【学习笔记】CTF PWN选手的养成（一）

原创

prettyX 于 2019-11-20 23:35:44 发布 1621 收藏 25
分类专栏： PWN 文章标签： PWN

本文链接：https://blog.csdn.net/prettyX/article/details/103173220

在ichunqiu上看的视频，对学习内容进行整理，对Atum老师表示感谢。

第一章 基础知识

0X01 PWN 简介

软件安全：软件安全专注于研究软件的设计和实现的安全

- 研究对象：代码（源码、字节码、汇编等）
- 研究目标：发掘漏洞、利用漏洞、修补漏洞
- 研究技术：逆向工程、漏洞挖掘与利用、漏洞防御技术

CTF PWN：软件安全研究的一个缩影

- 研究对象：可执行文件，主要是ELF文件
- 研究最终目标：夺取Flag

软件安全与CTF PWN特点：入门难、进阶难、精通难。

工具：

- 静态分析：IDA Pro
- 动态调试：gdb(with peda or gef)、windbg、ollydbg
- Exploit：pwntools、zio

前置技能：

- 汇编语言：程序执行、函数栈帧、函数调用等
- 编译、链接、装载、执行
- ELF文件结构
- Linux系统相关：文件描述符、系统调用、socket编程、shell命令

0X02 PWN学习方法

学习阶段一：学习套路

- 套路是有限的，假以时日一定会学完的招式
- 针对每种套路都练习1~2道习题

学习完所有常见套路，大多数国内比赛的中档题基本都可以随便切

学习阶段二：总结套路，变套路为艺术

- 漏洞利用是一门艺术，难以用套路完全概况，要想切难题不能全靠套路
- 多刷刷国际赛的难题，刷的慢没关系，刷多了自然就会融会贯通
- 多总结思考现有的套路的本质

资源：

- CTF Writeup Github：https://github.com/ctfs 聚合了各大国际比赛的习题文件以及writeup
- Googling:XXX writeup or XXX CTF 百度收录不了github pages

0X03 PWN 学习案例 ROP：

1、ROP：现代栈溢出中最基础的利用技术（最简单最基础的套路）

2、Googling 筛选到比较好的题目：r0pbaby

学习writeup，可以参考多篇writeup，根据writeup和学到的内容自己动手调试一下，尽量自己动手重写EXP

重写EXP成功：基本掌握了ROP。

第二章 CTF中漏洞挖掘的方法论：从逆向工程到漏洞挖掘

0X01 逆向工程简介

对于PWN来说，逆向工程的主要作用为发掘与分析漏洞

工具：

- 静态分析工具：IDA pro
- 动态调试工具：gdb、windbg、ollydbg、IDA Pro

IDA pro的F5大法

0X02 常见漏洞简介

在进行漏洞挖掘之前，必须对常见漏洞非常熟悉！！！

缓冲区溢出（Buffer Overflow）

- 堆溢出、栈溢出、bss溢出、data溢出（通常覆盖指针）
- wellpwn、AliCTF 2016 vss、Hitcon 2015 readable、stkof、zerostorage

整数溢出（Integer Overflow）

- 无符号型与有符号的转换（MMACTF 2016 shadow）
- 整数加减乘除法，如malloc(size*2) (pwnhub.cn calc)
- 整数溢出通常会进一步转换为缓冲区溢出、逻辑漏洞等其他漏洞

格式化字符串（Format String）

- printf(s)、sprintf(s)、fprintf(s)等，可能导致任意地址读写（MMACTF 2016 greeting）
- 可以用来leak（HCTF2016 fheap）

释放后使用（Use-After-Free）

- 释放掉的内存可能会被重新分配，释放后使用会导致重新分配的内存被旧的使用所改写
- Double free是一种特殊的UAF
- Defcon 2014 Qualifier shitsco、AliCTF 2016 router、0CTF2016 freenote（double free）、HCTF2016 fheap（double free）

逻辑漏洞

- 访问控制、协议漏洞、多线程竞态条件（fake fuzz）等

0X03 漏洞挖掘中的逆向技巧

关键数据结构分析：还原结构体、接口、类等

控制流分析：理清楚程序的执行逻辑，基本要做到从反汇编代码到源码的还原

数据流分析：理清楚数据的流向

CTF漏洞挖掘中的分析策略：

- 目标文件较小时，通常采用对整个目标文件进行控制流分析，做到整个程序从反汇编代码到接近源码级别的还原，还原的同时查找漏洞
- 目标文件较大时，逆向整个文件所需工作量太大，通常需要额外的关注数据流，并理清楚数据流所经之处的控制流，因为漏洞的触发与数据流离不开关系
- 无论是数据流分析和控制流分析，还原结构体、接口、类都会促进逆向工程

控制流分析的主要作用是理清楚程序的逻辑，对于规模较小的目标文件，一般选择理清整个目标文件。

代码以识别为主，不要硬逆。

善用标记，标记结构体、标记变量名、标记变量类型

F5大法好，但是F5不是万能的，当发现F5结果比较诡异时需要在汇编层分析（如mmactf 2016 shadow）

数据流分析

目标文件较大，全盘逆向不现实

- 追溯用户输入的走向，重点关注对用户输入数据处理的函数
- 可以在不用逆清楚控制流即可找到漏洞，需要一定的技巧性 （plaid CTF 2015 datastore）

加油，争取一周总结2~3篇！