

## Tim : Padepokan Siluman Kadal(49)

1. Ayala Septama Rahanda – 23081010071@student.upnjatim.ac.id
2. Billy Ramadhani – 22081010078@student.upnjatim.ac.id
3. Muhammad Rafli Alviro – rafli Alviro84@gmail.com

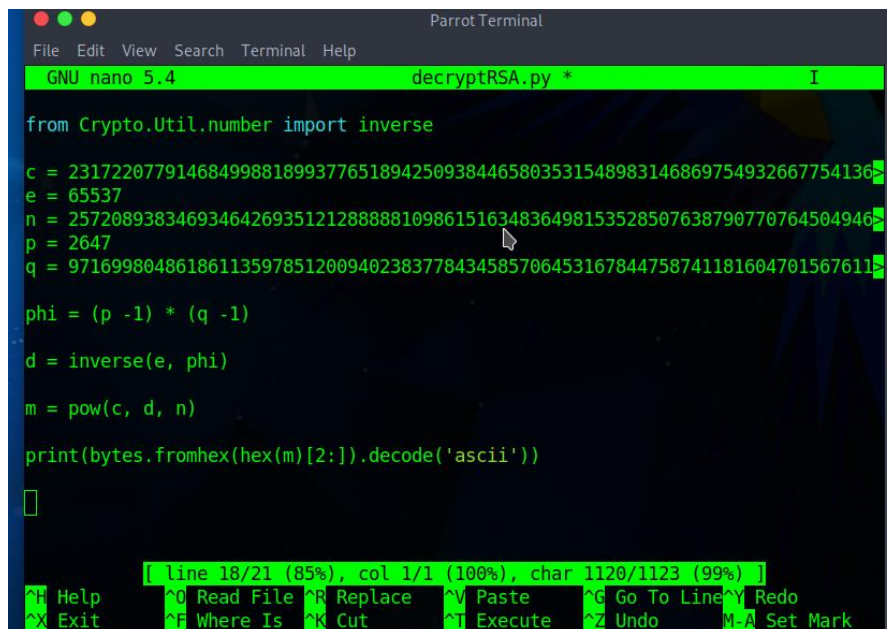
## Write Up

### Bonus:

Terdapat flag yang bisa langsung disubmit.

### 101 Cryptography:

Terdapat file chall.txt yang berisi cipher(c), public key E(e), public key value(n). Setelah itu mencari nilai 2 bilangan factor prima yang menghasilkan nilai sama dengan n, setelah berhasil mencari nilai 2 bilangan ini selanjutnya membuat script untuk menjalankan algoritma untuk mendecode enkripsi RSA ini.



```
from Crypto.Util.number import inverse

c = 231722077914684998818993776518942509384465803531548983146869754932667754136
e = 65537
n = 257208938346934642693512128888810986151634836498153528507638790770764504946
p = 2647
q = 971699804861861135978512009402383778434585706453167844758741181604701567611

phi = (p - 1) * (q - 1)
d = inverse(e, phi)
m = pow(c, d, n)

print(bytes.fromhex(hex(m)[2:]).decode('ascii'))
```

Kemudian script dijalankan dan menghasilkan output  
flag: **TSA{Crypto\_101\_d5b55ff525198ba6}**

### 101 Forensic:

Terdapat file pcap, kemudian kami melakukan analisis menggunakan wireshark. terdapat banyak paket dengan protocol ICMP serta informasi request dan reply. Setelah

menganalisanya kami mendapatkan bahwa terdapat banyak header dan footer PNG pada paket-paket ini dan ada 3 duplikasi pada setiap paket. Sehingga kami filter untuk mendapatkan semua raw data dari setiap paket reply ke dalam 1 file dengan script:

```
GNU nano 5.4      pcapICMP.py *
from scapy.all import *

ps = rdpcap("/home/xrrr/Downloads/packet.pcap")
i = 0
[ ]
for p in ps:
    if i % 2 == 0:
        paket = p.load.hex()[128:160]
        print(paket)
    i = i+1
```

[ line 5/13 (38%), col 1/1 (100%), char 79/167 (47%) ]

Help Read File Replace Paste Go To Line Redo  
Exit Where Is Cut Evacuate Undo Set Mark

[illegible]

