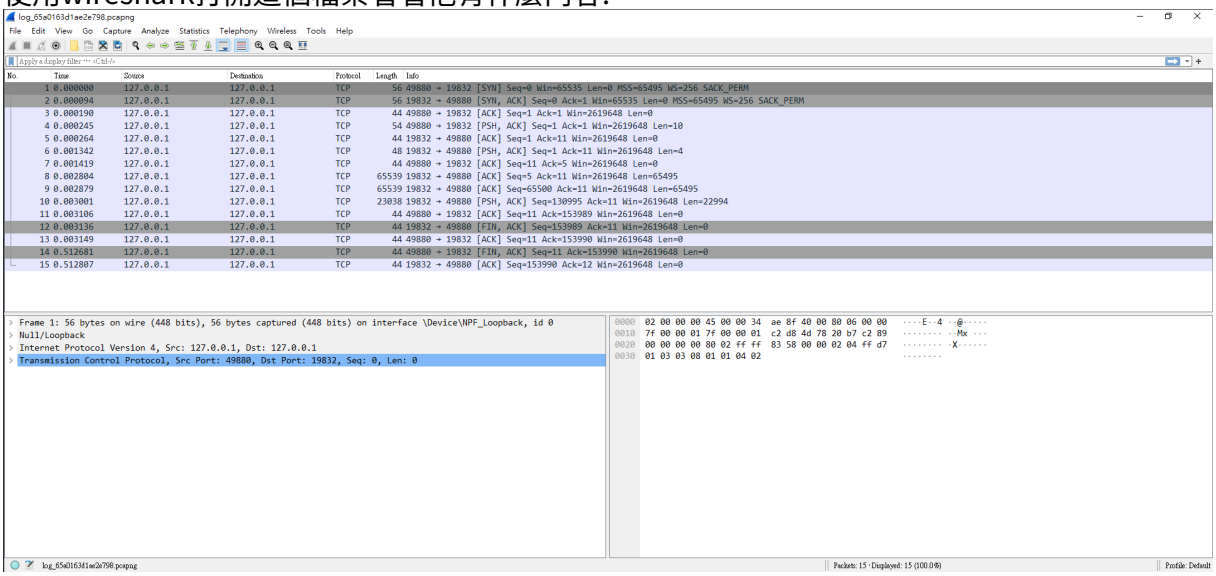
trojan

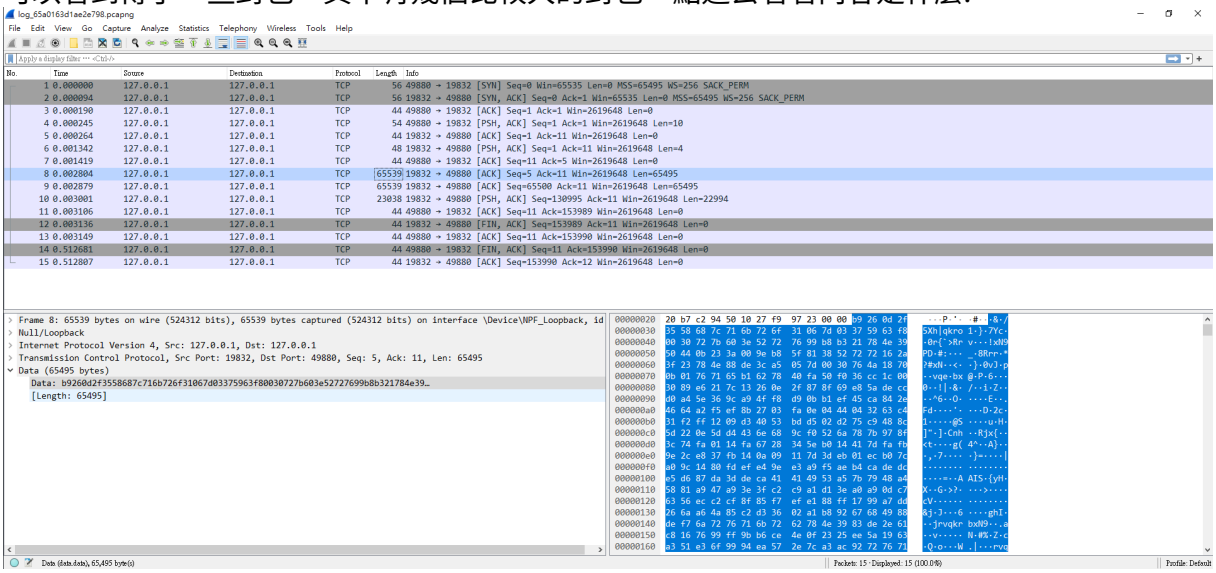
題目給了兩個檔案，一個是執行檔，一個是wireshark的檔案。

pcapng

使用wireshark打開這個檔案看看他有什麼內容:



可以看到傳了一些封包，其中有幾個比較大的封包，點進去看看內容是什麼:



看起來似乎是一些資料，但被加密過了，沒辦法看到想要的內容

reverse

看記錄檔無法看出什麼資訊，因此來看看反編譯出的main function，這邊根據理解已經對一些function重新命

```
3
4  void main(void)
5
6  {
7    void **ppvVar1;
8    undefined auStack184 [32];
9    undefined8 *local_98;
10   undefined8 local_90 [5];
11   undefined8 local_68 [10];
12   ulonglong local_18;
13
14   local_18 = DAT_14000e010 ^ (ulonglong)auStack184;
15   FUN_1400017e0((undefined *)local_68,0x48);
16   local_98 = local_90;
17   ppvVar1 = (void **)server(local_98,"127.0.0.1");
18                     // port 19832
19   listen(local_68,ppvVar1,19832);
20   FUN_140007d50((longlong)local_68,do_something);
21   FUN_140007e30((_String_val<struct_std::_Simple_types<char>_> *)local_68);
22   do {
23     FUN_140002fa0((longlong)&near_path);
24     Sleep(14400000);
25   } while( true );
26 }
27
```

看起來這邊是在建立socket連線，然後程式就會在後臺掛著，每隔14400秒做一次事情。
點進去看看這個function實際上在做什麼。

```
