Writeup PicoCTF
Category Reverse Engineering

Created by

rainxmei

Daftar isi

[EASY]

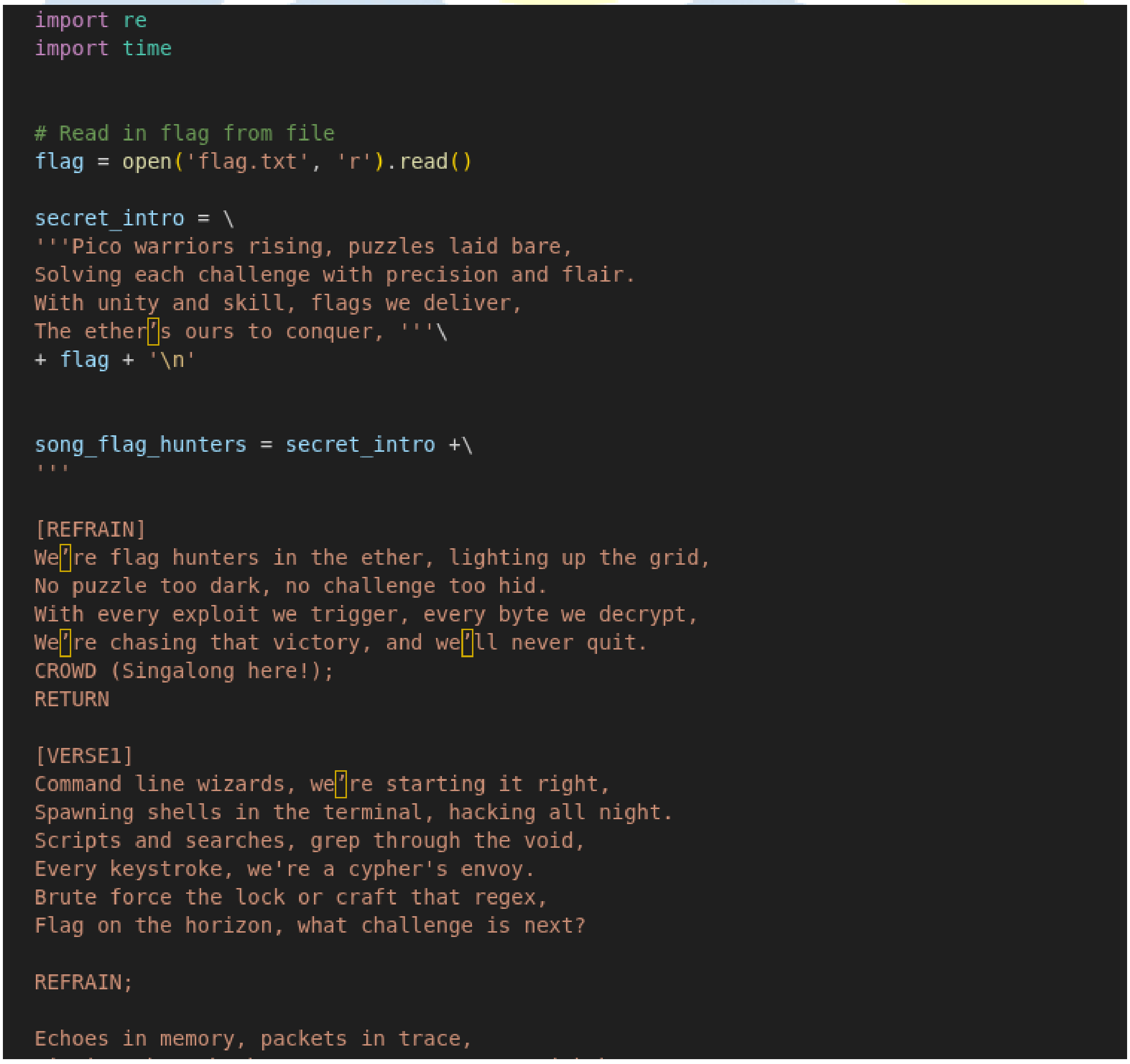
Flag HUnters

Lyrics jump from verses to the refrain kind of like a subroutine call. There's a hidden refrain this program doesn't print by default. Can you get it to print it? There might be something in it for you.The program's source code can be downloaded here. Additional details will be available after launching your challenge instance

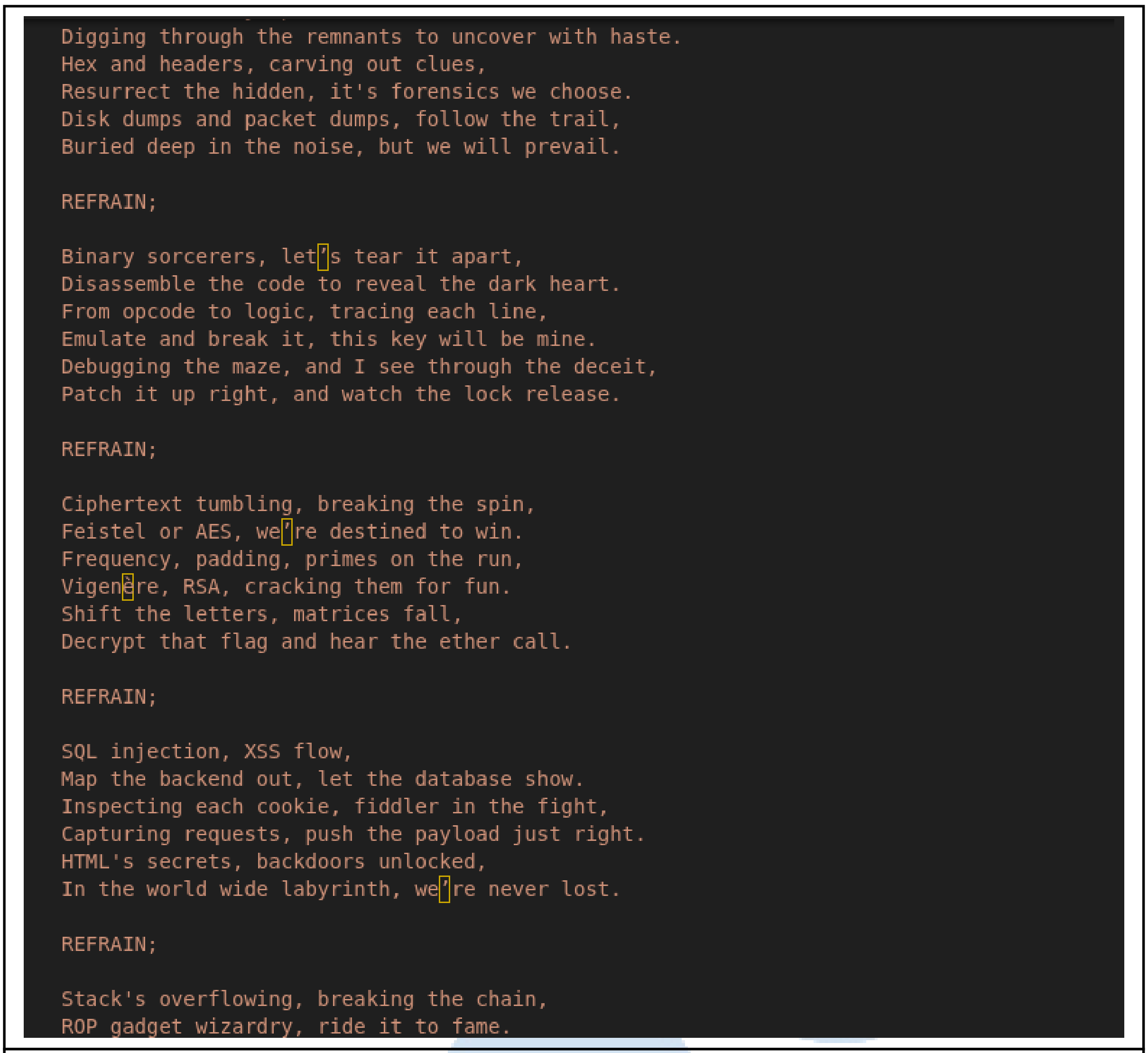
Attachment: lyric-reader.py

Kita diberi file python dan kita diberi tahu bahwa ada hidden refrain yang tidak diprint oleh program kemungkinan hidden refrain tersebut adalah flag, kita coba lihat source codenya