Untuk output yang dihasilkan sudah sesuai dari load semua paket reply yang telah difilter, tetapi masih belum fix karena file masih berisi gabungan beberapa file dan terdapat hex 0000 sebelum header PNG, maka akan dilakukan perulangan untuk mendecode dan memfilter untuk setiap file, sehingga script tadi perlu dimodifikasi:

```

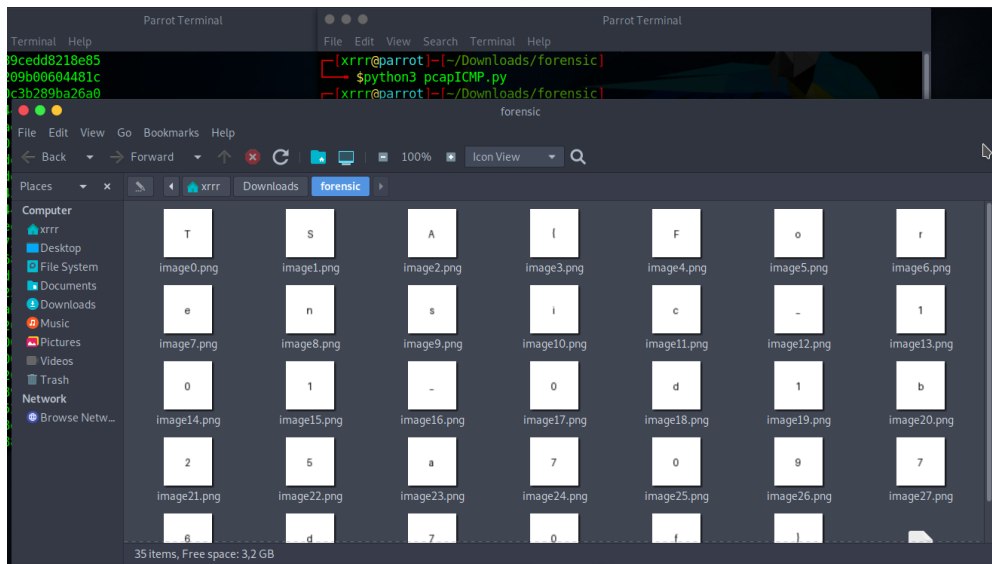
GNU nano 5.4.4 pcapICMP.py * 1
from scapy.all import *

ps = rdpcap("/home/xrrrr/Downloads/packet.pcap")
i = 0
png_data = []
count = 0

for p in ps:
    if i % 2 == 0:
        paket = p.load_hex()[128:160]
        if paket == "00000000000000000000000000000000":
            if png_data:
                with open(f"image{count}.png", "wb") as f:
                    f.write(bytes.fromhex("".join(png_data)))
                    count += 1
                    png_data = []
            else:
                png_data.append(paket)
        i = i + 1
    if png_data:
        with open(f"image{count}.png", "wb") as f:
            f.write(bytes.fromhex("".join(png_data)))

line 7/23 (38%), col 1/1 (100%), char 103/502 (20%)
[?] Help [F2] Read File [F3] Replace [F4] Paste [F5] Go To Line [F6] Redo
[Ctrl+H] Where Is [Ctrl+R] Cut [Ctrl+N] Execute [Ctrl+Z] Undo [Ctrl+M] Set Mark

```



Setelah script dijalankan maka akan menghasilkan beberapa file PNG yang urut dan setiap file berisi karakter flag yaitu: **TSA{Forensic\_101\_0d1b25a70976d70f}**

### 101 Reversing:

Terdapat file main yang jika dijalankan maka meminta input untuk Flag. Kami melakukan analisa menggunakan strings, gdb dan mendapati beberapa function yang ada pada file tersebut kemudian menggunakan ghidra untuk melakukan analisa lebih lanjut terkait file main ini. Pada ghidra kami mendapati bahwa cara kerja file ini adalah user melakukan input kemudian akan disimpan pada variable p yang kemudian akan program akan melakukan split dan membagi 2 Flag ke dalam variable l dan r kemudian program akan memanggil function cl dan cr untuk membandingkan apakah cl dan cr sama jika sama maka program akan menghasilkan CORRECT dan memberikan flagnya.