void FUN_140002fa0(longlong f_name_pre_pos)

{
  int cx;
  int cy;
  HDC hdc;
  HDC hdc_00;
  HBITMAP h;
  HGDIOBJ h_00;
  undefined8 path;
  undefined auStackY200 [32];
  undefined8 local_40 [3];
  undefined local_28 [16];
  ulonglong local_18;

  local_18 = DAT_14000e010 ^ (ulonglong)auStackY200;
  cx = *(int *)(f_name_pre_pos + 0x28);
  cy = *(int *)(f_name_pre_pos + 0x2c);
  if ((0 < cx) && (0 < cy)) {
                    // screenshot
    hdc = GetDC((HWND)0x0);
    hdc_00 = CreateCompatibleDC(hdc);
    h = CreateCompatibleBitmap(hdc,cx,cy);
    h_00 = SelectObject(hdc_00,h);
    BitBlt(hdc_00,0,0,cx,cy,hdc,0,0,0xcc0020);
    FUN_1400017e0((undefined *)local_40,0x18);
    FUN_1400025a0(local_40,h,0);
    save_img((longlong)L"image/png",local_28);
    path = FUN_140004c00((undefined8 *)(f_name_pre_pos + 8));
    save_to_file((longlong)local_40,path,local_28,0);
    SelectObject(hdc_00,h_00);
    DeleteDC(hdc_00);
    ReleaseDC((HWND)0x0,hdc);
    DeleteObject(h);
    FUN_1400026f0(local_40);
  }
  stack_chk(local_18 ^ (ulonglong)auStackY200);
  return;
}

