DAY8_advent_of_the_cyber

15

Don't know really how to explain it differently as I think the nmap page explains it ok, but basically...

Some hosts respond to pings if they are online (I.e. ping www.google.com and you get a reply.)

Nmap tries pinging them, if they respond nmap continues scanning and you get your result. If it gets no response it says 'hmm they must be down, well no point wasting time scanning someone who doesn't exist' and exits.

Now some hosts are alive, but configured not to answer to pings. Under normal circumstances this means nmap would simply not scan them thinking they were down. If you specify -Pn it skips this initial stage of checking if the host is up and basically says 'well I've been told to scan no matter what, so even if I think it's down* my stupid human operator is making me do it anyway'. It then tries to scan and actually gets results because in reality the host is alive.

*saying it thinks its down is slightly incorrect - if simply ignores the check in the first place.

Summary - skips checking if the host is alive which may sometimes cause a false positive and stop the scan.

- Pre-engagement Interactions
- Intelligence Gathering
- Threat Modeling
- Vulnerability Analysis
- Exploitation
- Post Exploitation
- Reporting

8.3. Intro to Nmap:

An open-source, extensible, and importantly free tool, Nmap is one of the industry standard's that everyone should have in their toolkit. Nmap is capable of a few things that are essential in the Information Gathering stages of a penetration testing methodology such as the Penetration Testing Execution Standard (PTES), including:

- Host discovery
- Vulnerability discovery
- Service/Application discovery

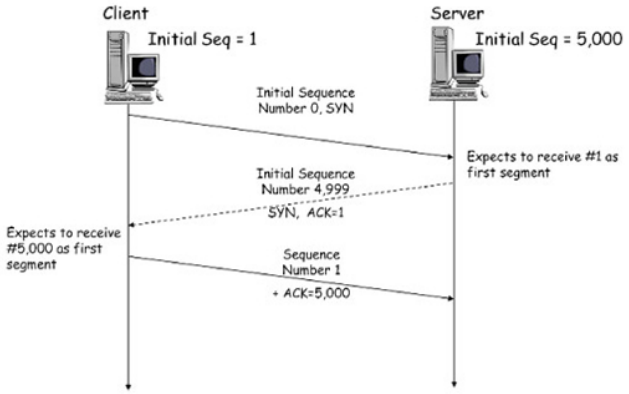
8.4. Basic Nmap Functionality

We'll quickly glaze over the basics of getting started with Nmap, the scan types, and the syntax for these types accordingly. We'll apply our networking knowledge learned yesterday in "Day 7 - The Grinch Really Did Steal Christmas" to help understand the differences between TCP and UDP scanning.

8.4.1 TCP Scanning

There are two common TCP scan types that you'll be using in Nmap. On the surface they seem to perform the same thing, however, they're very different. Before we break this down, let's illustrate TCP/IP's three-way-handshake again and recap the three stages of a "normal" three-way-handshake:

1. SYN
2. SYN/ACK
3. ACK



(The Open University., N.D)

- Connect Scan - `nmap -sT <ip>`
- SYN Scan - `nmap -sS <ip>`

8.4.1.1 SYN Scan:

The most favourable type of scan, Nmap uses the TCP SYN scan (`-sS`) if the scan is run with both administrative privileges and a different type isn't provided. Unlike a connect scan, this scan type doesn't fulfil the "three-way-handshake" as what would normally take place. Instead, after the "SYN/ACK" is received from the remote host, a "RST" packet is sent by the host that we are scanning from (never completing the connection).

This scan type is the most favourable method as Nmap can use all the information gathered throughout the handshake to determine port status based on the response that is given by the IP address that is being scanned:

- SYN/ACK = open
- RST = Closed
- Multiple attempts = filtered

Not only this but also since fewer packets are sent across a network, there is less likelihood of being detected.

8.4.1.2. Connect Scan:

Unlike a SYN scan, administrative privileges aren't required for this scan to run. This is as a result of Nmap using the *Berkeley Sockets API* which has quickly formed to be the standard method of how services like web applications communicate with an operating system. As a result of more packets being sent by Nmap, this scan is easier to detect and takes a longer time to complete. Moreover, as the "three-way-handshake" completes as if it were a normal connection, it can be logged a lot more conveniently.

