Jaimie Louie & Leah Shen
COSI 107a
May 14, 2024

A Deep Dive into Network Analyzers:
How Do They Protect Networks, and What Makes Their Usage Ethical?

Introduction

With the growing scale of the internet and its capabilities in the accessing and spreading of information, cybersecurity concerns also continue to grow as people around the world worry about who can see information that they would rather keep private and how they can protect themselves from such invasions of privacy. As the internet continues to expand, new attack surfaces also arise, giving attackers who seek to exploit people's private information more possible ways to breach security. A substantial concern has been the use of network analyzers, which are software tools that can be used to capture packet information and observe information about networks. Although these tools are often used by security professionals and administrators to monitor and keep their networks secure, attackers can also take advantage of these analyzers to find weaknesses and security flaws in a system, or to find out information about other users within the network without permission. This leads to a question of ethics and whether the use of these network scanning tools could be considered a breach of privacy, since they can reveal information about network traffic often without users knowing.

In order to discover more about this issue, we researched two popular network analyzers and how they are used, as well as tested them out safely. We primarily focused on Nmap, a network mapper, and Wireshark, a packet 'sniffer' (analyzer). We will discuss the functions of both these tools, the ethics tied to using them, as well as explore cases of possible breaches of privacy, laws, and terms of service. We will also explore the general rules and ethical principles that are applied within the field of cybersecurity. By synthesizing our understanding of these rules with our understanding of these network analyzers, we will make a conclusion about under what circumstances the usage of these tools can be ethical.

Nmap

Nmap is a free, open-source network mapper primarily used for network exploration, security auditing, and vulnerability assessment. Most commonly used through the command line, the user can perform specific types of scans using various flags. The main functions of Nmap include active host discovery, identifying the services running on the hosts, port scanning, and OS detection (Lyon, 2022). Though it was designed to scan larger networks, such as those of enterprises, smaller businesses are increasingly using it as more and more devices are requiring network connection - meaning a larger attack surface and more vulnerabilities. Not only used by network administrators to monitor network traffic and connected devices (Buckbee, 2022), this tool is also used by security professionals, ethical hackers, and IT security companies for penetration testing (Thelberg, 2023). Even personal website owners can use Nmap to check the security of their own website by simulating an attack. However, malicious attackers often use Nmap as a first step in order to get a layout of their target's attack surface - the network - and to check if there are any vulnerabilities such as open ports they can exploit to gain access to the system (Thelberg, 2023). Nmap can be used for a variety of purposes by both attackers and security experts, but the most common features of network mapping remain the same.

Host Discovery

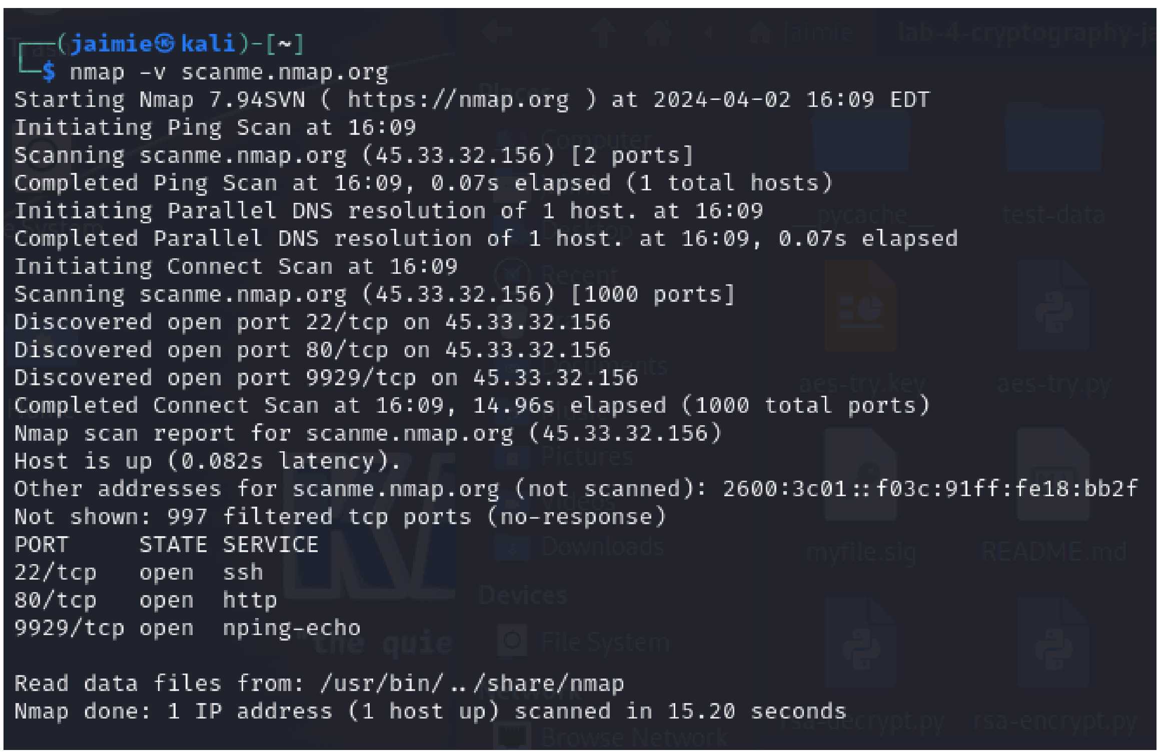
The first step to network mapping is host discovery or "ping scanning," which identifies all the active devices on the network, checking if an IP address is being used by a device (Lyon, 2022). This is extremely useful since there are usually not many IP addresses active on a network at once compared to the total number of existing IP addresses, and host discovery can narrow down which of those are active (Lyon, 2022). Based on the user's command specifications, Nmap can perform different types of scans, some more intrusive than others. Nmap's default ping scan, using -sn, sends four packets for host discovery without any port scanning: an ICMP echo request, a TCP SYN packet to port 443 (HTTPS), a TCP ACK packet to port 80 (HTTP), and an ICMP timestamp request (Everson & Cheng, 2024). If the host is active, an ICMP echo response is expected to be sent back; however, these ICMP requests are often blocked by firewalls due to security concerns such as DDoS attacks. The TCP packet with a SYN flag signals to the host that it is trying to establish a connection, and the host will reply with a RST packet (reset connection) or a SYN/ACK packet if it is responsive (Lyon, 2022). The TCP ACK packet, which has the ACK flag set, is meant to acknowledge data over the TCP connection with the host - yet this connection does not exist. Thus, the host will send back a RST packet, which reveals it is responsive (Lyon, 2022). Nmap sends multiple types of packets in order to increase the chances of discovering active hosts even if there is a firewall in place, since firewalls may be configured to block or drop these types of packets.

