In this reading, you'll re-examine IPv4 and IPv6 headers. Then, you'll explore how you can use Wireshark to investigate the details of packet capture files. Internet Protocol (IP) Packets form the foundation of data exchange over a network, which means that detection begins at the packet level. The Internet Protocol (IP) includes a set of standards used for routing and addressing data packets as they travel between devices on a network. IP operates as the foundation for all communications over the internet. IP ensures that packets reach their destinations. There are two versions of IP that you will find in use today: IPv4 and IPv6. Both versions use different headers to structure packet information. IPv4 is the most commonly used version of IP. There are thirteen fields in the header: • **Version**: This field indicates the IP version. For an IPv4 header, IPv4 is used. • Internet Header Length (IHL): This field specifies the length of the IPv4 header including any Options. • Type of Service (ToS): This field provides information about packet priority for delivery. • **Total Length**: This field specifies the total length of the entire IP packet including the header and the data. • Identification: Packets that are too large to send are fragmented into smaller pieces. This field specifies a unique identifier for fragments of an original IP packet so that they can be reassembled once they reach their Flags: This field provides information about packet fragmentation including whether the original packet has been fragmented and if there are more fragments in transit. • Fragment Offset: This field is used to identify the correct sequence of fragments. • Time to Live (TTL): This field limits how long a packet can be circulated in a network, preventing packets from being forwarded by routers indefinitely. • **Protocol**: This field specifies the protocol used for the data portion of the packet. Header Checksum: This field specifies a checksum value which is used for error-checking the header. • Source Address: This field specifies the source address of the sender. • **Destination Address**: This field specifies the destination address of the receiver. • **Options**: This field is optional and can be used to apply security options to a packet. IPv6 adoption has been increasing because of its large address space. There are eight fields in the header: • Version: This field indicates the IP version. For an IPv6 header, IPv6 is used. Traffic Class: This field is similar to the IPv4 Type of Service field. The Traffic Class field provides information about the packet's priority or class to help with packet delivery. • Flow Label: This field identifies the packets of a flow. A flow is the sequence of packets sent from a specific Payload Length: This field specifies the length of the data portion of the packet. • Next Header: This field indicates the type of header that follows the IPv6 header such as TCP. • Hop Limit: This field is similar to the IPv4 Time to Live field. The Hop Limit limits how long a packet can travel in a network before being discarded. Source Address: This field specifies the source address of the sender. • **Destination Address**: This field specifies the destination address of the receiver. $Header fields \, contain \, valuable \, information \, for \, investigations \, and \, tools \, like \, Wireshark \, help \, to \, display \, these \, fields \, in \, a \, display \, the \, contain \, valuable \, information \, for \, investigations \, and \, tools \, like \, Wireshark \, help \, to \, display \, these \, fields \, in \, a \, display \, the \, contain \, valuable \, information \, for \, investigations \, and \, tools \, like \, Wireshark \, help \, to \, display \, these \, fields \, in \, a \, display \, the \, contain \, valuable \, information \, for \, investigations \, and \, tools \, like \, Wireshark \, help \, to \, display \, the \, contain \, valuable \, information \, for \, investigations \, and \, tools \, like \, Wireshark \, help \, to \, display \, the \, contain \, valuable \, information \, for \, investigations \, and \, tools \, like \, Wireshark \, help \, to \, display \, the \, contain \, valuable \, information \, for \, investigations \, and \, tools \, like \, Wireshark \, help \, to \, display \, the \, contain \, the \,$ Wireshark Wireshark is an open-source network protocol analyzer. It uses a graphical user interface (GUI), which makes it easier to visualize network communications for packet analysis purposes. Wireshark has many features to explore that are beyond the scope of this course. You'll focus on how to use basic filtering to isolate network packets so that you can find what you need. The Wireshark Network Analyzer Learn User's Guide · Wild · Questions and Answers · Mailing Lists · SharkFest · Wireshark Discord · Denate Wireshark's display filters let you apply filters to packet capture files. This is helpful when you are inspecting packet captures with large volumes of information. Display filters will help you find specific information that's most relevant to your investigation. You can filter packets based on information such as protocols, IP addresses, ports, and virtually any other property found in a packet. Here, you'll focus on display filtering syntax and filtering for protocols, IP addresses, and ports. Comparison operators You can use different comparison operators to locate specific header fields and values. Comparison operators can be $expressed\ using\ either\ abbreviations\ or\ symbols.\ For\ example, this\ filter\ using\ the\ \verb§{==}\ equal\ symbol\ in\ this\ filter$ ip.src == 8.8.8.8 is identical to using the eqabbreviation in this filter ip.src eq 8.8.8.8. This table summarizes the different types of comparison operators you can use for display filtering. Operator type Symbol Not equal Greater than Less than Greater than or equal to Less than or equal to **Pro tip:** You can combine comparison operators with Boolean logical operators like **and** and **or** to create complex display filters. Parentheses can also be used to group expressions and to prioritize search terms. The contains operator is used to filter packets that contain an exact match of a string of text. Here is an example of a filter that displays all HTTP streams that match the keyword "moved". No. Time Source Destination Protocol Length Info 69 18.036927 142.250.1.139 172.21.224.2 HTTP 648 HTTP/1.1 301 Moved Permanently (text/html) 150 42.370267 142.250.1.102 172.21.224.2 HTTP 657 HTTP/1.1 301 Moved Permanently (text/html) The matches operator is used to filter packets based on the regular expression (regex) that's specified. Regular $expression\ is\ a\ sequence\ of\ characters\ that\ forms\ a\ pattern.