Module 3

<<صل علي محمد>>

– حاولت علي اد مقدر اجمع الاوامر الي في الموديول ممكن ابقا نسيت حاجة بس عموما دي معظم الحاجات المهمة

– Reconnaissance types: active, passive
— active => you send probes to the target system
— passive => the opposite (don't send probes ….)

– Passive reconnaissance categories:-

- Domain enumeration
- Packet inspection
- Open-source intelligence (OSINT)
- Recon-ng
- Eavesdropping

1- SpiderFoot tool:-

– spiderfoot -l 127.0.0.1:5001 => open spiderfoot in the localhost port 5001

2- Recon-ng tool:-

** To open the recon-ng terminal just type :

→ recon-ng

** to search for modules

→ modules search

** to load a module

→ modules load "module name"

** After loading a module you have to set the source (the target)

→ options set source "domain ex: google.com"

** To run it just type

→ run

** To provide summary for the results you got

→ dashboard

** To get back from the module you are in just type

→ back

** to list the available workspaces (a space that you're work saved in)

→ workspaces list

** to make a new workspace

→ workspaces create "name"

** to choose an available workspace

→ workspaces load "name"

** investigating the marketplace (something like google play, you install modules from it)

*** to search the marketplace

→ marketplace search "module name"

*** to install a module

→ marketplace install "module name"

– DNS Lookups

Ex1: dnsrecon provide detailed information about the dns servers and its IPs for the target domain

-> dnsrecon -d "domain ex: google.com"

Ex2: dig also provides detailed information but it's not as the deep of dnsrecon, dnsrecon is more deeper.

— By default dig return just the IP4, but you can use options to get more kind of outputs

-> dig google.com

#Notes:

1- A => stands for IP4

2- AAAA => IP6

3- mx => mail server

Another dig command options:-

-> dig google.com -mx

-> dig google.com AAAA

– You can combine them together also

-> dig google.com A google.com AAAA google.com -mx

dig for reflected dns enumeration (instead of giving it the domain name like "google.com", you give it the ip address)

-> dig +x 192.179.1.1

– Another tool is whois, it helps you identify domain technical and administrative contacts

-> whois google.com

-> whois 172.184.8.8 (reflected)

– Another nice tool is host, gives you IPv4, IPv6, and the mail server

→ host google.com

→ host 172.184.8.8 (reflected)

– To get the ssl certificate information of the target domain, we use a tool called sslscan

→ sslscan cisco.com

→ sslscan cisco.com | aha > scan.html

– To get the file metadata, we use exiftool (Exchangeable Image File Tool)

→ exiftool image.jpg

→ exiftool -csv > "path/out.csv" image.jpg

– Active reconnaissance:

- Host enumeration
- Network enumeration
- User enumeration
- Group enumeration
- Network share enumeration
- Web page enumeration
- Application enumeration
- Service enumeration
- Packet crafting

Port scan types:-

– first, let's describe the 3 way handshake process

1- the client send SYN request to the server

2- the server responds with SYN ACK

3- the client then sends ACK to the server

— then, let's dive into nmap tool scan types:-

→ sudo nmap -sS 192.168.1.1

-- As you see, this command needs a root privilege
-- it's called TCP SYN scan
– it's a half TCP scan, and it's stealthy (means it's not noisy)

– if the server is open you receive => SYN-ACK
– if it is closed => RST
– if filtered (firewalled) => No response

→ sudo nmap -sT 192.168.1.1
– Full TCP scan
– noisy
– if the sever is open => TCP SYN-ACK
– closed => TCP RST
– filtered => No response
→ sudo nmap -sU 192.168.1.1

– UDP scan
– it enumerates DNS, SNMP, or DHCP servers
– if the port is opened => Data returned
– closed => ICMP error message received
– filtered/open => No ICMP message received

→ sudo nmap -sF 192.168.1.1
– FIN scan
– You just probing the closed ports
– open => no response
– closed => RST
– filtered => ICMP unreachable error

→sudo nmap -sn 192.168.1.1/24
– Host discovery scan

– To probe the host to see if it's open or not like for example using "ping" but with nmap options
→ sudo nmap -sP 192.168.1.1

Timing options: from 0 to 5

- **-T0 (Paranoid)** : Very slow, used for IDS evasion
- **-T1 (Sneaky)** : Quite slow, used for IDS evasion
- **-T2 (Polite)** : Slows down to consume less bandwidth, runs about 10 times slower than the default
- **-T3 (Normal)** : Default, a dynamic timing model based on target responsiveness
- **-T4 (Aggressive)** : Assumes a fast and reliable network and may overwhelm targets
- **-T5 (Insane)** : Very aggressive; will likely overwhelm targets or miss open ports

User enumeration:
→ nmap --script smp-enum-users.nse 192.168.88.251
Group enumeration
→ nmap --script smp-enum-groups.nse 192.168.99.11

Network share enumeration:

→ nmap --script smp-enum-shares.nse 192.168.99.11

→ nmap -sC 192.168.88.251
– running against linux systems that have Samba

– Another tool to enumerate Samba shares is enum4linux

→ enum4linux 192.168.88.251

Web (page / application) enumeration

→ nmap -sV --script=http-enum -p 80 192.168.88.251
– sV => to get the version of running services includin os,...
– p 80 => port 80

– Another web server enumeration tool is nikto (for fast enumerations or scans)
→ nikto -h 192.168.88.251

Service enumeration
→ nmap --script smb-enum-processes.nse --script-args smbusername=<*username*>,smbpass=<*password*>
-p445 <*host*>

– Another options for nmap:
1) -O => for getting the operating system, need root privilege
2) -A => Enable OS detection, version detection, script scanning, and traceroute
3) --script vulners => to dicover the vulnerabilites

– Packet crafting with Scapy
→ sudo su
– open a terminal with a root privileges, you can omit it with typing sudo before each command
→ scapy
– open scapy terminal
→ sniff()
– means that scapy is listening now for any request (packets sent) to the current domain
→ sniff(iface="lo")

– you are listening for the network interface that called “io” (localhost)

You can use scapy to send packets also not just to listen
→ send(IP(dst=”127.0.0.1”)/ICMP/”You are hacked!”))
– This sent an ICMP message to the 127.0.0.1

– You can listen to those type of messages exactly by using
→ sniff(iface=”lo”,filter=”icmp”,count=10)
– you are now listening for the icmp messages sent for the network interface “lo”, the count means that number of packets is 10, and it’s optional in our case, you can set it as you want

– after finish listening press: CTRL + c , to see the results
– to save the results in a variable:
→ a =_
→ a.summary()

– This a Python code, by the way scapy is a tool written in Python, the first command save the result of the sniff() into the a variable, and then we used this variable to provide summary for the result using summary() function.
– you can also save the summary you made into a pcap file, why pcap? Well this type of files used by tools like wireshark to then anlyze the packets and do more….
→ wrpcap(“capture1.pcap”,a)

– last but not least, let’s send a tcp packet
→ send(IP(dst="127.0.0.1")/TCP(dport=445, flags="S"))

– dport => destination port
– flags=”S” => SYN

For network sniffing (listening), we use tools like WireShark, tcpdump, and tshark