8.5. Nmap Timing Templates

Nmap allows the user to determine Nmap's performance. Measured in aggressiveness, a user can use one of six profiles [0 to 5] to change the rate at which Nmap scans at. With `-T0` being the stealthiest, this profile scans one port every 5 minutes, whereas `-T5` is considered both the most aggressive and potential to be inaccurate. This is because the `-T5` waits a mere 0.3 seconds for the remote device to respond to a handshake. Factors such as those listed below determine how accurate these scans are:

- The resource usage a remote server is under. The higher usage, the slower it is to send a response to Nmap.
- The quality of the connection. If you have a slow or unstable connection, you are likely to miss responses if you are sending many packets at once.

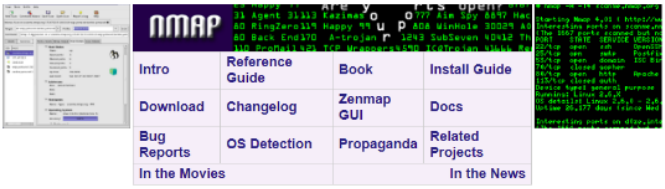
Generally speaking, you will want to use low-aggressive profiles for real-world scenarios, however, in a lab environment where noise doesn't matter - high-aggressive profiles prove to be the quickest. For perspective, Nmap uses `-T3` if no profile is provided. In a pentesting situation, you'd be inclined to use a lower value such as within in a lab environment, a higher value `-T4` will suffice as stealth is not as critical.

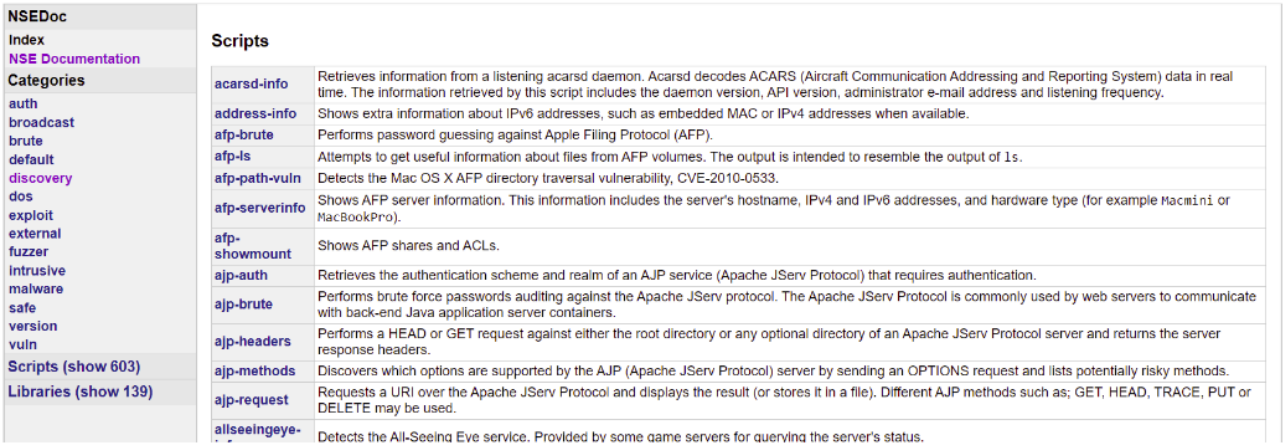
8.6. An Introduction to Nmap's Scripting Engine

A recent addition to Nmap is the Nmap Scripting Engine or NSE for short. This feature introduces a "plug-in" nature to Nmap, where scripts can be used to automate various actions such as:

- Exploitation
- Fuzzing
- Bruteforcing

At the time of writing, the NSE comes with 603 scripts, which can be found here.





Nmaps NSE Documentation Page

Take for example the FTP ProFTPD Backdoor script. This script attempts to exploit an FTP service that is running ProFTPD version 1.3.3c, the version of which is fingerprinted by Nmap itself.

We can provide the script that we want to use by using the `--script` flag in Nmap like so:

```
nmap --script ftp-proftpd-backdoor -p 21 <ip_address>
```

8.7. Additional Scan Types:

Not only can we use the Nmap's TCP Scan, but Nmap also boasts a combination of these types for various actions that are useful to us during the information gathering stage. I have assorted these into the table below, giving a brief explanation of their purpose.