Port Scanning

After host discovery, port scanning is used to check the status of each port. Nmap's main classifications for port status are open, closed, and filtered. Open ports, meaning the service on that port is listening for packets and connections, are commonly the central focus of both security professionals and attackers since they are vulnerable to attack (Lyon, 2022). Closed ports are responsive but do not have any applications listening on them, sending back a RST packet when probed (Singh, 2023). It may be useful to scan closed ports since they could open at any time. Filtered means that a firewall or a similar protective mechanism is blocking Nmap's packets from reaching the port, so Nmap cannot determine its status. Most often the filter drops the packets, so Nmap tries sending more packets in case it was due to a network problem, greatly slowing down the scan (Lyon, 2022). Thus, firewalls can be an effective security measure against attackers scanning a network.

Nmap's most widely used scan type is a TCP scan, which connects to a host with a three-way handshake. A SYN packet is sent to the server, the server responds with a SYN-ACK packet, and then the source sends an ACK packet to complete the connection (Everson & Cheng, 2024). This is the same as a SYN scan used for host discovery except with an extra ACK packet sent - a SYN scan is stealthier and more discreet since it does not acknowledge the connection with the server. On the other hand, the TCP scan is slower and could create a log entry documenting the scan, letting network administration know a scan was attempted (Singh, 2023). The other type of scan is a UDP scan, not as common as TCP but used for certain services such as DNS. Since this scan is connectionless, it is more difficult to figure out if a port is open. After a UDP packet is sent, the server sends back an ICMP error message if there is no service running (port is closed), and sends either nothing or a data packet only if the probe is the right structure for the service (port is open) (Everson & Cheng, 2024). Because UDP scans are much slower than TCP scans, security professionals often ignore them while auditing networks (Lyon, 2022). However, this underestimates attackers' use of UDP scans and could create security vulnerabilities regarding UDP ports.

In order to test Nmap out, we performed a basic host discovery scan on Kali Linux, which includes a ping scan followed by a port scan. We scanned the website scanme.nmap.org, a website provided by Nmap for practice scanning using the command nmap -v scanme.nmap.org, -v for verbosity. This way, we would not be scanning a website without permission. Below are the results of the scan.



```
┌──(jaimie㉿kali)-[~]
└─$ nmap -v scanme.nmap.org
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-02 16:09 EDT
Initiating Ping Scan at 16:09
Scanning scanme.nmap.org (45.33.32.156) [2 ports]
Completed Ping Scan at 16:09, 0.07s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 16:09
Completed Parallel DNS resolution of 1 host. at 16:09, 0.07s elapsed
Initiating Connect Scan at 16:09
Scanning scanme.nmap.org (45.33.32.156) [1000 ports]
Discovered open port 22/tcp on 45.33.32.156
Discovered open port 80/tcp on 45.33.32.156
Discovered open port 9929/tcp on 45.33.32.156
Completed Connect Scan at 16:09, 14.96s elapsed (1000 total ports)
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.082s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 997 filtered tcp ports (no-response)
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
9929/tcp open  nping-echo

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 15.20 seconds
```

The ping scan discovered 1 host, then scanned the 1,000 most common TCP ports at that host. Only 3 of those ports were open while the other 997 gave no response. Nmap displayed each of the ports along with their status and the service running on it.

OS Detection

Nmap can also identify a remote device's operating system through TCP/IP stack fingerprinting, sending TCP and UDP packets to the host and performing tests on the responses (Lyon, 2022). These test results are compared to Nmap's database of over 2,600 operating systems to attempt to find an OS match, in which Nmap prints information about the OS including vendor name and version number (Keary, 2023). If Nmap cannot find a match, the user may submit the OS information of the host if known (Lyon, 2022). OS detection is important for security testers since different exploits work against different OS and their versions, and being aware of these vulnerabilities can help predict and protect against various types of attacks. For example, Nmap's OS detection has been used on a botnet to discover the compromised devices' OS and see what vulnerabilities could have been exploited to infect the device (Everson & Cheng, 2024).

Below are snippets of the results from OS scanning scanme.nmap.org with the command sudo nmap -O -v scanme.nmap.org. Sudo is required since a SYN (stealth) scan involves sending raw packets.

```
┌──(jaimie㉿kali)-[~]
└─$ sudo nmap -O -v scanme.nmap.org
[sudo] password for jaimie:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-02 16:23 EDT
Initiating Ping Scan at 16:23
Scanning scanme.nmap.org (45.33.32.156) [4 ports]
Completed Ping Scan at 16:23, 0.01s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 16:23
Completed Parallel DNS resolution of 1 host. at 16:23, 0.00s elapsed
Initiating SYN Stealth Scan at 16:23
Scanning scanme.nmap.org (45.33.32.156) [1000 ports]
Discovered open port 22/tcp on 45.33.32.156
Discovered open port 80/tcp on 45.33.32.156
Discovered open port 31337/tcp on 45.33.32.156
Increasing send delay for 45.33.32.156 from 0 to 5 due to 11 out of 25 dropped pro
bes since last increase.
Discovered open port 9929/tcp on 45.33.32.156
Increasing send delay for 45.33.32.156 from 5 to 10 due to 11 out of 16 dropped pr
obes since last increase.
Increasing send delay for 45.33.32.156 from 10 to 20 due to 54 out of 178 dropped
probes since last increase.
```

