Module 10: Scanning and Enumeration

INSTALLING KIOPTRIX

A VULNERABLE VIRTUAL MACHINE

Kioptrix lvl had a big issue with the ping being blocked. I found a fix here:

https://www.youtube.com/watch?v=p2GGsmUxG2o

lvl 1 Login/Password - john/TwoCows2

arp-scan is an alternative to nmap

What is an arp packet???

Short for Address Resolution Protocol, a network layer protocol used to convert an IP address into a physical address (called a DLC address), such as an Ethernet address. ... There is also Reverse ARP (RARP)which can be used by a host to discover its IP address.

Scanning with nmap

nmap = network mapper (Stealth Scanning)

nmap -T4 -p- -A 172.16.4.132 ##can be any IP Address

-T4 - speed 1-5
-p- - Scan all ports (65,535 ports out there)
-A - Everything (Scans everything)

samba???
What Are Ports 139 And 445? SMB has always been a network file sharing protocol. As such, SMB requires network ports on a computer or server to enable communication to other systems. SMB uses either IP port 139 or 445.

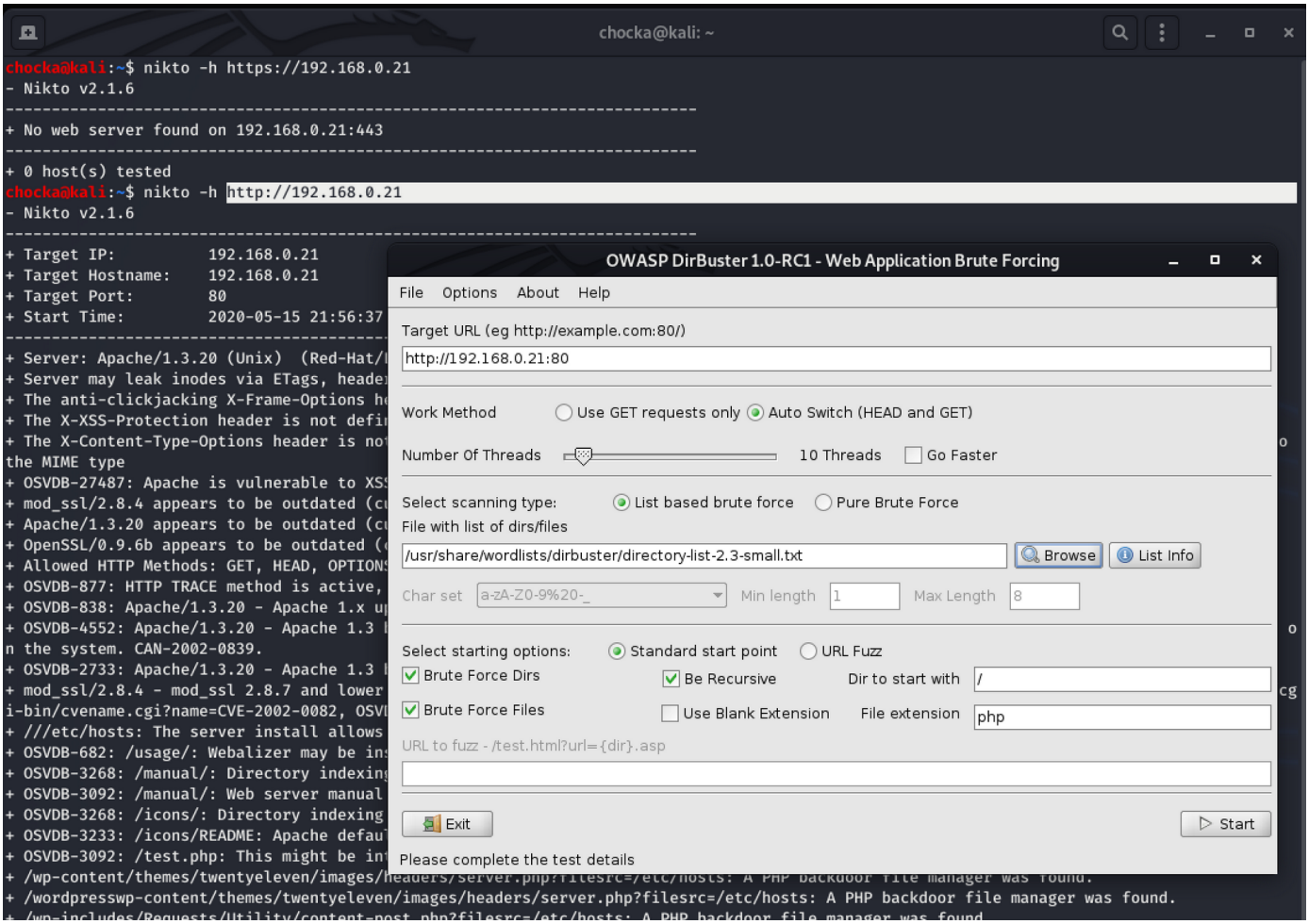
Smbd???
smbd is the server daemon that provides filesharing and printing services to Windows clients. The server provides filespace and printer services to clients using the SMB (or CIFS) protocol. This is compatible with the LanManager protocol, and can service LanManager clients.