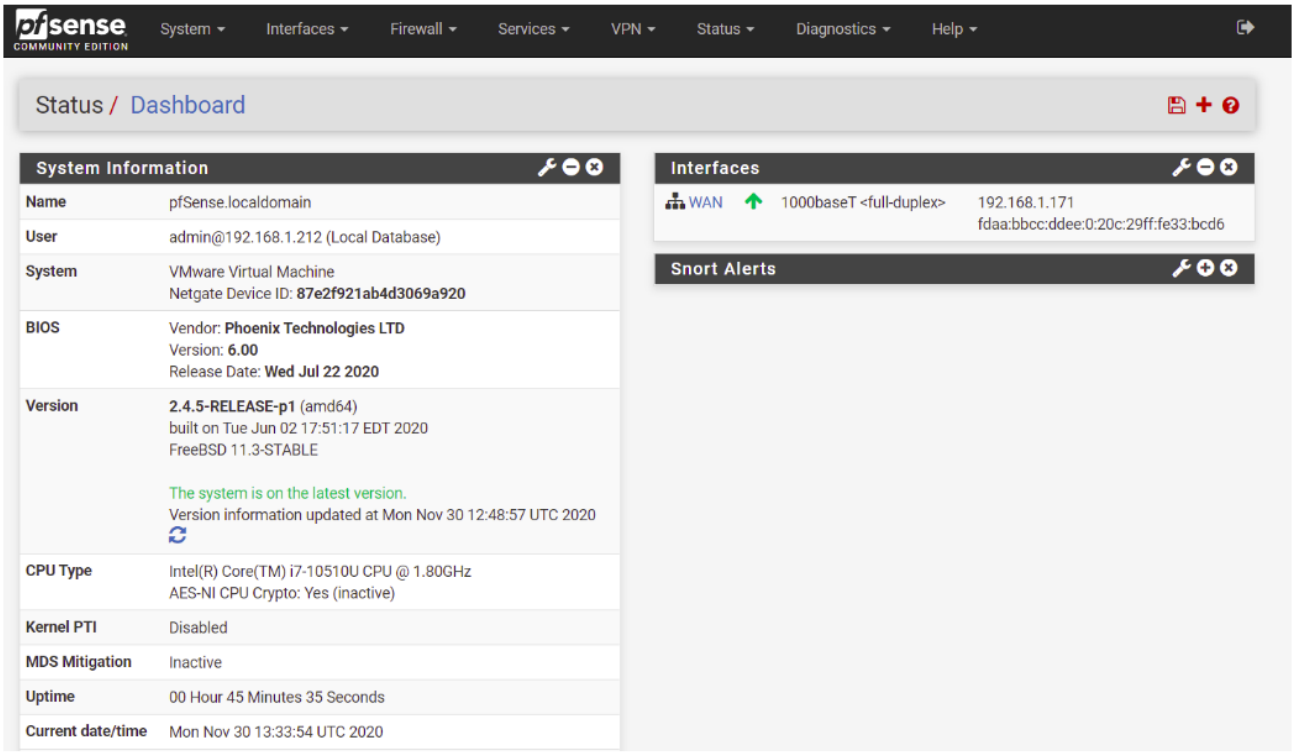
(Where x.x.x.x == 10.10.142.107)

Flag	Usage Example	Description
-A	`nmap -A x.x.x.x`	Scan the host to identify services running by matching against Nmap's database with OS detection
-O	`nmap -O x.x.x.x`	Scan the host to retrieve and perform OS detection
-p	`nmap -p 22 x.x.x.x`	Scan a specific port number on the host. A range of ports can also be provided (i.e. 10-100) by using the first and last value of the range like so: `nmap -p 10-100 x.x.x.x`
-p-	`nmap -p- x.x.x.x`	Scan all ports (0-65535) on the host
-sV	`nmap -sV x.x.x.x`	Scan the host using TCP and perform version fingerprinting

8.8. Defending against Nmap Scans

The practice of security through obscurity doesn't work here. Whilst it may seem logical to attempt to hide a service by changing its port number to something other than the standard (such as changing SSH from port 22 to 2222), the service will still be fingerprinted during an Nmap scan (albeit slightly later on). Unfortunately, you cannot get the best of both worlds in having a service available yet hidden.