This SYN scan to discover open ports took much longer than the host discovery scan - about 16 minutes compared to 15 seconds - because a majority of probes were dropped every time they were sent. This could indicate a firewall configured to drop untrusted packets.

```
Initiating OS detection (try #1) against scanme.nmap.org (45.33.32.156)
Retrying OS detection (try #2) against scanme.nmap.org (45.33.32.156)
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.31s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 988 closed tcp ports (reset)
PORT      STATE    SERVICE
21/tcp    filtered ftp
22/tcp    open     ssh
42/tcp    filtered nameserver
80/tcp    open     http
135/tcp   filtered msrpc
139/tcp   filtered netbios-ssn
445/tcp   filtered microsoft-ds
514/tcp   filtered shell
3283/tcp  filtered netassistant
3372/tcp  filtered msdtc
9929/tcp  open     nping-echo
31337/tcp open     Elite
Aggressive OS guesses: Actiontec MI424WR-GEN3I WAP (99%), DD-WRT v24-sp2 (Linux 2.4.37) (98%), Li
nux 3.2 (96%), Microsoft Windows XP SP3 or Windows 7 or Windows Server 2012 (96%), Linux 4.4 (94%
), Microsoft Windows XP SP3 (94%), BlueArc Titan 2100 NAS device (90%)
No exact OS matches for host (test conditions non-ideal).
TCP Sequence Prediction: Difficulty=251 (Good luck!)
IP ID Sequence Generation: Incremental

Read data files from: /usr/bin/../share/nmap
OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 966.76 seconds
           Raw packets sent: 2923 (131.804KB) | Rcvd: 2890 (116.172KB)
```

After completing the SYN scan, Nmap attempted the OS detection process, listing the filtered or open ports and their services. Even though Nmap did not find any exact OS matches, it provided a few guesses and their accuracy percentages. The highest confidence guess was the Actiontec MI424WR-GEN3I router at 99%.

Service and Version Detection

Along with OS detection, Nmap can also identify the service listening on a port, including what software is running and its version number. To detect a service, Nmap connects to a port and listens for a response without sending any probes, otherwise known as a NULL probe. If a response is received, Nmap compares this response with its NULL probe signatures file to try and identify the service (Lyon, 2022). If no response is received, Nmap sends probes based on the port number and uses regular expressions to try and match responses with the database file in order to recognize the service (Lyon, 2022). Although services typically listen on certain ports, most services can run on any port, making service detection important for validating services and ensuring ports are not used to run malicious software. Version detection helps attackers figure out which exploits are possible for a specific version of a service and aids security testers in understanding vulnerabilities and which bugs need patching (Everson & Cheng, 2024). This form of detection can also help the scanner gain insight into the machine's overall purpose by discovering the combination of services running on it.

Nmap Scripting Engine (NSE)

Another powerful feature, the Nmap Scripting Engine (NSE), allows users to write and run their own scripts or Nmap provided scripts to automate tasks. Scripts are written in the Lua programming language and are used for a variety of purposes including gathering extra information about ports, vulnerability detection and exploitation, and backdoor detection (Lyon, 2022). Nmap has many useful script categories such as auth: authentication or bypassing of credentials to access a target, brute: brute force attacks on protocols, dos: testing a target with denial of service attacks, malware: testing if the target is infected by malware or a backdoor, and vuln: checking for vulnerabilities (Buckbee, 2022). The NSE is especially helpful for security testers looking to simulate attacks or find vulnerabilities in their network. However, since the NSE allows users to create whatever script they want, an attacker could run their own malicious scripts on a target. Thus, the flexibility of the scripting engine comes with more robust security possibilities as well as the danger of more potential threats.

Defenses

Being aware of defenses against Nmap are crucial to understanding Nmap from a security point of view. The greatest threats concerning scans are attackers directly targeting a certain network rather than random people scanning the internet, but it is still important regardless to ensure good network security. According to the Nmap manual, the best defense is using Nmap to proactively scan one's own network to figure out vulnerabilities and what information attackers could find through a scan (Lyon, 2022). Also, it suggests closing unnecessarily open ports and disabling unneeded services, and using a firewall to prevent the public from accessing services that should be private (Lyon, 2022). Firewalls can be effective to protect against scanning because of the deny-by-default principle, which denies everything by standard and only lets certain, trusted traffic through. This way, it is much harder for attackers to access information about a network. Firewalls that drop untrusted packets are difficult for attackers, since it takes longer for them to get a reading on a port status - extra probes may be sent in case the packet was dropped in error, or a response might not be received at all (Lyon, 2022). However, Nmap also has options attackers can use to evade firewalls. Hiding the scans, such as encoding the packets or making them resemble legitimate traffic (stealth scan) can let them bypass the firewall (Everson & Cheng, 2024). Along with firewalls, administrators can use an intrusion prevention system (IPS) to detect network scans, since some scans like host discovery follow a specific pattern. Neural networks can also be trained on different types of scans and network traffic to classify a scan as an intrusion (Everson & Cheng, 2024). Ultimately, the best defense is having trusted network security rather than trying to trick attackers since most scans are harmless if the network has no overt vulnerabilities.

Wireshark

Wireshark is a network packet analyzer - also known as a 'packet sniffer' - capable of capturing packets and presenting detailed data about the packet (Wireshark Foundation, "User's Guide" ch.1, n.d.). Network packets are separate units of data that are delivered within ethernet networks, which make them the perfect target for analysis for the purpose of finding out what might be wrong with a network if there happens to be a problem with data transmission. Wireshark helps IT professionals, government agencies, educational institutions, and more to manage their networks and troubleshoot any problems that come up. It is used mainly as an analysis tool for fixing performance issues in networks, but can also be used for purposes such as tracing connections, viewing the contents of suspicious network communications/transactions, and investigating any sudden increases in network traffic (CompTIA, n.d.). While Wireshark's functions help security professionals with noticing and investigating possible attacks on a network, the program is unable to alert an individual about these attacks - only providing information about the attack. It is more of an assistance and analysis tool than a security tool, and therefore requires the user of the program to have sufficient knowledge about information security to use it effectively. Most of the information provided by Wireshark's analysis can only be understood by individuals who understand network traffic analysis and network protocols.