\ You'll\ explore\ more\ about\ regular\ expressions\ later\ in\ this$ Filter toolbar You can apply filters to a packet capture using Wireshark's filter toolbar. In this example, **dns** is the applied filter, which means Wireshark will only display packets containing the DNS protocol. 0. Time Source Destination Protocol Length Info 9 8.637619 172.21.224.2 169.254.169.254 DNS 81 Standard query 0x0c26 A op 10 8.637625 172.21.224.2 169.254.169.254 DNS 81 Standard query 0x0c26 A op 11 8.641838 169.254.169.254 172.21.224.2 DNS 193 Standard query response 0x 12 8.641978 169.254.169.254 172.21.224.2 DNS 177 Standard query response 0x 19 8.644093 172.21.224.2 169.254.169.254 DNS 86 Standard query ox0549 PTR 20 8.647339 169.254.169.254 172.21.224.2 DNS 120 Standard query response 0x 27 9.645214 172.21.224.2 169.254.169.254 DNS 86 Standard query response 0x 28 9.645859 169.254.169.254 172.21.224.2 DNS 120 Standard query response 0x 33 10.646715 172.21.224.2 169.254.169.254 DNS 86 Standard query ox0540 PTR Filter toolbar **Pro tip**: Wireshark uses different colors to represent protocols. You can customize colors and create your own filters. Filter for protocols $Protocol\ filtering\ is\ one\ of\ the\ simplest\ ways\ you\ can\ use\ display\ filters.\ You\ can\ simply\ enter\ the\ name\ of\ the\ protocol\ protocol\$ $to \ filter. For example, to \ filter \ for \ DNS \ packets \ simply \ type \ \textbf{dns} \ in \ the \ filter \ toolbar. Here \ is \ a \ list \ of \ some \ protocols \ you \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not be a \ filter \ for \ do not \ filter \ for \ filter \ for \ do not \ filter \ for \ filter \ for \ filter \ for \ filter \ filter \ for \ filter \ filter \ for \ filter \ for \ filter \ filter \ for \ filter \ for \ filter \ for \ filter \ filte$ telnet icmp Filter for an IP address You can use display filters to locate packets with a specific IP address. For example, if you would like to filter packets that contain a specific IP address use ip.addr, followed by a space, the equal == comparison operator, and the IP address. Here is an example of a display filter that filters for the IP address ip.addr == 172.21.224.2 To filter for packets originating from a specific source IP address, you can use the ip.src filter. Here is an example that looks for the 10 . 10 . 10 . 10 source IP address: ip.src == 10.10.10.10 To filter for packets delivered to a specific destination IP address, you can use the ip.dst filter. Here is an example that searches for the 4.4.4.4 destination IP address: ip.dst == 4.4.4.4 You can also filter packets according to the Media Access Control (MAC) address. As a refresher, a MAC address is a $unique\ alphanumeric\ identifier\ that\ is\ assigned\ to\ each\ physical\ device\ on\ a\ network.$ Here's an example: eth.addr == 00:70:f4:23:18:c4 Filter for ports Port filtering is used to filter packets based on port numbers. This is helpful when you want to isolate specific types of traffic. DNS traffic uses TCP or UDP port 53 so this will list traffic related to DNS queries and responses only. For example, if you would like to filter for a UDP port: udp.port == 53 Likewise, you can filter for TCP ports as well: tcp.port == 25 Wireshark provides a feature that lets you filter for packets specific to a protocol and view streams. A stream or conversation is the exchange of data between devices using a protocol. Wireshark reassembles the data that was transferred in the stream in a way that's simple to read. ✓ Wireshark · Follow TCP Stream (tcp.stream eq 4) · sample.pcap GET / HTTP/1.1 Host: opensource.google.com User-Agent: curl/7.74.0 Accept: */* HTTP/1.1 301 Moved Permanently Location: https://opensource.google/ Cross-Origin-Resource-Policy: cross-origin Content-Type: text/html; charset=UTF-8 X-Content-Type-Options: nosniff Date: Wed, 23 Nov 2022 12:38:34 GMT Expires: Wed, 23 Nov 2022 13:08:34 GMT Cache-Control: public, max-age=1800 Server: sffe Content-Length: 223 X-XSS-Protection: 0 <HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8"> <TITLE>301 Moved</TITLE></HEAD><BODY> <H1>301 Moved</H1> The document has moved here. </BODY></HTML> Following a protocol stream is useful when trying to understand the details of a conversation. For example, you can examine the details of an HTTP conversation to view the content of the exchanged request and response messages. Key takeaways In this reading, you explored basic display filters with Wireshark. Packet analysis is an essential skill that you will $continue\ to\ develop\ over\ time\ in\ your\ cybersecurity\ journey.\ Put\ your\ skills\ to\ practice\ in\ the\ upcoming\ activity\ and$ explore investigating the details of a packet capture file using Wireshark! • To learn more about Wireshark's full features and capabilities, explore the Wireshark official user guide [2]. Mark as completed 🖒 Like 🖓 Dislike 🏳 Report an issue

Understand network traffic

Capture and view network traffic

(D) Video: Packets and packet captures

Reading: Learn more about packet

(D) Video: Interpret network communications with packets

Video: Reexamine the fields of a packet header

Reading: Investigate packet details 20 min

Reading: Resources for completing labs

(Z) Ungraded App Item: Activity: Analyze your first packet 30 min

Ungraded App Item: Optional Exemplar: Analyze your first

Practice Quiz: Test your knowledge:
Capture and view network traffic

Review: Network monitoring and analysis

4 questions

Packet inspection

captures 20 min Investigate packet details

So far, you've learned about how network protocol analyzers (packet sniffers) intercept network communications. You've also learned how you can analyze packet captures (p-caps) to gain insight into the activity happening on a

network. As a security analyst, you'll use your packet analysis skills to inspect network packets and identify suspicious

?