Just pwn it for fun 00

前言

`Hello`, 欢迎来到`Just pwn it for fun`系列的第`0`篇, 这一篇的主要内容是介绍这个系列的主要内容, 阅读博客的注意事项. 以及自己常用的一些小技巧.

本文传送门如下:

正文

关于博客的起源

这个系列来源于我现在正在做的另外一个项目. 本来准备两个月之后一起发出来的, 但是由于一些不可抗力:

> 比如电脑坏了, 得整点稿费买电脑....
>
> 
>

就提前放出来了. 如果不出意外的话, 这篇博客应该算是非常简陋的一份草稿. 后面估计还要大改.

这一个月主要是`Just pwn it for fun`. 花了一个月在自己喜欢的事情上, 想去见识更多新奇的东西, 所以看了浏览器, adobe.

学习期间, 做了一些零零碎碎的笔记, 后来越来越发现浏览器的学习真是痛苦不堪, 所以想好好写一个系列, 如果后面的人有人学浏览器的话, 对于初学者而言. 可能会有一点小用就好.

所以重新思考了一下自己所学, 花了五天时间整理了一下草稿. 就有了这个系列.

另外, 因为时间不够, adobe相关的东西我只放了一个case study, 如果有机会再去adobe玩玩, 应该也会独立出一个系列出来.

windows kernel部分, TJ学弟说他看的时候能看懂, 但是感觉有点乱, 所以我想抽时间重做一下...也不知道何年何月...

关于这个系列的目标

> [+] 我想写一些东西给和我一样的初学者看, 想在博客里面表达出怎么去碰坑, 爬坑. 以及怎么放弃(误).

老实说, 学安全的话, 我算是偏笨的那种类型. 所以每次看到诸如小刀师傅, doar-e团队所写的博客就很感动. 如果你有看过的话, 就会发现里面全是分享两个词, 讲的也特别仔细, 生怕我这种笨蛋无法去理解. 所以我想把这个博客做到接近他们的样子.实在是帮了大忙. 但是大概现在的内容全是学习, 称不上研究. 所以类似小刀师傅那种翻译为英文版的文章, 我感觉没有必要.... 所以只有中文版.

关于英文版, 由于第7篇的内容对我来说比较有趣, 算是我自己最喜欢的一篇, 所以你在这里你能看到唯一一个Changlish的版本.

所以一个是想把博客做简单和详细一点, 我不太懂厉害的人会怎么思考问题, 但由于我自己就是菜鸡, 所以我想大概能懂如我这种类型的菜鸡怎么想问题.

另外一个, 浏览器的学习痛苦不堪(比如搭建v8.)

所以我想让这个博客尽量有趣一点, 我不会讲笑话, 所以会在博客里面穿插一些表情包或者冷笑话. 大概觉得你看到某个点会觉得难过的时候就会穿插一个这种...

详细 + 有趣 + 简单, 大概就是我这个博客最初的目标了, 我本来准备加上便于理解的, 但是发现人和人的理解方式不太一样... 所以又不是很自信... 就舍掉了这个目标.

关于博客的阅读方式

唔, 与其说是博客的阅读方式, 其实更多的是关于我自己的学习方式.

> [+] 请注意我说的是方式, 并不是方法. 方法是证明了行之有效的. 而方式只是一种不断的测试不断的改正的东西. 大多数情况下也只针对和我类似的人.

我是偏向于笨的那种人, 这种笨会体现在理论的部分. 如果一直让我看理论的知识. 很容易找不到重点. 然后淹死在知识的海洋当中. 所以我偏向的方法是欺骗和麻痹自己(知识屏蔽). 比如我碰到一个理论不会的时候, 我会先假装自己是会的. 然后把接下来的部分做了. 再去看理论. 大概是由于调试器调试的时候我总会犯错, 对于正确的东西我记忆不深, 对错误的东西却是一朝被蛇咬，十年怕井蛇.

由调试的时候引起的错误出发，去积累相应的知识点. 这个方式对于我是行之有效的. 所以我的博客基于这个思想而组织... 但我不确定这个方式对别人会不会有效...