Packet Sniffers

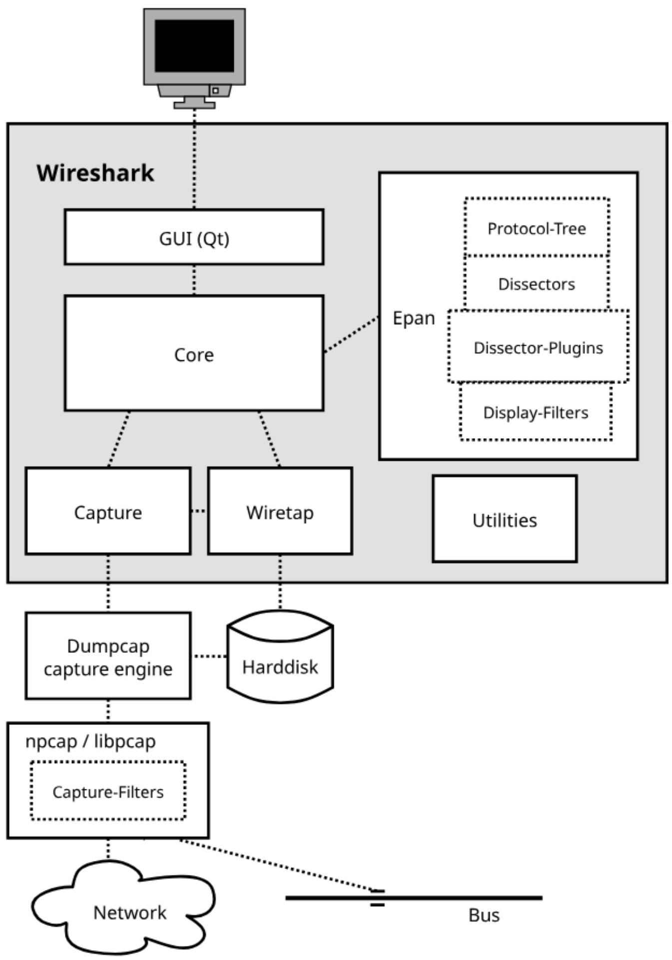
One main reason that we are investigating network analyzers is the potential for them to be used as malicious tools by hackers, and - despite Wireshark being used by many security professionals as a network maintenance tool - it is still very possible for intruders to use them as a method of infiltration. Packet sniffers are capable of capturing all incoming and outgoing packets within a network, so they can be used by hackers for accessing data being transmitted in a network without authorization, making them viable tools for stealing passwords and other sensitive information (Tuli, 2023). However, packet sniffers do not come with the capability of decrypting information, so any packet data that have been encrypted are not immediately at risk of being accessed by unauthorized users once captured.

Two types of packet sniffers exist, active sniffers and passive sniffers (Tuli, 2023). Wireshark is classified as a passive sniffer, as it can only collect data from a network but is undetectable by other systems within the same network. Vice versa, active sniffers are also able to send data within the network, but they can be detected by other systems.

Packet sniffers function by utilizing the workings of packet transmission within networks. Whenever a packet is traveling through a network, every device in the network receives the data of the packet because packets have to pass through these intermediate devices to reach their destination (Tuli, 2023). However, the packet is usually not captured and read by the device unless it is the destination. Every device in the network is connected to the network through their Network Interface Card/Controller (NIC). Packet sniffing programs put the NIC of devices in 'promiscuous mode,' which causes the NIC to actually both capture and let the device read packets that are passing through.

The process of packet sniffing can be explained through splitting it into three steps (Henry & Agana, 2019). The first step is collection, where the NIC is put into promiscuous mode to listen to all network traffic and pick up raw binary data from packets being transmitted. The next step is conversion, where the raw data is converted into a readable format. Packet sniffers operated using the command line usually stop at this step, and a majority of the analysis of data is left to the user of the packet sniffer. However, network packet analyzers like Wireshark provide assistance with analysis. The third step is analysis, where programs like Wireshark use the captured data to verify the protocol used and analyze its features.

The design and structure of the Wireshark program follows the above described three step process. Below is a diagram of the program's structure, separated into blocks, provided by Wireshark Foundation (n.d.) in Chapter 7 of their Developer's Guide.

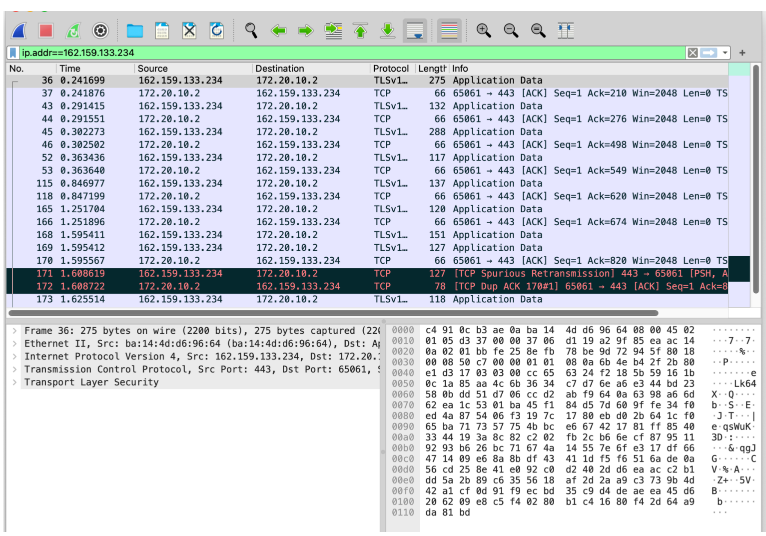


As seen in the diagram, the main function included in Wireshark is the analysis of packets, and the capturing and conversion of packets into readable formats is completed through using external libraries. Wireshark only has a block that utilizes these libraries to capture packets, rather than the program having the ability to capture packets. Therefore, an important part of the program is the Epan block - the Enhanced Packet Analyzer, which is the engine used by Wireshark for dissecting packets (Wireshark Foundation, “Developer’s Guide” ch. 7, n.d.). The protocol tree provides the dissection information from individual packets, the dissectors help dissect the protocols of packets, the dissector plugins provide support for the implementation of dissectors as distinct modules, and the display filter engine is also included in the Epan block. The usage of display filters is explained in the next section, which introduces how to use Wireshark and what libraries Wireshark uses to capture packets.