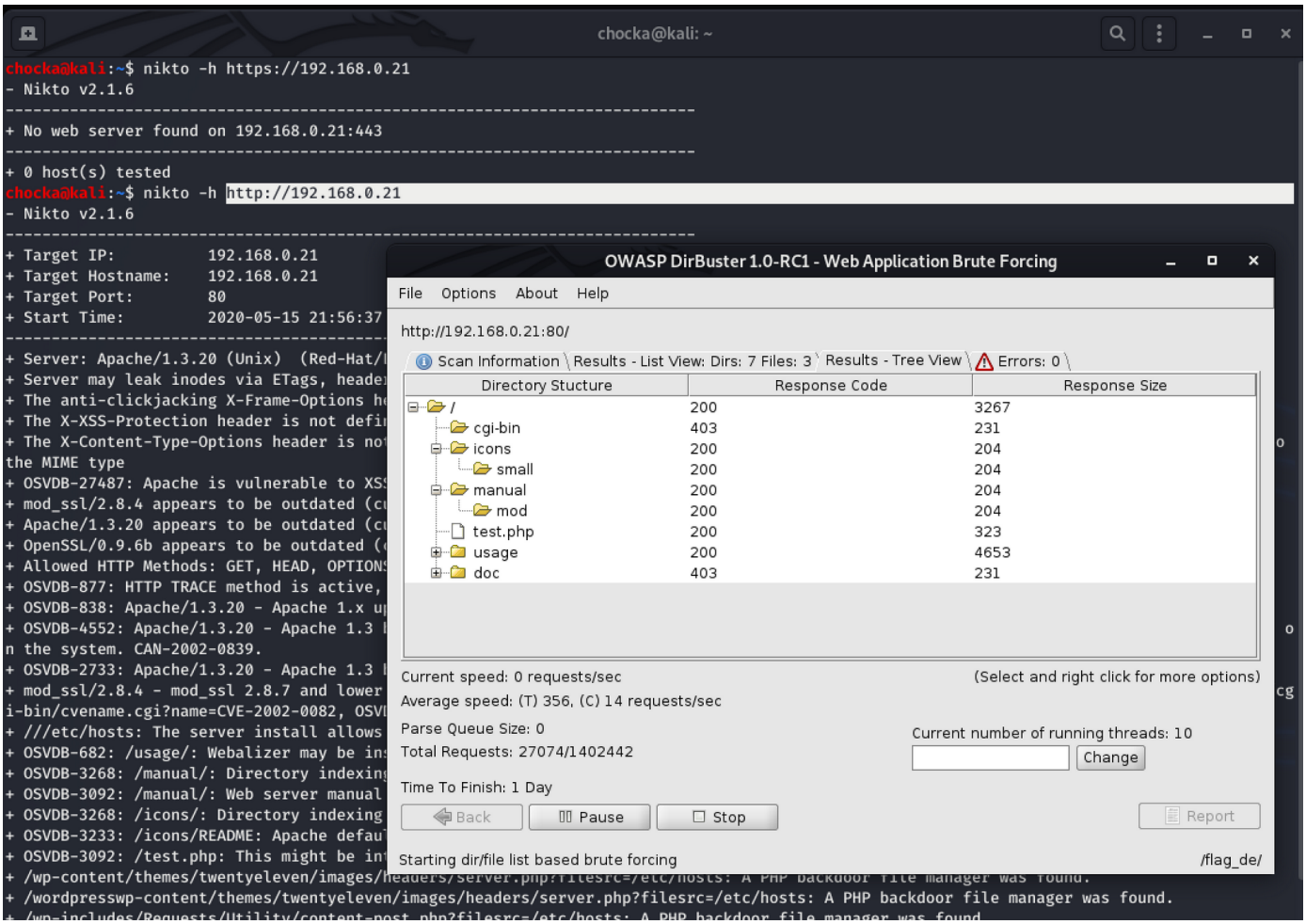
STEPS: (See Kioptrix 1 File for findings and notes)
1.arp-scan -l
2.nmap -T4 -p- -A (found ip from step 1)
3.Look at open ports, what is running on them and how we can exploit them
4.Determine plan of attack. smbd (samba) is historically back, websites are historically bad. Ports 80,139,443
5.Explore port 80/443 by copying the address into a web browser.
*Note: If a test page comes up from the address that is an indication of poor hygiene and it is considered a finding and should be a write up on a test.
**Example of a finding note:

