# 逆向工程(RE)

#### 来自维基

逆向工程(英语: Reverse Engineering),又称反向工程,是一种技术仿造过程。 即对一项目标产品进行逆向分析及研究,从而演绎并得出该产品的处理流程、组织结构、功能性能规 其主要目的是,在无法轻易获得必要的生产信息下,直接从成品的分析,推导产品的设计原理

#### 那么,我们要做的是对于软件的逆向工程

一般,CTF中的逆向工程题目形式为:程序接收用户的一个输入,并在程序中进行一系列校验算法,如通过校验则提示成功,此时的输入即flag。这些校验算法可以是已经成熟的加解密方案,也可以是作者自创的某种算法。比如,一个小游戏将用户的输入作为游戏的操作步骤进行判断等。这类题目要求参赛者具备一定的算法能力、思维能力,甚至联想能力。

我们知道,我们可以阅读源代码,了解一个程序的功能

但是一个程序不一定需要源代码才能运行。 事实上,我们在运行可执行文件,那么他们打开来实际上是一些二进制数据。 例如下面这一段程序源码

```
#include <stdio.h>
int main(){
         printf("Hello world");
}
```

我们使用记事本打开就能看到逻辑。但是真正编译出来的,我们用来运行的程序,使用记事本打开是这样的

```
☑2? ???L?This program cannot be run in DOS mode.
                 @/4
?idata
                                                                       @ B/29
                                                           @ B/67
            B/116
             @ MZ?\蠤 🗈
                          ?T蠤 @ u@?<@ 伜 @ PE 崐 @ tz?蓋 @墓 吚t<?$@ 桫f 棼f ?|蠤 ?桷f ?d蠤 ?鑁@ ?@怈 @tH1纼?呢
        悆?1纅?
                     ?x蠤 🗈
          1绤?暲閥四四峷?$?@?
    ②v媼戻
                           峵& 悆?四萬 茭$四
 1纼?脥?
       焦t22員222歳?
                1绤?暲锽嵈&
蠤 ?蠤 ?怈 荄$ュュ 遘$
荄$@ 蠤 ?$$蠤 ?f 兡,胒悕L$@冧??q黆夂WV峌®壶Q?
                            发x?h蠤 螳叮₪厾₪ d? ?X酅 煸శ1垭®峵& ???७ ?$? ®謨?壺??H蠤 呀u蕖L蠤 1腿?®?® ®蘿 呀®剏® ?®薑
    @墓 座@@@@ 呟@?@ @燖 吚t@荄$@
                 婨控 蠤 鑑□ ?蠤 ?t鄭 ?堪$□?蠤 堪$□?簋 ?$鑦□ ?□蠤 ?簋 吷囘匀
                                                     ?0簋 呉0劇 - 峞餣[^_]峚0峵& ?香蚤0000嵈&
?c ?囍 峞餣[^ ]崒@f惽D$@@餈 ?$
 ?8鄠 发8?蠤 荄$80時 ?$8襞?壠芨$8)燖 ?$®祝®世 发8呀Jt®荄$8,蠤 ?$8盅 8智®$?@ 鑎8006陽^_]脥?
盄 卟(蠤 呀t
                                           椤?ඕ念?婦$?$鑙b 呀嗯斃兡郮独髻脨悙∪夂wvs冹@?$ 燖 ₪®0鄘 冹₪呼ts壝?$ 燖 ₪
                                                                          怳夂发图?怈 吚t
```

这些二进制数据直接无法识读,但是包含了程序执行所需要的东西,而我们的工作就是,识读他们,找到flag!

## 如何开始逆向呢

#### 来自含树

ctf中的逆向 (reverse) 涉及很多编程语言,最常见的是C、C++、python、java、C#, 或者一常见的编程语言golang、rust、lisp等等。

除了windows与linux系统下可执行文件的逆向,还经常遇到安卓的( apk )逆向、系统驱动、固件逆向等等。

但是作为一个 beginner , 你不需要在刚入门的时候就去掌握这么多的编程语言

从C语言以及C语言逆向开始,是一个比较好的选择

因为C语言十分接近高级语言,又有充分的系统底层的支持

同时入门的逆向题很多是用C语言写的

并且当你学会了C语言,尽管你无法使用java、python等语言进行编程,但是你会发现,看懂这些高语言写的代码还是比较容易的。

## 常用工具

#### 常用工具介绍

#### 1. IDA Pro

IDA (Interactive DisAssembler) Pro (以下简称IDA) 是一款强大的可执行文件分析工具,可以对包括但不限于x86/x64、ARM、MIPS等架构,PE、ELF等格式的可执行文

件进行静态分析和动态调试。IDA集成了Hex-Rays Decompiler,提供了从汇编语言到C语言伪代码的反编译功能,可以极大地减少分析程序时的工作量

#### 2. OllyDbg和x64dbg

OllyDbg是Windows 32位环境下一款优秀的调试器,最强大的功能是可扩展性,许多开发者为其开发了具备各种功能的插件,能够绕过许多软件保护措施。但OllyDbg在64位环境下已经不能使用,许多人因此转而使用了x64dbg。

### 这些怎么下载?

https://down.52pojie.cn/

### 下好了怎么用呢?

推荐查阅一些书籍,如《从0到1: CTFer成长之路》 re篇

或者直接b站搜ctf re入门

以及第一题的bin在压缩包里, 试试ida吧, 你会需要它的。