Capturing Packets

To begin capturing packets in a network using Wireshark, all one has to do is download, install, and launch the program, then click the blue shark fin button in the top left corner of the window. Captured packets will then begin rolling in, color coded based on packet type - this visualization helps individuals easily identify entire conversations of packet exchanges within the network, and to discover possible problems in the network (CompTIA, n.d.). While there is a default set of coloring rules, users are allowed to change this and apply custom coloring rules to their own convenience. This can be done through navigating the menus from View >> Coloring Rules. The default coloring rules are: Light purple for TCP, light blue for UDP, black for packet with errors, light green for HTTP traffic, light yellow for Windows-specific traffic (such as SMB or NetBIOS traffic), dark yellow for routing, and dark gray for TCP SYN, FIN, and ACK traffic.

A user may enter display filters into the filter text bar at the top of the screen so that the program will only display packets captured from specific addresses, from specific sources, or to specific destinations. The keyphrases used for these filters are “ip.addr”, “ipv6.addr”, “src”, and “dst” respectively (CompTIA, n.d.). The first two keyphrases allow the program to only capture packets from specific addresses - ipv4 addresses use the first keyphrase, and ipv6 addresses use the second keyphrase. The logical syntaxes “&&”, “==”, and “!” are used when entering a filter. Below is an example of entering a display filter, ‘ip.addr==162.159.133.234’.



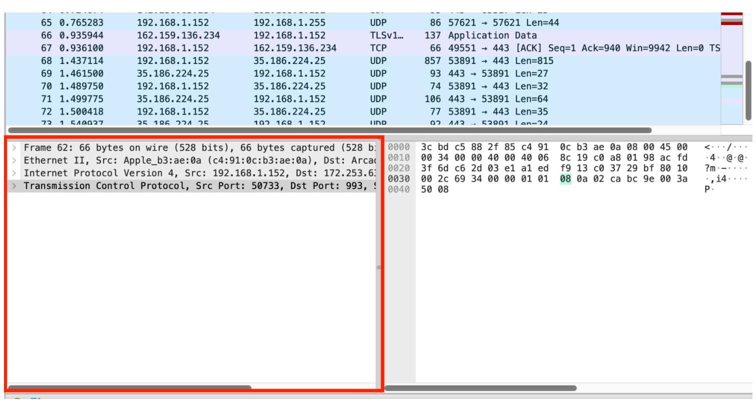
The program still captures all packets being transmitted through the network, but the visualization only displays packets that match this filter (i.e. packets that are either from or to the ipv4 address 162.159.133.234). If you change the filter mid process, then packets that were filtered out before will be displayed.

Alternatively, users may also enter capture filters instead of display filters before beginning the packet capture process. Capture filters, unlike display filters, will actually limit what packets are captured, instead of just adjusting what packets are displayed (Wireshark Foundation, “User’s Guide” ch. 4, n.d.). The Wireshark GitHub page provides many examples of available capture filters. Another difference between display filters and capture filters is that capture filters use primitive expressions. Conjunctions such as “and”, “or”, and “not” are used for defining capture filters. The reason why capture filters and display filters use different syntaxes is because the functionality of capture filters are supported by external libraries rather than Wireshark’s display filters, and these libraries work at a lower level compared to Wireshark’s display filters (Wireshark Foundation, “Developer’s Guide” ch.7, n.d.).

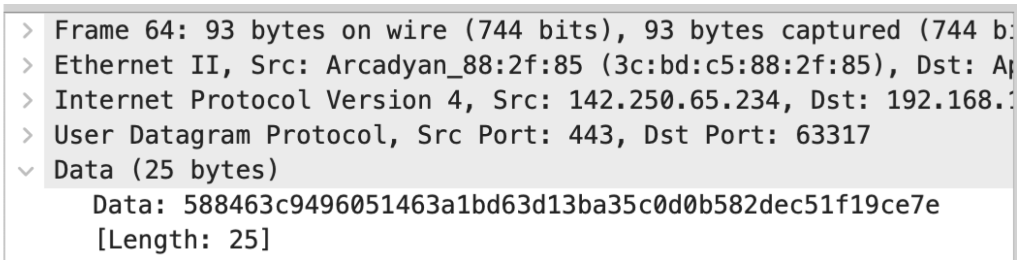
While usage of the program’s main function for capturing packets is simple enough to explain, how is Wireshark able to achieve this functionality? Wireshark accomplishes packet capture through using Dumpcap, a network traffic dump tool that uses the pcap library for capturing packets (Wireshark Foundation, “User’s Guide” Appendix D, n.d.). The capture filter syntax used in the program is also from this same library. Raw packet data and time stamps for each packet are written into a pcapng file by default, but when using Dumpcap in the command line, the user can specify the -P option for the data to be written into a pcap file instead (Wireshark Foundation, “Dumpcap(1),” n.d.).

Network Analysis and Packet Dissection

In the main Wireshark window, the user can view the details of a packet through the lower left box when a packet is selected, as highlighted in the image below.



