U30495-19YR

Cyber Security and Digital Forensics

UP939720

Identifying Information about a remote machine

Active information gathering is the most involved and intrusive form of reconnaissance as it requires directly engaging with the victim. Active recon techniques include Port Scans and banner grabbing. Whilst the information this approach provides is very accurate, this comes at the cost of a high risk of detection **("Active Vs Passive Reconnaissance - ASM , Rockville , Maryland", 2020)**. Passive reconnaissance does not require interacting with the victim, resulting in a minimal detection chance. Passive reconnaissance typically makes use of forensic and OSINT tools such as Shodan. The reliance upon OSINT tools can limit the accuracy of information **("Passive vs. Active Reconnaissance", 2020)**.

Active recon typically starts with a port scan to see what services are running on the target. This initial scan is typically performed with NMAP. By default NMAP will only check ports 1-1000 by default **("Port Scanning Techniques | Nmap Network Scanning", 2020)** and as a result will not check the windows remote desktop protocol port at 3389 or the teamviewer remote desktop protocol port at 5938 **("List of TCP and UDP port numbers", 2020)**. Stealth scanning is currently the default option for NMAP scans as it is less intrusive. There are several techniques for performing stealth scanning although the most common is known as “Half Open Scanning”. As opposed to the traditional port scan that completes a connection on every port, this method sends a packet with just the SYN Flag and 1 byte of data to all ports. If you receive a packet with SYN/ACK that port is open and if you receive an RST packet you know that the port is closed **("TCP SYN (Stealth) Scan (-sS) | Nmap Network Scanning", 2020)**. Whilst this specific method of stealth scanning is quicker and less obtrusive when compared to traditional “Full Connection Scanning” it is still possible to detect the half open scan using a firewall with aggressive rules.

Passive recon primarily relies upon the usage of Open Source Intelligence Tools (OSINT) to get the information from the target without ever interacting with them. The first step of an attack is usually to check what ports are open, although the hands-off nature of passive recon means we cannot interact with the target in any way. Port scanning is not possible by virtue of it asking the victim if a port is available. Whilst an extremely stealthy technique for port scanning exists and is called idle zombie scanning, It still requires sending some packets although you use another host as a proxy for your packets **(“TCP Idle Scan (-sI) | Nmap Network Scanning”, 2020)**.

Packets broadcast from the victim can give a strong indication as to what operating system the victim is using based on two parameters in the packets. These two parameters are the Time To Live (TTL) and the packet window size for the first packet in a handshake. This is because RFC 791 **("RFC 791 - Internet Protocol", 2020)** and RFC 793 **("RFC 793 - Transmission Control Protocol", 2020)** do not define a value for either of those parameters. This means every OS has to pick what these values are by default. For example, Windows 10 has a default TTL of 128 **(system?, 2020)** whereas Linux and Mac OSX have a default TTL of 64 **("Hacker Geek: OS Fingerprinting With TTL and TCP Window Sizes", 2020)**. Shodan is an open source search engine for IOT devices, and can be used to find devices connected to the internet. Their web crawlers are always searching for IOT devices and cataloguing information about them in a searchable database **("Shodan", 2020)**.

Once you have learned enough about the victim you can start enumerating their network and learning about any services they may have running on their network. The primary objective of this stage is to learn version numbers of software so you can find any vulnerabilities in that software. To do this, we will be using a technique called “Banner Grabbing” **("Banner grabbing", 2020)**. Nikto is a powerful web scanner that can perform many tests upon a server to identify vulnerable versions of software. It does this through a combination of many approaches. NMAP has a flag “-sV” that identifies services and version numbers automatically **("Service and Version Detection | Nmap Network Scanning", 2020)**. Curl is a command line tool that is used to download information from web servers. Telnet is defined in RFC 15 and expanded upon in RFC 855. Telnet is an 8-bit raw data protocol and that allows it to connect to a wide variety of services. You can manually check banners with Curl and Telnet. If there is an SSH server on the host, you can fetch it’s server version using Telnet to connect to it, where it will then give you it’s version in the MOTD and it’s packets. Using Curl and the flags “-s-I” you can get the http server backend and version **(Chandel, 2020)**.

