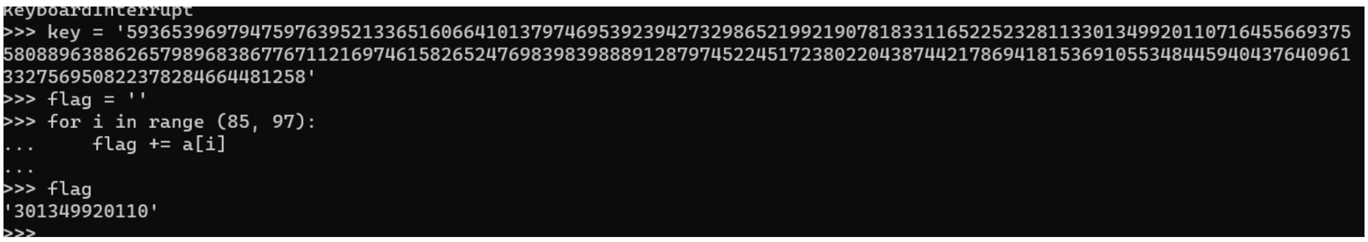
Bosch-TheThree

> Install "wine" if you in Linux and want to execute file `.exe`

Challenge 1:

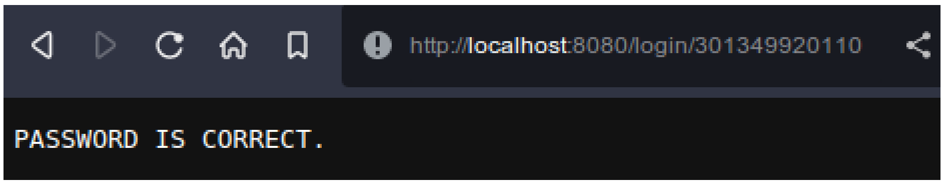
- Run file `server.exe` and go to "http://localhost:8080/login/{password}".

- Solution:



```
KeyboardInterrupt
>>> key = '5936539697947597639521336516066410137974695392394273298652199219078183311652252328113301349920110716455669375580889638862657989683867767112169746158265247698398398889128797452245172380220438744217869418153691055348445940437640961332756950822378284664481258'
>>> flag = ''
>>> for i in range (85, 97):
...     flag += a[i]
...
>>> flag
'301349920110'
>>>
```

- Check password



http://localhost:8080/login/301349920110

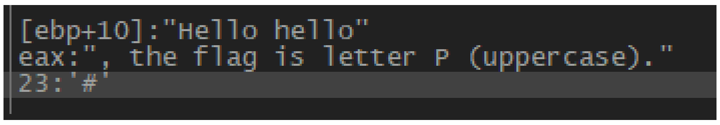
```
PASSWORD IS CORRECT.
```

> The flag would be `301349920110`

Challenge 2:

- Run file `get-flag.exe` with the arguments is the name of the user

- Note that we reverse the file `get-flag.exe` and find this line



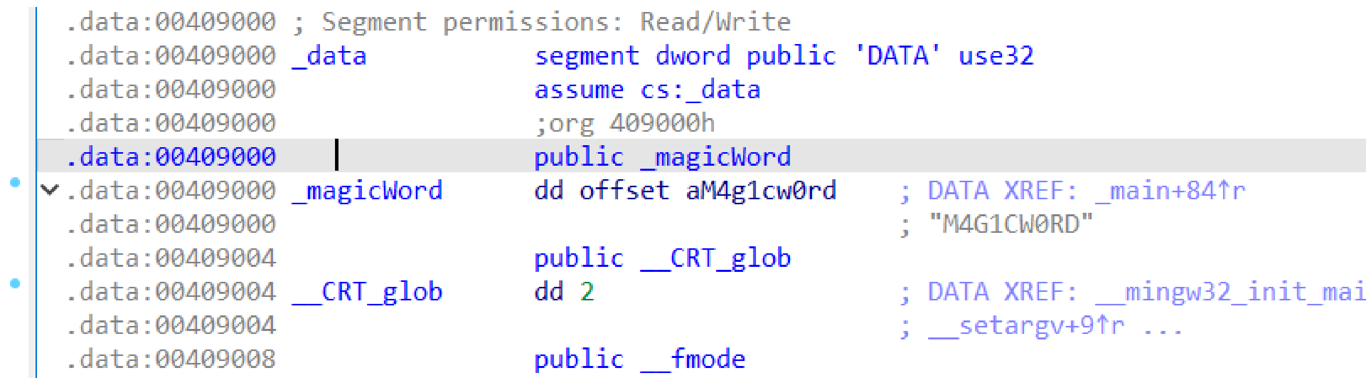
```
[ebp+10]:"Hello hello"
eax:", the flag is letter P (uppercase)."
23:'#'
```

> The flag would be `P`

Challenge 3:

- Run file `get-flag.exe` with the arguments is the name of the user name

- Again, reverse the file `get-flag.exe` we got this

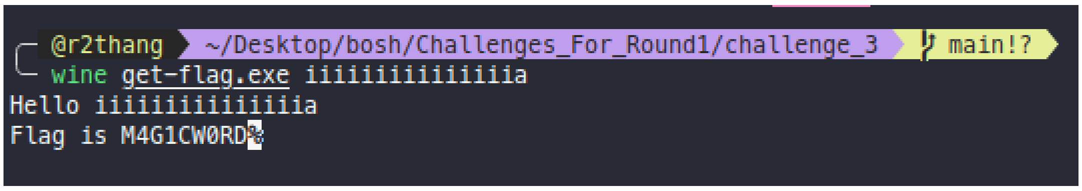


```
.data:00409000 ; Segment permissions: Read/Write
.data:00409000 _data           segment dword public 'DATA' use32
.data:00409000                 assume cs:_data
.data:00409000                 ;org 409000h
.data:00409000                 public _magicWord
.data:00409000 _magicWord      dd offset aM4g1cw0rd    ; DATA XREF: _main+84↑r
.data:00409000                                         ; "M4G1CW0RD"
.data:00409004                 public __CRT_glob
.data:00409004 __CRT_glob      dd 2                    ; DATA XREF: __mingw32_init_mai
.data:00409004                                         ; __setargv+9↑r ...
.data:00409008                 public __fmode
```

- Easy to buffer overflow by passing a string

```
get-flag.exe iiiiiiiiiiiia
```

- It will be return the flag `Flag is M4G1CW0RD`



```
@r2thang  ~/Desktop/bosh/Challenges_For_Round1/challenge_3  main!?
wine get-flag.exe iiiiiiiiiiiiiia
Hello iiiiiiiiiiiiiia
Flag is M4G1CW0RD%
```

> The flag would be `M4G1CW0RD`

CHallenge 4:

- Run file `exploit.py`

> cat `exploit.py`

```
record = [{
    "plaintext": "001333b95c1edb3ef145a4a9f52f7b9a",
    "ciphertext": "bb28b7e3f49355b7a236ad2745d68cb25",
    "iv" : "70359350f329603f0da99a1f9151d844"
}]

from pwn import xor
p_1 = bytes.from_hex("001333b95c1edb3ef145a4a9f52f7b9a")
c_1 = bytes.from_hex("bb28b7e3f49355b7a236ad2745d68cb25")
c_2 = bytes.from_hex("fdfa29ef6547ef37a64a1bb2c629c5cc")

p_2 = xor(xor(c_1, c_2), p_1)

print(p_2)
```

> python `exploit.py`

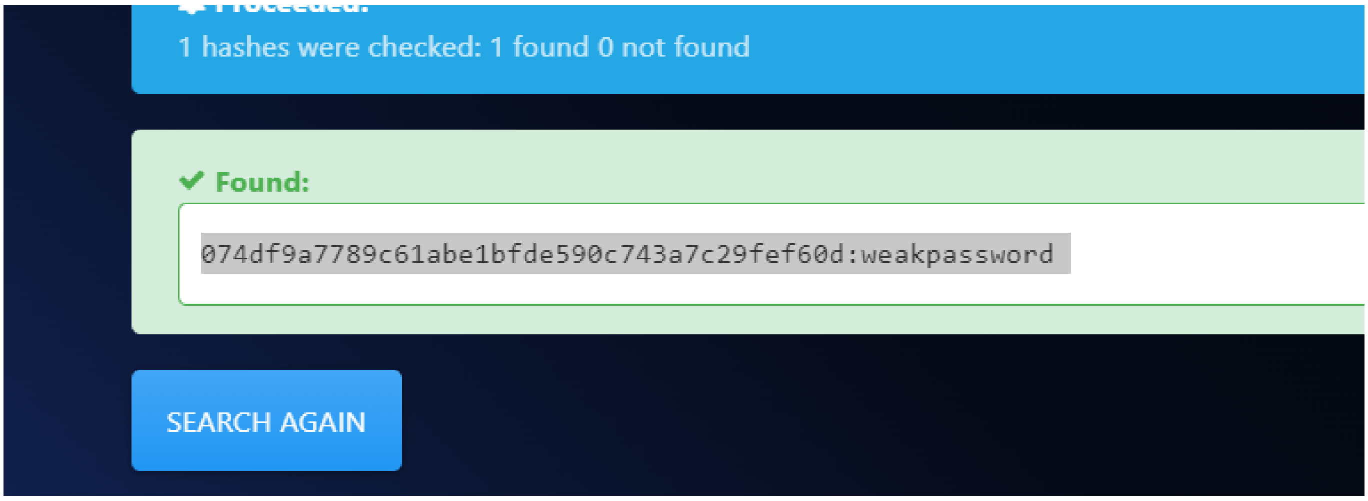
- The output would look like

```
b'Obdiplostemonnus'
```

> The flag would be `Obdiplostemonnus`

Challenge 5

- SHA-1 digest of the flag is 0x074df9a7789c61abe1bfde590c743a7c29fef60d
- After reverse the given digest


1 hashes were checked: 1 found 0 not found

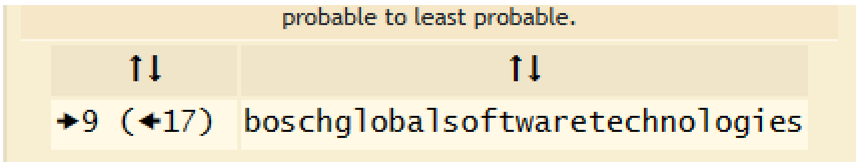
Found:

```
074df9a7789c61abe1bfde590c743a7c29fef60d:weakpassword
```

> The flag would be `weakpassword`

Challenge 6

- The ciphertext is `kxblqpuxkjubxocfjancnlqwxuxprnb`
- After decrypting the ciphertext, we got


probable to least probable.

⇅	⇅
→9 (←17)	boschglobalsoftwaretechnologies

> The flag would be `boschglobalsoftwaretechnologies`