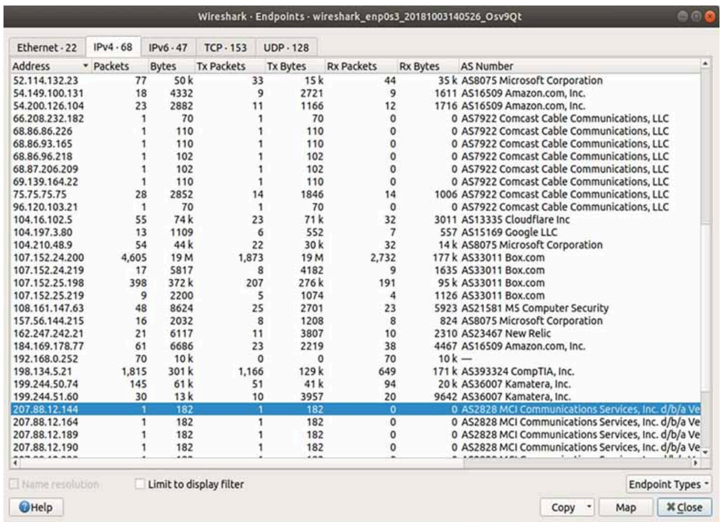
For some packet types, the user can view the data included within the packet through this interface, although in raw form. As mentioned previously, Wireshark does not decrypt data.



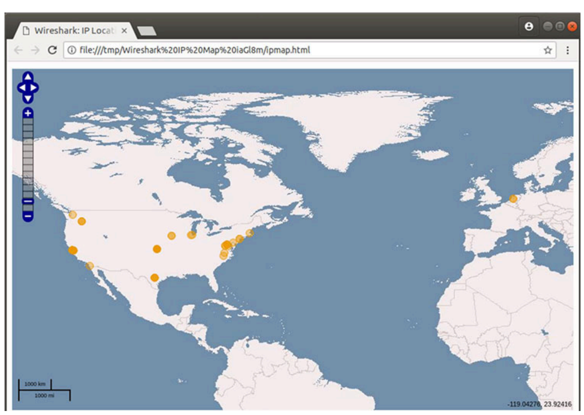
Double clicking a packet creates a new window to view the packet details. This window provides information about the packet's size, source, destination, protocols used, timestamp, and other information that could be useful to professionals maintaining a network's security. Without using Wireshark or other packet analyzers, an individual would usually have to obtain all this information through manual analysis of a packet, but this program makes the process much easier.

As a network analyzer, Wireshark also provides functionality for analyzing network traffic. By monitoring both input and output of traffic during the packet capture session, Wireshark is capable of providing users with a graph (demonstrated in the image below, through clicking Statistics >> I/O Graphs in the menu bar) of traffic statistics during the session (CompTIA, n.d.). This graph is useful to IT professionals for spotting spikes in traffic during the packet capture session, which allows them to easily spot possible attacks, especially DDoS attacks which cause abnormally high levels of traffic in a network.

A summary of the conversations that have occurred during the capture session may also be viewed by the user through Statistics >> Conversations (CompTIA, n.d.). This displays useful information about every conversation that has happened during the capture session, such as the amount of packets transferred, which addresses have had conversations during the session, the amount of bytes exchanged through their conversations, and the total duration of the conversations.



Only in some cases, users can also find the geographical location of the source and destination of traffic, accessible through this window (CompTIA, n.d.). There would be a "Map" button in the lower right of the Conversations window, as seen in the image above from CompTIA. The button opens a window displaying a map of the program's estimates of the geographical locations for the conversations that occurred during the capture session. Below is another image from CompTIA demonstrating this.



It is clear that Wireshark's capabilities allow its users to access a lot of information about what is being transmitted in a network, from where, and to where. As a packet sniffer and analyzer, Wireshark gives users access to every packet that goes through the network the sniffer is in, and the contents of these packets, which could allow unauthorized access to sensitive information. The functionality for seeing the geographical information about the sources and destinations of packets is especially something that might be seen by most as a breach of privacy, or as having other ethical concerns. So how might individuals protect themselves against packet sniffers like Wireshark?

Defenses

There are a lot of possible ways to defend against packet sniffers, including both direct countermeasures and techniques that can be used for detecting packet sniffers. While passive sniffers like Wireshark are hard to detect because of their passive nature, there are still ways to find them within the network (Tuli, 2020). Once they are detected, they can be handled by the network manager. Restricting physical access to the network to prevent the installation of packet sniffers is one effective way to stop unauthorized packet sniffers from being used within a network. Other than this, since packet sniffers cannot decrypt data, using end-to-end encryption for sensitive info, always encrypting wireless traffic, using encrypted sessions like SSH for conversations, and other encryption measures will help to protect users and their data against malicious packet sniffers even if they are present within the network. The most straightforward way for detecting passive packet sniffers within a network is checking for devices within the network which have their NIC in promiscuous mode. One simple example of a method to do this is the ping method, where a ping request would be sent to the suspect device with the correct IP address but the incorrect MAC address. If the NIC of the device is in promiscuous mode, the sniffer will respond to the ping request, because NICs in promiscuous mode do not reject packets that have the wrong MAC (Tuli, 2020).

Network Analyzer Summary

Network analyzers, although used primarily by professionals for the purpose of network security, can also be used by malicious attackers for infiltrating a network. In the cases of the two network analyzers we have investigated, they are capable of allowing attackers to find weaknesses within networks and gaining unauthorized access to sensitive information. For Nmap, the tool provides methods for users to find the vulnerabilities of a network, which can be used by both attackers and security professionals alike. For attackers, it gives them suitable pathways for infiltration. For security professionals, it allows them to find points to strengthen through penetration testing. As for Wireshark, the program gives users a tool for monitoring network traffic, capturing transmitted packets within a network, and reading the contents of these packets. Attackers can use this tool for gaining access to sensitive information, while security professionals can use Wireshark for detecting attacks or problems within the network.