```

GetDC這個function是Windows的API，給與參數0代表是做螢幕截圖，到這邊可以推斷剛才wireshark內比較大
可以推測題目之所以叫trojan，就是因爲這個程式模仿了一個木馬的行爲，只是資料是送到本機端。

send

回到main function，跳到do_something的function內，看看做了什麼事情:

```
void do_something(undefined8 param_1)

{
  undefined auStack664 [32];
  char local_278;
  int local_274;
  void *key_data1;
  undefined4 local_268;
  void *local_260;
  undefined8 *local_258;
  undefined8 *local_250;
  undefined8 local_248;
  void *local_240 [4];
  int key_data2 [2];
  undefined local_218 [512];
  ulonglong local_18;

  local_18 = DAT_14000e010 ^ (ulonglong)auStack664;
  local_268 = 0x200;
  local_274 = send(param_1,local_218,0x200,0);
  if (0 < local_274) {
    local_258 = server(local_240,"cDqr0hUUz1");
    local_250 = local_258;
    local_278 = operator!=<>((_String_val<struct_std::_Simple_types<char>_> *)local_258,local_218);
    ~basic_string<>(local_240);
    if (local_278 != '\0') {
      Ordinal_3(param_1);
      goto LAB_1400016f3;
    }
    key_data1 = (void *)FUN_140003180((longlong)&near_path,key_data2);
    local_274 = Ordinal_19(param_1,key_data2,4,0);
    if (local_274 == -1) {
      Ordinal_3(param_1);
      goto LAB_1400016f3;
    }
    maybe_flag((longlong)key_data1,key_data2[0]);
    local_274 = Ordinal_19(param_1,key_data1,key_data2[0],0);
    if (local_274 == -1) {
      Ordinal_3(param_1);
      goto LAB_1400016f3;
    }
    local_260 = key_data1;
    free(key_data1);
    if (local_260 == (void *)0x0) {
      local_248 = 0;
    }
    else {
      key_data1 = (void *)0x8123;
      local_248 = 0x8123;
    }
  }
  Ordinal_3(param_1);
LAB_1400016f3:
  stack_chk(local_18 ^ (ulonglong)auStack664);
  return;
}
```

仔細看過之後我覺得這邊應該是在處理送資料的部分。

encrypt

繼續往下觀察，可以看到一個可疑的function，裡面做了XOR運算，大致可以猜測他是對輸入的參數(指標)做加

```
void maybe_flag(longlong param_1,int param_2)

{
  longlong lVar1;
  byte *pbVar2;
  byte *pbVar3;
  int local_48 [2];
  longlong local_40;
  byte local_38 [24];
  ulonglong local_20;

  local_20 = DAT_14000e010 ^ (ulonglong)local_48;
  pbVar2 = (byte *)"0vCh8RrvqkrbxN9Q7Ydx";
  pbVar3 = local_38;
  for (lVar1 = 0x15; lVar1 != 0; lVar1 = lVar1 + -1) {
    *pbVar3 = *pbVar2;
    pbVar2 = pbVar2 + 1;
    pbVar3 = pbVar3 + 1;
  }
  for (local_48[0] = 0; local_48[0] < param_2; local_48[0] = local_48[0] + 1) {
    local_40 = (longlong)local_48[0];
    *(byte *)(param_1 + local_48[0]) = *(byte *)(param_1 + local_40) ^ local_38[(ulonglong)(longlong)local_48[0] % 0x15];
  }
  stack_chk(local_20 ^ (ulonglong)local_48);
  return;
}

```

decrypt

跟著這個想法，試著將wireshark抓到的資料拿出來，照著他的流程嘗試拿來解密。

```python
with open("data", "r") as f:
    data = f.read()

# split data to bytes
data = [data[i:i+2] for i in range(0, len(data), 2)]

x = "0vCh8RrvqkrbxN9Q7Ydx"
x = [ord(i) for i in x]
x.append(0) # '\0' in C str

data = [int(i, base=16) for i in data]
```

```
for i, v in enumerate(data):
    data[i] = v ^ x[i%0x15]

with open("output", "wb") as f:
    f.write(bytes(data))
```

solve

執行上面的程式，並觀察output，是一張圖片，裡面有flag:

```
$ python3 test.py
$ file output
output: PNG image data, 1920 x 1080, 8-bit/color RGBA, non-interlaced
```