NaCl Sandbox - Homework

03 - Basic Linux

67011178 Nuker (Com-Inter)

My OverTheWire: Bandit walkthroughs.

Bandit00

Use **SSH** command to login.

```
nukerducker@MSI:~$ ssh bandit0.labs.overthewire.org -p 2220
```

The password is `bandit0`

```
bandit0@bandit:~$ ls
```

Use **ls** to list all the file in the directory.

```
bandit0@bandit:~$ ls
readme
bandit0@bandit:~$ cat readme
```

Then, use **cat** command to read readme file.

The password is `ZjLjTmM6FvvyRnrb2rfNWOZOTa6ip5If`

proof

```
bandit0@bandit:~$ exit
```

Bandit01

Login using password from previous level.

```
nukerducker@MSI:~$ ssh bandit1.labs.overthewire.org -p 2220
```

```
bandit1@bandit:~$ ls
-
bandit1@bandit:~$ cat < -
263JGJPfgU6LtdEvgfWU1XP5yac29mFx
bandit1@bandit:~$ exit
```

The `<` operator tells the shell to read input from a file name `-`.

proof The password is `263JGJPfgU6LtdEvgfWU1XP5yac29mFx`

Bandit02

Login using password from previous level.

```
nukerducker@MSI:~$ ssh bandit2.labs.overthewire.org -p 2220
```

```
bandit2@bandit:~$ ls
spaces in this filename
bandit2@bandit:~$ cat < 'spaces in this filename'
MNk8KNH3Usiio41PRUEoDFPqfxLPlSmx
```

Use `' '` to create a string of filename.

proof The password is `MNk8KNH3Usiio41PRUEoDFPqfxLPlSmx`

Bandit03

Login using password from previous level.

```
nukerducker@MSI:~$ ssh bandit3.labs.overthewire.org -p 2220
```

```
bandit3@bandit:~$ ls
inhere
bandit3@bandit:~$ cd inhere
bandit3@bandit:~$ ls -a
bandit3@bandit:~$ cat < '...Hiding-From-You'
2WmrDFRmJIq3IPxneAaMGhap0pFhF3NJ
```

Use `cd` to change the working directory. Use `ls -a` to see all the files including hidden files.

proof The password is `2WmrDFRmJIq3IPxneAaMGhap0pFhF3NJ`

Bandit04

Login using password from previous level.

```
nukerducker@MSI:~$ ssh bandit4.labs.overthewire.org -p 2220
bandit4@bandit:~$ ls
inhere
bandit4@bandit:~$ cd inhere
bandit4@bandit:~/inhere$ ls -a
bandit4@bandit:~/inhere$ file -- *
-file07: ASCII text
bandit4@bandit:~/inhere$ cat < '-file07'
4oQYVPkxZOOEOO5pTW81FB8j8lxXGUQw
```

Use `file` to check file types, then use `--` to tell the command that the next arguments are file names, then use `*` to check all files in the folder.

proof The password is `4oQYVPkxZOOEOO5pTW81FB8j8lxXGUQw`

Bandit05

After logged in,

```
bandit5@bandit:~$ ls
inhere
bandit5@bandit:~$ cd inhere
bandit5@bandit:~/inhere$ ls -a
bandit5@bandit:~/inhere$ find -type f -size 1033c
./maybehere07/.file2
bandit5@bandit:~/inhere$ cat ./maybehere07/.file2
HWasnPhtq9AVKe0dmk45nxy20cvUa6EG
```

Use `find` to search, then use `-type f` to look only for files, and use `-size 1033c` (`c` is for bytes) to find specifics file size.

proof The password is `HWasnPhtq9AVKe0dmk45nxy20cvUa6EG`

Bandit06

After logged in,

```
bandit6@bandit:~$ ls -la
bandit6@bandit:~$ find / -type f -user bandit7 -group bandit6 -size 33c
2>/dev/null
/var/lib/dpkg/info/bandit7.password
```

```
bandit6@bandit:~$ cat /var/lib/dpkg/info/bandit7.password
morbNTDkSW6jIlUc0ymOdMaLnOlFVAaj
```