Mitigating information exposure on the remote machine

There are several techniques that can stop people scanning ports and enumerating your services, the most common of which is a firewall with aggressive rules or an intrusion detection system. A firewall can log how many requests are being made by a client and if necessary place a timeout on requests being made by that client. The growing popularity of Intrusion detection systems and firewalls has caused stealth scanning to become the default method for port scanning. You can modify the banners of most services on your system to prevent the enumeration and probing of your server. On apache you can edit a config file **(Kili, 2020)** in order to prevent most of the information transmission. On an SSH server you can only change the MOTD as the version number contains the protocol version within it. One way around this limitation is to introduce port knocking in order to stop unauthorised connections in the first place **("OpenSSH Hide Version Number From Clients - nixCraft", 2020)**. Your packet TTL values are easily changed within your operating system and are a very effective way to stop someone packet sniffing from being able to identify your

operating system. The minimum, default and maximum packet window size can be changed for Linux **(Riddell, 2020)** whereas Windows only allows you to change the maximum size, or the Maximum Transmission Unit **("Changing the MTU size in Windows Vista, 7 or 8 (mtu, speed, windows, Windows 7, Windows 8, Windows Vista)", 2020)**. Stopping an FTP server from broadcasting its version number is typically possible through the use of the server's configuration and as a result will vary depending on the software being used. An HTTP server can be prevented from sending out sensitive version numbers through the use of a custom error page **("Custom Error Responses - Apache HTTP Server Version 2.4", 2020)**. This will stop the server providing it's OS and Version number when an error occurs. Overall, port knocking is one of the best ways to mitigate information exposure, especially to unknown systems. Port knocking ensures that only authorised clients can connect and access version numbers, vastly reducing information exposure.

Analysis of PCAP file

We can determine from the capture file properties that the packet capture lasted 6 minutes and 22 seconds. To gain an overview of the hosts involved in this pcap we can look at the ARP packets recorded. Using the "arp" filter we can look at the MAC addresses of all hosts on the network. All of the MAC addresses have the same initial 3 bytes of 08:00:27, meaning that their NICs are all produced by the same vendor. Upon searching this MAC address, we can determine that these hosts are all Virtualbox virtual machines **("How to recognise an Oracle Virtual machine by its MAC address? | FAQ | MAC Address Vendor Lookup", 2020)**. In order to get a better insight into what happened in the PCAP, we can export objects to reconstruct some of what was transmitted. There are a number of notable instances of a TCP stealth scan being performed. This stealth scan can be seen on tcp streams 1030, 1039 and 1044.

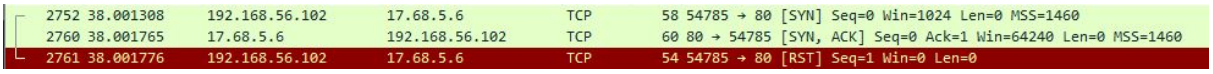

Fig. A - Open port found in TCP Stream 1030

Within these streams 192.168.56.102 sends a syn packet to port 21 on 17.56.5.6, who responds with a SYN, ACK that confirms that port 21 is open. The attacker then closes the connection with an RST flag as seen in *Fig. A.* There are many failed scan attempts starting from TCP stream 12 and stopping at TCP stream 2020. From these we can ascertain that ports 21, 22 and 80 are open on 17.68.5.6. This may give us an indication that 17.68.5.6 has FTP, SSH and HTTP services active on their machine. In addition to that, we can determine the version of these services. The SSH server is "OpenSSH 7.6p1 Ubuntu-4ubuntu0.3", the FTP server is "vsFTPd 3.0.3" and the HTTP server is "Apache/2.4.29 (Ubuntu)". These version numbers allow us to look for vulnerabilities within these pieces of software.

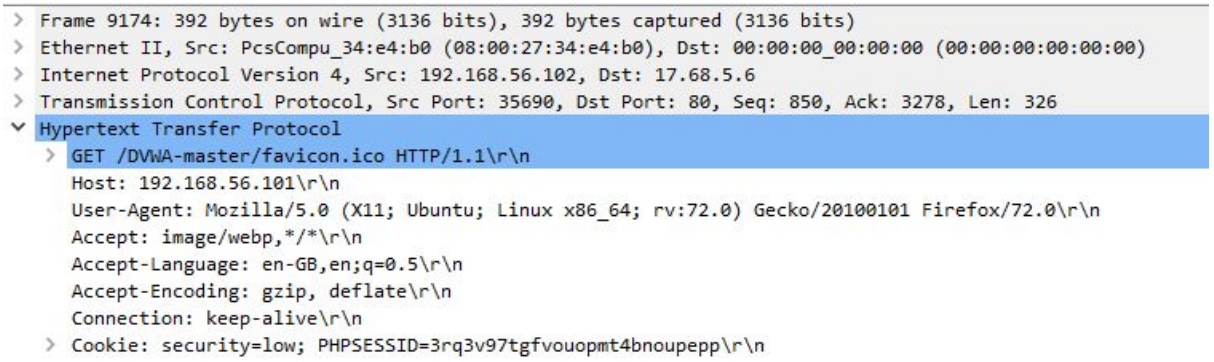

Fig. B - HTTP Get request from 192.168.56.102 showing Useragent, cookies and directory