博客依赖于我自己的理解顺序, 我想让他相互独立一点, 以及憋的不解越多, 在后面的时候我看到的时候才能记得更深. 所以大概需要耐心一点. 碰到看不懂的问题可以先思考是不是真的需要在此时此刻去理解这个问题. 如果不需要的话, 可以先假装会, 再去看看接下来的部分...

我并没有刻意的去控制我每一篇博客的长度一致, 我想让他们分开, 想让思路更为清晰一些. 所以请不要介意我是否偷懒... 嘤嘤嘤.

关于地址.

你可以在这些地方找到这个系列.

- blog: wjllz 是人间大笨蛋

> 我的个人博客. 我自己还蛮喜欢粉红色的主题... 但是粉红色不太好找... 就只有他了...

- github: Just pwn it for fun

> 我的`github`项目. 博客的相关的内容, 资料, 文件. 全在这里.

- 先知: wjllz是人间大笨蛋

> 向阿里爸爸鞠躬.

- pediy: wjllz

> 看雪是我蛮喜欢的做安全的网站. 因为我更多的是参考的国外的资料, 所以看雪参考的资料其实不是很多... 但是, 在里面认识了很多厉害的人. 所以也想慢慢做一点和前辈们一样的事.

wjllz's debug trick

我学安全的时候, 总是有一种奇怪的感觉. 比如我明明在写bug, 但是在某一分钟和自己的打篮球或者做数学题的感觉有点类似. 所以这些方式来源于自己调试的时候不断地纠错(比如觉得调试或者学习不顺畅, 又去找个方法改正.), 慢慢的试出来对自己有效的方法.

需要注意的是, 这一小节有个关键词是`wjllz's`, 所以我只是总结对我有用的一些手段, 无法保证会对其他人有用...

装疯

这个方法最开始用到的时候是做高考数学压轴题的第二问, 第一次见到专业的描述词汇是在王爽老师的汇编语言(第三版)里面, 他所给的词汇叫做知识屏蔽, 我自己翻译一下就是:

> 把所要学习的东西分为`A, B, C`几个部分. 碰到不会的东西, 可以先假装自己是会的. 然后看一下能不能建立在这种假设之上把剩余的部分给推通. 再去解决难的部分.

我之前的一篇博客采用的就是这个方法. 包括我接下来的文章也会长期的采用这个方法. 如果深究于某个点, 很有可能会淹死在知识的海洋当中.

举个例子, 如果你之前有看过ROP相关的知识的话. 你会发现我们在我们需要`ROP`的时候基本是这个情况:

> [+] 1. 我们已经可以控制`RIP`了.
> [+] 2. 我们已经能够编写`shellcode`了, 但是如果直接跳到`shellcode`. 会由于某些缓解措施无法正常的运行`shellcode`, 这里我们记为`缓解措施very_hard`.

现在, 我们想一下我们的`ROP`会怎么做.

> [+] 1. 我们会把`RIP`控制到某个`ROP链`, 该`ROP链`记为`ROP_NIUBI`.
> [+] 2. 我们期待我们正常`ROP_NIUBI`之后, 能够跳到`SHELCODE`. 并且`ROP_NIUBI`当中会做一些骚操作, 让我们关掉`缓解措施very_hard`.
> [+] 3. 执行想要执行的`shellcode`, pwn the world!

这种情况下, 我们可以装疯了, 首先, 我们需要控制`RIP`, 正规的做法是利用漏洞, 编写一些代码, 然后来完成. 但是呢, 我们可以假装我们已经有这个漏洞了. 用什么假装呢, 用调试器.

我常用的调试器, `windbg`, `gdb`都是提供了直接修改`RIP`的命令.

> [+] windbg: r rip=0x4141414141414141
> [+] gdb: set $rip = 0x4141414141414141

我们可以直接利用调试器模仿代码来直接修改`RIP`, 让他直接指向我们想要执行的`ROP_NIUBI`. 所以我们简化了工作, 剩下的事情就是研究哪一条`ROP_NIUBI`合适. 做完这一部分之后, 我们再来做控制`RIP`的工作.

把所有的工作细分, 一部分一部分完成, 在做完所有的工作之后, 再去结合具体的代码, 尝试用代码模仿我们手工的行为.

对于大部分的缓解措施的学习, 我基本都会采用上面的方式, 大大的简化了我写代码的过程.

