moectf2024 Reverse 方向 入门指北！

0xcafebabe *QQ：782816967*

什么是逆向工程

《逆向工程》是你即将体验的一款开放世界冒险游戏。该游戏秉持着根据已有的东西和结果，通过分析来推导出具体的实现方法，包括但不限于：编译后的二进制程序、汇编代码分析、甚至是编译前的代码分析。游戏设定在一个名为"调试器"的真实工具中，你将扮演一位名为"逆向人"的角色，在自由的代码执行中邂逅性格各异、能力独特的指令、特性们，与他们一起拿下CTF比赛，找回失散的flag，同时逐步发掘"二进制"的真相。

本文讲解的是狭义逆向工程（偏向于CTF比赛），你需要深入分析代码片段/二进制文件，获取其中加密后的一段flag文本并提交到比赛平台，就算**你过关!**

一般来说，比赛主办方都会给你一个exe(win下的可执行文件)/elf(linux下的可执行文件)，还有的会直接给你.asm(汇编语言文件)或者apk(手机安装包文件)，甚至更离谱的，scratch编程工具的代码文件，不常见架构的程序（riscv、甚至是上世纪的红白机的文件），你需要有正确的思考能力和肝能力，从史山代码中提取珠宝，定位正确的地点，从而解决这个问题。

怎么快速入门逆向工程

逆向工程是从比较底层的观点对程序运行过程进行分析的过程，我们平时遇到的C语言、Python、Java都是高级语言，它们可以被编译/解释为可以被CPU"看懂"的二进制汇编代码（低级语言），从而CPU可以直接执行它们，并修改CPU内部的一些变量或者是内存中的一些值，使得程序进行正常的逻辑，从而正常和你进行交互，而你要做的，是从这些二进制汇编中，恢复原来的加密算法或者分析程序逻辑。

入门逆向工程，请你从C语言开始，那么为什么呢？因为C语言是比较贴近底层的一门语言，而Python这种高度封装的语言，你虽然写着非常舒服，但是你不懂它的原理是什么，而我们逆向工程就是要研究这些原理（当然Python在写解密算法的时候很好用，不过本段是推荐你学习C语言），总之，你了解的语言越多，在逆向工程这上面的拦路虎就会越少，而且你只要掌握C语言（能做完这一套题并且了解其原理），你的大一上学期的C语言导论必修课程就一定能满分通过，如此收益的过程，相信你一定会认真学习的 。

当你能看懂C语言了（数组，指针，函数调用），你就应该从更底层来思考这是如何实现的，你可以使用IDA这个软件（点击这里下载IDA）对可执行文件进行逆向分析，这个软件可以把汇编语言"反编译"为对应的C语言代码，让你可以直接看懂，你可以了解汇编语言与C语言每行代码之间的对应关系，并且在了解"位运算"的知识后你就可以了解现代的加密算法是如何实现的；在了解指令的时候，你就会了解现代编译器为什么在编译的时候，让除法运算为什么要用乘法来代替；在了解栈和堆后，你就会知道你的局部变量，全局变量的确切位置，了解为何局部变量只能在局部用、全局变量为何与线程密切相关；最终，你将成为逆向带神，在阅读汇编语言上游刃有余，在破解软件上手到擒来（当然，学到了技能，就不能做违法的事情）。

CTF比赛中的逆向题是什么

CTF比赛中一般会给你一个可执行文件或其他文件，可能是C/C++写的，可能是Python写的，Java写的，C#写的.....不管出什么，你都得硬着头皮看下去，不懂的东西就查，接下来我就以一个简单的例子来介绍我们CTF中的逆向题

一般CTF，主办方都会给你一个让你输入东西的程序，输入东西后，程序会进行一系列加密或者取数

据摘要，然后把面目全非的结果和一个常量（正确的flag进行加密或者取数据摘要后的结果）进行比较，如果两个值一样，则你就拿到了正确的flag，否则就给你说flag错误。

对于简单题，我们喜欢使用strcmp(input_flag, “real_flag”);这样的C语言函数进行字符串比较，注意这里并没有进行加密，所以非常简单，我们只要在IDA中看到了这样的代码就可以很容易察觉到它进行了比较，然后我们就可以获取到real_flag。