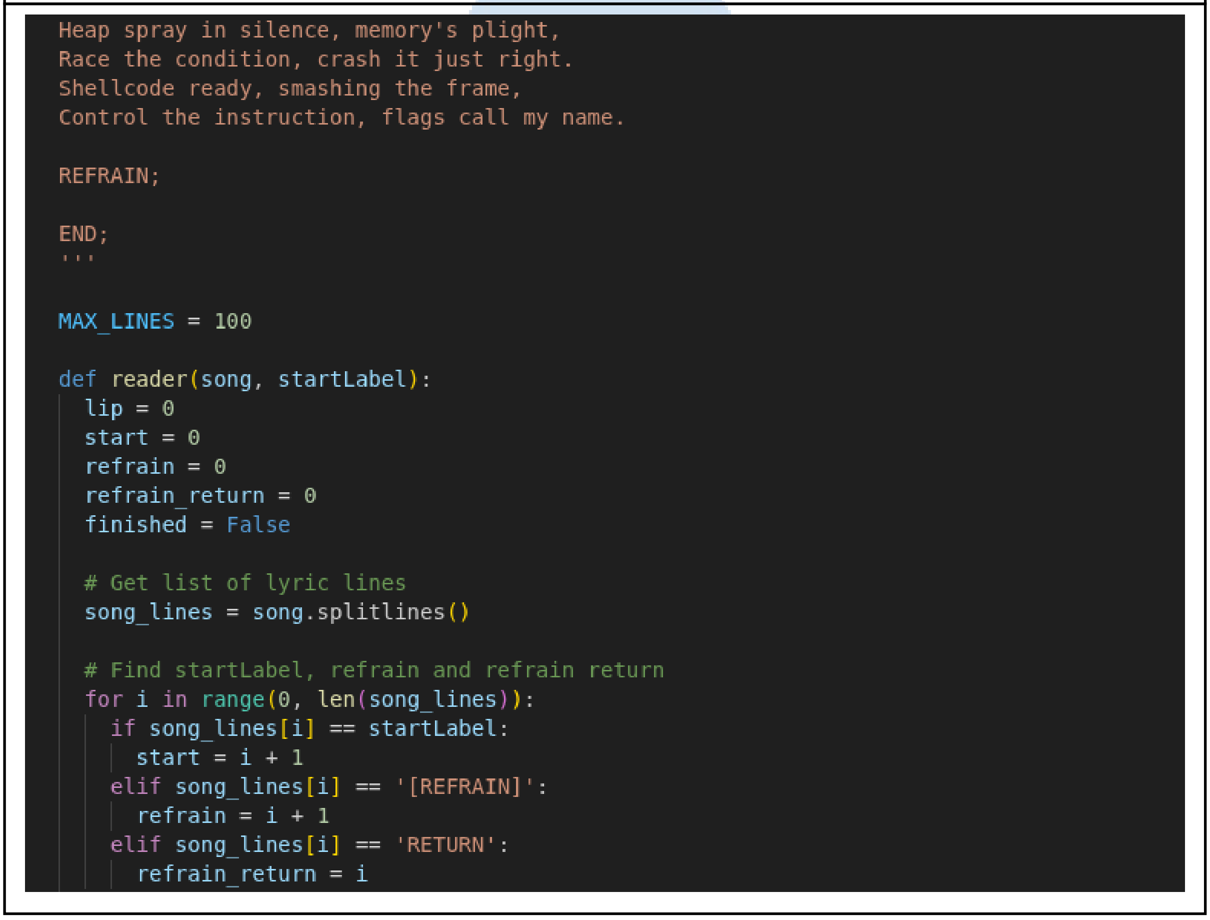
lyric-reader.py



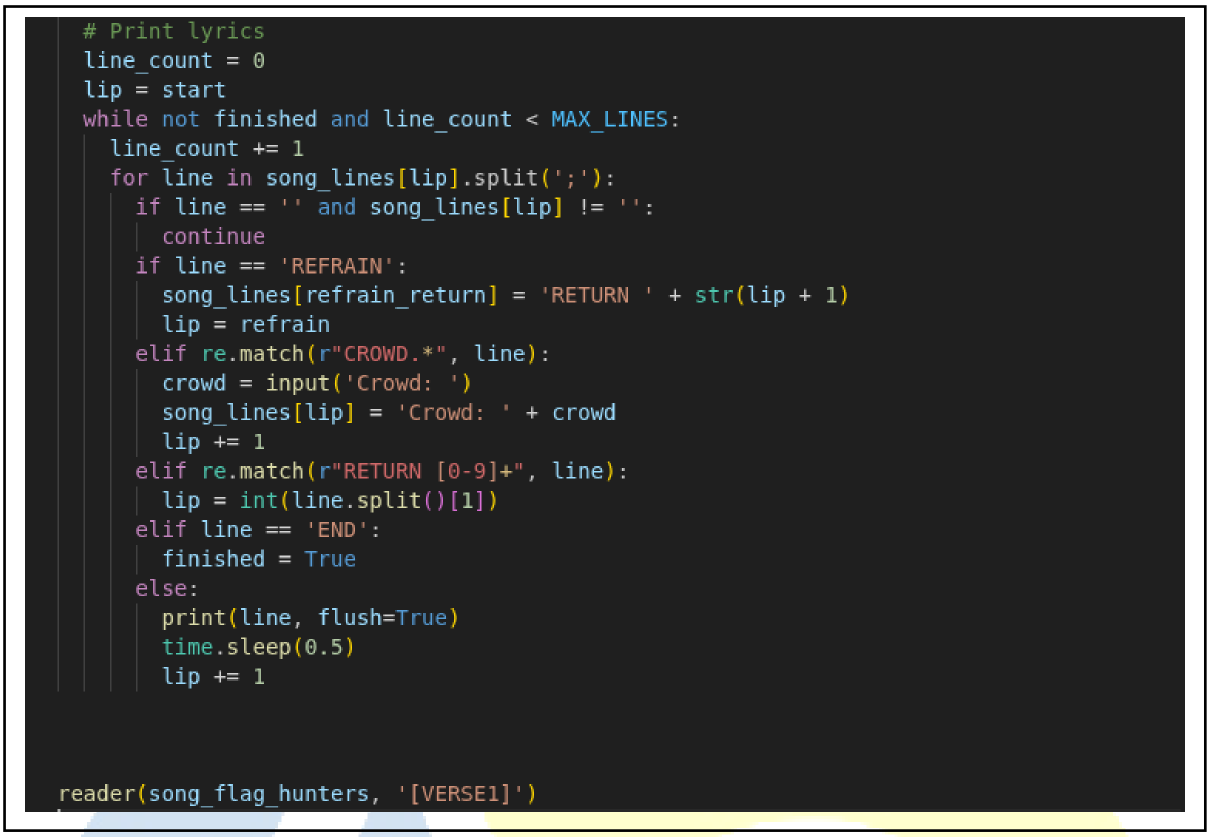
```
import re
import time


# Read in flag from file
flag = open('flag.txt', 'r').read()

secret_intro = \
'''Pico warriors rising, puzzles laid bare,
Solving each challenge with precision and flair.
With unity and skill, flags we deliver,
The ether’s ours to conquer, '''\
+ flag + '\n'


song_flag_hunters = secret_intro +\
'''


[REFRAIN]
We’re flag hunters in the ether, lighting up the grid,
No puzzle too dark, no challenge too hid.
With every exploit we trigger, every byte we decrypt,
We’re chasing that victory, and we’ll never quit.
CROWD (Singalong here!);
RETURN

[VERSE1]
Command line wizards, we’re starting it right,
Spawning shells in the terminal, hacking all night.
Scripts and searches, grep through the void,
Every keystroke, we're a cypher's envoy.
Brute force the lock or craft that regex,
Flag on the horizon, what challenge is next?

REFRAIN;

Echoes in memory, packets in trace,
```



```
Digging through the remnants to uncover with haste.
Hex and headers, carving out clues,
Resurrect the hidden, it's forensics we choose.
Disk dumps and packet dumps, follow the trail,
Buried deep in the noise, but we will prevail.

REFRAIN;

Binary sorcerers, let’s tear it apart,
Disassemble the code to reveal the dark heart.
From opcode to logic, tracing each line,
Emulate and break it, this key will be mine.
Debugging the maze, and I see through the deceit,
Patch it up right, and watch the lock release.

REFRAIN;

Ciphertext tumbling, breaking the spin,
Feistel or AES, we’re destined to win.
Frequency, padding, primes on the run,
Vigenère, RSA, cracking them for fun.
Shift the letters, matrices fall,
Decrypt that flag and hear the ether call.

REFRAIN;

SQL injection, XSS flow,
Map the backend out, let the database show.
Inspecting each cookie, fiddler in the fight,
Capturing requests, push the payload just right.
HTML's secrets, backdoors unlocked,
In the world wide labyrinth, we’re never lost.

REFRAIN;

Stack's overflowing, breaking the chain,
ROP gadget wizardry, ride it to fame.
```

