The Bandit WarGame

Website:

<https://overthewire.org/wargames/bandit/>

1. Logged into the server

(base) basundharachakrabarty@Basundharas-MacBook-Air ~ % ssh bandit0@bandit.labs.overthewire.org -p 2220

The authenticity of host '[bandit.labs.overthewire.org]:2220 ([176.9.9.172]:2220)' can't be established.

ECDSA key fingerprint is SHA256:98UL0ZWr85496EtCRkKlo20X3OPnyPSB5tB5RPbhczc.

Are you sure you want to continue connecting (yes/no/[fingerprint])? yes

Warning: Permanently added '[bandit.labs.overthewire.org]:2220,[176.9.9.172]:2220' (ECDSA) to the list of known hosts.

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

bandit0@bandit.labs.overthewire.org's password:

Linux bandit.otw.local 5.4.8 x86\_64 GNU/Linux

1. Ran a cat ./<filename>

**bandit0@bandit**:**~**$ cat readme

boJ9jbbUNNfktd78OOpsqOltutMc3MY1

(base) basundharachakrabarty@Basundharas-MacBook-Air ~ % ssh bandit1@bandit.labs.overthewire.org -p 2220

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

bandit1@bandit.labs.overthewire.org's password:

Linux bandit.otw.local 5.4.8 x86\_64 GNU/Linux

**bandit1@bandit**:**~**$ cat ./-

CV1DtqXWVFXTvM2F0k09SHz0YwRINYA9

2. Ran the cat command with quotes to get the password

(base) basundharachakrabarty@Basundharas-MacBook-Air ~ % ssh bandit2@bandit.labs.overthewire.org -p 2220

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

bandit2@bandit.labs.overthewire.org's password:

Linux bandit.otw.local 5.4.8 x86\_64 GNU/Linux

**bandit2@bandit**:**~**$ cat 'spaces in this filename'

UmHadQclWmgdLOKQ3YNgjWxGoRMb5luK

3. Ran ls -al to see all hidden files

(base) basundharachakrabarty@Basundharas-MacBook-Air ~ % ssh bandit3@bandit.labs.overthewire.org -p 2220

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

bandit3@bandit.labs.overthewire.org's password:

Linux bandit.otw.local 5.4.8 x86\_64 GNU/Linux

**bandit3@bandit**:**~**$ cd inhere/

**bandit3@bandit**:**~/inhere**$ ls

**bandit3@bandit**:**~/inhere**$ ls -al

total 12

drwxr-xr-x 2 root root 4096 May 7 2020 **.**

drwxr-xr-x 3 root root 4096 May 7 2020 **..**

-rw-r----- 1 bandit4 bandit3 33 May 7 2020 .hidden

**bandit3@bandit**:**~/inhere**$ cat .hidden

pIwrPrtPN36QITSp3EQaw936yaFoFgAB

4.

(base) basundharachakrabarty@Basundharas-MacBook-Air ~ % ssh bandit4@bandit.labs.overthewire.org -p 2220

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

bandit4@bandit.labs.overthewire.org's password:

Linux bandit.otw.local 5.4.8 x86\_64 GNU/Linux

**bandit4@bandit**:**~/inhere**$ file ./-\*

./-file00: data

./-file01: data

./-file02: data

./-file03: data

./-file04: data

./-file05: data

./-file06: data

./-file07: ASCII text

./-file08: data

./-file09: data

**bandit4@bandit**:**~/inhere**$ cat ./-file07

koReBOKuIDDepwhWk7jZC0RTdopnAYKh

5.

**bandit5@bandit**:**~/inhere**$ find . -type f -size 1033c ! -executable -exec file {} + | grep ASCII

./maybehere07/.file2: ASCII text, with very long lines

**bandit5@bandit**:**~/inhere**$ cd maybehere07/

**bandit5@bandit**:**~/inhere/maybehere07**$ ls

**-file1** -file2 **-file3** **spaces file1** spaces file2 **spaces file3**

**bandit5@bandit**:**~/inhere/maybehere07**$ cat .file2

DXjZPULLxYr17uwoI01bNLQbtFemEgo7

6.

**bandit6@bandit**:**~**$ find / -size 33c -group bandit6 -user bandit7

find: ‘/root’: Permission denied

find: ‘/home/bandit28-git’: Permission denied

find: ‘/home/bandit30-git’: Permission denied

find: ‘/home/bandit5/inhere’: Permission denied

.

.

.

