

LẬP TRÌNH HỆ THỐNG– LỚP NT209.L21.ANTN

RE CHALLENGES 3: Catalina

Giảng viên hướng dẫn	Phạm Văn Hậu		ĐIỂM
Sinh viên thực hiện 1	Trần Đức Lương	19521815	

Đây là file thực thi PE 64-bit. Có thể thấy chương trình yêu cầu nhập password như là tham số đầu vào. Thực hiện chạy thử với password “*test*” thì thấy chương trình báo flag không hợp lệ.

```
(janlele91@kali)-[~/Documents/RE Challenges/Release_3]
$ ./crackme
Welcome to crackme N1
Usage:
./crackme <password>

(janlele91@kali)-[~/Documents/RE Challenges/Release_3]
$ ./crackme test
Invalid flag, try again
```

Mở file *crackme* bằng IDA Pro thực hiện quá trình dịch ngược. Mở Subview -> Strings thì thấy xuất hiện dòng chữ “*Congratulations...*” chính là mục tiêu chúng ta cần hướng đến.

```
.rodata:000000... 00000033      C      Congratulations !! you solved the first challenge.
.rodata:000000... 00000018      C      Invalid flag, try again
```

Thực hiện truy xuất strings trên thì thấy nó nằm trong hàm *main*. Bắt đầu phân tích hàm *main*:

```
v12 = "JTQSRyZKSB05Dh9JgH6fQJIVjJ04UpA7ezxMIHcvpX6X70NjHW4x1xSHHMuLDjCJbz19ITfgeLbTDLExZENyYrAzn7ehjAMuZf1siTB4HBLgy"
      "gpK38LHCq4Uvpqg0xeoh72AVgDOYS8HU9xg";
```

```

50  v11[29] = 2;
51  v11[30] = 4;
52  v11[31] = 5;
53  v9[0] = 0LL;
54  v9[1] = 0LL;
55  v9[2] = 0LL;
56  v9[3] = 0LL;
57  v10 = 0;
58  v15 = 0;
59  for ( i = 0; i <= 31; ++i )
60  {
61      *((_BYTE *)v9 + i) = v12[v15];
62      v15 += v11[i] + 1;
63  }
64  sub_55DA95032482((__int64)v9, (__int64)v8, 32);

```

Chương trình thực hiện tạo giá trị **v8** từ **v12** cho trước thông qua **v9**. Ta không cần quan tâm phần này vì không liên quan đến password chúng ta nhập vào.

```

if ( v13 == 24 )
    puts("Congratulations !! you solved the first challenge.");
else
    puts("Invalid flag, try again");

```

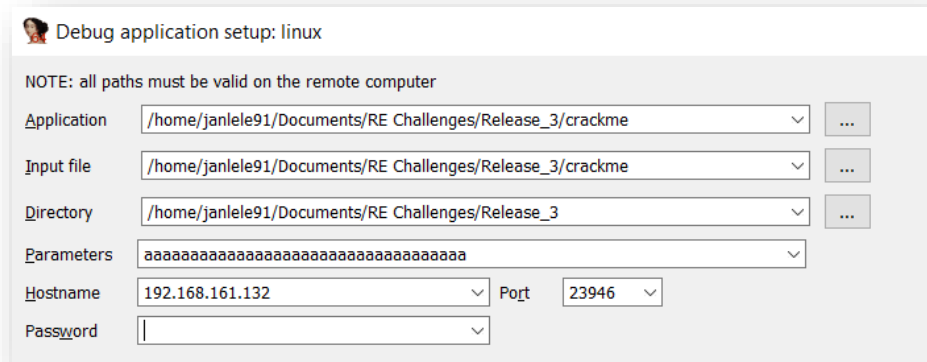
Có thể nhìn thấy rằng, chương trình sẽ chỉ in ra dòng chữ thành công khi giá trị **v13** == 24.

```

v15 = 0;
v13 = 0;
while ( v15 <= 23 )
{
    v8[v15] ^= 0x41u;
    v4 = (unsigned __int8)v8[v15];
    v5 = a2[1];
    v6 = strlen(v5);
    if ( v6 >= v15 )
        v7 = v15;
    else
        v7 = strlen(a2[1]);
    v13 += v4 == v5[v7];
    ++v15;
}

```

Giá trị $v13 == 24$ khi và chỉ khi $v4 == v5[v7]$ trong cả 24 lần lặp của vòng while. Cụ thể $v5 = a2[1]$ chính là giá trị password chúng ta nhập vào. Cứ mỗi lần lặp, chương trình lấy từng kí tự của $v8$ sau khi xor với $0x41$ gán vào $v4$ và so sánh với kí tự ở vị trí tương ứng của password $v5$. Nếu bằng nhau thì giá trị $v13$ tăng thêm 1. Điều này có nghĩa là password $v5$ cần tìm sẽ có tối thiểu 24 kí tự và chứa chuỗi 24 kí tự đầu tiên chính bằng $v8$ sau khi vòng while kết thúc.



Debug application setup: linux

NOTE: all paths must be valid on the remote computer

Application: /home/janlele91/Documents/RE Challenges/Release_3/crackme

Input file: /home/janlele91/Documents/RE Challenges/Release_3/crackme

Directory: /home/janlele91/Documents/RE Challenges/Release_3

Parameters: aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa

Hostname: 192.168.161.132 Port: 23946

Password:

Thực hiện Linux remote debug với password đầu vào như trên tại breakpoint ngay sau vòng while vừa rồi, thì thấy giá trị của chuỗi $v8$ tại đó là “ $flag\{2020_sana_sa3ida:\}$ ”

```
[stack]:00007FFC3BD21CC0 aFlag2020SanaSa db 'flag{2020_sana_sa3ida:})'
```

Vậy password cần tìm là bắt đầu bằng 24 kí tự trên. Thực hiện chạy với password = “ $flag\{2020_sana_sa3ida:\}$ ” hay “ $flag\{2020_sana_sa3ida:\})aa$ ” hay “ $flag\{2020_sana_sa3ida:\})aaaaa$ ” đều thành công.

```
(janlele91@kali) - [~/Documents/RE Challenges/Release_3]
$ ./crackme "flag{2020_sana_sa3ida:})"
Congratulations !! you solved the first challenge.

(janlele91@kali) - [~/Documents/RE Challenges/Release_3]
$ ./crackme "flag{2020_sana_sa3ida:})aa"
Congratulations !! you solved the first challenge.

(janlele91@kali) - [~/Documents/RE Challenges/Release_3]
$ ./crackme "flag{2020_sana_sa3ida:})aaaaa"
Congratulations !! you solved the first challenge.
```