Function m (main dari program ini):

```

1  void m(void *param_1,int param_2,char *param_3,int param_4,int param_5)
2  {
3      int iVar1;
4      size_t sVar2;
5      undefined4 in_register_00000034;
6
7      printf("[>] FLAG: ",CONCAT44(in_register_00000034,param_2));
8      _isoc99_scanf(5DAT_0010200f,p);
9      sVar2 = strlen(p);
10     iVar1 = (int)sVar2 / 2;
11     for (i = 0; i < iVar1; i = i + 1) {
12         l[i] = p[i];
13     }
14     l[i] = 0;
15     k = 0;
16     for (i = iVar1; i <= (int)sVar2; i = i + 1) {
17         r[k] = p[i];
18         k = k + 1;
19     }
20     cl();
21     cr();
22     printf("[CORRECT] %s\n",p);
23     return;
24 }
25
26
27

```

Function cl pada program:

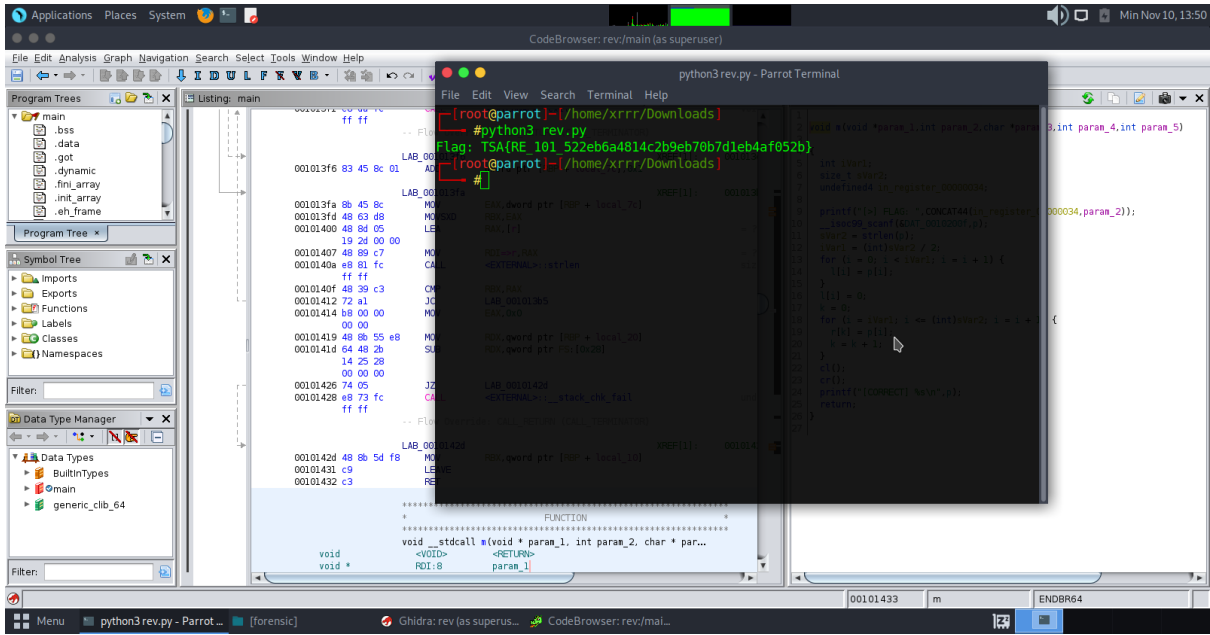
The screenshot shows a debugger window with two panes. The left pane displays assembly code for function 'cl', including instructions like 'AL:1', 'frags: frags', 'CWD: CWD', 'JMP', and 'Flow Override: CALL\_RETURN (CALL\_TERMINATOR)'. The right pane shows the decompiled code for 'cl', which includes a loop structure with 'Stack' and 'local' variables, and a 'long' variable.

Function cr pada program:

The screenshot shows a debugger window with two panes. The left pane displays assembly code for function 'cr', including instructions like 'MOV', 'LEA', 'MOVZX', 'XOR', 'MOV', 'CMP', 'JZ', 'CALL', and 'Flow Override: CALL\_RETURN (CALL\_TERMINATOR)'. The right pane shows the decompiled code for 'cr', which includes a loop structure with 'local' variables and a 'long' variable.

Setelah mengetahui variabel2 yang ada pada kedua function tersebut selanjutnya membuat script untuk melakukan looping berdasarkan logika yang ada pada function m.

The screenshot shows a code editor window with a Python script. The script is titled 'nano rev.py' and contains a loop structure that iterates over a range of values, performing calculations and printing results. The script is being executed in a Parrot Terminal window.



Setelah script jadi dan dijalankan maka didapatkan flag berikut  
flag: **TSA{RE\_101\_522eb6a4814c2b9eb70b7d1eb4af052b}**