**bandit6@bandit**:**~**$ find / -size 33c -group bandit6 -user bandit7 2>/dev/null

/var/lib/dpkg/info/bandit7.password

I then used ‘2>/dev/null’ to redirect STDOUT errors like “Permission denied” to /dev/null. /dev/null is a special file that’s present in every single Linux system, Whatever you write to /dev/null will be discarded, forgotten into the void. It’s known as the null device in a UNIX system.

**bandit6@bandit**:**~**$ cat /var/lib/dpkg/info/bandit7.password

HKBPTKQnIay4Fw76bEy8PVxKEDQRKTzs

7.

A simple grep gave the password:

**bandit7@bandit**:**~**$ cat data.txt | grep -C 1 millionth

Halley H7Mg53D6bPDpleFYGp1KF1SKTQh7jiNl

millionth cvX2JJa4CFALtqS87jk27qwqGhBM9plV

shied OfMT7PpeOvra4NWlZz7JOzyjL236NFVF

**bandit7@bandit**:**~**$ cat data.txt | grep -i millionth

millionth cvX2JJa4CFALtqS87jk27qwqGhBM9plV

8. I used the sort command to sort the lines lexicographically:

**bandit8@bandit**:**~**$ sort data.txt | more

07KC3ukwX7kswl8Le9ebb3H3sOoNTsR2

07KC3ukwX7kswl8Le9ebb3H3sOoNTsR2

07KC3ukwX7kswl8Le9ebb3H3sOoNTsR2

07KC3ukwX7kswl8Le9ebb3H3sOoNTsR2

.

.

I then used uniq along with sort to get the unique lines. The -u extension can then be used to retrieve only the unique values:

**bandit8@bandit**:**~**$ sort data.txt | uniq -u

UsvVyFSfZZWbi6wgC7dAFyFuR6jQQUhR

9.

I used the strings command to get the strings in the file data.txt, and grep-ed with ‘==’ to filter out strings with multiple =s:

**bandit9@bandit**:**~**$ strings data.txt | grep -i "=="

========== the\*2i"4

========== password

Z)========== is

&========== truKLdjsbJ5g7yyJ2X2R0o3a5HQJFuLk

10.

I used the base64 command to decode the file with a -d extension:

**bandit10@bandit**:**~**$ base64 -d data.txt

The password is IFukwKGsFW8MOq3IRFqrxE1hxTNEbUPR

11.

The task is to find the password where the password characters have been substituted by Rot13 values. We can use the tr command to retrieve the password as follows:

**bandit11@bandit**:**~**$ cat data.txt | tr '[a-z]' '[n-za-m]' | tr '[A-Z]' '[N-ZA-M]'

The password is 5Te8Y4drgCRfCx8ugdwuEX8KFC6k2EUu

12.

**bandit12@bandit**:**/tmp/basu**$ xxd -r data.txt > output

**bandit12@bandit**:**/tmp/basu**$ file output

output: gzip compressed data, was "data2.bin", last modified: Thu May 7 18:14:30 2020, max compression, from Unix

**bandit12@bandit**:**/tmp/basu**$ gunzip output

gzip: output: unknown suffix -- ignored

**bandit12@bandit**:**/tmp/basu**$ cp output output.gzip

**bandit12@bandit**:**/tmp/basu**$ cp output output.gz

**bandit12@bandit**:**/tmp/basu**$ gunzip output.gz

gzip: output already exists; do you wish to overwrite (y or n)? y

**bandit12@bandit**:**/tmp/basu**$ ls

data.txt output output.gzip

**bandit12@bandit**:**/tmp/basu**$ rm -rf output.gzip

**bandit12@bandit**:**/tmp/basu**$ ls

data.txt output

**bandit12@bandit**:**/tmp/basu**$ ls

data.txt output

**bandit12@bandit**:**/tmp/basu**$ file output

output: bzip2 compressed data, block size = 900k

**bandit12@bandit**:**/tmp/basu**$ bzip2 -d output

bzip2: Can't guess original name for output -- using output.out

**bandit12@bandit**:**/tmp/basu**$ ls

data.txt output.out

**bandit12@bandit**:**/tmp/basu**$ file output.out

output.out: gzip compressed data, was "data4.bin", last modified: Thu May 7 18:14:30 2020, max compression, from Unix

**bandit12@bandit**:**/tmp/basu**$ cp output.out output.out.gz

**bandit12@bandit**:**/tmp/basu**$ gunzip output.out.gz

gzip: output.out already exists; do you wish to overwrite (y or n)? y

**bandit12@bandit**:**/tmp/basu**$ ls

data.txt output.out

**bandit12@bandit**:**/tmp/basu**$ tar -xvf output.out

data5.bin

**bandit12@bandit**:**/tmp/basu**$ file data5.bin

data5.bin: POSIX tar archive (GNU)