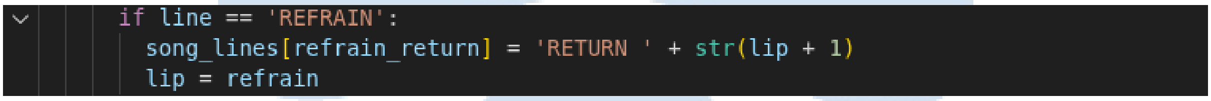


```
Heap spray in silence, memory's plight,
Race the condition, crash it just right.
Shellcode ready, smashing the frame,
Control the instruction, flags call my name.

REFRAIN;

END;
'''

MAX_LINES = 100

def reader(song, startLabel):
  lip = 0
  start = 0
  refrain = 0
  refrain_return = 0
  finished = False

  # Get list of lyric lines
  song_lines = song.splitlines()

  # Find startLabel, refrain and refrain return
  for i in range(0, len(song_lines)):
    if song_lines[i] == startLabel:
      start = i + 1
    elif song_lines[i] == '[REFRAIN]':
      refrain = i + 1
    elif song_lines[i] == 'RETURN':
      refrain_return = i
```



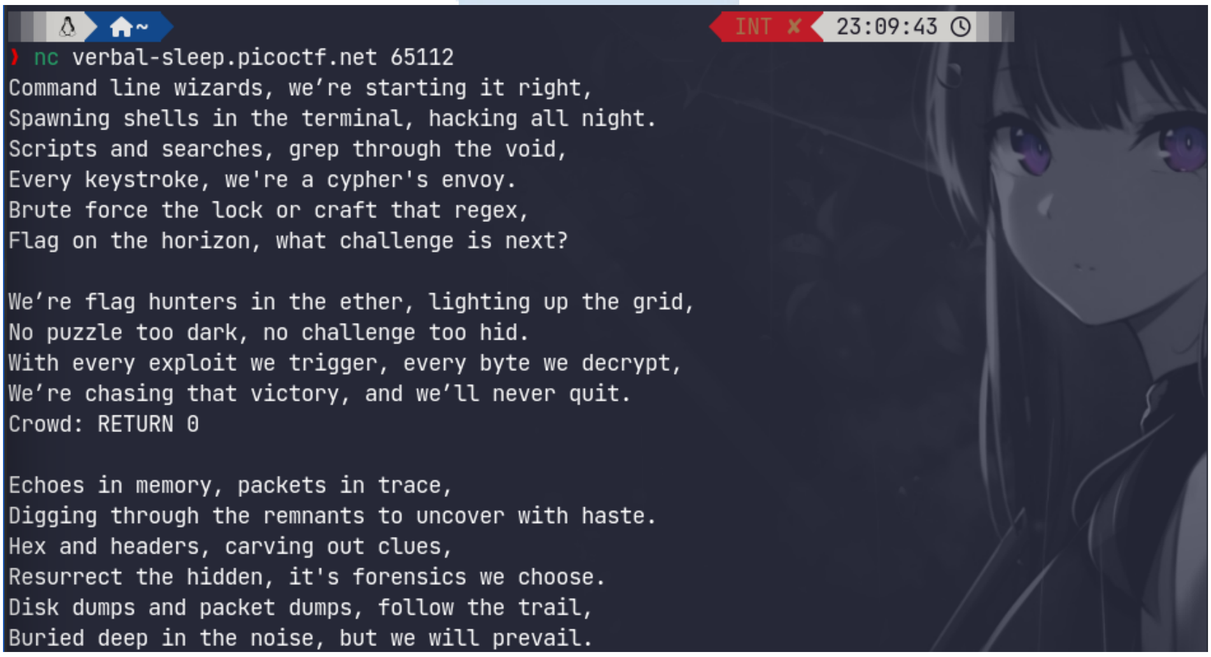
```
    # Print lyrics
    line_count = 0
    lip = start
    while not finished and line_count < MAX_LINES:
      line_count += 1
      for line in song_lines[lip].split(';'):
        if line == '' and song_lines[lip] != '':
          continue
        if line == 'REFRAIN':
          song_lines[refrain_return] = 'RETURN ' + str(lip + 1)
          lip = refrain
        elif re.match(r"CROWD.*", line):
          crowd = input('Crowd: ')
          song_lines[lip] = 'Crowd: ' + crowd
          lip += 1
        elif re.match(r"RETURN [0-9]+", line):
          lip = int(line.split()[1])
        elif line == 'END':
          finished = True
        else:
          print(line, flush=True)
          time.sleep(0.5)
          lip += 1



reader(song_flag_hunters, '[VERSE1]')
```

Bagian yang tersembunyi itu termasuk flag tersebut, tetapi bagian tersebut tidak diprint coba kita cari apakah ada celah agar bisa print flag tersebut