> [+] 关于这个方法, 你可以在我的第一篇博客里面看到一个具体的例子

卖傻

我大概经常会用一些笨方法, 但又好像总有他的好处在里面. 比如虽然它笨, 但胜在有用.

问题是这样的, 我想针对和我一样的菜鸡, 大部分的东西我们的积累都是有限的. 光是`gdb, windbg`的调试指令就有很多. 更别提这些调试器都有许多有关他的调试技巧.

依然举个例子:

> 在前期学习windows kernel的时候, 我需要去学习heap spray. kernel heap spray指的是去控制堆的布局, 所以在学习的过程中我需要去观察堆的布局. 但是这个时候, 我对windbg的!heap指令并不熟悉, 另外, 我不确信!heap对内核态的堆是否有用. 我想去观察某个handle对应的堆具体在内存当中分配在哪. 所以我无法知道我下一步该怎么做.

这个时候卖傻就派上了用场, 一共两个思路:

> [+] 1. 去查阅!heap的官方文档. 看一下是不是有具体的命令.
> [+] 2. github上面有一个项目(windows address leak)可以通过handle来泄露一个对象. 泄露出来之后我可以使用printf给打印出来. 相当于我自己多了一个指令.

我想说的是第二个思路, 它很笨, 但是呢, 他是有效的. 另外一方面, 他是在我的知识体系类唯一能想出来的替代方法.

我前期学习使用会大量尝试卖傻这种行为, 使用一些笨但是有效的方法, 当然, 也是有好处的, 比如你可以更清晰的了解到我们的指令的原理. 所以万事皆有好处在其中.

尽信书

这方面其实不太好说, 主要说的是折腾. 大概是这样的.

我自己做这些东西的时候, 大多数都是上手去做, 也就是说会需要环境. 很多时候, 版本不太一致, 就会出现很多不能解释的行为. 然后就是这种不能解释的行为, 会去折腾很久很久的时间. 但是很久之后呢. 积累更多的时候, 就算版本再不一致, 凭借着积累的知识. 发现解决那些问题好像又不再是写在旗帜上的目标.

也就是说, 如果是前期学习的话, 尽量和作者环境一致, 能够大概率的避免摸不着头脑的错误. 所以尽信书.

不如无书

与前面对应的, 不如无书主要是说加点自己的判断和思考进去, 在走不通的时候, 可以去适当想一想, 为什么走不通, 单凭自己现在的东西能够推出来么. 有没有更好的解决方案.

我自己学windows kernel的时候, 学会一个GDI primitive的东西, 后来在做浏览器, adobe的利用的时候, 很多东西觉得看着不太舒服. 就会去移值借用GDI primitive的思路去自己写. 所以也学到了很多有趣的东西(虽然也碰到了很多坑...)

这两个方法各有好处, 我是两者都用... 所以我分不清谁差谁好.

> [+] 第五篇博客对应尽信书.
> [+] 第六篇博客博客对应不如无书.

我们不一样

假期认识了TJ学弟, 因为他经常会问我一些windows kernel的东西. 所以我偶尔会帮他调一下bug. 我发现我和他调试的思路是不一样的.

学弟擅长的是把东西跟进去, 然后想本源. 这方面我不经常用... 所以了解不深.

我自己采用的是对比.

> [+] 如果我的情况是错误的. 本来是应该先执行A+B+C, 但是结果是A+D+E.
>
> > [+] 1. A 为什么能正常出现.
> > [+] 2. B/D 有什么不同. 什么会导致他们不同.

我一般会从这两个思路出发. 简而言之, 预想一下正常的流程, 再对比一下错误的流程. 不断的调试着比对. 特别是开发`exp`的过程, 为什么这个能运行, 为什么加了这一个语句就不能运行. 带着很多问题对比思考, 能够收获更多新的东西.

我大概很多`bug`的调试都依赖于此.

但是每个人的思路都不一样, 习惯, 角度都不一样. 所以这算是我对自己的博客不太自信的原因之一. 所以TJ学弟和我说能够看懂的时候真是谢天谢地.

想不到标题了.