**bandit12@bandit**:**/tmp/basu**$ tar xvf data5.bin

data6.bin

**bandit12@bandit**:**/tmp/basu**$ file data6.bin

data6.bin: bzip2 compressed data, block size = 900k

**bandit12@bandit**:**/tmp/basu**$ ls

data5.bin data6.bin data.txt output.out

**bandit12@bandit**:**/tmp/basu**$ bzip2 -d data6.bin

bzip2: Can't guess original name for data6.bin -- using data6.bin.out

**bandit12@bandit**:**/tmp/basu**$ ls

data5.bin data6.bin.out data.txt output.out

**bandit12@bandit**:**/tmp/basu**$ file data6.bin.out

data6.bin.out: POSIX tar archive (GNU)

**bandit12@bandit**:**/tmp/basu**$ tar xvf data6.bin.out

data8.bin

**bandit12@bandit**:**/tmp/basu**$ file data8.bin

data8.bin: gzip compressed data, was "data9.bin", last modified: Thu May 7 18:14:30 2020, max compression, from Unix

**bandit12@bandit**:**/tmp/basu**$ gzip data8.bin

**bandit12@bandit**:**/tmp/basu**$ ls

data5.bin data6.bin.out **data8.bin.gz** data.txt output.out

**bandit12@bandit**:**/tmp/basu**$ gzip data8.bin finalOutput

gzip: data8.bin: No such file or directory

gzip: finalOutput: No such file or directory

**bandit12@bandit**:**/tmp/basu**$ gunzip data8.bin.gz finalOutput

gzip: finalOutput.gz: No such file or directory

**bandit12@bandit**:**/tmp/basu**$ gunzip data8.bin.gz

gzip: data8.bin.gz: No such file or directory

**bandit12@bandit**:**/tmp/basu**$ ls

data5.bin data6.bin.out data8.bin data.txt output.out

**bandit12@bandit**:**/tmp/basu**$ file data8.bin

data8.bin: gzip compressed data, was "data9.bin", last modified: Thu May 7 18:14:30 2020, max compression, from Unix

**bandit12@bandit**:**/tmp/basu**$ gunzip data8.bin

gzip: data8.bin: unknown suffix -- ignored

**bandit12@bandit**:**/tmp/basu**$ ls

data5.bin data6.bin.out data8.bin data.txt output.out

**bandit12@bandit**:**/tmp/basu**$ file data8.bin

data8.bin: gzip compressed data, was "data9.bin", last modified: Thu May 7 18:14:30 2020, max compression, from Unix

**bandit12@bandit**:**/tmp/basu**$ cp data8.bin data8.bin.gz

**bandit12@bandit**:**/tmp/basu**$ gunzip data8.bin.gz

gzip: data8.bin already exists; do you wish to overwrite (y or n)? y

**bandit12@bandit**:**/tmp/basu**$ ls

data5.bin data6.bin.out data8.bin data.txt output.out

**bandit12@bandit**:**/tmp/basu**$ file data8.bin

data8.bin: ASCII text

**bandit12@bandit**:**/tmp/basu**$ cat data

cat: data: No such file or directory

**bandit12@bandit**:**/tmp/basu**$ cat data8.bin

The password is 8ZjyCRiBWFYkneahHwxCv3wb2a1ORpYL

13. On logging in, I notice a SSH private key:

**bandit13@bandit**:**~**$ ls

sshkey.private

**bandit13@bandit**:**~**$ cat sshkey.private

-----BEGIN RSA PRIVATE KEY-----

MIIEpAIBAAKCAQEAxkkOE83W2cOT7IWhFc9aPaaQmQDdgzuXCv+ppZHa++buSkN+

gg0tcr7Fw8NLGa5+Uzec2rEg0WmeevB13AIoYp0MZyETq46t+jk9puNwZwIt9XgB

ZufGtZEwWbFWw/vVLNwOXBe4UWStGRWzgPpEeSv5Tb1VjLZIBdGphTIK22Amz6Zb

ThMsiMnyJafEwJ/T8PQO3myS91vUHEuoOMAzoUID4kN0MEZ3+XahyK0HJVq68KsV

ObefXG1vvA3GAJ29kxJaqvRfgYnqZryWN7w3CHjNU4c/2Jkp+n8L0SnxaNA+WYA7

jiPyTF0is8uzMlYQ4l1Lzh/8/MpvhCQF8r22dwIDAQABAoIBAQC6dWBjhyEOzjeA

J3j/RWmap9M5zfJ/wb2bfidNpwbB8rsJ4sZIDZQ7XuIh4LfygoAQSS+bBw3RXvzE

I copied the key to my local machine, changed permissions (chmod 600: allowing only the owner of the file to be permitted to read/write) and SSH-ed using the key:

(base) basundharachakrabarty@Basundharas-MacBook-Air ~ % ssh -i sshkey.private bandit14@bandit.labs.overthewire.org -p 2220

Now, the password can be seen:

**bandit14@bandit**:**~**$ cd /etc/bandit\_pass/

**bandit14@bandit**:**/etc/bandit\_pass**$ ls

bandit0 bandit11 bandit14 bandit17 bandit2 bandit22 bandit25 bandit28 bandit30 bandit33 bandit6 bandit9

bandit1 bandit12 bandit15 bandit18 bandit20 bandit23 bandit26 bandit29 bandit31 bandit4 bandit7

bandit10 bandit13 bandit16 bandit19 bandit21 bandit24 bandit27 bandit3 bandit32 bandit5 bandit8

**bandit14@bandit**:**/etc/bandit\_pass**$ cat bandit14

4wcYUJFw0k0XLShlDzztnTBHiqxU3b3e

14.

I submitted the password to localhost using the netcat command to localhost (the host machine) on port=30000

**bandit14@bandit**:**/etc/bandit\_pass**$ echo 4wcYUJFw0k0XLShlDzztnTBHiqxU3b3e | nc localhost 30000

Correct!

BfMYroe26WYalil77FoDi9qh59eK5xNr

15. I used openssl to create a ssl-client (s\_client) and connect to localhost on port 30001, using -ign\_eof as mentioned in the comments to prevent closing of the conn when EOF is reached and got the password:

**bandit14@bandit**:**/etc/bandit\_pass**$ echo 'BfMYroe26WYalil77FoDi9qh59eK5xNr' | openssl s\_client -connect localhost:30001 -ign\_eof

CONNECTED(00000003)

depth=0 CN = localhost

verify error:num=18:self signed certificate

verify return:1

depth=0 CN = localhost

verify return:1

---

.

.

.

Start Time: 1650032601

Timeout : 7200 (sec)

Verify return code: 18 (self signed certificate)

Extended master secret: yes

---

Correct!

cluFn7wTiGryunymYOu4RcffSxQluehd

closed

16. I obtained the open ports between 31000-3200 by running a nmap scan:

**bandit16@bandit**:**~**$ nmap -p 31000-32000 localhost

Starting Nmap 7.40 ( https://nmap.org ) at 2022-04-15 16:31 CEST

Nmap scan report for localhost (127.0.0.1)

Host is up (0.00027s latency).

Not shown: 996 closed ports

PORT STATE SERVICE

31046/tcp open unknown

31518/tcp open unknown

31691/tcp open unknown

31790/tcp open unknown

31960/tcp open unknown

Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds

**bandit16@bandit**:**~**$

I tried sending the current password to all the ports above using openssl s\_client, port=31790 gave the key for the next level:

**bandit16@bandit**:**~**$ cat /etc/bandit\_pass/bandit16

cluFn7wTiGryunymYOu4RcffSxQluehd

**bandit16@bandit**:**~**$ echo 'cluFn7wTiGryunymYOu4RcffSxQluehd' | openssl s\_client --connect localhost:31790 -ign\_eof

CONNECTED(00000003)

depth=0 CN = localhost

verify error:num=18:self signed certificate

verify return:1

depth=0 CN = localhost

verify return:1

---

Certificate chain

0 s:/CN=localhost

i:/CN=localhost

---

Server certificate

.

.

.

Start Time: 1650033411

Timeout : 7200 (sec)

Verify return code: 18 (self signed certificate)

Extended master secret: yes

---

Correct!