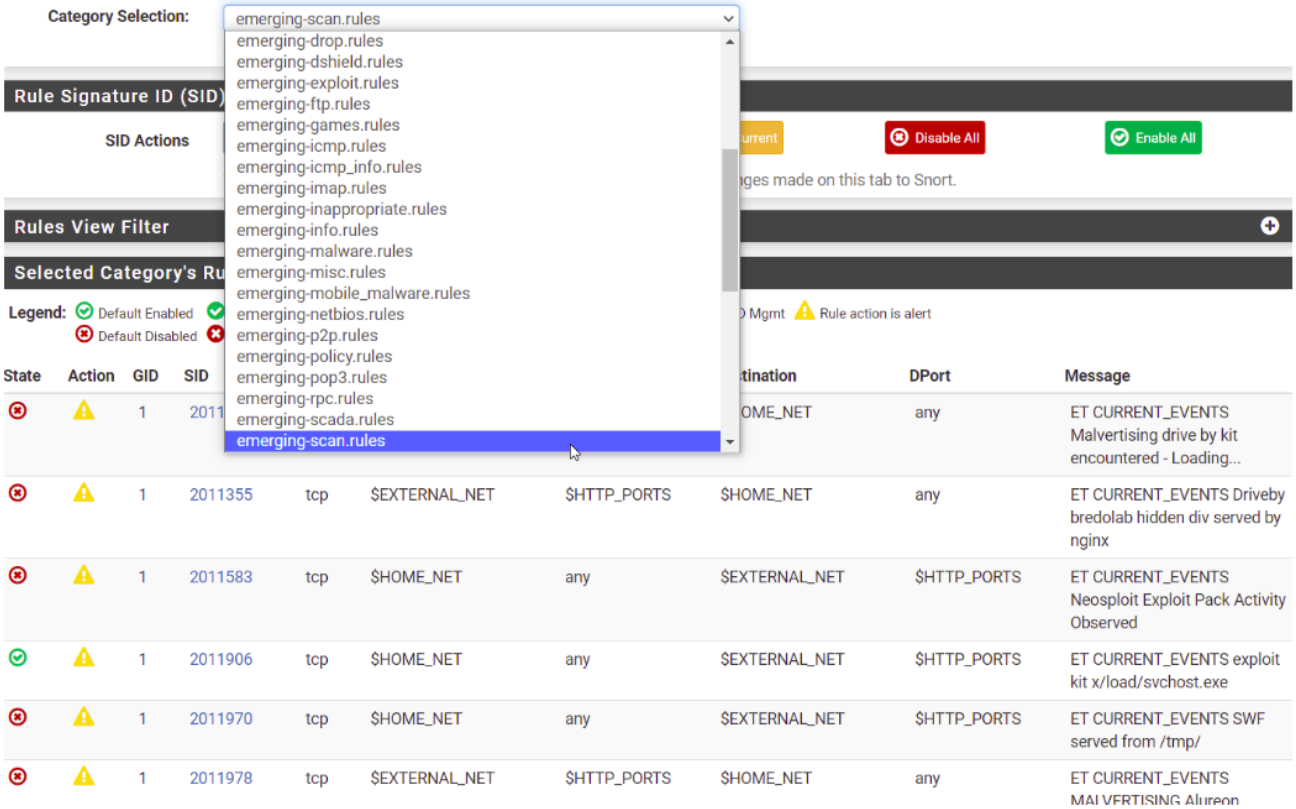
Fortunately, open-source Intrusion Detection (IDS) & Prevention Systems (IPS) such as Snort and Suricata allows blue-teamers to protect their networks using network monitoring. For example, you would install these services on firewalls such as pfSense:



The dashboard of a pfSense Firewall.

Rulesets such as the emerging threats for Snort and Suricata are capable of detecting and blocking a wide variety of potentially malicious traffic - including:

- Malware traffic
- Reverse shells
- Metasploit payloads
- Nmap scans



A list of Snort rules installed on a pfSense firewall.

For example, detecting the Metasploit payload for CVE 2013-3205:

$HTTP_PORTS	$HOME_NET	any	ET EXPLOIT Metasploit CVE-2013-3205 Exploit Specific

The emerging threat rule to detect the Metasploit payload for CVE-2013-3205.

If properly configured, a majority of Nmap scans can be detected. This is especially true when using an aggressive timing template such as `-T4` or `-T5`. Let's take a look at the following Nmap scan being detected: `nmap -A 192.168.1.171`

```
cmnatic@danny ~ nmap -A 192.168.1.171
Starting Nmap 7.80 ( https://nmap.org ) at 2020-11-30 13:52 GMT
Stats: 0:00:13 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 8.40% done; ETC: 13:54 (0:02:33 remaining)
Stats: 0:00:15 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 9.00% done; ETC: 13:54 (0:02:32 remaining)
Stats: 0:00:15 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 9.25% done; ETC: 13:54 (0:02:37 remaining)
```

Starting an Nmap scan to the pfSense firewall.

