


Angstrom CTF 2021

Infinity Gauntlet

-  Challenge's Description: All **clam** needs to do is snap and finite will turn into **infinite**...

```
InfinityGauntlet:/>$. /infinity_gauntlet
Welcome to the infinity gauntlet!
If you complete the gauntlet, you'll get the flag!
=== ROUND 1 ===
foo(64832, 352) = ?
_
```

TL; DR

- Trả lời đúng các giá trị còn thiếu trong các phép toán mà chương trình đưa ra(bởi hai hàm **foo()** và **bar()**)

```
__int64 __fastcall bar(int a1, int a2, int a3)
{
    return (unsigned int)(a1 + a2 * (a3 + 1));
}
```

cách tính của hàm foo

```
1 __int64 __fastcall foo(int a1, int a2)
2 {
3     return a1 ^ (a2 + 1) ^ 0x539u;
4 }
```

cách tính của hàm bar

- Nội dung flag sẽ được đọc vào, và nó đã được tính toán qua một bước khởi tạo (các ký tự flag được xor lần lượt với [0, 17, 34, 51...], ta gọi mảng này là *i_xor*) trước khi đi vào vòng lặp chứa foo, bar (tên mặc định do IDA tạo đã được thay đổi để dễ dàng cho phân tích)

```
fgets(flagContent, 256, v7);
fclose(v8);
len_flag = strlen(flagContent, "\n");
_len_flag = len_flag;
flagContent[len_flag] = 0;
if ( len_flag )
{
    v10 = flagContent;
    v11 = 17 * len_flag;
    v12 = 0;
    do
    {
        *v10 ^= v12;
        v12 += 17;
        ++v10;
    }
    while ( (_BYTE)v12 != v11 );
}
```

các ký tự flag được xor lần lượt với i_xor = [0, 17, 34, 51...]

- Như vậy, sau khi tìm được flag, ta phải thực hiện thêm bước xor khởi tạo này để lấy lại flag ban đầu

```

77 LABEL_11:
78 printf("=== ROUND %d ===\n", (unsigned int)round_Rth);
79 result = rand();
80 if ( round_Rth <= 49 )
81 {
82     v18 = rand();
83     v19 = (unsigned __int16)(v18 + ((unsigned __int64)v18 >> 48)) - ((unsigned int)(v18 >> 31) >> 16);
84     goto LABEL_13;
85 }
86 do
87 {
88     v19 = flagContent[_len_flag & result] | ((unsigned __int8)((_len_flag & result) + round_Rth) << 8);

```

- Chú ý, sau 49 round đầu(49 câu hỏi phép tính đầu), byte cuối biến v19(gồm 2 byte, đóng vai trò như là kết quả các phép toán) sẽ chứa một byte **flagContent[unknown]**(với *unknown* là một số random, vì result= rand() ở dòng 79)
- Từ **flagContent[unknown]**, ta có thể khôi phục lại **flag[unknown]** bằng: **flag[unknown] = i_xor[unknown]^flagContent[unknown]**

Solve Script



- Trong bài này, ta sẽ sử dụng thêm thư viện pwn của bộ công cụ **pwntools** để tự động tương tác với server
- Vì ban đầu không biết được độ dài flag nên phải tìm đến khi nào mà ký tự '}' xuất hiện, lúc đó mới xác định được length của flag, trong script này, vì đã biết độ dài flag nên mình set dừng ngay sau khi đã tìm đủ tất cả các phần tử flag luôn.

```

Desktop:/>$python3 solve_InfinityGauntlet.py
[+] Opening connection to shell.actf.co on port 21700: Done
actf{snapped_away_the_end}
Desktop:/>$_

```

Bạn có thể tìm thấy source của challenge ở [đây](#) nếu không thể truy cập đến file đặt ở server, ngoài ra đây là một solve [script](#) tham khảo