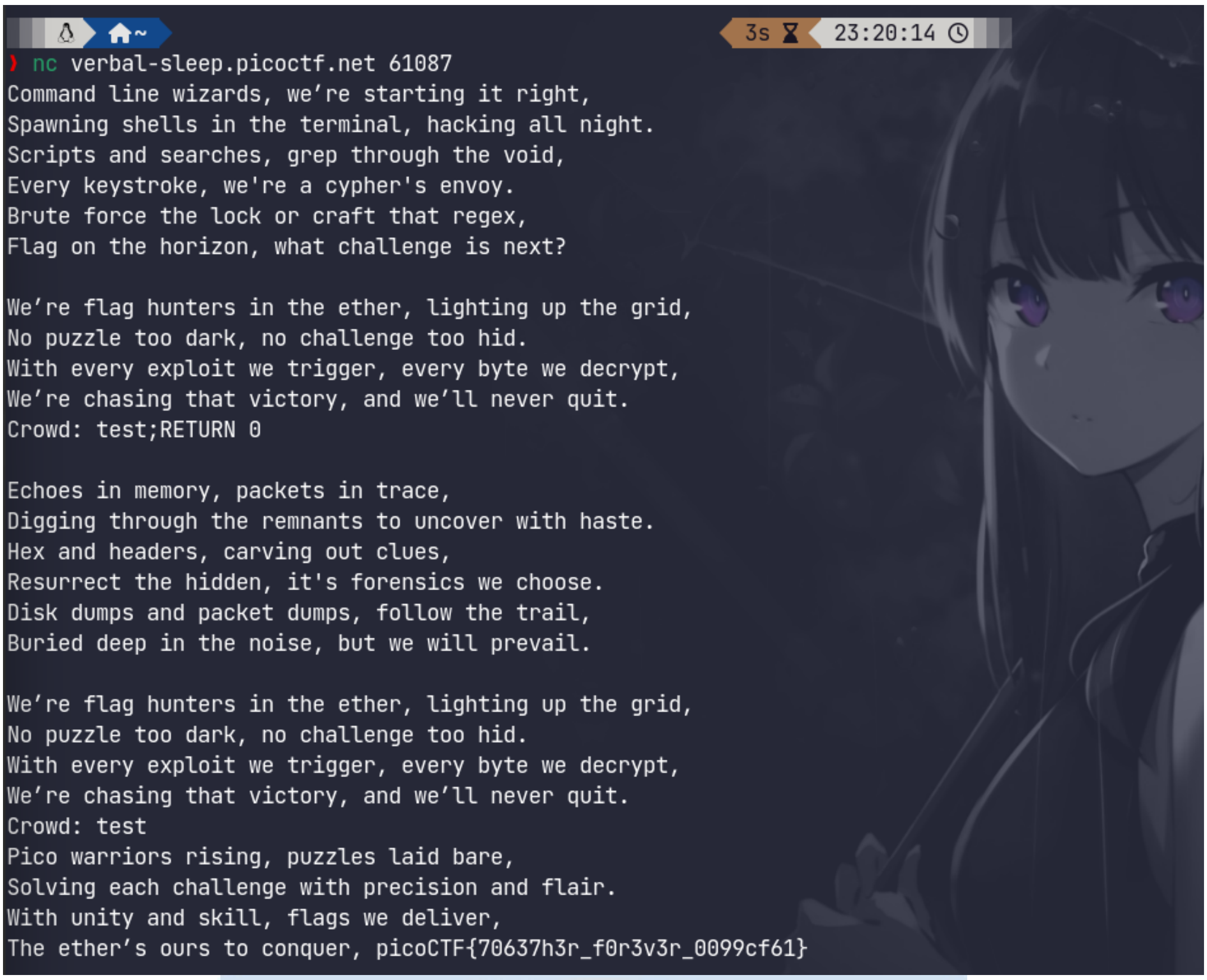
```
        if line == 'REFRAIN':
          song_lines[refrain_return] = 'RETURN ' + str(lip + 1)
          lip = refrain
```

Sepertinya bagian ini dia setelah setalah print REFRAIN dia akan menyimpan baris terakhir ke baigan RETURN kita coba untuk RETURN 0 karena flag berada pada bagian awal



```
❯ nc verbal-sleep.picoctf.net 65112
Command line wizards, we’re starting it right,
Spawning shells in the terminal, hacking all night.
Scripts and searches, grep through the void,
Every keystroke, we're a cypher's envoy.
Brute force the lock or craft that regex,
Flag on the horizon, what challenge is next?

We’re flag hunters in the ether, lighting up the grid,
No puzzle too dark, no challenge too hid.
With every exploit we trigger, every byte we decrypt,
We’re chasing that victory, and we’ll never quit.
Crowd: RETURN 0

Echoes in memory, packets in trace,
Digging through the remnants to uncover with haste.
Hex and headers, carving out clues,
Resurrect the hidden, it's forensics we choose.
Disk dumps and packet dumps, follow the trail,
Buried deep in the noise, but we will prevail.
```

Sepertinya RETURN 0-nya tidak tereksekusi karena dia tergabung dengan Crowd: coba kita pisah menggunakan semicolon karena program tersebut memisah berdasarkan semicolon



```
❯ nc verbal-sleep.picoctf.net 61087
Command line wizards, we’re starting it right,
Spawning shells in the terminal, hacking all night.
Scripts and searches, grep through the void,
Every keystroke, we're a cypher's envoy.
Brute force the lock or craft that regex,
Flag on the horizon, what challenge is next?

We’re flag hunters in the ether, lighting up the grid,
No puzzle too dark, no challenge too hid.
With every exploit we trigger, every byte we decrypt,
We’re chasing that victory, and we’ll never quit.
Crowd: test;RETURN 0

Echoes in memory, packets in trace,
Digging through the remnants to uncover with haste.
Hex and headers, carving out clues,
Resurrect the hidden, it's forensics we choose.
Disk dumps and packet dumps, follow the trail,
Buried deep in the noise, but we will prevail.

We’re flag hunters in the ether, lighting up the grid,
No puzzle too dark, no challenge too hid.
With every exploit we trigger, every byte we decrypt,
We’re chasing that victory, and we’ll never quit.
Crowd: test
Pico warriors rising, puzzles laid bare,
Solving each challenge with precision and flair.
With unity and skill, flags we deliver,
The ether’s ours to conquer, picoCTF{70637h3r_f0r3v3r_0099cf61}
```

Horee kita sudah dapat flag tersebut

picoCTF{70637h3r_f0r3v3r_0099cf61}

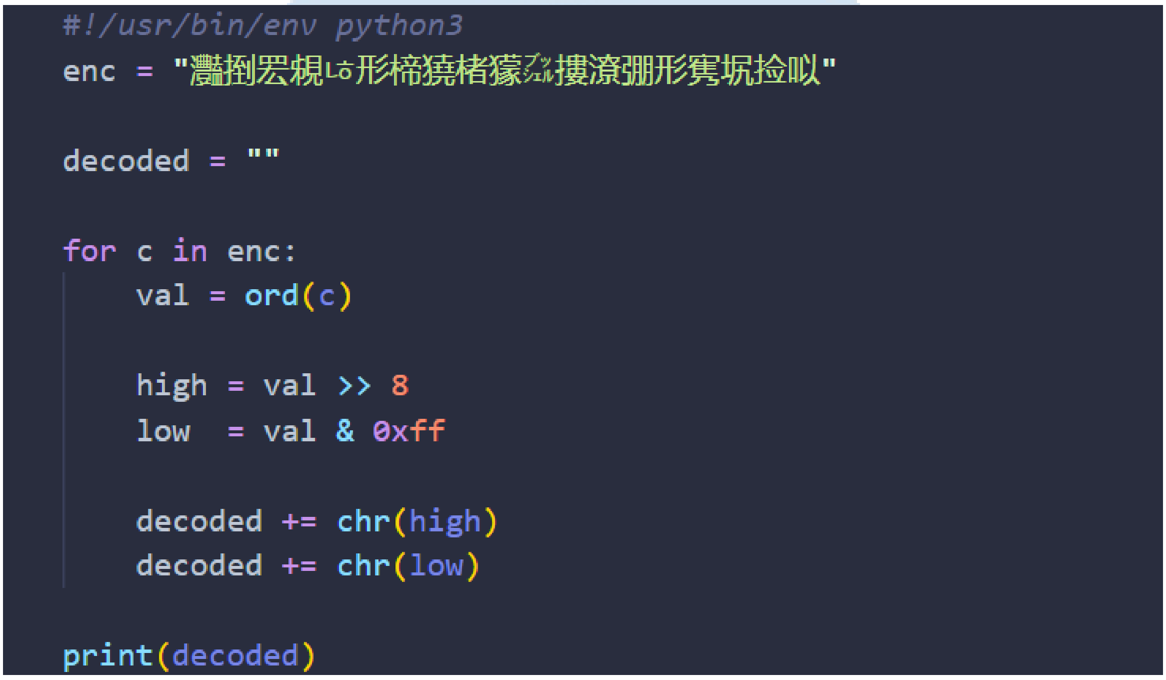
Transformation
I wonder what this really is...enc ''.join([chr((ord(flag[i]) << 8) + ord(flag[i + 1])) for i in range(0, len(flag), 2)])
Attachment: env

