Scanning & Enumeration

Iker M. Canut

July 18, 2020

1 Install a Vulnerable VM (Kioptix)

To start off, a Kioptix Level 1, from Vulnhub, should work fine. Vulnhub is a webpage where you could find a lot of vulnerable machines, with their level, flags, operating systems... It's like a puzzle.

First, you download the rar and unzip it. Then open it with VMWare or VBox. Configurations:

- Network Settings: **NAT**: Used to share the host's IP address.
- Memory: **256 MB** should be fine, we're gonna run it in the background.
- In the **Kioptix Level 1.vmx** we need to change ethernet0.networkName = "Bridged" to "nat".
- Power it on, and select **I copied it**.

2 Starting with Nmap

Once we have the OS up and running, and we get to know the IP, we could start using Nmap: e.g. **nmap -T4 -p- -A [IP]**. -A stands for *All*: Fingerprinting, OS...

- **-sS**: TCP
- **-sU**: UDP

BUT, if we run **nmap -sU -T4 -p- -A [IP]** it's going to take hours. Because it is a connection-less protocol, we do not have that instant response. We should run **nmap -sU -T4 -p [IP]**.

Also, typically it is much faster to run a **-p-** rather than a **-A**.

3 Results

The first thing we notice are the open ports:

- **22/TCP**: OpenSSH 2.9p2
- **80/TCP**: Apache httpd 1.3.20
- **111/TCP**: Samba
- **139/TCP**: Samba
- **443/TCP**: Apache httpd 1.3.20
- **32768/TCP**:

And the system information:

- Running **Linux 2.4.X**

4 Possible exploits

The ports that should light you up are *"80 and 443"* and *"111 and 139"*. Those are commonly found with exploits: There are a lot of SMB exploits (e.g Wannacry, MS 17.0.1.0 aka Eternal-Blue). SMB and Websites have been historically BAD. SSH is not that bad, we could try to brute force it, try default credentials... But there is not much to do, it is hard to get a shell back. It is not common to attack SSH because it is not a low hanging fruit, and that is what we are after.

4.1 Investigation 80 and 443

First things first, go to the websites, why not? You can try the **[ip]** and **https://[ip]**, for ports 80 and 443 are open. In this case, we have a **default webpage**. This is an *automatic finding*. It is not exploitable by default, but it tells us a little bit about the architecture that's running behind the scenes and a little bit about the client's potential hygine:

- Apache
- Redhat Linux

This should bring us a question: Are there other web directories behind this? **Directory busting**.

If we click on the documentation button, we see that it really is a *bad link*, because it didn't find anything. Also, this is called *information disclosure* because it tell us the version of Apache: 1.3.20 and a name: *kioptrix.level*1. Moreover, we can get naming conventions on their internal network. Take a screenshot and add it to your notes.

4.1.1 Nikto

Nikto is a web vulnerability scanner. It is good for begginers. If a client has really good security (not common), you might run into some issues and get auto blocked (firewall).

In this example, check **nikto -h http://[ip]**. This is detecting that the server is Apache/1.3.20, mod_ssl/2.8.4, OpenSSL/0.9.6b. It also tells us some vulnerabilities back: like for example ETags, anti-clickjacking X-Frame-Options... On an external penetration test, that is not really important, but if we're doing a web penetration test, that becomes more interesting. Now we should focus on Apache, mod_ssl and OpenSSL, for they appear to be outdated. This is another discovery. We got an Apache/1.3.20 in the system, and we got AT LEAST Apache/2.4.37 available, so maybe that's gold. Also it is vulnerable to a remote DoS, possible code execution, vulnerable to overflows, vulnerable to a **remote** buffer overflow (the remote is important, may allow a remote shell). TRACE is allowed (exploitable through XST).

SAVE ALL THE SCANS AND HAVE THEM AVAILABLE

5 Researching Potential Vulnerabilities

Recopying the notes, we have this:

- Default webpage - Apache - PHP
- Information Disclosure - 404 page
- Information Disclosure - Server headers (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/2.8.4 to 2.8.7 - Vulnerable to a remote buffer overflow, may allow a remote shell.
- SMB Unix (Samba 2.2.1a) (juicy)
- Webalizer Version 2.01 (not juicy)
- SSH OpenSSH 2.9p2 (not juicy)

First, we can google "mod_ssl 2.8.4 exploit". Some good pages are exploit-db and rapid7. Then "Apache 1.3.20 exploit". In cvedetails, just look at the score. Red is "good".

Vulnerabilities:

- **80/443 - Potentially vulnerable to OpenLuck**.
  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/22469

. . . . . .

To do this locally, you can just **searchsploit Samba 2.2** (do NOT be too specific). Then look at the path. E.g, osx is not the os. It's Unix/Linux. Also, remote is HUGE.