CTF入门--题目介绍

原创

网络安全研发随想 于 2021-02-04 16:10:46 发布 1589 收藏 8

分类专栏： 网络安全

本文链接：https://blog.csdn.net/gengzhikui1992/article/details/113646191

CTF 题目类型一般分为 Web 渗透、RE 逆向、Misc 杂项、PWN 二进制漏洞利用、Crypto 密码破译。

1. Web类题目

在传统的CTF线上比赛中，Web类题目是主要的题型之一，相较于二进制、逆向等类型的题目，参赛者不需掌握系统底层知识；相较于密码学、杂项问题，不需具特别强的编程能力，故入门较为容易。Web类题目常见的漏洞类型包括注入、XSS、文件包含、代码执行、上传、SSRF等。

信息搜集

前期的题目信息搜集可能对于解决CTF线上比赛的题目有着非常重要的作用

敏感目录泄露

通过敏感目录泄露，我们往往能获取网站的源代码和敏感的URL地址，如网站的后台地址等。

比如：

- 攻击者可以通过.git文件夹中的信息获取开发人员提交过的所有源码，进而可能导致服务器被攻击而沦陷。
- admin之类的敏感后台文件夹

敏感备份文件

通过一些敏感的备份文件，我们往往能获得某一文件的源码，亦或网站的整体目录等。

比如：

- vim的备份文件

常规文件

比如

- robots.txt：记录一些目录和CMS版本信息。
- readme.md：记录CMS版本信息，有的甚至有Github地址。
- www.zip/rar/tar.gz：往往是网站的源码备份。

Banner识别

一个网站的Banner信息（服务器对外显示的一些基础信息）对解题有着十分重要的作用，选手往往可以通过Banner信息来获得解题思路，如得知网站是用ThinkPHP的Web框架编写时，我们可以尝试ThinkPHP框架的相关历史漏洞。

SQL注入

而由于开发者在程序编写过程中，对传入用户数据的过滤不严格，将可能存在的攻击载荷拼接到SQL查询语句中，再将这些查询语句传递给后端的数据库执行，从而引发实际执行的语句与预期功能不一致的情况。这种攻击被称为SQL注入攻击。

SQL注入漏洞也是现实场景下最常见的漏洞类型之一。

SQL注入，包括数字型注入、UNION注入、字符型注入、布尔盲注、时间注入、报错注入和堆叠注入等注入方式。

任意文件读取漏洞

所谓文件读取漏洞，就是攻击者通过一些手段可以读取服务器上开发者不允许读到的文件。

文件读取漏洞常常意味着被攻击者的服务器即将被攻击者彻底控制。

SSRF漏洞

SSRF（Server Side Request Forgery，服务端请求伪造）是一种攻击者通过构造数据进而伪造服务器端发起请求的漏洞。因为请求是由内部发起的，所以一般情况下，SSRF漏洞攻击的目标往往是从外网无法访问的内部系统。

SSRF漏洞形成的原因多是服务端提供了从外部服务获取数据的功能，但没有对目标地址、协议等重要参数进行过滤和限制，从而导致攻击者可以自由构造参数，而发起预期外的请求。

命令执行漏洞

通常情况下，在开发者使用一些执行命令函数且未对用户输入的数据进行安全检查时，可以注入恶意的命令，使整台服务器处于危险中。

XSS漏洞

跨站脚本（Cross-Site Scripting，XSS）是一种网站应用程序的安全漏洞攻击，是代码注入的一种，允许恶意用户将代码注入网页，其他用户在观看网页时会受到影响。

根据XSS漏洞点的触发特征，XSS可以粗略分为反射型XSS、存储型XSS。

反射型XSS通常是指恶意代码未被服务器存储，每次触发漏洞的时候都将恶意代码通过GET/POST方式提交，然后触发漏洞。

存储型XSS则相反，恶意代码被服务器存储，在访问页面时会直接被触发（如留言板留言等场景）。

Web文件上传漏洞

文件上传在Web业务中很常见，如用户上传头像、编写文章上传图片等。在实现文件上传时，如果后端没有对用户上传的文件做好处理，会导致非常严重的安全问题，如服务器被上传恶意木马或者垃圾文件。