80/443 - http://192.168.0.21/ - 9:51pm 5/15/2020
Default webpage - Apache - PHP (Include screnshot)

6.Vulnerability Scanning using nikto (web vulnerability scanner)
nikto -h(host) Web address
Notate outdated services

7.Directory Busting (Tools: dirbuster, dirb, gobuster)
*Response Codes:
-200s: okay
-400s: some kind of error
-300s: redirect
-500s: server errors/other





8.Look in the websites source code for random passwords, keys, etc

NMAP --help

##

Nmap 7.80 (https://nmap.org)

Usage: nmap [Scan Type(s)] [Options] {target specification}

TARGET SPECIFICATION:

Can pass hostnames, IP addresses, networks, etc.

Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254

-iL : Input from list of hosts/networks

-iR : Choose random targets

--exclude <host1[,host2][,host3],...>: Exclude hosts/networks

--excludefile <exclude_file>: Exclude list from file

HOST DISCOVERY:
-sL: List Scan - simply list targets to scan
-sn: Ping Scan - disable port scan
-Pn: Treat all hosts as online -- skip host discovery
-PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
-PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
-PO[protocol list]: IP Protocol Ping
-n/-R: Never do DNS resolution/Always resolve [default: sometimes]
--dns-servers <serv1[,serv2],...>: Specify custom DNS servers
--system-dns: Use OS's DNS resolver
--traceroute: Trace hop path to each host

SCAN TECHNIQUES:
-sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
-sU: UDP Scan
-sN/sF/sX: TCP Null, FIN, and Xmas scans
--scanflags : Customize TCP scan flags
-sI <zombie host[:probeport]>: Idle scan
-sY/sZ: SCTP INIT/COOKIE-ECHO scans
-sO: IP protocol scan
-b : FTP bounce scan

PORT SPECIFICATION AND SCAN ORDER:
-p : Only scan specified ports
Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9
--exclude-ports : Exclude the specified ports from scanning
-F: Fast mode - Scan fewer ports than the default scan
-r: Scan ports consecutively - don't randomize
--top-ports : Scan most common ports
--port-ratio : Scan ports more common than

SERVICE/VERSION DETECTION:
-sV: Probe open ports to determine service/version info
--version-intensity : Set from 0 (light) to 9 (try all probes)
--version-light: Limit to most likely probes (intensity 2)
--version-all: Try every single probe (intensity 9)
--version-trace: Show detailed version scan activity (for debugging)

SCRIPT SCAN:
-sC: equivalent to --script=default
--script=: is a comma separated list of

directories, script-files or script-categories
--script-args=<n1=v1,[n2=v2,...]>: provide arguments to scripts
--script-args-file=filename: provide NSE script args in a file
--script-trace: Show all data sent and received
--script-updatedb: Update the script database.
--script-help=: Show help about scripts.
is a comma-separated list of script-files or
script-categories.
#
OS DETECTION:
-O: Enable OS detection
--osscan-limit: Limit OS detection to promising targets
--osscan-guess: Guess OS more aggressively

TIMING AND PERFORMANCE:
Options which take are in seconds, or append 'ms' (milliseconds),
's' (seconds), 'm' (minutes), or 'h' (hours) to the value (e.g. 30m).
-T<0-5>: Set timing template (higher is faster)
--min-hostgroup/max-hostgroup : Parallel host scan group sizes
--min-parallelism/max-parallelism : Probe parallelization
--min-rtt-timeout/max-rtt-timeout/initial-rtt-timeout : Specifies
probe round trip time.
--max-retries : Caps number of port scan probe retransmissions.
--host-timeout : Give up on target after this long
--scan-delay/--max-scan-delay : Adjust delay between probes
--min-rate : Send packets no slower than per second
--max-rate : Send packets no faster than per second