Kita diberikan file env, sepertinya soal menyuruh kita untuk mendecode dari file env, kita coba lihat isi file enc

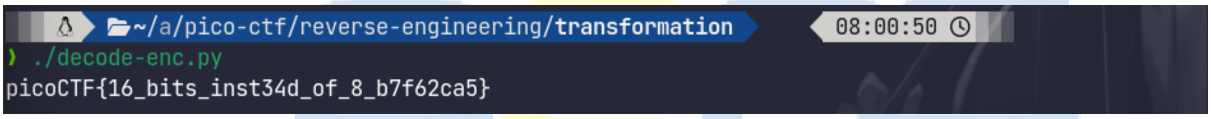
```
灩捯䍔䙻ㄶ形楴獟楮獴㌴摟潦弸形㝦㘲捡㕽
```

Coba kita coba bikin script untuk decode tulisan tersebut

Decode-enc.py



```python
#!/usr/bin/env python3
enc = "灩捯䍔䙻ㄶ形楴獟楮獴㌴摟潦弸形㝦㘲捡㕽"

decoded = ""

for c in enc:
    val = ord(c)

    high = val >> 8
    low  = val & 0xff

    decoded += chr(high)
    decoded += chr(low)

print(decoded)
```



```
~/a/pico-ctf/reverse-engineering/transformation          08:00:50
❯ ./decode-enc.py
picoCTF{16_bits_inst34d_of_8_b7f62ca5}
```

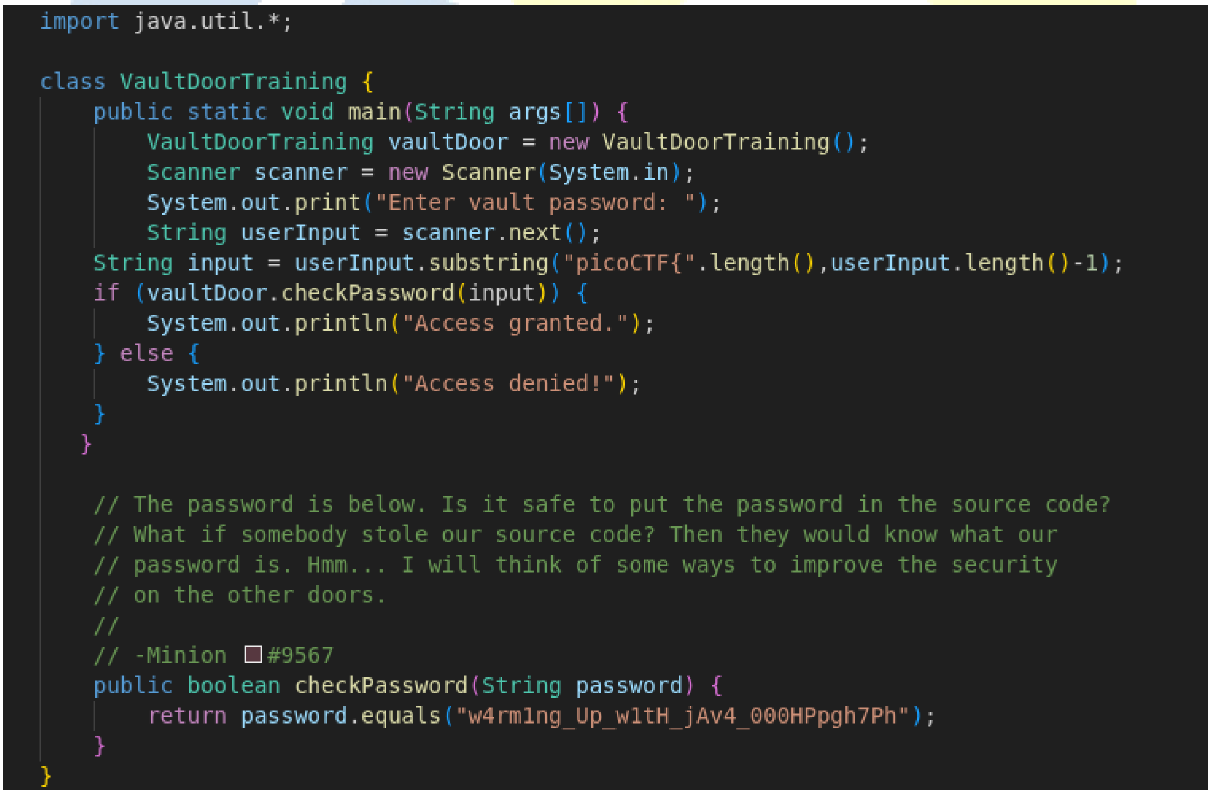
picoCTF{16_bits_inst34d_of_8_b7f62ca5}

vault-door-training

Your mission is to enter Dr. Evil's laboratory and retrieve the blueprints for his Doomsday Project. The laboratory is protected by a series of locked vault doors. Each door is controlled by a computer and requires a password to open. Unfortunately, our undercover agents have not been able to obtain the secret passwords for the vault doors, but one of our junior agents obtained the source code for each vault's computer! You will need to read the source code for each level to figure out what the password is for that vault door. As a warmup, we have created a replica vault in our training facility.The source code for the training vault is here:

Attachment: VaultDoorTraining.java

Kita diberikan file java petunjuk yang diberikan adalah kita disuruh lihat source code tersebut apakah ada yang salah