Ethics & Privacy

Considering the capabilities of these tools and their ability to acquire extensive network traffic information, we must consider if they are ethical to use, especially in terms of user privacy. Privacy is the socially defined ability of people to determine whether, when, and to whom information can be shared with. If network scans reveal information about user network traffic to unknown parties, this could be considered a breach of privacy. However, since U.S. laws do not provide concrete rules defining what unauthorized access of internet information is, ethics in the cybersecurity field is an ambiguous and controversial topic. In his paper “Finding Fences in Cyberspace,” Preston argues that the internet should be open access, but there also needs to be laws, or rather property rights, in place to protect against abuse (Preston, 2001). He believes that computer security laws would assign liability more predictably and create better protections if the internet was approximated as a real, physical place with property lines or “fences” (Preston, 2001). Having clearer laws regarding cybersecurity access built on ethical principles would definitely help, since many states also have such broad definitions of access they could be interpreted in various ways. Because of this, these laws could be used to criminalize any form of communication between computers on the internet, from everyday tasks such as searching the web to dangerous phishing attacks (Preston, 2001). Thus, the law does not provide a well-defined set of ethical standards users should approach the internet or cybersecurity with.

If we look at research instead of laws, Kenneally and Bailey’s paper on a cybersecurity research ethics workshop supplies some insight into how to approach ethics in the realm of cybersecurity. With the growing gap between the capabilities of technologies and user expectations, ethical challenges are increasing as these technologies become more powerful. The paper explores the difficulties of applying ethics to security research - for example, if researchers studying phishing have a right to try and stop the criminals, or if researchers allowing themselves to be infected by a botnet to study it is ethical (Kenneally & Bailey, 2014). Since research is more nuanced than just answering whether something is “right or wrong” and researchers need to understand the impacts of their work to minimize harm, the workshop suggested creating common principles or guidelines that can help guide ethics in cybersecurity research (Kenneally & Bailey, 2014). A more positive ethics system can build trust with users, decrease ambiguity, and as a result get more accurate data. This could also include open case studies and discussion boards organized by the community (Kenneally & Bailey, 2014). Although there are no official ethical standards, we can still make informed judgements about whether or not a cybersecurity practice is ethical or not.

This leads to the question of the ethics of network scanning. Though users may think all network scanning is a breach of privacy, it is used by internet service providers (ISPs) and organizations to keep their networks safe. When signing up with an ISP, allowing scans of the users’ devices is often part of the user agreement. The Verizon Customer Agreement (2023) states: “You agree to permit us and our applicable third party suppliers to access and scan your device, network ports, and Equipment and to monitor, adjust and record data, profiles and settings for the purpose of providing Services, managing Equipment software, and managing the security and performance of our Networks.” Organizations also disclose their reasons for scanning and have users comply with their policies before joining their networks. For example, Brandeis

University’s Information Technology Services (ITS) website (n.d.-a) includes “ITS regularly scans for vulnerabilities and works to improve and strengthen Brandeis's core infrastructure, such as servers, network equipment, and web applications.” Since scanning is clearly stated in these descriptions and users are agreeing to it, scanning is not a breach of privacy as long as the providers only use the gathered information for the listed purposes.

Furthermore, these providers also ban their users from unauthorized scanning. The Xfinity Comcast Terms of Service (n.d.) state that “Prohibited conduct includes…probing the security of other hosts, networks, or accounts without express permission to do so.” They also explicitly forbid the use of network analyzer tools: “You will not use or distribute tools or devices designed or used for compromising security…This includes…analyzers, cracking tools, packet sniffers, encryption circumvention devices, and Trojan Horse programs. Unauthorized port scanning is strictly prohibited” (Comcast, n.d.). Similarly, Brandeis University ITS policy prohibits students from misusing technological resources: “students must not send unsolicited bulk communications (spam), use disproportionate amounts of network resources, conduct unauthorized network scans or probes” (Brandeis University, n.d.-b). In these cases, network probing without permission is not allowed - providers do not want their users to threaten network security in any way. ISPs may ban or even sue a user for performing a scan without approval (Lyon, 2022). Unlike attackers, security professionals use scan information to protect and ensure the safety of their network and users. Thus, it makes sense for their security teams to perform scans while also preventing unauthorized scanning.

The Nmap manual provides some guidance for users on how to use Nmap in a safe manner. Before attempting a scan on someone else’s network, it is crucial to get written authorization from the network administration first. This should include a legitimate reason and a clear description of the scan, and how the information gathered will be used (Lyon, 2022). Although stealth (SYN) scans are harder to track, they look more suspicious especially if the scanner is caught. As mentioned, it is also important to read ISP terms of service in case scans are prohibited. Court cases involving scanning may be rare, but different countries have their own computer abuse laws with their own rules about network analyzing that should be abided by.

While Wireshark’s manual does not provide guidance for the same subject, the first chapter of its User Guide does describe its intended purposes and a brief overview of the program’s functions. This provides a high level of transparency about Wireshark’s design and functionality, allowing anyone who searches it up to know about what can be done using the program. The intended purposes listed in the manual include troubleshooting networks, examining security problems, verifying network applications, debugging protocol implementations, and learning network protocols (Wireshark Foundation, “User’s Guide” ch.1, n.d.). Even though the manual does not encourage or suggest users to use the program for malicious purposes, it is still lacking for the manual not to mention the potential risks of attackers trying to use the program for malicious infiltration.

According to the idea of Open Design in the security design principles devised by Saltzer & Schroeder, the transparency within the documentation for both Nmap and Wireshark and the fact that both security tools are open source are good design choices made by the developers. These factors ensure the users understand what these tools can do and cannot do. This also ties into the ethics of usage of these tools - since individuals can easily find out about the functionality of these tools with a search, ethical usage of them may be easily achieved as well. Letting the individuals (who will be scanned by these network analyzers) understand what information about them and their devices is going to be collected when these analyzers are used in networks is key to the ethical usage of tools like these.

Conclusion

In the ever growing world of cybersecurity, increasingly more powerful network analysis tools are being created. While they may be built for the purpose of maintaining networks and ensuring their security during usage, it is impossible to ensure that no one would use the same tools for more malicious purposes. These security tools can equally be used as defense methods and as attack methods - attackers can use them to gain the same information that security professionals would have access to. Despite this, many network analysis tools, like Nmap and Wireshark, have an open design - being open source and having publicly available documentation - allowing anyone to read through how they function and what they can do. This makes it easier for users to find ways to protect themselves from attackers who might be using these tools if need be.

