Hexhunt

Question 1

- What is the name of the vulnerability found when using the `-o` option in the `hexhunt.out` binary?
- What is the size of the buffer used?
- What is the flag obtained?

Hint

- Try sending different values as the input!

```
$ ./hexhunt.out -o example.com

  _    ____    _            ____
| |_ |__ /_ _| |_ _  _ _ _|__  |
| ' \ |_ \ \ / ' \ || | ' \ / /
|_||_|___/_\_\_||_\_,_|_||_/_/
  Hexhunt -- a not so dangerous application

Sending payload to example.com

Target overflowed!
```

Question 2

- When using the `-d` option, you are given a 'payload quota' of 16. Increase this payload quota in order to obtain the flag!
- The flag also contains a date. Investigate why this date is important with regards to computer systems.

Hint

- Find out what happens when you send additional payloads even if the quota is 0...

```
$ ./hexhunt.out -d example.com

  _    ____    _            ____
| |_ |__ /_ _| |_ _  _ _ _|__  |
| ' \ |_ \ \ / ' \ || | ' \ / /
|_||_|___/_\_\_||_\_,_|_||_/_/
  Hexhunt -- a not so dangerous application

Entering DDOS Session

You have 16 payloads left. Please subscribe to Premium tier for higher
payload quota!

Please enter the number of payloads to send: 16
Sent 16 packets!
```

```
You have 0 payloads left. Please subscribe to Premium tier for higher
payload quota!

Please enter the number of payloads to send:
```

Question 3

- When using the `-r` option, you are asked to send some data to the target. Exploit this vulnerability to call the `returnAFlag()` function present in the binary.

Hint

- It is easy to find out an input that will cause a segmentation fault...

```
$./hexhunt.out -r example.com

  _    ____    _            ____
| |_ |__ /_ _| |_ _  _ _ _|__  |
| ' \ |_ \ \ / ' \ || | ' \ / /
|_||_|___/_\_\_||_\_,_|_||_/_/
  Hexhunt -- a not so dangerous application

Enter data (in bytes) to return to the target:
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

Data has been returned!

Segmentation fault
```

- The Python `pwn` library can help with exploiting this vulnerability. Do keep in mind the issue of stack alignment when crafting the payload!

```python
import pwn

args = ['./hexhunt.out', '-r', 'ntu.edu.sg']
p = pwn.process(args)
elf = pwn.ELF(p.argv[0])

returnAFlag_addr = elf.symbols['returnAFlag']
print(f"win_addr={hex(returnAFlag_addr)}")

# write payload here
# payload =
# print(payload)

p.sendline(payload)
p.interactive()
```