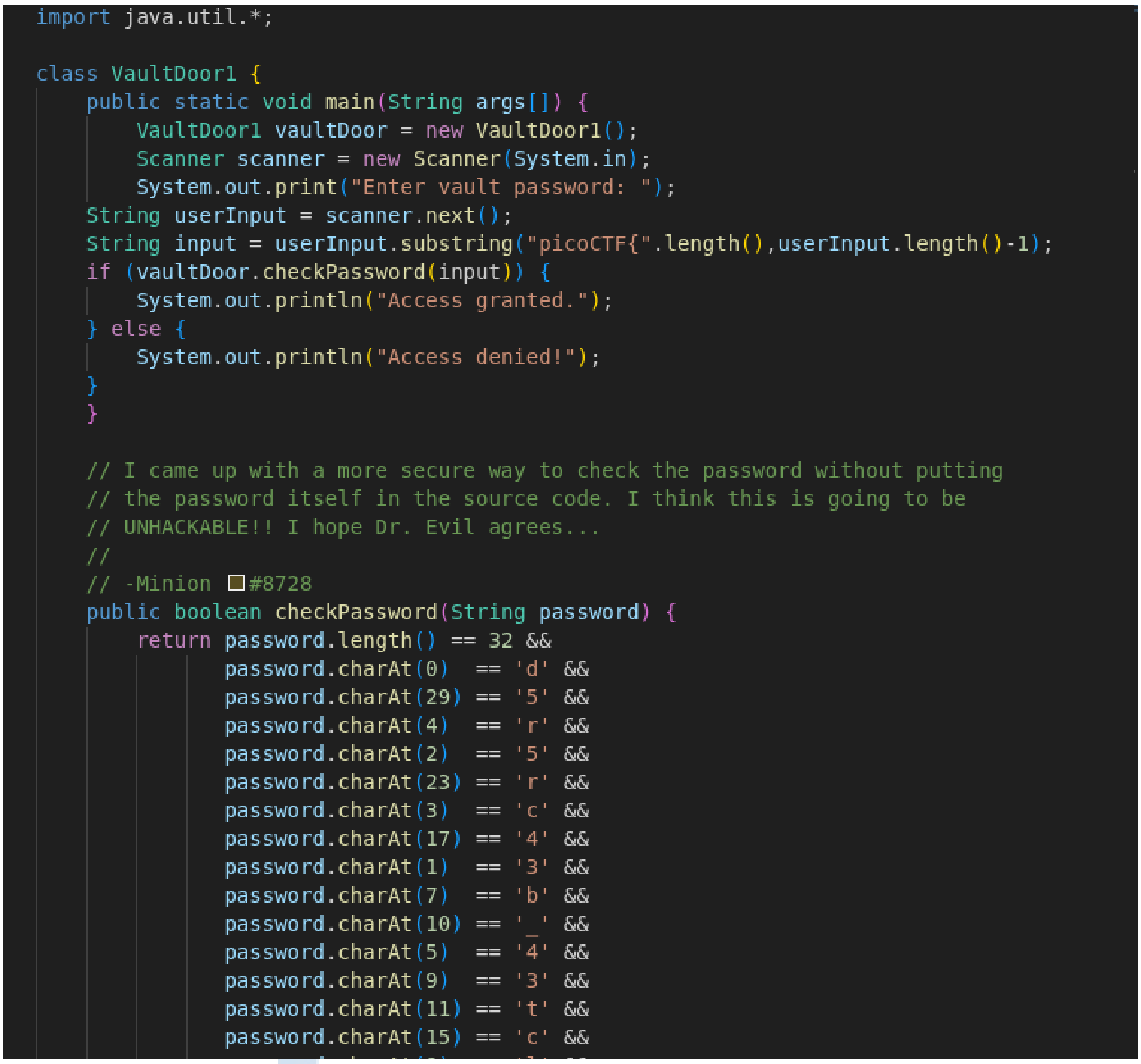
```java
import java.util.*;

class VaultDoorTraining {
    public static void main(String args[]) {
        VaultDoorTraining vaultDoor = new VaultDoorTraining();
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter vault password: ");
        String userInput = scanner.next();
    String input = userInput.substring("picoCTF{".length(),userInput.length()-1);
    if (vaultDoor.checkPassword(input)) {
        System.out.println("Access granted.");
    } else {
        System.out.println("Access denied!");
    }
   }

    // The password is below. Is it safe to put the password in the source code?
    // What if somebody stole our source code? Then they would know what our
    // password is. Hmm... I will think of some ways to improve the security
    // on the other doors.
    //
    // -Minion ■#9567
    public boolean checkPassword(String password) {
        return password.equals("w4rm1ng_Up_w1tH_jAv4_000HPpgh7Ph");
    }
}
```

Kesalahan kode tersebut adalah menaruh check password pada source code

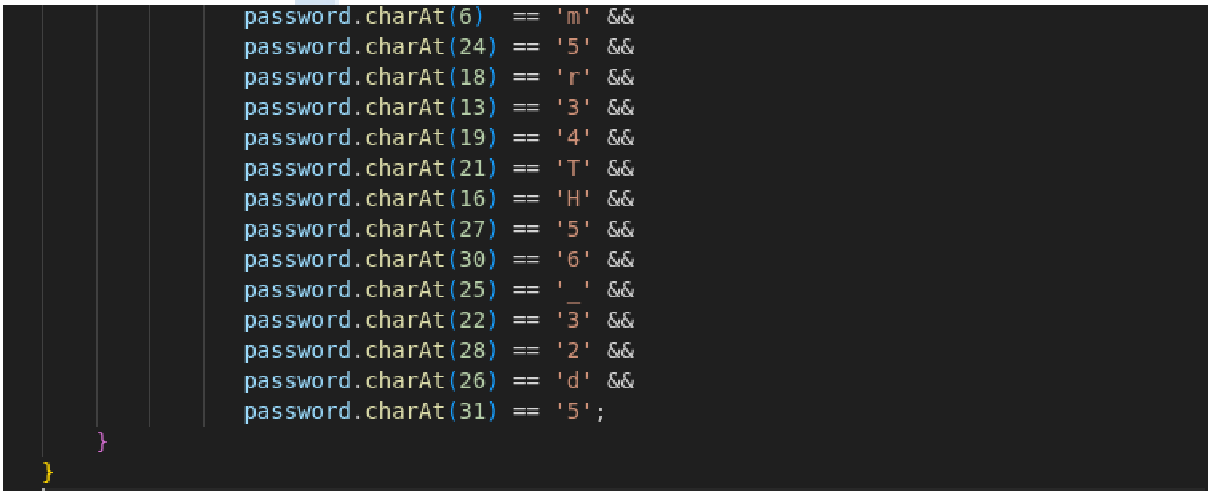
picoCTF{w4rm1ng_Up_w1tH_jAv4_000HPpgh7Ph}

vault-door-1
This vault uses some complicated arrays! I hope you can make sense of it, special agent. The source code for this vault is here
Attachment: VaultDoor1.java

Kita diberikan file java petunjuk yang diberikan adalah kita disuruh lihat source code tersebut apakah ada yang salah



```java
import java.util.*;

class VaultDoor1 {
    public static void main(String args[]) {
        VaultDoor1 vaultDoor = new VaultDoor1();
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter vault password: ");
    String userInput = scanner.next();
    String input = userInput.substring("picoCTF{".length(),userInput.length()-1);
    if (vaultDoor.checkPassword(input)) {
        System.out.println("Access granted.");
    } else {
        System.out.println("Access denied!");
    }
    }

    // I came up with a more secure way to check the password without putting
    // the password itself in the source code. I think this is going to be
    // UNHACKABLE!! I hope Dr. Evil agrees...
    //
    // -Minion ■#8728
    public boolean checkPassword(String password) {
        return password.length() == 32 &&
               password.charAt(0)  == 'd' &&
               password.charAt(29) == '5' &&
               password.charAt(4)  == 'r' &&
               password.charAt(2)  == '5' &&
               password.charAt(23) == 'r' &&
               password.charAt(3)  == 'c' &&
               password.charAt(17) == '4' &&
               password.charAt(1)  == '3' &&
               password.charAt(7)  == 'b' &&
               password.charAt(10) == '_' &&
               password.charAt(5)  == '4' &&
               password.charAt(9)  == '3' &&
               password.charAt(11) == 't' &&
               password.charAt(15) == 'c' &&
```



```java
               password.charAt(6)  == 'm' &&
               password.charAt(24) == '5' &&
               password.charAt(18) == 'r' &&
               password.charAt(13) == '3' &&
               password.charAt(19) == '4' &&
               password.charAt(21) == 'T' &&
               password.charAt(16) == 'H' &&
               password.charAt(27) == '5' &&
               password.charAt(30) == '6' &&
               password.charAt(25) == '_' &&
               password.charAt(22) == '3' &&
               password.charAt(28) == '2' &&
               password.charAt(26) == 'd' &&
               password.charAt(31) == '5';
    }
}
```

