CLI Commands by Suteerth Subramaniam

- Bandit Website Game
- Main Github Repo
- Exercise Repo

Level 0 -> Level 1

`ZjLjTmM6FvvyRnrb2rfNWOZOTa6ip5If`

```
ssh bandit0@bandit.labs.overthewire.org -p 2220
ls
cat readme
```

Level 1 -> Level 2

`263JGJPfgU6LtdEvgfWU1XP5yac29mFx`

- The filename is - so it has to be accessed in a different way.

```
cat < -
cat ./-
```

Level 2 -> Level 3

`MNk8KNH3Usiio41PRUEoDFPqfxLPlSmx`

- The filename has spaces in it's name so it needs to be accessed with it's name in quotes.

```
ls
cat < "spaces in this filename"
```

Level 3 -> Level 4

`2WmrDFRmJIq3IPxneAaMGhap0pFhF3NJ`

- The filename is hidden so we have to first access it (they start with `.`)

```
ls -a
cat ./...Hiding-From-You
```

Level 4 -> Level 5

`4oQYVPkxZOOEOO5pTW81FB8j8lxXGUQw`

- The files are having a common prefix and only the last number changes so we can use * to make this easy for us.

```
cat ./-file0*
```

Level 5 -> Level 6

`HWasnPhtq9AVKe0dmk45nxy20cvUa6EG`

- The password for the next level is stored in a file somewhere under the inhere directory and has all of the following properties:
  - human-readable (ASCII only)
  - 1033 bytes in size
  - not executable

```
cd inhere
find . -type f -size 1033c ! -executable
```

- We can specify the exact size using the `-size` option. Adding `+/-` can be used to find it in a range.

Level 6 -> Level 7

`morbNTDkSW6jIlUc0ymOdMaLnOlFVAaj`

- The password for the next level is stored somewhere on the server and has all of the following properties:
  - owned by user bandit7
  - owned by group bandit6
  - 33 bytes in size

```
find / -group bandit6 -user bandit7 -size 33c
cat /var/lib/dpkg/info/bandit7.password
```

Level 7 -> Level 8

`dfwvzFQi4mU0wfNbFOe9RoWskMLg7eEc`

- The password for the next level is stored in the file data.txt next to the word millionth

```
grep --help
grep -F "millionth" data.txt
```

Level 8 -> Level 9

`4CKMh1JI91bUIZZPXDqGanal4xvAg0JM`

- We have to find the password but it occurs only ONCE. This can be demonstrated as well.

```
sort data.txt | uniq -c
sort data.txt | uniq -c | grep "1 "
sort data.txt | uniq -u
```

Level 9 -> Level 10

`FGUW5ilLVJrxX9kMYMmlN4MgbpfMiqey`

- The password for the next level is stored in the file data.txt in one of the few human-readable strings, preceded by several '=' characters.
- `strings` command can be used to distinguish human-readable strings

```
strings data.txt | grep "======*"
```

Level 10 -> Level 11

`dtR173fZKb0RRsDFSGsg2RWnpNVj3qRr`

- Pretty easy as you just gotta decode the base64 string to human readable text.

```
cat data.txt | base64 -d
```

Level 11 -> Level 12

`7x16WNeHIi5YkIhWsfFIqoognUTyj9Q4`

- The password for the next level is stored in the file data.txt, where all lowercase (a-z) and uppercase (A-Z) letters have been rotated by 13 positions
- Decode ROT13 encryption which can be achieved by `tr old new`

```
cat data.txt | tr 'A-Za-z' 'N-ZA-Mn-za-m'
```

Level 12 -> Level 13

`FO5dwFsc0cbaIiH0h8J2eUks2vdTDwAn`

- `data.txt` is a hexdump of a file that has been repeatedly compressed
- Might be useful to create a directory under /tmp in which you can work. Use mkdir with a hard to guess directory name. Or better, use the command "mktemp -d". Then copy the datafile using cp, and rename it using mv (read the manpages!)
- `1f8b` is gzip as we can see in the hexdump data. File signature will be the first few bytes.
- After decompressing first time, the new signature is `425a 68` which is bzip.s
- Once we see `.bin` file, its an archive so we can use tar
- `data6.bin` is again bz2

```
cd /tmp
mktemp -d
cd /tmp/tmp.agG7Rxjoj6
cp ~/data.txt /tmp/tmp.agG7Rxjoj6/data.txt
mv data.txt hexdump_data
xxd -r hexdump_data reversed
mv reversed reversed.gz
gzip -d reversed.gz
xxd reversed
mv reversed reversed.bz2
bzip2 -d reversed.bz2
mv reversed reversed.gz
gzip -d reversed.gz
xxd reversed.gz | head
```

```
mv reversed.gz reversed.tar
tar -xf data5.bin
xxd data6.bin.out | head
tar -xf data6.bin.out
xxd data8.bin | head
mv data8.bin data8.bin.gz
gzip -d data8.bin.gz
```