那么进阶一点的题目会怎么出呢？我们可以使用位运算中的异或（xor）！

```
#include <iostream>

int main()
{
	char input[] = "moectf{??????????????????????????????????}"; // 这是假的flag
	char password_enc[] = {
		123, 121, 115, 117, 98, 112, 109, 100, 37, 96, 37, 100, 101, 37, 73, 39,
		101, 73, 119, 73, 122, 121, 120, 113, 73, 122, 121, 120, 113, 73, 97, 119,
		111, 73, 98, 121, 73, 115, 110, 102, 122, 121, 100, 115, 107, 22 };
	// password_enc的每一位和22进行异或，就能得到真实的密码
	for (int i = 0; i < 46; i++) { // 46是 input的长度，也是flag的长度！
		if ((input[i] ^ 22) != password_enc[i]) {
			printf("Password is wrong!\n");
			exit(0);
		}
	}
	printf("Password is right!\n");

	return 0;
}
```

由于异或运算的性质，a^b = c的时候，c^b=a，是一个可逆的操作，加密的时候可以异或22，解密的时候我们也异或22，这是一个最基础的“对称加密算法”，因为加密解密的“密钥”都是22

在做这题的时候，我们需要分析：

1. 加密是如何实现的，我们是否可以把加密算法搬到我们自己的代码上正确运行？
2. 加密是否对称？加密的密钥和解密的密钥是否一样？
3. 如何逆向进行加密算法，或者对ASCII码进行爆破（如果一个字节变换不会引起其他很多字节的变换，这种情况下暴力往往是非常快的）。

在这里，我们可以很容易写出解密脚本：

```
#include <iostream>

int main()
{
	char password_enc[] = {
		123, 121, 115, 117, 98, 112, 109, 100, 37, 96, 37, 100, 101, 37, 73, 39,
		101, 73, 119, 73, 122, 121, 120, 113, 73, 122, 121, 120, 113, 73, 97, 119,
		111, 73, 98, 121, 73, 115, 110, 102, 122, 121, 100, 115, 107, 22 };
	// 因为a^b=c时，b^c=a，所以我们可以这样还原数据：
	char password[47];
	for (int i = 0; i < 46; i++) {
		password[i] = password_enc[i] ^ 22;
	}
	password[46] = 0; // 使用0字符来截断掉%s的无尽输出..
	printf("%s\n", password); // 哈哈，这就是本题的f l a g，自己运行一下交上去吧！

	return 0;
}
```

最终通过printf函数，我们得到了我们想要的“flag”，解决了这个题！

当然，随着你对逆向工程的研究越发深入，你会发现，这种题过于简单了，其实难题是简单题的复合，一堆简单的操作混在一起，就显得非常困难了，这其实是你的惯性思维，逆向工程是从一小块一小块分析，最后组合成一个目标结果的，一般来说，你看到的可能是：flag的第3位异或了2，第四位异或了4....，或者是第n位进行了其他位运算，甚至是异或了一整个数组，或者是循环异或，前后异或（第一个和最后一个，第二个和倒数第二个...），这些很可能都在接下来的题目中显现，有待你的深入探索。

最后，贴出一些本人常用的工具，每一个工具都经过百般打磨，相当好用。
为了防止新手们使用《P2P下崽器》，本人特意贴出了所有较为官方的下载链接

1. x96dbg(x64dbg+x32dbg)

https://x64dbg.com/

2. IDA

https://down.52pojie.cn/Tools/Disassemblers/IDA_Pro_v8.3_Portable.zip

3. 010 Editor

https://down.52pojie.cn/Tools/Editors/010_Editor_All_Versions_For_Windows_CracKed.7z

4. Python

https://python.org/

5. Visual studio 2022

https://visualstudio.microsoft.com/vs/

6. dnSpy

https://down.52pojie.cn/Tools/NET/dnSpy.zip

7. GDA (Android/Java Reverse)

https://down.52pojie.cn/Tools/Android_Tools/GDA4.10.zip

8. Jadx (Android/Java Reverse)

https://github.com/skylot/jadx/releases/

9. upx

https://github.com/upx/upx/releases/

10.wsl/linux environment

https://learn.microsoft.com/zh-cn/windows/wsl/setup/environment

11. notepad++

https://notepad-plus.en.softonic.com/download

好了，接下来就是本题flag了，什么？你说你看到这里了还没发现flag，那你再看一遍吧。😷