We are also able to discern what the apache server is hosting. In this case we can look at what GET requests that were being processed by the server. As seen in *Fig. B*, packet 9174 is an HTTP GET request where 192.168.56.102 asks for "/DVWA-master/favicon.ico", we can tell from the first folder in the server's directory that the server is hosting "Damn Vulnerable Web Application". This packet betrays information about the client requesting the data in the form of the user agent, although the user agent can be spoofed. The user agent for 192.168.56.102 shows that it is an Ubuntu linux computer using Firefox 72.0. We can also view any cookies included in this request and in this case there's one of note to us, that being the security value for DVWA which is "low" in this case.

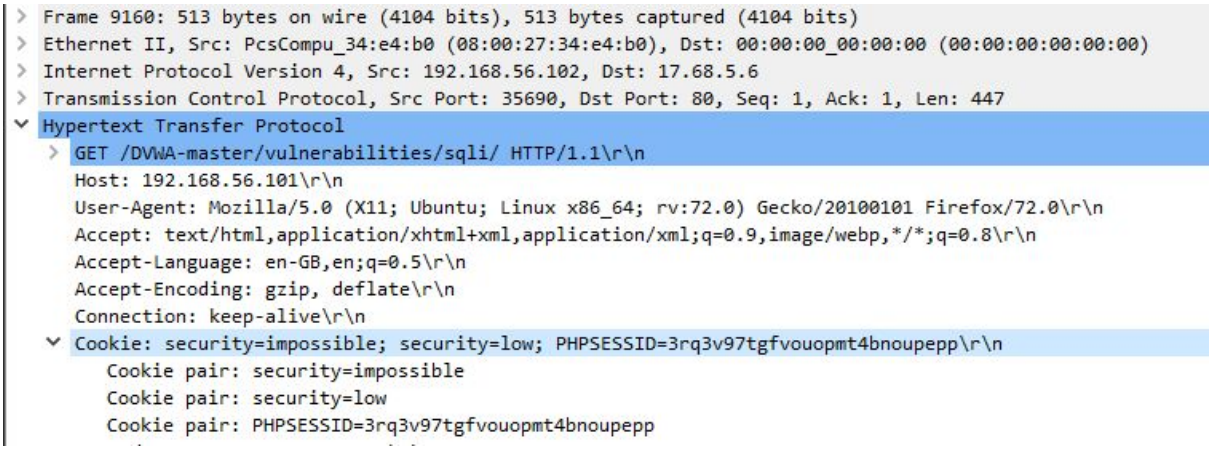

Fig. C showing both impossible and low security cookies on packet 9160

Looking at TCP stream 2055, packets 9160 and 9125 show two instances of the security cookie, the first containing a value of "impossible" and the second containing a value of "low" in both packets as seen in *Fig. C*. These packets are both in TCP stream 2055 indicating that the values were changed, likely after performing a successful SQL injection on each difficulty level.

Fig. D Showing SQL database contents after successful low security SQL injection

In this case, packets 9216 and 10612 are of interest as they are the HTTP streams that contain the webpage with the contents of the SQL database dumped upon them as seen in *Fig. D*. If we export these packets as objects and view the HTML pages we are able to see that packet 9216 showing the difficulty set to impossible along with the administrator account username and password on the screen. Packet 10612 is similar in that when exported and opened up in a browser it shows the difficulty level as low, the database ID as well as the administrator's username and password.

Fig. E Showing Query, ID and database table schema after successful injection

Packet 10612, as seen in *Fig. E* also contains the query used in the attack as well as 3 other entries under the username and password. Within the names we can see "varchar" followed by a set of brackets with an integer inside. This gives us an indication that these are column names from the database. Printing these is likely a consequence of including the database

schema in the SQL injection. These two HTTP streams are just the successful SQL injections, there were 45 attempts made to dump the contents of the DB with only 2 working.