For the field of network scanning, intention matters. Though security professionals and attackers have access to the same scan information, they are using it for distinct purposes. Even people who just want to explore these tools must be careful in case they accidentally scan something they are not supposed to, as this can get them into trouble. The usage of these tools can be a risk both to those being scanned, and those using them.

While we have conducted in-depth investigation into the functionality of two widely used network analyzers, there are still many other network analyzers available out there with different capabilities which we are curious about. Although we were able to come to the conclusion that the ethical usage of network analyzers is based on the intentions of the user, it is still possible that looking into other network analyzers would change our conclusion. Another curious idea is the possibility of experimenting using both of these network analyzers as tools to go against attackers also using network analyzers - to see whether or not this would balance out the ethical concerns tied to their usage. Other than this, even though defenses against network analyzers was not a main subject within this paper, it is one subject that intrigues us deeply, and one of our stretch goals for this project was to look more into the defenses and possibly create a program of our own that would have the functions to defend a device against network scanners like Nmap and Wireshark. Nevertheless, we did not have the time to pursue this goal, but from the research we have conducted we can deduce that it would be possible.

The main challenge we faced for this project was our initial lack of familiarity with network analyzers in general. Due to this, we spent a lot of time and effort researching the network analysis tools we chose and their functions, and a lot of investigation was required just for us to understand how to use them and what they are capable of. Approaching the topic was intimidating at first, but through our research process we eventually gained a comfortable amount of understanding towards the tools we investigated, albeit it left us with less time. A relevant difficulty to this was ensuring that we could make sure that our experimentations using the tools (especially Nmap) was done safely, which we were able to overcome through finding concise guidelines for doing so. Other than the challenges relevant to the network analysis tools, we also encountered difficulties in our research into ethics and laws relevant to network scanning. Papers that are relevant to both network scanning and the ethics of it are rare, and looking into terms of services from different service providers to investigate terms related to network scanning was difficult as well. Lastly, the lack of clearly defined guidelines on ethics and laws related to network scanning also was an obstacle for our research. Even though this was the case, we incorporated this fact into our research, and included it as a discussion point within our section about ethics.

References

Brandeis University. (n.d.-a). *Information security*. Brandeis University. https://www.brandeis.edu/its/information-security/index.html

Brandeis University. (n.d.-b). *Section 10. Library and technology services*. Brandeis University. https://www.brandeis.edu/student-rights-community-standards/rights-responsibilities/current/section-10.html

Buckbee, M. (2022, May 4). *How to use nmap: Commands and tutorial guide*. Varonis. https://www.varonis.com/blog/nmap-commands#:~:text=Nmap%20builds%20on%20previous%2

Comcast. (n.d.). *Web services terms of service*. Xfinity. https://www.xfinity.com/terms/web

CompTIA. (n.d.). *What Is Wireshark and How to Use It | Cybersecurity*. CompTIA. https://www.comptia.org/content/articles/what-is-wireshark-and-how-to-use-it

Everson, D., & Cheng, L. (2024). A survey on network attack surface mapping. *ACM,* 1-22. https://dl.acm.org/doi/epdf/10.1145/3640019

Henry, O. N., & Agana, M. A. (2019). Intranet security using a LAN packet sniffer to monitor traffic. *9th International Conference on Computer Science and Information Technology*, 57–68. https://doi.org/10.5121/csit.2019.90806

Jacobson, V., Leres, C., & McCanne, S. (n.d.). *Pcap(3PCAP) man page | TCPDUMP & LIBPCAP*. The Tcpdump Group. https://www.tcpdump.org/manpages/pcap.3pcap.html

Kenneally, E., & Bailey, M. (2014). Cyber-security research ethics dialogue & strategy workshop. *ACM SIGCOMM Computer Communication Review, 44*(2), 76-79. https://dl.acm.org/doi/epdf/10.1145/2602204.2602217

Keary, T. (2023, April 27). *Definitive guide to nmap: How it works & scanning basics*. Comparitech. https://www.comparitech.com/net-admin/the-definitive-guide-to-nmap/

Lyon, G. (2022). *Nmap network scanning: The official Nmap project guide to network discovery and security scanning.* Nmap Software LLC. https://nmap.org/book/toc.html

Preston, E. (2001). Finding fences in cyberspace: privacy and open access on the internet. *Journal of Technology Law & Policy*, *6*(1), Article 3, 57-99. https://scholarship.law.ufl.edu/cgi/viewcontent.cgi?article=1213&context=jtlp

Singh, S. (2023, July 17). *Inside nmap, the world's most famous Port Scanner*. Pentest Tools. https://pentest-tools.com/blog/nmap-port-scanner

Thelberg, S. (2023, June 9). *What is nmap & how does it work?*. Holm Security. https://www.holmsecurity.com/blog/what-is-nmap

Tuli, R. (2023). Analyzing network performance parameters using Wireshark. *International Journal of Network Security & Its Applications*, 15(01), 01–13. https://doi.org/10.5121/ijnsa.2023.15101

Tuli, R. (2020). Packet sniffing and sniffing detection. *International Journal of Innovations in Engineering and Technology*, 16, 22. https://doi.org/10.21172/ijiet.161.04

Verizon, (2023, September 1). *Verizon customer agreement*. Verizon. https://www.verizon.com/about/terms-conditions/verizon-customer-agreement

Wireshark Foundation. (n.d.). *Dumpcap(1)*. Dumpcap(1) Manual Page. https://www.wireshark.org/docs/man-pages/dumpcap.html

Wireshark Foundation. (n.d.). *Wireshark developer's guide.* Wireshark Developer's Guide. https://www.wireshark.org/docs/wsdg_html_chunked/index.html.

Wireshark Foundation. (n.d.). *Wireshark user's guide.* Wireshark User's Guide. https://www.wireshark.org/docs/wsug_html_chunked/index.html.