocol’ inside the packet details window and open up the sub-data that reads ‘SSH Version 2…’.

Do you see how there is no eligible data? That is because the data has been encrypted because SSH is a secure application that will not send your data in plain-text. SSH uses AES-128 encryption which is a strong encryption standard (the government approves it to encrypt Top-Secret data) that is very, very difficult to break.

**11Take a screenshot.**

==END==

**Questions:**

1. What is packet analysis? Identifying what is inside each packet, and the seeing the conversation between the client to server
2. What does filtering do in Wireshark? It filters out the unwanted information through different search parameters (like searching for only TCP from a specific address)
3. What is a dropped packet? A packet that wasn’t successfully transmitted
4. What is the expression to filter packets with the UDP port of 6432? udp.dstport == 6432
5. What is the expression to filter packets with a TCP port greater than 1024? tcp.port > 1024
6. Why is it important to use HTTPS and not HTTP (in reference to this lab)? http is unencrypted and can easily be seen on wireshark, while https only displays the encrypted information so you can’t easily see private information on wireshark
7. What is the encryption that SSH uses that you found in your analysis? AES-128 encryption
8. What is the version of HTTP that you found in your analysis? 1.1
9. What FTP server did you find running in your analysis?
10. What are TCP sequence numbers and their importance?
11. What is the significance if SYN/ACK/FIN in reference to TCP?
12. What happens if there is too much data to hold in a single packet? Does the network just get rid of that data?

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement | **Points Allowed** | **Points Actual** | **Comments** |
|  |  |  |  |
| **Title page** | **5** |  |  |
| **Screen shots** | **5** |  |  |
| **Questions** | **10** |  |  |
| **Conclusion** | **5** |  |  |
|  |  |  |  |
| **Extra Credit** |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **Total Points** | **25** |  |  |