-----BEGIN RSA PRIVATE KEY-----

MIIEogIBAAKCAQEAvmOkuifmMg6HL2YPIOjon6iWfbp7c3jx34YkYWqUH57SUdyJ

imZzeyGC0gtZPGujUSxiJSWI/oTqexh+cAMTSMlOJf7+BrJObArnxd9Y7YT2bRPQ

Ja6Lzb558YW3FZl87ORiO+rW4LCDCNd2lUvLE/GL2GWyuKN0K5iCd5TbtJzEkQTu

DSt2mcNn4rhAL+JFr56o4T6z8WWAW18BR6yGrMq7Q/kALHYW3OekePQAzL0VUYbW

JGTi65CxbCnzc/w4+mqQyvmzpWtMAzJTzAzQxNbkR2MBGySxDLrjg0LWN6sK7wNX

x0YVztz/zbIkPjfkU1jHS+9EbVNj+D1XFOJuaQIDAQABAoIBABagpxpM1aoLWfvD

KHcj10nqcoBc4oE11aFYQwik7xfW+24pRNuDE6SFthOar69jp5RlLwD1NhPx3iBl

J9nOM8OJ0VToum43UOS8YxF8WwhXriYGnc1sskbwpXOUDc9uX4+UESzH22P29ovd

d8WErY0gPxun8pbJLmxkAtWNhpMvfe0050vk9TL5wqbu9AlbssgTcCXkMQnPw9nC

YNN6DDP2lbcBrvgT9YCNL6C+ZKufD52yOQ9qOkwFTEQpjtF4uNtJom+asvlpmS8A

vLY9r60wYSvmZhNqBUrj7lyCtXMIu1kkd4w7F77k+DjHoAXyxcUp1DGL51sOmama

+TOWWgECgYEA8JtPxP0GRJ+IQkX262jM3dEIkza8ky5moIwUqYdsx0NxHgRRhORT

8c8hAuRBb2G82so8vUHk/fur85OEfc9TncnCY2crpoqsghifKLxrLgtT+qDpfZnx

SatLdt8GfQ85yA7hnWWJ2MxF3NaeSDm75Lsm+tBbAiyc9P2jGRNtMSkCgYEAypHd

HCctNi/FwjulhttFx/rHYKhLidZDFYeiE/v45bN4yFm8x7R/b0iE7KaszX+Exdvt

SghaTdcG0Knyw1bpJVyusavPzpaJMjdJ6tcFhVAbAjm7enCIvGCSx+X3l5SiWg0A

R57hJglezIiVjv3aGwHwvlZvtszK6zV6oXFAu0ECgYAbjo46T4hyP5tJi93V5HDi

Ttiek7xRVxUl+iU7rWkGAXFpMLFteQEsRr7PJ/lemmEY5eTDAFMLy9FL2m9oQWCg

R8VdwSk8r9FGLS+9aKcV5PI/WEKlwgXinB3OhYimtiG2Cg5JCqIZFHxD6MjEGOiu

L8ktHMPvodBwNsSBULpG0QKBgBAplTfC1HOnWiMGOU3KPwYWt0O6CdTkmJOmL8Ni

blh9elyZ9FsGxsgtRBXRsqXuz7wtsQAgLHxbdLq/ZJQ7YfzOKU4ZxEnabvXnvWkU

YOdjHdSOoKvDQNWu6ucyLRAWFuISeXw9a/9p7ftpxm0TSgyvmfLF2MIAEwyzRqaM

77pBAoGAMmjmIJdjp+Ez8duyn3ieo36yrttF5NSsJLAbxFpdlc1gvtGCWW+9Cq0b

dxviW8+TFVEBl1O4f7HVm6EpTscdDxU+bCXWkfjuRb7Dy9GOtt9JPsX8MBTakzh3

vBgsyi/sN3RqRBcGU40fOoZyfAMT8s1m/uYv52O6IgeuZ/ujbjY=

-----END RSA PRIVATE KEY-----

Closed

I copied the key to my local machine just like in the previous challenge, and used it to ssh for challenge 17

(base) basundharachakrabarty@Basundharas-MacBook-Air ~ % chmod 600 sshkey.private

(base) basundharachakrabarty@Basundharas-MacBook-Air ~ % ssh -i sshkey.private bandit17@bandit.labs.overthewire.org -p 2220

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

17.

The simplest way is to use the diff utility on linux to show the difference between the two files:

**bandit17@bandit**:**~**$ diff passwords.old passwords.new

42c42

< w0Yfolrc5bwjS4qw5mq1nnQi6mF03bii

---

> kfBf3eYk5BPBRzwjqutbbfE887SVc5Yd

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

18.

On trying to log into level 18, I was forced out as the .bashrc file has been modified to log the user out

(base) basundharachakrabarty@Basundharas-MacBook-Air ~ % ssh bandit18@bandit.labs.overthewire.org -p 2220

Therefore, I used -t to force a pseudo terminal as follows:

(base) basundharachakrabarty@Basundharas-MacBook-Air ~ % ssh -t bandit18@bandit.labs.overthewire.org -p 2220 /bin/sh

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

bandit18@bandit.labs.overthewire.org's password:

$

$

$ ls

readme

$ cat readme

IueksS7Ubh8G3DCwVzrTd8rAVOwq3M5x

19.

I ran the file as explained in the challenge, and noted that it can be used to run a command as bandit20. I cannot view the password in /etc/bandit\_pass/bandit20 ordinarily because of permissions, however we can use bandit20-do to view the password as shown below:

**bandit19@bandit**:**~**$ ./bandit20-do

Run a command as another user.