Kesalahan kode tersebut adalah menaruh check password pada source code

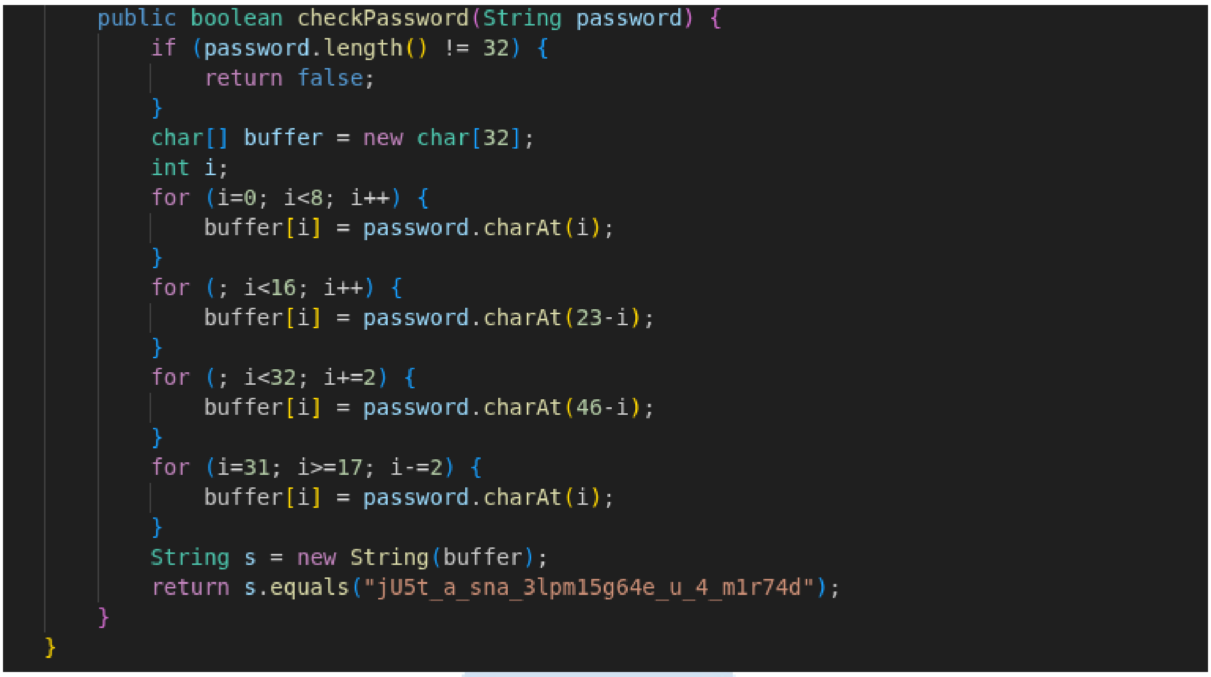
picoCTF{d35cr4mbl3_tH3_cH4r4cT3r5_5d2f65}

vault-door-3

This vault uses for-loops and byte arrays.The source code for this vault is here:

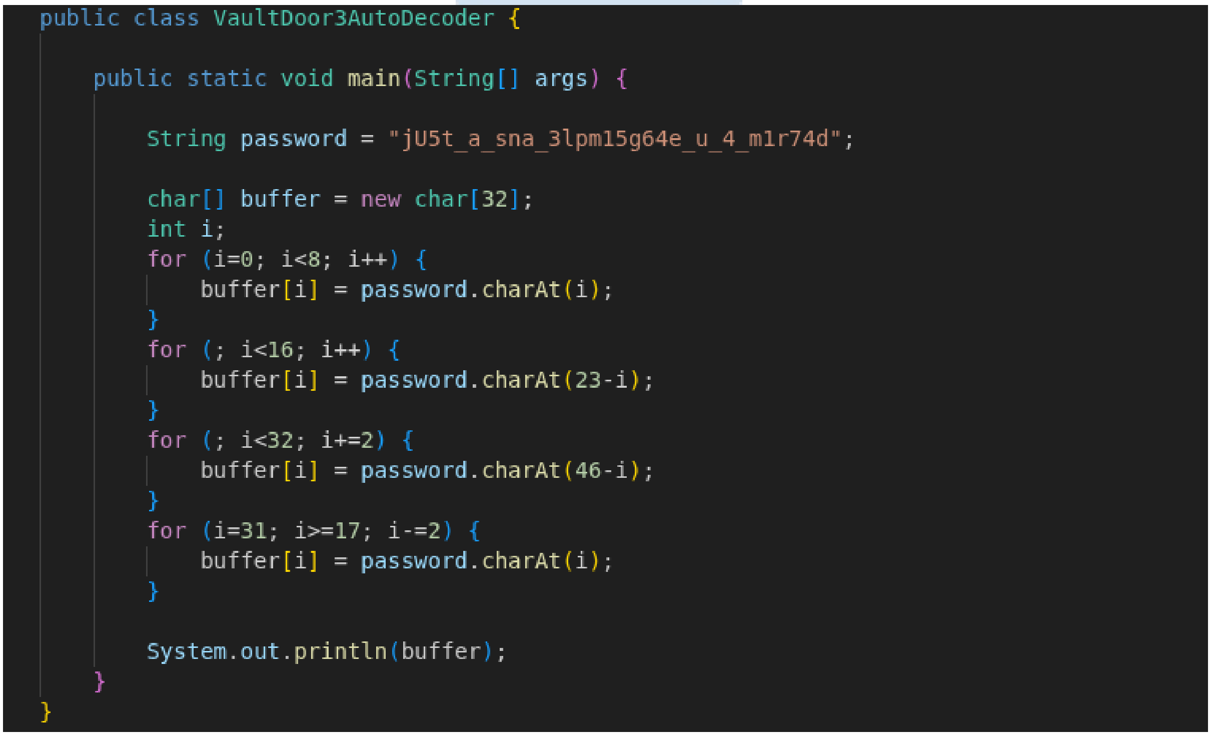
Attachment: VaultDoor3.java

Kita diberikan file java petunjuk yang diberikan adalah kita disuruh lihat source code tersebut apakah ada yang salah

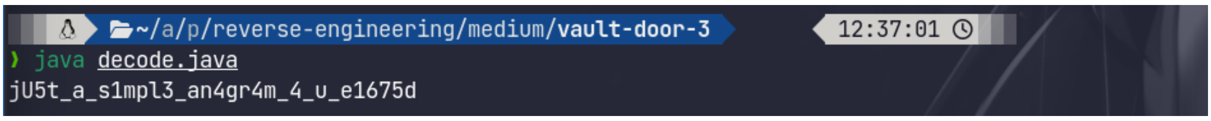


```java
    public boolean checkPassword(String password) {
        if (password.length() != 32) {
            return false;
        }
        char[] buffer = new char[32];
        int i;
        for (i=0; i<8; i++) {
            buffer[i] = password.charAt(i);
        }
        for (; i<16; i++) {
            buffer[i] = password.charAt(23-i);
        }
        for (; i<32; i+=2) {
            buffer[i] = password.charAt(46-i);
        }
        for (i=31; i>=17; i-=2) {
            buffer[i] = password.charAt(i);
        }
        String s = new String(buffer);
        return s.equals("jU5t_a_sna_3lpm15g64e_u_4_m1r74d");
    }
}
```

sepertinya untuk level ini posisi passwordnya diacak, kita coba ubah script tersebut untuk mengetahui apa password yang sebenarnya



```java
public class VaultDoor3AutoDecoder {

    public static void main(String[] args) {

        String password = "jU5t_a_sna_3lpm15g64e_u_4_m1r74d";

        char[] buffer = new char[32];
        int i;
        for (i=0; i<8; i++) {
            buffer[i] = password.charAt(i);
        }
        for (; i<16; i++) {
            buffer[i] = password.charAt(23-i);
        }
        for (; i<32; i+=2) {
            buffer[i] = password.charAt(46-i);
        }
        for (i=31; i>=17; i-=2) {
            buffer[i] = password.charAt(i);
        }

        System.out.println(buffer);
    }
}
```