FIREWALL/IDS EVASION AND SPOOFING:
-f; --mtu : fragment packets (optionally w/given MTU)
-D <decoy1,decoy2[,ME],...>: Cloak a scan with decoys
-S <IP_Address>: Spoof source address
-e : Use specified interface
-g/--source-port : Use given port number
--proxies <url1,[url2],...>: Relay connections through HTTP/SOCKS4 proxies
--data : Append a custom payload to sent packets
--data-string : Append a custom ASCII string to sent packets
--data-length : Append random data to sent packets
--ip-options : Send packets with specified ip options
--ttl : Set IP time-to-live field
--spoof-mac <mac address/prefix/vendor name>: Spoof your MAC address
--badsum: Send packets with a bogus TCP/UDP/SCTP checksum

OUTPUT:
-oN/-oX/-oS/-oG : Output scan in normal, XML, sl<rIpt kIddi3,
and Grepable format, respectively, to the given filename.
-oA : Output in the three major formats at once
-v: Increase verbosity level (use -vv or more for greater effect)
-d: Increase debugging level (use -dd or more for greater effect)
--reason: Display the reason a port is in a particular state
--open: Only show open (or possibly open) ports
--packet-trace: Show all packets sent and received
--iflist: Print host interfaces and routes (for debugging)
--append-output: Append to rather than clobber specified output files
--resume : Resume an aborted scan
--stylesheet <path/URL>: XSL stylesheet to transform XML output to HTML
--webxml: Reference stylesheet from Nmap.Org for more portable XML
--no-stylesheet: Prevent associating of XSL stylesheet w/XML output

---

MISC:
-6: Enable IPv6 scanning
-A: Enable OS detection, version detection, script scanning, and traceroute
--datadir : Specify custom Nmap data file location
--send-eth/--send-ip: Send using raw ethernet frames or IP packets
--privileged: Assume that the user is fully privileged
--unprivileged: Assume the user lacks raw socket privileges
-V: Print version number
-h: Print this help summary page.

EXAMPLES:
nmap -v -A scanme.nmap.org
nmap -v -sn 192.168.0.0/16 10.0.0.0/8
nmap -v -iR 10000 -Pn -p 80
##
##############

ENUMERATING SMB

SMB is a fileshare (Most often in the workplace)
-look for version information

-Try to connect to the machine.

TOOL (METASPLOIT)
Search smb
Metasploit is an exploitation framework.
smb -RHOSTS - TARGET ADDRESS/VICTIM
-THREADS

Unix (Samba 2.2.1a)

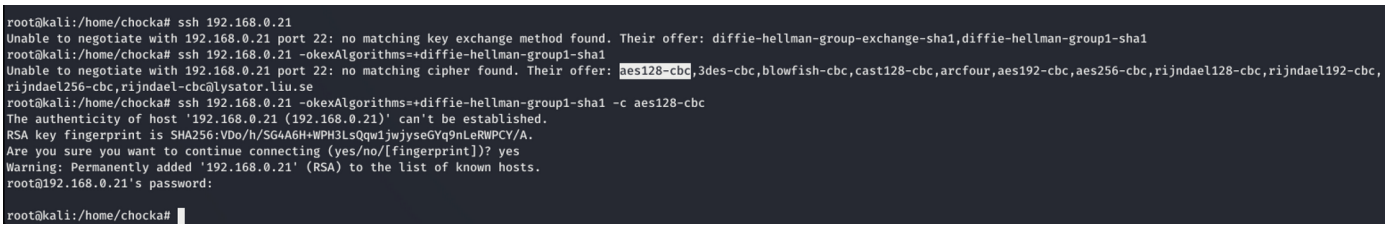
After using the victim IP with RHOSTS to find the smb version use *SMBCLIENT* tool to try to access the fileshare.

Syntax: smbclient -L \\IP.IP.IP.IP\

ENUMERATING SSH

OpenSSH 2.9p2

There is nothing here. However, sometimes using this command will show us a banner that may give away version information and/or who the version is made by etc.



RESEARCHING VULNERABLITIES

##########
Current Notes on Kioptrix lvl 1:

80/443 - 192.168.0.21 - 8:30pm
Default webpage - Apache - PHP
Information Disclosure - 404 page
Information Disclosure - server headers disclose version information

80/tcp open http Apache httpd 1.3.20 (Unix) (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b)

mod_ssl/2.8.7 and lower are vulnerable to a remote buffer overflow which may allow a remote shell. http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2002-0082. OSVDB-756.