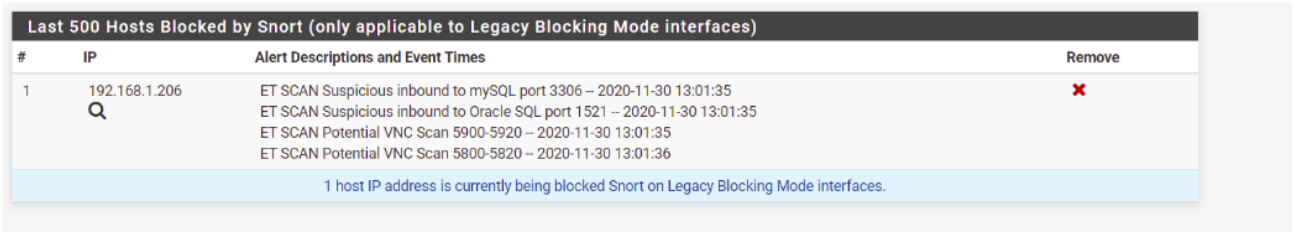
After returning to pfSense a few seconds later, we notice that alerts are being generated by Snort:

6 Entries in Active Log

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2020-11-30 13:01:36		2	TCP	Attempted Information Leak	192.168.1.206	52661	192.168.1.171	5802	1:2002910	ET SCAN Potential VNC Scan 5800-5820
2020-11-30 13:01:35		2	TCP	Attempted Information Leak	192.168.1.206	52662	192.168.1.171	5906	1:2002911	ET SCAN Potential VNC Scan 5900-5920
2020-11-30 13:01:35		2	TCP	Potentially Bad Traffic	192.168.1.206	52662	192.168.1.171	1521	1:2010936	ET SCAN Suspicious inbound to Oracle SQL port 1521
2020-11-30 13:01:35		2	TCP	Potentially Bad Traffic	192.168.1.206	52662	192.168.1.171	3306	1:2010937	ET SCAN Suspicious inbound to mySQL port 3306
2020-11-30 13:01:35		2	TCP	Potentially Bad Traffic	192.168.1.206	52661	192.168.1.171	1521	1:2010936	ET SCAN Suspicious inbound to Oracle SQL port 1521
2020-11-30 13:01:35		2	TCP	Potentially Bad Traffic	192.168.1.206	52661	192.168.1.171	3306	1:2010937	ET SCAN Suspicious inbound to mySQL port 3306

Viewing newly created alerts by Snort as a result of the Nmap scan.

Even with a timing template of `-T3`, Snort is capable of detecting the port scan, where after 6 alerts (in this case) the attacker is then blocked by the firewall.



After 6 alerts, Snort blocks the IP address running the Nmap scan from contacting the pfSense firewall.

```
cmnatic@danny ~ nmap -A 192.168.1.171
Starting Nmap 7.80 ( https://nmap.org ) at 2020-11-30 13:51 GMT
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 3.62 seconds
cmnatic@danny ~
```

Confirming that the IP address running the Nmap scan can no longer contact the pfSense firewall.

exercise:

```
'.
Open 10.10.142.107:80
Open 10.10.142.107:2222
Open 10.10.142.107:3389
^C
```

nmap ‘http-title’ script scan:

```
[kafka@kafka ~]$ sudo nmap --script http-title -sV -p80 10.10.142.107
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-10 03:09 IST
Nmap scan report for tbfc.blog (10.10.142.107)
Host is up (0.52s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: TBFC&#39;s Internal Blog
```

nmap vuln script scan:

```
PORT     STATE SERVICE       VERSION
80/tcp   open  http          Apache httpd 2.4.29 ((Ubuntu))
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-enum:
|   /css/: Potentially interesting directory w/ listing on 'apache/2.4.29 (ubuntu)'
|   /images/: Potentially interesting directory w/ listing on 'apache/2.4.29 (ubuntu)'
|   /js/: Potentially interesting directory w/ listing on 'apache/2.4.29 (ubuntu)'
|   /page/: Potentially interesting directory w/ listing on 'apache/2.4.29 (ubuntu)'
|_  /src/: Potentially interesting directory w/ listing on 'apache/2.4.29 (ubuntu)'
| http-internal-ip-disclosure:
|_  Internal IP Leaked: 10
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
2222/tcp open  ssh           OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
3389/tcp open  ms-wbt-server xrdp
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
|_rdp-vuln-ms12-020: ERROR: Script execution failed (use -d to debug)
|_ssl-ccs-injection: No reply from server (TIMEOUT)
|_sslv2-drown:
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```