2. 二进制类题目

移动端题目

CTF中的移动端题目普遍偏少，Android类的题目主要偏向杂项（Misc）和逆向（Reverse）。

前者通常根据Android系统特性隐藏相关数据，考察参赛者对系统特性的熟悉程度；

后者主要考察参赛者的Java、C/C++逆向能力，出题人常常会加入混淆（ollvm等）、加固、反调试等技术，以增加应用的逆向难度。这类题目往往需要参赛者具备一定的逆向和开发能力，熟悉常用调试逆向工具，知道常见反调试及加壳脱壳方法。

在CTF中，逆向工程一般是指软件逆向工程，即对已经编译完成的可执行文件进行分析，研究程序的行为和算法，然后以此为依据，计算出出题人想隐藏的flag。

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。

静态分析

逆向工程的最基本方法是静态分析，即不运行二进制程序，而是直接分析程序文件中的机器指令等各种信息。目前，静态分析最常用的工具是IDA Pro。

动态分析

所谓动态分析，就是将程序实际运行起来，观察程序运行时的各种行为，从而对程序的功能和算法进行分析。这需要被称为调试器的软件，调试器可以在程序运行时观察程序的寄存器、内存等上下文信息，还可以让程序在指定的地址停止运行等。

二进制代码保护和混淆

在现实生活中，攻与防的博弈无处不在。为了防止自己编写的二进制程序被逆向分析，许多软件会采取各种手段，为程序加上重重壁垒。

二进制代码的保护手段种类繁多，且运用极其灵活，例如：对汇编指令进行一定程度的混淆变换，可以干扰静态分析中的反汇编过程；在程序中穿插各种反调试技术，能有效地抵御动态分析；对程序中的关键算法进行虚拟化保护，可以给逆向工作者带来极大的阻力。

符号执行

符号执行（Symbolic Execution）是一种程序分析技术，可以通过分析程序来得到让特定代码区域执行的输入。

使用符号执行分析一个程序时，该程序会使用符号值作为输入，而非一般执行程序时使用的具体值。在达到目标代码时，分析器可以得到相应的路径约束，然后通过约束求解器来得到可以触发目标代码的具体值。

在实际环境下，符号执行被广泛运用到了自动化漏洞挖掘测试的过程中。

在CTF中，符号执行很适合解决各种逆向题，只需让符号执行引擎自动分析，找到让程序执行到输出flag正确的位置，然后求解出所需的输入即可。

二进制插桩

插桩（Instrumentation）是在保证程序原有逻辑完整性的基础上，在程序中插入探针，通过探针的执行来收集程序运行时信息的技术。

插桩往往用在以下两方面：

- 程序分析，性能分析，错误检测、捕获和重放。
- 程序行为模拟，改变程序的行为，模拟不支持的指令。

与源码级插桩相比，二进制插桩与语言无关，不需要程序的源码，也不需要重新编译链接程序，它直接对程序的机器码进行插桩，因此在逆向工程和实际漏洞挖掘中一般可以使用二进制插桩。

二进制动态插桩相对于二进制静态插桩更加强大，可以在程序运行时进行插桩，可以处理动态生成的代码，如加壳，适用的场景更加广泛。

二进制漏洞PWN

通过二进制漏洞获取计算机权限的方法或者过程被称为PWN。

在CTF中，PWN主要通过利用程序中的漏洞造成内存破坏以获取远程计算机的shell，从而获得flag。

PWN题目比较常见的形式是把一个用C/C++语言编写的可执行程序运行在目标服务器上，参赛者通过网络与服务器进行数据交互。因为题目中一般存在漏洞，攻击者可以构造恶意数据发送给远程服务器的程序，导致远程服务器程序执行攻击者希望的代码，从而控制远程服务器。

逆向工程是PWN的基础，二者的知识结构差不多。所以，有时会用二进制安全来指代逆向工程和PWN。

3. 密码学题目

近年来，CTF中密码学题目的难度不断增大，占比也越来越高。相比于Web和二进制，密码学更考验参赛者的基础知识，对数学能力、逻辑思维能力与分析能力都有很高的要求。