Over the wire solutions
Bandit

Nathan Warner

Northern Illinois
University

Computer Science
Northern Illinois University
November 5, 2023
United States

Contents

Passwords

0. bandit0
1. NH2SXQwcBdpmTEzi3bvBHMM9H66vVXjL
2. rRGizSaX8Mk1RTb1CNQoXTcYZWU6lgzi
3. aBZ0W5EmUfAf7kHTQeOwd8bauFJ2lAiG
4. 2EW7BBsr6aMMoJ2HjW067dm8EgX26xNe
5. lrIWWI6bB37kxfiCQZqUdOIYfr6eEeqR
6. z7WtoNQU2XfjmMtWA8u5rN4vzqu4v99S
7. TESKZC0XvTetK0S9xNwm25STk5iWrBvP
8. EN632PlfYiZbn3PhVK3XOGSlNInNE00t
9. G7w8LIi6J3kTb8A7j9LgrywtEUlyyp6s
10. 6zPeziLdR2RKNdNYFNb6nVCKzphlXHBM
11. VNBBFSmZwKKOP0XbFXOoW8chDz5yVRv

Commands used

- **xclip -selection clipboard < filename**: used to save contents of a file into clipboard (can't use on bandit servers)
- **File:** The file command is used to determine the type of a file. In this game, we use it to determine if a file is human-readable (ASCII text), as apposed to DATA.
- **Find:**
  - Exec flag
  - ! -executable (negation of -executable)
  - -size 1033c
- **grep**
  - -e (match pattern)
  - -v (match NOT pattern)
- **Exclude "permission denied" output from grep**
  - find / [expressions] 2>&1 | grep -v "Permission denied"
- **awk**

– awk '{print 2}′

Over The Wire Solutions

Level 4

The password for the next level is stored in the only human-readable file in the inhere directory

The file command

The way we solve this level is by use of the file command. The file command will output whether a file is DATA or ASCII text. We are looking for the file that contains ASCII text

```
for i in *;do file "./$i";done
```

Looping with the ”find” command

Instead of using a simple for loop, we can instead make use of the *find* command with the -exec flag. The general syntax for this method is

```
find [path] [expression] -exec [command] {} \;
// or
find [path] [expression] -exec [command] {} +
```

Where:

- [path] is the directory find starts searching from. If omitted, find uses the current directory.
- [expression] is used to specify search criteria such as name, type, size, etc.
- [command] is the command that find will execute on each file that matches the search criteria.
- {} is a placeholder that find replaces with the current file name being processed.
- \; is used to terminate the command.

When using at the end of the -exec command, find will execute the command once for each file found. However, when you use + at the end of the -exec command, find will pass all the matched files to the command at once, rather than one by one. This is often more efficient, especially when dealing with a large number of files, because it minimizes the number of times the command is called.

Thus, the solution would be

```
find . -type f -exec file {} \; | grep -e 'text'
// or
find . -type f -exec file {} + | grep -e 'text'
```

We grep 'text' because file with either output the file as being data, or ASCII text, so we only want to display the file that has ASCII text.

Level 5

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
- 1033 bytes in size
- not executable

The way in which we solve this problem is by use of:

- For loop
- Find command
- file command

```
for i in *;do find "./${i}" -type f -size 1033c ! -executable -exec file {} \; | grep -e 'ASCII' ;done
```

Level 6

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 / -type f -size 33c -user bandit7 -group bandit6 2>&1 | grep -v "Permission denied"
```

Note:-

The reason we also redirect standard error and standard output to the grep command (btw grep -v excludes matches) is because "permission denied" matches are errors, thus we need to send them to our grep

Level 7

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

This one is pretty simple, we just grep with awk

```
grep -e 'millionth' data.txt | awk '{print $2}'
```

Level 8

The password for the next level is stored in the file data.txt and is the only line of text that occurs only once

For this level we use the *uniq* command with the -u flag, this will output unique lines. However, the uniq command does not detect repeated lines unless they are adjacent. You may want to sort the input first

```
sort data.txt | uniq -u
```

Level 9

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.

To solve this problem we can use the *strings* command. The *strings* command will print the sequences of printable characters in files

```
strings data.txt | grep -e '==='
```

Level 10

The password for the next level is stored in the file data.txt, which contains base64 encoded data

For this level, we use the *base64* command to decode the data.

```
base64 -d data.txt
```

Level 11

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

he file data.txt contains text that has been encrypted using the ROT13. To decrypt the text, we can use the *tr* command, which allows us translate or delete characters

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

Which will map the character set that has been shifted by 13 characters, back by 13 characters