Example: ./bandit20-do id

**bandit19@bandit**:**~**$ ./bandit20-do cat /etc/bandit\_pass/bandit20

GbKksEFF4yrVs6il55v6gwY5aVje5f0j

20.

**bandit20@bandit**:**~**$ ls -l

total 12

-rwsr-x--- 1 bandit21 bandit20 12088 May 7 2020 suconnect

**bandit20@bandit**:**~**$ ./suconnect

Usage: ./suconnect <portnumber>

This program will connect to the given port on localhost using TCP. If it receives the correct password from the other side, the next password is transmitted back.

Therefore, I used netcat to listen on a port (1055), opened another terminal, run ./suconnect 1055 to connect to localhost, and passed the bandit20 password on the server’s side. This gave me the bandit21 password.

**bandit20@bandit**:**~**$ nc -lp 1055

GbKksEFF4yrVs6il55v6gwY5aVje5f0j

gE269g2h3mw3pwgrj0Ha9Uoqen1c9DGr----Next password

**bandit20@bandit**:**~**$ ./suconnect 1055

Read: GbKksEFF4yrVs6il55v6gwY5aVje5f0j

Password matches, sending next password

21. On logging in and navigating to /etc/cron.d, I find a couple of crontab files, out of which I observe cronjob\_bandit22, which has two cron jobs, one running cronjob\_bandit22.sh once after every reboot (@reboot) and another running every minute (\*\*\*\*\*).

On observing the shell script cronjob\_bandit22.sh, it writes bandit22’s password to a file under the /tmp directory. Therefore, I am able to get level 22’s password.

**bandit21@bandit**:**/etc/cron.d**$ ls

cronjob\_bandit15\_root cronjob\_bandit17\_root cronjob\_bandit22 cronjob\_bandit23 cronjob\_bandit24 cronjob\_bandit25\_root

**bandit21@bandit**:**/etc/cron.d**$

**bandit21@bandit**:**/etc/cron.d**$ ls -al

total 36

drwxr-xr-x 2 root root 4096 Jul 11 2020 **.**

drwxr-xr-x 87 root root 4096 May 14 2020 **..**

-rw-r--r-- 1 root root 62 May 14 2020 cronjob\_bandit15\_root

-rw-r--r-- 1 root root 62 Jul 11 2020 cronjob\_bandit17\_root

-rw-r--r-- 1 root root 120 May 7 2020 cronjob\_bandit22

-rw-r--r-- 1 root root 122 May 7 2020 cronjob\_bandit23

-rw-r--r-- 1 root root 120 May 14 2020 cronjob\_bandit24

-rw-r--r-- 1 root root 62 May 14 2020 cronjob\_bandit25\_root

-rw-r--r-- 1 root root 102 Oct 7 2017 .placeholder

**bandit21@bandit**:**/etc/cron.d**$ cat cronjob\_bandit22

@reboot bandit22 /usr/bin/cronjob\_bandit22.sh &> /dev/null

\* \* \* \* \* bandit22 /usr/bin/cronjob\_bandit22.sh &> /dev/null

**bandit21@bandit**:**/etc/cron.d**$ cat /usr/bin/cronjob\_bandit22.sh

#!/bin/bash

chmod 644 /tmp/t7O6lds9S0RqQh9aMcz6ShpAoZKF7fgv

cat /etc/bandit\_pass/bandit22 > /tmp/t7O6lds9S0RqQh9aMcz6ShpAoZKF7fgv

**bandit21@bandit**:**/etc/cron.d**$ cat /tmp/t7O6lds9S0RqQh9aMcz6ShpAoZKF7fgv

Yk7owGAcWjwMVRwrTesJEwB7WVOiILLI

22. I observed the scripts in /etc/cron.d and obtained the shell script as below. It seems like the script is getting the current user (myname), constructing the string “I am user $myname” and writing a modified md5 hashed version of it as the target directory name and copying the password to it.

**bandit22@bandit**:**~**$ cd /etc/cron.d

**bandit22@bandit**:**/etc/cron.d**$ ls

cronjob\_bandit15\_root cronjob\_bandit17\_root cronjob\_bandit22 cronjob\_bandit23 cronjob\_bandit24 cronjob\_bandit25\_root

**bandit22@bandit**:**/etc/cron.d**$

**bandit22@bandit**:**/etc/cron.d**$ cat cronjob\_bandit23

@reboot bandit23 /usr/bin/cronjob\_bandit23.sh &> /dev/null

\* \* \* \* \* bandit23 /usr/bin/cronjob\_bandit23.sh &> /dev/null

**bandit22@bandit**:**/etc/cron.d**$ cat /usr/bin/cronjob\_bandit23.sh

#!/bin/bash

myname=$(whoami)

mytarget=$(echo I am user $myname | md5sum | cut -d ' ' -f 1)

echo "Copying passwordfile /etc/bandit\_pass/$myname to /tmp/$mytarget"

cat /etc/bandit\_pass/$myname > /tmp/$mytarget

We can reverse engineer this by running the same operation for “I am user bandit23”, getting the target filename, and from there we can get the password for level 23!

