逆向工程之入门指北

逆向工程是什么

- 逆向工程(Reverse Engineering)，又称反向工程，是一种技术过程，即通过对产品的分析窥见其深藏的设计原理，从而进行更深层次的破解与利用。
- 我们所要给大家介绍的CTF竞赛（Capture The Flag）中的逆向工程指的都是**软件逆向技术**，所谓软件包括但不限于windows或者linux平台的`二进制文件`（如exe文件）、安卓平台的`apk`文件。总的来说，一切能隐藏逻辑并具有执行特定功能的文件，都是我们逆向的对象。

二进制文件的前世今生

- 以我们比较熟悉的exe文件为例，当你双击运行它时，程序就能成功地跑起来。但是如果你尝试过用记事本来打开一个exe文件，你就会发现，所看到的都是乱码。exe文件（executable file）是二进制文件的一种，二进制文件大多是给计算机看的，而不是给人类看的。
- 在生成exe文件之前，首先需要程序员进行软件开发，程序员所编写的一般称为源代码。以一个C语言程序为例（或许你还没有学过C语言，但这并不要紧），C源代码是人类能看懂的语言，但执行各种命令、进行复杂运算的CPU却无法理解。这就需要对源代码进行“翻译”，使其变成CPU所能理解的语言。这个“翻译”的过程被称为`编译`，进行这项工作的部件就被称为`编译器`。
- 由于计算机语言发展的历史原因（这里不做详细介绍，参考阅读），一个C语言源程序会首先被`编译器`翻译成更接近于底层的`汇编语言`，接着被`汇编编译器`（简称`汇编器`）翻译成二进制文件（如exe文件），然后CPU才能读懂并执行。

如何进行逆向分析

- 上面我们介绍了一份`源代码`变成`二进制文件`的过程，通常我们对一个软件进行逆向分析的时候，只能拿到一个软件的`二进制文件`，但我们想要分析其设计原理，就必须将其变成我们所能看懂的`汇编代码`或者`源代码`，即要把上面的过程倒过来，由此便产生了与编译器和`汇编器`相对应的两个强大工具
  - **反汇编器**：将可执行文件中的`机器码`转化成`汇编代码`
  - **反编译器**：将`汇编代码`转化成高级语言的`源代码`
- 借由反汇编器和反编译器我们可以得到类似于开发者所写的源代码的代码，之后我们就可以开始分析程序的逻辑了，在CTF竞赛中，我们常常需要逆写程序的逻辑，拿到flag

- 下面给出一个简单的例子（涉及简单C语言语法）

```c
#include<stdio.h>                               // 包含头文件
int main(){                                     // main函数
    unsigned char flag[30], enc_flag[30];       // 定义了两个字符数组，分别是加密前后的flag
    int i = 0;                                  // 定义并初始化一个循环变量
    scanf("%s", flag);                          // 输入字符串flag
    for(i = 0; i < 30; i++){
        enc_flag[i] = flag[i] ^ 0x86;           // 循环将flag字符串中的每一位异或上
        printf("%d ", enc_flag[i]);             // 循环输出enc_flag的每一位
    }
    return 0;                                   // 使程序正常退出
}
```

- 运行程序得到输出

```
224 234 231 225 253 207 217 237 232 233 241 217 238 233 241 217 242 233 217 244 227
240 227 244 245 227 167 167 167 251
```

- 由于异或(^)具有可逆的性质，即若`a = b ^ c`，则有`b = a ^ c`和`c = a ^ b`，故我们可以逆写程序逻辑，拿到flag

```c
#include<stdio.h>
int main(){
    unsigned char enc_flag[] = {224, 234, 231, 225, 253, 207, 217, 237, 232, 233, 241,
217, 238, 233, 241, 217, 242, 233, 217, 244, 227, 240, 227, 244, 245, 227, 167, 167,
167, 251};
    int i = 0;
    unsigned char flag[30];
    for(i = 0; i < 30; i++){
        flag[i] = enc_flag[i] ^ 0x86;
        printf("%c", flag[i]);
    }
}
```

- 得到输出即为flag：`flag{I_know_how_to_reverse!!!}`
- 到这里，恭喜你已经完成了第一次逆向分析！

一些入门建议

- **语言基础**：你可能已经注意到了，C语言在逆向学习中是不可或缺的，更进一步说，在你的接下来许多与计算机打交道的日子里都是极为重要的，所以从学好C语言开始吧。

- **学会用搜索引擎**：在学习逆向的过程中，你可能会遇到各种各样的问题，善用搜索引擎将会让你减少很多麻烦
- **学会智慧地提问**：参见提问的智慧
- **坚持下来**：学习逆向的路可能并不一帆风顺，但只要你坚持下来，一定会收获满满

最后

- Have fun in moeCTF 2022!
- 本题flag：`moectf{0hhhhhhh_I_kn0w_hoW_t0_R3v3rs3!}`