还有很多的调试`trick`需要自己慢慢积累. 比如:

> [+] 什么时候用条件断点能够让我们的调试如德芙纵享丝滑.
> [+] 什么时候用硬件断点, 什么时候用软件断点.
> [+] 哪一些命令是有用的, 经常在哪些地方用.
> [+] 以及, 自己调试的时候. log哪些语句会方便调试.
> [+] 哪些工具好用. 适用于什么地方.
> [+] ...

这些东西无法诉诸于语言. 我的积累方式就是还没用的时候没啥感觉, 某一瞬间体会到了就一瞬间记住... 所以大概需要时间的积累.

传送门

- Just pwn it for fun 00 - Just pwn it for fun

> [+] 讲一些关于博客相关的事情.

- Just pwn it for fun 01 -- firefox: 从弹计算器出发学习浏览器

> [+] 从弹计算器出发, 学习浏览器的调试.

- Just pwn it for fun 02 -- firefox: 从调试`simple.js`出发看`JS`对象分配

> [+] 理解一些关于`JS`对象在内存当中如何分配的问题.

- Just pwn it for fun 03 -- firefox: 从构建`R/W`学习浏览器.

> [+] 利用第2篇的东西, 尝试去学习`R/W`的编写.

- Just pwn it for fun 04 -- chrome: 从弹计算器出发学习浏览器

> [+] 从弹计算器出发, 学习浏览器的调试.

- Just pwn it for fun 05 -- chrome: 从调试`simple.js`出发看`JS`对象分配

> [+] 理解一些关于`JS`对象在内存当中如何分配的问题.

- Just pwn it for fun 06 -- chrome: 从构建R/W学习浏览器.

> [+] 利用第5篇的东西, 尝试去学习R/W的编写.

- Just pwn it for fun 07 -- chrome: 从一次爬坑到脱坑到放弃学习浏览器

> [+] 一个exp开发的学习, 碰到的坑. 以及碰到坑的时候怎么去思考解决问题.

- Just pwn it for fun 08 -- chrome: 从调试源码和阅读文献出发学习浏览器

> [+] 怎么调试chrome, 怎么利用调试来帮助我们理解浏览器的一些原理性的东西.

- Just pwn it for fun 09 -- firefox: 从调试源码和阅读文献出发学习浏览器

> [+] 怎么调试firefox, 怎么利用调试来帮助我们理解浏览器的一些原理性的东西.

- Just pwn it for fun 10 -- chrome: 从理解漏洞原理到理解漏洞模型 00

> [+] 漏洞原理分析, 以及漏洞模型(00)构建.

- Just pwn it for fun 11 -- chrome: 从理解漏洞原理到理解漏洞模型 01

> [+] 漏洞原理分析, 以及漏洞模型(01)构建.

- Just pwn it for fun 12 -- chrome: case study(cve-2019-9810)

> [+] 一个缓解措施和一个特性的理解, 可以帮助我们的浏览器学习. 依然是爬坑之旅

- Just pwn it for fun 13 -- chrome: case study(cve-2019-5786)

> [+] FileReader的漏洞的学习与分析.

- Just pwn it for fun 14 -- adobe: case study(cve-2018-4990)

> [+] adobe相关的调试和学习.

- Just pwn it for fun 15 -- browser: 环境搭建

> [+] 填坑补充环境搭建部分.

后记

thanks

做安全还蛮奇怪的... 莫名其妙认识很多人, 被很多人帮助. 虽然总觉的这样不是很妥当, 毕竟没做成什么太大的事, 但对我而言, 至少这个系列是花了很多时间的... 慢慢改慢慢写完成. 所以总是想感激很多人.

- special thx to 学长.

> ADLab的实习对我来说是一段很舒服的时光, 因为很自由. 学长帮我解决了很多生活方面的问题, 垫了房租, 生活费... 还有很多. 除了没帮我找一个女朋友, 安排相亲. 我每天的生活就是学习, 挖洞, 抽烟. 睡觉. 能够专心的闭关, 全靠学长. 所以. 蛮感激的.

- special txh to doare-team

> I read their blog and got tooooooooooooooo much!