SMB
Unix (Samba 2.2.1a)

Webalizer Version 2.01 - http://192.168.0.21/usage/usage_211.html

SSH
OpenSSH 2.9p2

Vulnerabilities:

Look for Rapid 7 (metasploit creator) or exploit DB

80/443 - potentially vulnerable to OpenFuck (https://www.exploit-db.com/exploits/764), https://github.com/heltonWernik/OpenLuck

139- potentially vulnerable to trans2open (https://www.rapid7.com/db/modules/exploit/linux/samba/trans2open) ,(https://www.exploit-db.com/exploits/7), (https://www.exploit-db.com/exploits/10)
##########
TOOLS:
Search Sploit
##########

```
chocka@kali:~$ sudo su
[sudo] password for chocka:
root@kali:/home/chocka# searchsploit Samba 2.2.1a
-------------------------------------------------------------------------------------------------
-------------------------------------------------------------- ----------------------------
-----
 Exploit Title
                                                              |  Path
-------------------------------------------------------------------------------------------------
-------------------------------------------------------------- ----------------------------
-----
Samba 2.2.0 < 2.2.8 (OSX) - trans2open Overflow (Metasploit)
                                                              | osx/remote/9924.rb
Samba < 2.2.8 (Linux/BSD) - Remote Code Execution
                                                              | multiple/remote/10.c
Samba < 3.0.20 - Remote Heap Overflow
                                                              | linux/remote/7701.txt
Samba < 3.6.2 (x86) - Denial of Service (PoC)
                                                              | linux_x86/dos/36741.py
-------------------------------------------------------------------------------------------------
-------------------------------------------------------------- ----------------------------
-----
Shellcodes: No Results
root@kali:/home/chocka#
```

What is a buffer overflow?

Buffers are memory storage regions that temporarily hold data while it is being transferred from one location to another. A buffer overflow (or buffer overrun) occurs when the volume of data exceeds the storage capacity of the memory buffer. As a result, the program attempting to write the data to the buffer overwrites adjacent memory locations.

For example, a buffer for log-in credentials may be designed to expect username and password inputs of 8 bytes, so if a transaction involves an input of 10 bytes (that is, 2 bytes more than expected), the program may write the excess data past the buffer boundary.

Buffer overflows can affect all types of software. They typically result from malformed inputs or failure to allocate enough space for the buffer. If the transaction overwrites executable code, it can cause the program to behave unpredictably and generate incorrect results, memory access errors, or crashes.
buffer overflow

Buffer overflow example

What is a Buffer Overflow Attack

Attackers exploit buffer overflow issues by overwriting the memory of an application. This changes the execution path of the program, triggering a response that damages files or exposes private information. For example, an attacker may introduce extra code, sending new instructions to the application to gain access to IT systems.

If attackers know the memory layout of a program, they can intentionally feed input that the buffer cannot store, and overwrite areas that hold executable code, replacing it with their own code. For example, an attacker can overwrite a pointer (an object that points to another area in memory) and point it to an exploit payload, to gain control over the program.

Types of Buffer Overflow Attacks

Stack-based buffer overflows are more common, and leverage stack memory that only exists during the execution time of a function.

Heap-based attacks are harder to carry out and involve flooding the memory space allocated for a program beyond memory used for current runtime operations.

What Programming Languages are More Vulnerable?

C and C++ are two languages that are highly susceptible to buffer overflow attacks, as they don't have built-in safeguards against overwriting or accessing data in their memory. Mac OSX, Windows, and Linux all use code written in C and C++.

Languages such as PERL, Java, JavaScript, and C# use built-in safety mechanisms that minimize the likelihood of buffer overflow.

How to Prevent Buffer Overflows

Developers can protect against buffer overflow vulnerabilities via security measures in their code, or by using languages that offer built-in protection.

In addition, modern operating systems have runtime protection. Three common protections are:

-Address space randomization (ASLR)—randomly moves around the address space locations of data regions. Typically, buffer overflow attacks need to know the locality of executable code, and randomizing address spaces makes this virtually impossible.
-Data execution prevention—flags certain areas of memory as non-executable or executable, which stops an attack from running code in a non-executable region.
-Structured exception handler overwrite protection (SEHOP)—helps stop malicious code from attacking Structured Exception Handling (SEH), a built-in system for managing hardware and software exceptions. It thus prevents an attacker from being able to make use of the SEH overwrite exploitation technique. At a functional level, an SEH overwrite is achieved using a stack-based buffer overflow to overwrite an exception registration record, stored on a thread's stack.

Security measures in code and operating system protection are not enough. When an organization discovers a buffer overflow vulnerability, it must react quickly to patch the affected software and make sure that users of the software can access the patch.