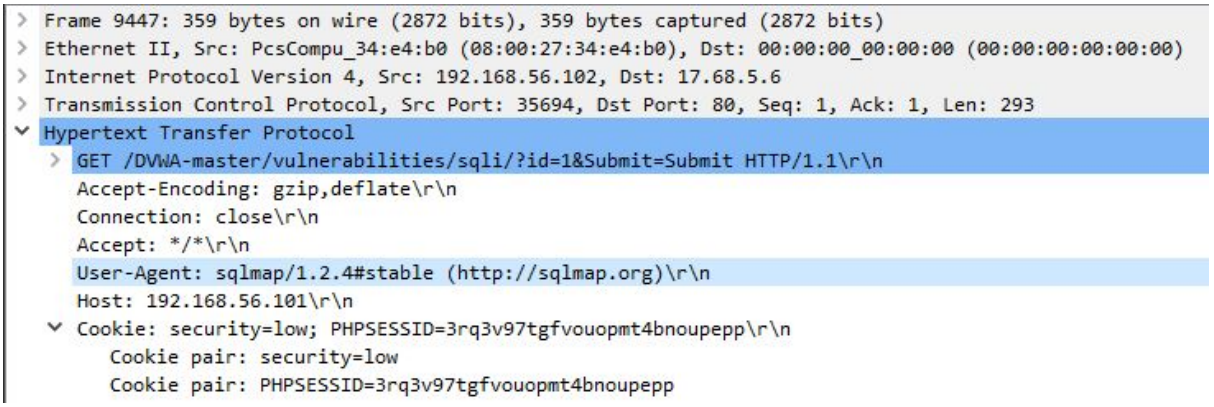

Fig. F showing SQLmap useragent

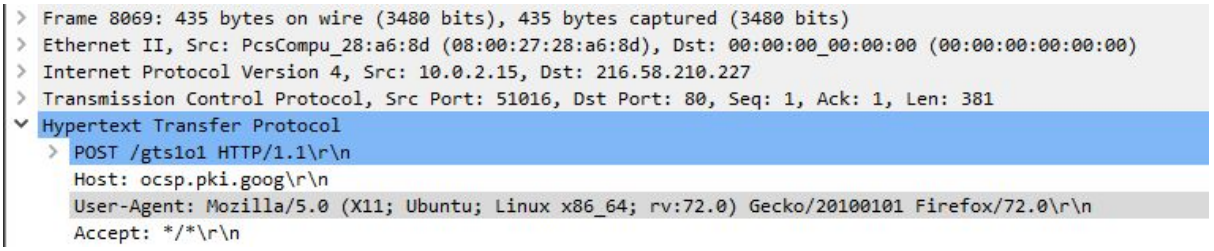

Fig. G showing Firefox 72.0 useragent

It appears that SQLmap was used in order to perform these injections, packet 10610 shows the injection query for the low security setting and it's useragent is "sqlmap 1.2.4#stable" as seen in *Fig. F*. When we examine the Impossible security injection, we can go to packet 9215 in order to see the injection query. This packet differs from the last in it's user agent, the user agent of this query defines it as originating from a Firefox 72.0 browser such as in *Fig. G*. This means that this injection was likely done manually due to the complexity involved in the impossible challenge.