**bandit22@bandit**:**/etc/cron.d**$ echo I am user bandit23 | md5sum | cut -d ' ' -f 1

8ca319486bfbbc3663ea0fbe81326349

**bandit22@bandit**:**/etc/cron.d**$ cd /tmp/8ca319486bfbbc3663ea0fbe81326349

-bash: cd: /tmp/8ca319486bfbbc3663ea0fbe81326349: Not a directory

**bandit22@bandit**:**/etc/cron.d**$ cat /tmp/8ca319486bfbbc3663ea0fbe81326349

jc1udXuA1tiHqjIsL8yaapX5XIAI6i0n

23. The cron job runs the following shell script:

**bandit23@bandit**:**/etc/cron.d**$ cat cronjob\_bandit24

@reboot bandit24 /usr/bin/cronjob\_bandit24.sh &> /dev/null

\* \* \* \* \* bandit24 /usr/bin/cronjob\_bandit24.sh &> /dev/null

**bandit23@bandit**:**/etc/cron.d**$ cat /usr/bin/cronjob\_bandit24.sh

#!/bin/bash

myname=$(whoami)

cd /var/spool/$myname

echo "Executing and deleting all scripts in /var/spool/$myname:"

for i in \* .\*;

do

if [ "$i" != "." -a "$i" != ".." ];

then

echo "Handling $i"

owner="$(stat --format "%U" ./$i)"

if [ "${owner}" = "bandit23" ]; then

timeout -s 9 60 ./$i

fi

rm -f ./$i

fi

done

The script scans /var/spool/{current user} and runs all the scrips, then deletes them. If we put a script in that directory and redirect the output to a file, we can obtain the password:

**bandit23@bandit**:**/etc/cron.d**$ mkdir /tmp/challenge23

**bandit23@bandit**:**/etc/cron.d**$ chmod 777 /tmp/challenge23

**bandit23@bandit**:**/etc/cron.d**$ vim /tmp/challenge23/23script.sh

**bandit23@bandit**:**/etc/cron.d**$ cat /tmp/challenge23/23script.sh

#!/bin/sh

cat /etc/bandit\_pass/bandit24 > /tmp/challenge23/output.txt---Our script

**bandit23@bandit**:**/etc/cron.d**$ touch /tmp/challenge23/output.txt

**bandit23@bandit**:**/etc/cron.d**$ chmod 777 /tmp/challenge23/output.txt

**bandit23@bandit**:**/etc/cron.d**$ chmod 777 /tmp/challenge23/23script.sh

**bandit23@bandit**:**/etc/cron.d**$ cp /tmp/challenge23/23script.sh /var/spool/bandit24

**After waiting for a minute:**

**bandit23@bandit**:**/etc/cron.d**$ cat /tmp/challenge23/output.txt

UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ

24.

Running a netcat for port=30002 shows and testing using pin=0000 shows how the server works:

**bandit24@bandit**:**~**$ nc localhost 30002 UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ 0000

I am the pincode checker for user bandit25. Please enter the password for user bandit24 and the secret pincode on a single line, separated by a space.

UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ 0000

Wrong! Please enter the correct pincode. Try again.

UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ 0000

Timeout. Exiting.

invalid port UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ

**bandit24@bandit**:**~**$ UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ 0000

-bash: UoMYTrfrBFHyQXmg6gzctqAwOmw1IohZ: command not found

We need to brute force the password. I wrote a shell script to generate all combinations for 0000-9999 to a file (combinations.txt) and then attempt netcat using that file.