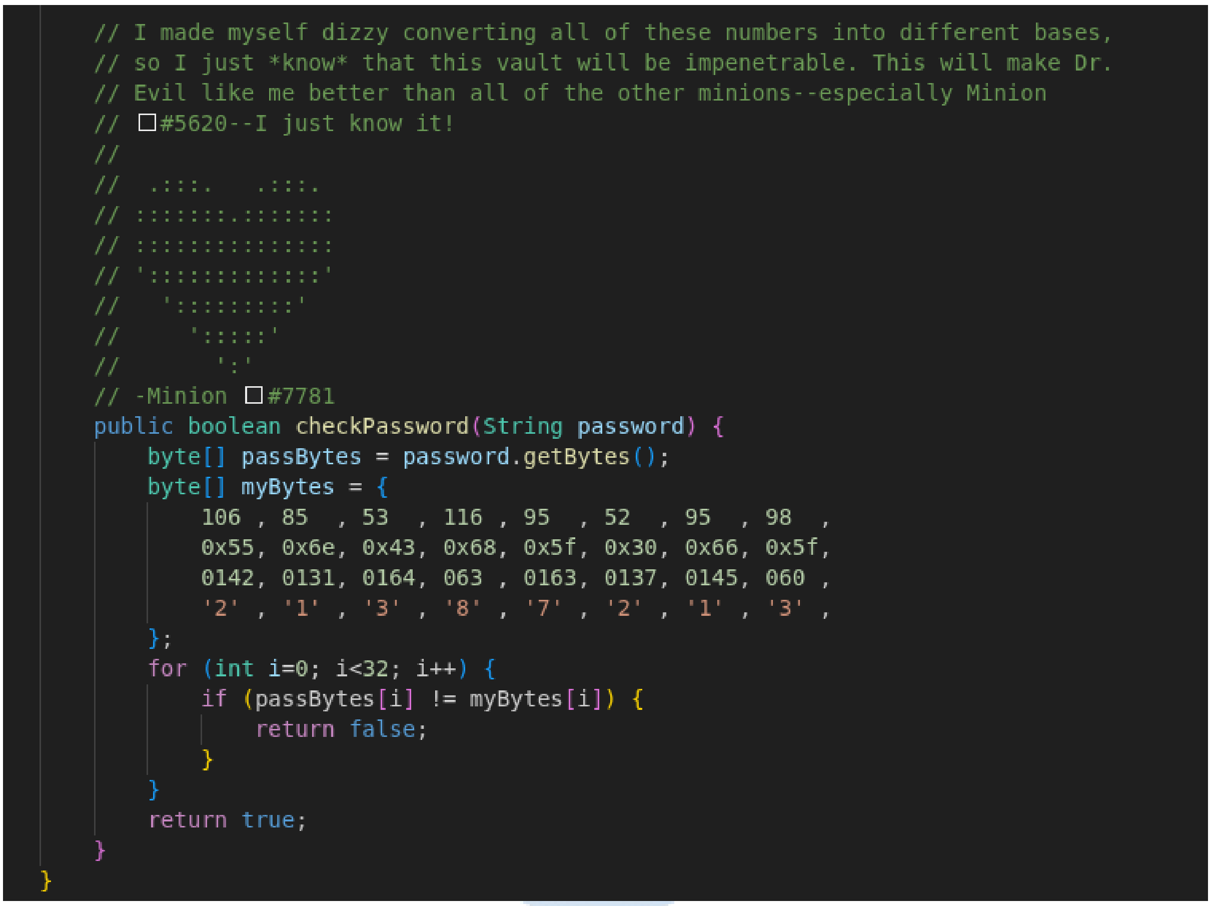


```
~/a/p/reverse-engineering/medium/vault-door-3          12:37:01
❯ java decode.java
jU5t_a_s1mpl3_an4gr4m_4_u_e1675d
```

picoCTF{jU5t_a_s1mpl3_an4gr4m_4_u_e1675d}

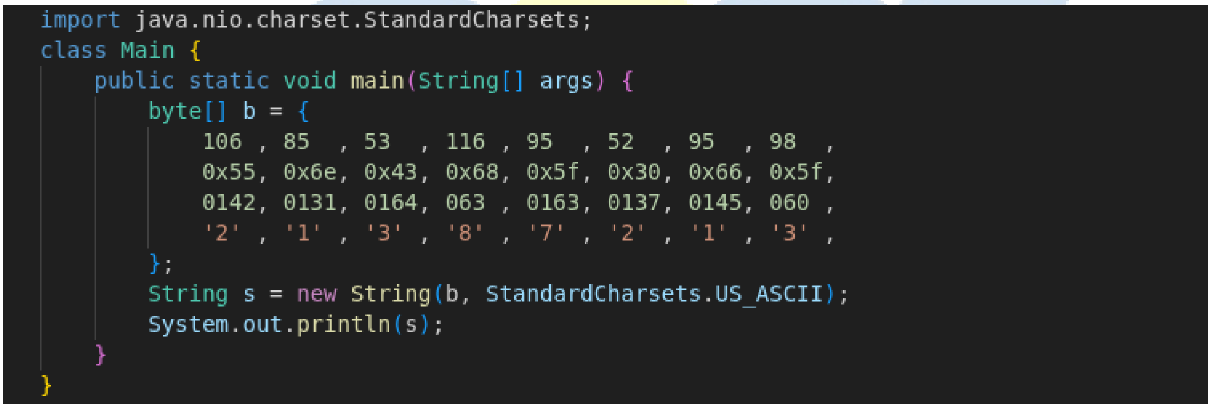
vault-door-4
This vault uses ASCII encoding for the password.The source code for this vault is here:
Attachment: VaultDoor4.java

Kita diberikan file java petunjuk yang diberikan adalah kita disuruh lihat source code tersebut apakah ada yang salah

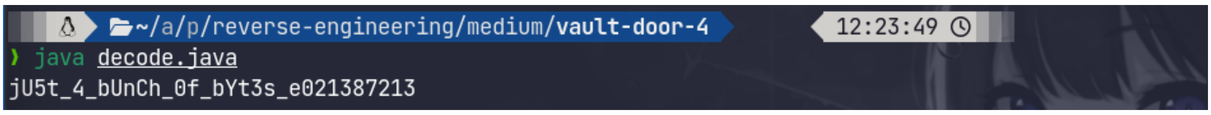


```
    // I made myself dizzy converting all of these numbers into different bases,
    // so I just *know* that this vault will be impenetrable. This will make Dr.
    // Evil like me better than all of the other minions--especially Minion
    // □#5620--I just know it!
    //
    //  .:::.   .:::.
    // :::::::.:::::::
    // :::::::::::::::
    // ':::::::::::::'
    //   ':::::::::'
    //     ':::::'
    //       ':'
    // -Minion □#7781
    public boolean checkPassword(String password) {
        byte[] passBytes = password.getBytes();
        byte[] myBytes = {
            106 , 85  , 53  , 116 , 95  , 52  , 95  , 98  ,
            0x55, 0x6e, 0x43, 0x68, 0x5f, 0x30, 0x66, 0x5f,
            0142, 0131, 0164, 063 , 0163, 0137, 0145, 060 ,
            '2' , '1' , '3' , '8' , '7' , '2' , '1' , '3' ,
        };
        for (int i=0; i<32; i++) {
            if (passBytes[i] != myBytes[i]) {
                return false;
            }
        }
        return true;
    }
}
```

Sepertinya fungsi check password pada source code dia tidak menggunakan ascii coba kita bikin script untuk menjadi ke ascii



```
import java.nio.charset.StandardCharsets;
class Main {
    public static void main(String[] args) {
        byte[] b = {
            106 , 85  , 53  , 116 , 95  , 52  , 95  , 98  ,
            0x55, 0x6e, 0x43, 0x68, 0x5f, 0x30, 0x66, 0x5f,
            0142, 0131, 0164, 063 , 0163, 0137, 0145, 060 ,
            '2' , '1' , '3' , '8' , '7' , '2' , '1' , '3' ,
        };
        String s = new String(b, StandardCharsets.US_ASCII);
        System.out.println(s);
    }
}
```



```
~/a/p/reverse-engineering/medium/vault-door-4                12:23:49
❯ java decode.java
jU5t_4_bUnCh_0f_bYt3s_e021387213
```

picoCTF{jU5t_4_bUnCh_0f_bYt3s_e021387213}

vault-door-5

In the last challenge, you mastered octal (base 8), decimal (base 10), and hexadecimal (base 16) numbers, but this vault door uses a different change of base as well as URL encoding!The source code for this vault is here:

Attachment: VaultDoor5.java