Use `find /` to start from root directory, then use `-type f` to look only for files, use `-user bandit7`find files that are owned by the user **bandit7**, use `-group bandit6` search for files that belong to the group **bandit6**, use `-size 33c` to find specifics file size, and use `2>/dev/null` to ignore error messages. proof The password is `morbNTDkSW6jIlUc0ymOdMaLnOlFVAaj`

Bandit07

After logged in,

```
bandit7@bandit:~$ ls
data.txt
bandit7@bandit:~$ grep -w 'data.txt' -e 'millionth'
millionth       dfwvzFQi4mU0wfNbFOe9RoWskMLg7eEc
```

Use `grep` to search text within files, use `-w` ensures that `grep` matches whole words only, `'data.txt'` name of the file, and use `-e 'millionth'` to find specifics term.

proof The password is `dfwvzFQi4mU0wfNbFOe9RoWskMLg7eEc`

Bandit08

After logged in,

```
bandit8@bandit:~$ ls
data.txt
bandit8@bandit:~$ sort data.txt | uniq -u
4CKMh1JI91bUIZZPXDqGanal4xvAg0JM
```

Use `sort` to sort lins in file in alphabetical order, use `|` to take output from previous command and uses it as input for the next command, use `uniq -u` to filters out duplicate lines.

proof The password is `4CKMh1JI91bUIZZPXDqGanal4xvAg0JM`

Bandit09

After logged in,

```
bandit8@bandit:~$ ls
data.txt
bandit8@bandit:~$ strings data.txt | grep '=========='
```

Use `string` to output human-readable texts, use `grep '=========='` to searches for lines that contain the specific string in the text extracted by strings. proof The password is `FGUW5ilLVJrxX9kMYMmlN4MgbpfMiqey`

Bandit10

After logged in,

```
bandit10@bandit:~$ ls
data.txt
bandit10@bandit:~$ cat data.txt | base64 -d
The password is dtR173fZKb0RRsDFSGsg2RWnpNVj3qRr
```

Use `base64 -d` to decode the output text from `cat` command.

proof The password is `dtR173fZKb0RRsDFSGsg2RWnpNVj3qRr`

Bandit11

After logged in,

```
bandit10@bandit:~$ ls
data.txt
bandit10@bandit:~$ cat data.txt | tr 'A-Za-z' 'N-ZA-Mn-za-m'
The password is 7x16WNeHIi5YkIhWsfFIqoognUTyj9Q4
```

Assuming that this is ***ROT13*** cipher I sought out to find command that could decrypt this and found `tr 'A-Za-z'`. `tr` is command for translate or replace characters, `'A-Za-z'` this specifies the set of characters to be translated. `'N-ZA-Mn-za-m'` It maps each letter to another letter with a shift. proof The password is `7x16WNeHIi5YkIhWsfFIqoognUTyj9Q4`

Bandit12

Oh boy this is so brutal to see and explain but here we go. After logged in,

```
bandit12@bandit:~$ mktemp -d
/tmp/tmp.H1dCj0UGDn
```

First, start with create a temp directory using `mktemp -d`

```
bandit12@bandit:~$ cp data.txt /tmp/tmp.H1dCj0UGDn
bandit12@bandit:~$ cd /tmp/tmp.H1dCj0UGDn
```

then we `cd` into the temp directory and use `cp` to copy over **data.txt** into the temp directory.

```
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ xxd -r data.txt > data.bin
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ file data.bin
data.bin: gzip compressed data, was "data2.bin", last modified: Thu Sep 19
07:08:15 2024, max compression, from Unix, original size modulo 2^32 574
```

use `xxd -r data.txt > data.bin` to convert hexdump file in txt to binary format as **data.bin**, then we use `file` to check type of **data.bin**.

```
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ mv data.bin data.gz
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ gunzip data.gz
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ ls
data  data.txt
```