**bandit24@bandit**:**~**$ vim passwordCracker.sh

**bandit24@bandit**:**~**$ ls

**bandit24@bandit**:**~**$ touch passwordCracker.sh

touch: cannot touch 'passwordCracker.sh': Permission denied

**bandit24@bandit**:**~**$ touch /tmp/passwordCracker.sh

**bandit24@bandit**:**~**$ chmod 777 /tmp/passwordCracker.sh

**bandit24@bandit**:**~**$ vim /tmp/passwordCracker.sh

**bandit24@bandit**:**~**$ touch /tmp/combinations.txt

**bandit24@bandit**:**~**$ chmod 777 /tmp/combinations.txt

**bandit24@bandit**:**~**$ cd tmp

-bash: cd: tmp: No such file or directory

**bandit24@bandit**:**~**$ cd /tmp/

**bandit24@bandit**:**/tmp**$ ./passwordCracker.sh > combinations.txt

I then ran netcat with the combinations file to try all combinations:

**bandit24@bandit**:**/tmp**$ nc localhost 30002 < combinations.txt

I am the pincode checker for user bandit25. Please enter the password for user bandit24 and the secret pincode on a single line, separated by a space.

Wrong! Please enter the correct pincode. Try again.

Wrong! Please enter the correct pincode. Try again.

Wrong! Please enter the correct pincode. Try again.

.

.

Correct!

The password of user bandit25 is uNG9O58gUE7snukf3bvZ0rxhtnjzSGzG

Exiting.

25, 26.

**bandit25@bandit**:**~**$ ls

bandit26.sshkey

**bandit25@bandit**:**~**$

**bandit25@bandit**:**~**$ ssh -i bandit26.sshkey bandit26@localhost

Could not create directory '/home/bandit25/.ssh'.

The authenticity of host 'localhost (127.0.0.1)' can't be established.

ECDSA key fingerprint is SHA256:98UL0ZWr85496EtCRkKlo20X3OPnyPSB5tB5RPbhczc.

Are you sure you want to continue connecting (yes/no)? yes

Failed to add the host to the list of known hosts (/home/bandit25/.ssh/known\_hosts).

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

A look at /etc/passwd shows that bandit26 user is mapped with the shell /usr/bin/showtest.

The showtext script is as follows:

**bandit25@bandit**:**~**$ cat /usr/bin/showtext  
#!/bin/shexport TERM=linuxmore ~/text.txt  
exit 0  
bandit25@bandit:~$

It runs more on a file, and then exits. When we ssh to bandit26, we can see the file and the more prompt. I ssh-ed to the file in a small terminal, and upon getting the file, pressed ‘v’ to enable vim, followed by using :e to open another /etc/bandit\_pass/bandit26 (:e filename, <https://linuxhint.com/opening_switching_multiple_files_vim/>)

This gives me the password 5czgV9L3Xx8JPOyRbXh6lQbmIOWvPT6Z

I also changed the shell variable back to /bin/bash by running “:set shell=/bin/bash”.

Upon logging into the shell, I notice the following file:

bandit26@bandit:~$ ls

bandit27-do text.txt

bandit26@bandit:~$ ./bandit27-do

Run a command as another user.

Example: ./bandit27-do id

I can run a cat for the password similar to what we’ve done in one of the previous challenges:

bandit26@bandit:~$ ./bandit27-do cat /etc/bandit\_pass/bandit27

3ba3118a22e93127a4ed485be72ef5ea

bandit26@bandit:/tmp$ git clone ssh://bandit27-git@localhost/home/bandit27-git/rep

Cloning into 'rep'...

The authenticity of host 'localhost (127.0.0.1)' can't be established.

ECDSA key fingerprint is SHA256:98UL0ZWr85496EtCRkKlo20X3OPnyPSB5tB5RPbhczc.

Are you sure you want to continue connecting (yes/no)? yes

Failed to add the host to the list of known hosts (/home/bandit26/.ssh/known\_hosts).

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

bandit27-git@localhost's password:

Permission denied, please try again.

bandit27-git@localhost's password:

fatal: '/home/bandit27-git/rep' does not appear to be a git repository

fatal: Could not read from remote repository.

Git clone fails in the root directory, thereby ran it from /tmp, and obtained the password from a READme file within.

bandit26@bandit:/tmp$ git clone ssh://bandit27-git@localhost/home/bandit27-git/rep

Cloning into 'rep'...

The authenticity of host 'localhost (127.0.0.1)' can't be established.

ECDSA key fingerprint is SHA256:98UL0ZWr85496EtCRkKlo20X3OPnyPSB5tB5RPbhczc.

Are you sure you want to continue connecting (yes/no)? yes

Failed to add the host to the list of known hosts (/home/bandit26/.ssh/known\_hosts).

This is a OverTheWire game server. More information on http://www.overthewire.org/wargames

bandit26@bandit:/tmp$ ls

ls: cannot open directory '.': Permission denied

bandit26@bandit:/tmp$ cd repo

bandit26@bandit:/tmp/repo$ ls

README

bandit26@bandit:/tmp/repo$ cat README

The password to the next level is: 0ef186ac70e04ea33b4c1853d2526fa2