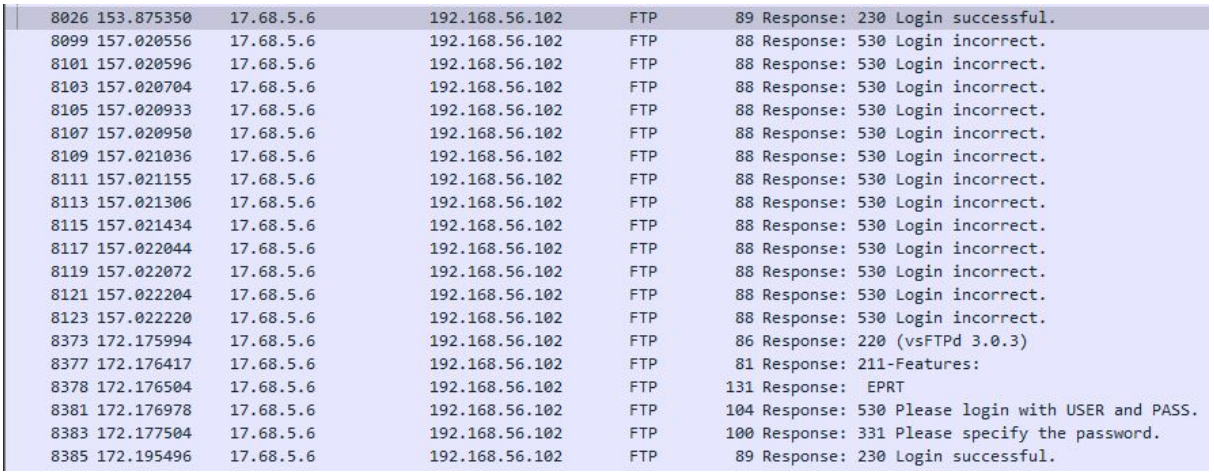

Fig. H showing FTP brute force succeeding

Fig. H shows one of many attempts to log into the ftp server on 17.56.5.6. This is done through a brute force attack that can be seen by a series of password packets being sent to port 21 in succession. First, 192.168.56.102 tries to access the server with no authentication by sending “ls” in packet 6689, the server replied with error 530 asking for a username and password, the attacker then ends the session. Viewing TCP streams 2031 to 2044 shows 192.168.56.102 trying 13 passwords. TCP stream 2043 shows that 192.168.56.102 was authenticated with the username and password both being “admin”. FTPS is not being used so we can see what commands are being executed. TCP stream 2047 shows the commands sent to the FTP server including the authentication process. The client invokes passive mode after every command sent to the server meaning that the client could be behind a firewall with port 21 closed. The client traverses through the directory tree until reaching “/var/www/html/DVWA-master/vulnerabilities/sqli”, they then list the contents of the directory and exit the session. We can see the directory listings of the web server and the root directory of 17.56.5.6 in TCP streams 2048 to 2055.

Sources:

-Shodan. (2020). Retrieved 7 May 2020, from https://www.shodan.io/

-TCP Idle Scan (-sl) | Nmap Network Scanning. (2020). Retrieved 6 May 2020, from https://nmap.org/book/idlescan.html

-How to recognise an Oracle Virtual machine by its MAC address? | FAQ | MAC Address Vendor Lookup. (2020). Retrieved 7 May 2020, from https://macaddress.io/faq/how-to-recognise-an-oracle-virtual-machine-by-its-mac-address

-Changing the MTU size in Windows Vista, 7 or 8 (mtu, speed, windows, Windows 7, Windows 8, Windows Vista). (2020). Retrieved 7 May 2020, from https://support.zen.co.uk/kb/Knowledgebase/Changing-the-MTU-size-in-Windows-Vista-7-or-8

-Riddell, M. (2020). How to set the maximum TCP receive window size in Linux?. Retrieved 7 May 2020, from https://serverfault.com/questions/775837/how-to-set-the-maximum-tcp-receive-window-size-in-linux

-OpenSSH Hide Version Number From Clients - nixCraft. (2020). Retrieved 7 May 2020, from https://www.cyberciti.biz/faq/howto-ssh-server-hide-version-number-sshd_config/

-Kili, A. (2020). How to Hide Apache Version Number and Other Sensitive Info. Retrieved 7 May 2020, from https://www.tecmint.com/hide-apache-web-server-version-information/

-Active Vs Passive Reconnaissance - ASM , Rockville , Maryland. (2020). Retrieved 6 May 2020, from https://asmed.com/active-vs-passive-reconnaissance/

-Banner grabbing. (2020). Retrieved 6 May 2020, from https://en.wikipedia.org/wiki/Banner_grabbing

-Chandel, R. (2020). 5 ways to Banner Grabbing. Retrieved 6 May 2020, from https://www.hackingarticles.in/5-ways-banner-grabbing/

-Custom Error Responses - Apache HTTP Server Version 2.4. (2020). Retrieved 6 May 2020, from https://httpd.apache.org/docs/2.4/custom-error.html

-List of TCP and UDP port numbers. (2020). Retrieved 6 May 2020, from https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers

-TCP SYN (Stealth) Scan (-sS) | Nmap Network Scanning. (2020). Retrieved 6 May 2020, from https://nmap.org/book/synscan.html

-Hacker Geek: OS Fingerprinting With TTL and TCP Window Sizes. (2020). Retrieved 6 May 2020, from https://www.howtogeek.com/104337/hacker-geek-os-fingerprinting-with-ttl-and-tcp-window-sizes/

-Passive vs. Active Reconnaissance. (2020). Retrieved 6 May 2020, from https://medium.com/@jharve08/passive-vs-active-reconnaissance-c2974913237f

-Port Scanning Techniques | Nmap Network Scanning. (2020). Retrieved 6 May 2020, from https://nmap.org/book/man-port-scanning-techniques.html

-system?, H. (2020). How to find the initial values of TTL for your current operating system?. Retrieved 6 May 2020, from https://serverfault.com/questions/749175/how-to-find-the-initial-values-of-ttl-for-your-current-operating-system

-RFC 791 - Internet Protocol. (2020). Retrieved 6 May 2020, from https://tools.ietf.org/html/rfc791

-RFC 793 - Transmission Control Protocol. (2020). Retrieved 6 May 2020, from https://tools.ietf.org/html/rfc793

-Service and Version Detection | Nmap Network Scanning. (2020). Retrieved 6 May 2020, from https://nmap.org/book/man-version-detection.html