I renamed it using `mv data.bin data.gz`. Given the output said that it is **gzip** compressed file I Used `gunzip data.gz`to decomress **data.gz**

```
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ file data
data: bzip2 compressed data, block size = 900k
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ mv data data.bz2
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ bunzip2 data.bz2
```

then after checking and knowing that the file is **bzip2** type I renamed it using `mv data data.bz2` to convert it back to **.bz2** file.

```
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ ls
data  data.txt
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ file data
data: gzip compressed data, was "data4.bin", last modified: Thu Sep 19 07:08:15
2024, max compression, from Unix, original size modulo 2^32 20480
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ mv data data.gz
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ gunzip data.gz
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ ls
data  data.txt
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ file data
data: POSIX tar archive (GNU)
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ mv data data.tar
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ tar -xvf data.tar
data5.bin
```

```
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ ls
data5.bin  data.tar  data.txt
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ file data5.bin
data5.bin: POSIX tar archive (GNU)
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ mv data5.bin data.tar
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ tar -xvf data.tar
data6.bin
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ mv data6.bin data.tar
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ tar -xvf data.tar
data8.bin
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ mv data8.bin data.tar
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ ls
data.tar  data.txt
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ tar -xvf data.tar
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ ls
data.tar  data.txt
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ tar -xvf data.tar
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ tar -xvf data.tar
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ file data.tar
data.tar: gzip compressed data, was "data9.bin", last modified: Thu Sep 19
07:08:15 2024, max compression, from Unix, original size modulo 2^32 49
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ mv data.tar data.gzip
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ ls
data.gzip  data.txt
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ gunzip data.gzip
gzip: data.gzip: unknown suffix -- ignored
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ mv data.gzip data.gz
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ gunzip data.gzip
gzip: data.gzip.gz: No such file or directory
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ gunzip data.gz
```

I repeated this processes until I got new type of file which is **data.tar** so I used `tar -xvf data.tar` to extracts the contents of the tar archive **data.tar**

```
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ ls
data  data.txt
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ file data
data: ASCII text
bandit12@bandit:~$ /tmp/tmp.H1dCj0UGDn$ cat data
```

I repeat all of this until I found the file with **ASCII text** data type and `cat` it.

proof The password is `FO5dwFsc0cbaIiH0h8J2eUks2vdTDwAn`

Bandit13

After logged in,

```
bandit13@bandit:~$ ls
sshkey.private
bandit13@bandit:~$ ssh -i sshkey.private bandit14@localhost -p 2220
```

Use `ssh` to login, use `-i sshkey.private` to specify private SSH key file.

After getting in Bandit14, just use simple `cat` followed by directories.

```
bandit14@bandit:~$ cat /etc/bandit_pass/bandit14
MU4VWeTyJk8ROof1qqmcBPaLh7lDCPvS
```

proof proof

The password is `MU4VWeTyJk8ROof1qqmcBPaLh7lDCPvS`

Bandit14

Still in Bandit14,

```
bandit14@bandit:~$ nc localhost 30000
MU4VWeTyJk8ROof1qqmcBPaLh7lDCPvS
Correct!
8xCjnmgoKbGLhHFAZlGE5Tmu4M2tKJQo
```

I used `nc` or **Netcat** which is command to read or write to network connections, followed by `localhost` which refers to the local manchine, and then `30000` which is the given port number.

proof

The password is `8xCjnmgoKbGLhHFAZlGE5Tmu4M2tKJQo`

Bandit15

Still in Bandit14,

```
bandit14@bandit:~$ openssl s_client -connect localhost:30001 -quiet
Can't use SSL_get_servername
depth=0 CN = SnakeOil
verify error:num=18:self-signed certificate
verify return:1
depth=0 CN = SnakeOil
verify return:1
8xCjnmgoKbGLhHFAZlGE5Tmu4M2tKJQo
```

```
Correct!
kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx
```

I used `openssl` which is command-line for managing **SSL/TLS** connections, then I used `s_client` to initiates an SSL/TLS connection to remote server, use `-connect localhost:3000` to specifies the server `localhost` and the port number `30001`, and ended with `-quiet` to show only essential data of the connection.

proof

The password is `kSkvUpMQ7lBYyCM4GBPvCvT1BfWRy0Dx`