- special thx to 小刀师傅

> 其实我以前蛮反感写博客的, 大概觉得浪费时间. 后来第一次想写博客是因为北归姐和我说不如写写. 但是第一次想好好写博客是因为小刀师傅. 在小刀师傅的博客里总是能够感觉到用心呀, 分享呀之类的字眼. 后来与小刀师傅慢慢交流我才明白. 原来厉害的人是应该长这个样子. 寄希望于自己的优秀, 也能够帮助别人优秀. 我对于优秀没有追求, 但是能做一些和小刀师傅他们一样的事就好. 所以天知道看见自己小刀师傅加了自己的友链的时侯我高兴了一个下午, 恍若少女怀春!

关于安全的一些小想法

我对安全的理解不深, 如我的理解之下. 我觉得泉哥呀, 吴石老师, 古河老师他们这些老师. 是能够很有资格的说说安全, 给后辈一些指导和建议, 我单方面的在泉哥的文章里面获益很多, 特别是这两篇文章:

- 安全人员的自我修养

> [+] 帮助自己修正了一些自己挖洞的思路. 也帮助自己改进了一下学习方式.

- 安全人员的自我修养(续)

> [+ 帮助自己修正了`fuzz`的看法, 明白自己不需要走的那么急.

至于我, 我连自己都没有教好, 所以是没有资格资格教人的.

所以我所会不多. 大概就只有一个东西我是确定的.

> 安全是AB制的东西. 所谓的AB制, 指的大概是如果你能够完成A, 那么你就一定也完成B. 这个世界上有很多东西不是AB制. 你把天日了该做的事可能还是做不到, 但安全不是. 花点时间砸下去, 总感觉该做的事一定能做完. 这是我在安全身上所能感受到的幸福感之一.

另外一个是前些天看到看雪的一个小伙伴的一个帖子. 主要是讨论一些关于工作的话题. 因为秋招临近, 所以大概也会想想这方面的东西.

学安全大概很容易饿死.

但我其实不太懂这些... 我所知道的就是`just pwn it for fun...`.

我是那种, 觉得一件事情有点难就会觉得麻烦或者无聊的人. 然而所做的事基本难度都比较大. 所以最后, 放一张每次挖洞挖到无聊的时候, 总会浮在脑海里的一张图. 给和我一样初学的小伙伴:

后续

博客不出意外的话应该还会继续更新, 因为还有些想做的内容没有做完. 但是:

> [+] 如果这些文章被喷的很惨的话,.. 除了github和我的个人博客之外... 看雪和先知我应该是不会更新了...
> [+] 接下来的日子, 我大概率不会做安全了, 所以我可能只有在我完成正经工作的闲暇之余写下安全相关的东西. 所以频率可能会放低.
>
> 
>

还有一点重要的事

唔, 虽然很啰嗦, 但是我希望就是... 我大概在五十岁以前, 我的文章不会加上`tutorial`和`教程`这种词汇, 大概是源于在那之前我的积累都是不够的, 太早的加上这些词汇教人会很误导人. 所以没有资格加上这两个词汇. 初衷只是如前面所说. 我想做一点和我一样的`newbee`有关的事, 大概大家一起菜的时候反而能更理解彼此.

> 以一个初学者身份, 思考的问题卡在那里, 在哪些地方会犯错, 怎么去思考问题, 理解问题. 怎么去解决问题,

我想把这些东西在自己的博客慢慢写出来.

教程的话我想国内无论是同辈(同辈我所认识的都比我厉害)的亦或者是前辈, 他们都会更合适一些,。 加上网上有一些神奇的师傅们不厌其烦的分享他们的知识, 所以找起来会稍微麻烦一些, 但是查资料查的多了... 就能够慢慢的学会一点关于查资料的方法... 所以应该是问题不大的.

copyright

- CopyRight of ADLabs

> [+] 这些代码, 报告, 全是在实习期间写的. 除了博客之外... 其余的东西全是在公司做的... 所以, 应该是属于`adlab`.(如果你要是觉得这博客很菜(虽然到现在我的博客访问量还是0). 那就肯定是我写的. 要是偶尔觉得哪个地方还行, 那就是`adlab`了)

- CopyRight of 先知

> [+] 这些文章(除了这篇)会投稿到先知. 先知提供稿费, 所以如果转载的话请加上本文首发于先知.

wjllz's life

最后, `wjllz`是人间大笨蛋.