

MYSQL 注入天书

-----Sql-labs 使用手册



结合 **sql-labs**
讲解 **mysql** 注入方法

在线 <http://www.cnblogs.com/lcamry/category/846064.html>

Made By Lcamry 2016 年 7 月

MYSQL 注入天书.....	1
写在前面的一些内容.....	1
SQLI 和 sql-labs 介绍.....	2
Sql-labs 下载.....	2
Sql-labs 安装.....	2
第一部分/page-1 Basic Challenges.....	3
Background-1 基础知识.....	3
Less-1.....	10
Less-2.....	14
Less-3.....	15
Less-4.....	16
Background-2 盲注的讲解.....	18
Less-5.....	21
Less-6.....	29
Background-3 导入导出相关操作的讲解.....	29
Less-7.....	32
Less-8.....	35
Less-9.....	36
Less-10.....	37
Less-11.....	37
Less-12.....	39
Less-13.....	40
Less-14.....	41
Less-15.....	42
Less-16.....	42
Background-4 增删改函数介绍.....	43
Less-17.....	44
Background-5 HTTP 头部介绍.....	46
Less-18.....	50
Less-19.....	51
Less-20.....	51
Less-21.....	52
Less-22.....	53
第二部分/page-2 Advanced injection.....	54
Less-23.....	54
Less-24.....	56
Less-25.....	58
Less-25a.....	59
Less-26.....	59
Less-26a.....	60
Less-27.....	61
Less-27a.....	62
Less-28.....	62
Less-28a.....	62

Background-6 服务器（两层）架构.....	63
Less-29.....	64
Less-30.....	65
Less-31.....	66
Background-7 宽字节注入.....	66
Less-32.....	67
Less-33.....	67
Less-34.....	68
Less-35.....	70
Less-36.....	70
Less-37.....	72
第三部分/page-3 Stacked injection.....	73
Background-8 stacked injection.....	73
Less-38.....	80
Less-39.....	81
Less-40.....	82
Less-41.....	83
Less-42.....	84
Less-43.....	86
Less-44.....	86
Less-45.....	87
Background-9 order by 后的 injection.....	88
Less-46.....	88
Less-47.....	93
Less-48.....	96
Less-49.....	98
Less-50.....	98
Less-51.....	99
Less-52.....	100
Less-53.....	100
第四部分/page-4 Challenges.....	101
Less-54.....	101
Less-55.....	103
Less-56.....	103
Less-57.....	104
Less-58.....	105
Less-59.....	105
Less-60.....	106
Less-61.....	106
Less-62.....	106
Less-63.....	107
Less-64.....	107
Less-65.....	107
后记.....	108

写在前面的一些内容

请允许我叨叨一顿：

最初看到 `sql-labs` 也是好几年前了，那时候玩了前面的几个关卡，就没有继续下去了。最近因某个需求想起了 `sql-labs`，所以翻出来玩了下。从每一关卡的娱乐中总结注入的方式，这就是娱中作乐，希望能保持更大的兴趣学下去。而很多的东西不用就忘记了，有些小的知识点更是这样，网上搜了一下相关资料，基本上同仁前辈写的博客中讲解基础关。脑门一热决定给后面的人留下点东西（或许糟粕或许精华，全在你的角度），遂决定写 `blog`。`Blog` 写完了后决定总结成一个 `pdf` 文档，更方便查看。本来想着录制视频，可是时间要求太长了，所以只能先放下，此处看后期时间安排或者需求，可能的话对原作者的视频进行消音重新配音。最后希望能对后面参考该文档的人有帮助，不枉此作。

为什么要写这个？

(1) `sql` 的危害，多少的网站是被此攻破的，危害不必细讲，同样的今天网络安全形势一片大好，依旧有很多的网站存在漏洞。具体不表，可以去各大 `src` 看看。

(2) 很多的人觉得 `sql` 是如此的简单，同时很多人是华而不实，对 `sql` 注入的理解到底有多深，决定了你对此漏洞的利用方式有多么变幻莫测。个人就想着利用自己对 `sql` 注入的理解，来完成这一个游戏。当然了，`sql` 注入的内容有很多，这个是真的！因为我写的手酸。。

(3) 以前自己在学习的时候太过于痛苦，大部分的人入门的时候通过 `sql` 走进来。同时呢，`sql` 注入的世界真的很精彩，当你看到别人用智慧构成的 `payload` 时，你会感触颇多。所有形成你自己的理解和使用方法，而不是生搬硬套。此处希望通过该文档帮助到正在学习的人。**这项工作要怎么做？**

现在大致的思路分为三个部分，但是不知道有没有时间和精力能够写完。确实写的过程比较耗费时间。

(1)、通过源码和手工的方式，将所有的注入方式和造成漏洞的原因找出来，并进行学习。此处要求是对每一个类型的注入进行“深刻”的了解，了解其原理和可能应用到的场景。

(2) 通过工具进行攻击，我们此处推荐使用 `sqlmap`。此过程中，了解 `sqlmap` 的使用方法，要求掌握 `sqlmap` 的流程和使用方法，精力较足的话，针对一些问题会附 `sqlmap` 的源码分析。

(3) 自己实现自动化攻击，这一过程，我们根据常见的漏洞，自己写脚本来进行攻击。此处推荐 `python` 语言。同时，`sql-labs` 系统是 `php` 写的，这里个人认为可以精读一下每关的源码，同时针对有些关卡，可以尝试着添加一些代码来增强安全性。

Ps：工具注入和自动化注入可能延后，见第二个版本。请关注博客。

你该怎么去学？

(1) 安装环境后，动手实验。实践中遇到问题才能更大的激发兴趣。

(2) 遇到不会查资料，可以在我的博客 (www.cnblogs.com/lcamry) 中找到一些资料。或者可以请教别人，虚心请教，不耻下问。三人行必有我师焉！

(3) 书山有路勤为径，勤奋是唯一的一条路。

我的相关介绍

Blog: www.cnblogs.com/lcamry

联系 QQ: 646878467

课余时间和工作之外时间来写，时间仓促，同时个人实力有限，望各位批评指正。



SQLI 和 sql-labs 介绍

SQLI, sql injection, 我们称之为 sql 注入。何为 sql, 英文: Structured Query Language, 叫做结构化查询语言。常见的结构化数据库有 MySQL, MS SQL, Oracle 以及 Postgresql。Sql 语言就是我们在管理数据库时用到的一种。在我们的应用系统使用 sql 语句进行管理应用数据库时, 往往采用拼接的方式形成一条完整的数据库语言, 而危险的是, 在拼接 sql 语句的时候, 我们可以改变 sql 语句。从而让数据执行我们想要执行的语句, 这就是我们常说的 sql 注入。

原理性的东西我们这里就不进行详细的讲解了, 从 sql-labs 以下的每一个关卡中, 你能真正体会到什么是 sql 注入。Ps:有些朋友对工具比较熟悉, 例如 sqlmap, 可以从 sqlmap 的日志中分析每个关卡的原理。但是我个人建议先去了解原理, 再去使用工具。这样在使用工具的时候你也能深刻的理解工具起到了什么样的作用。更进一步你应该想着如果让你自己写代码实现攻击, 你应该如何写。

Ps: 因为本项工作我是在多个平台和多个浏览器下进行测试的, 所以截图等可能会有不同的环境, 但是都能说明原理。这里就不要吹毛求疵了。图片刚开始有很多都是 chrome 下截取的, 后来发现不是很好看, 所以在图片上面的 url 全部粘贴。尽量用 firefox, 有ackbar。

Sql-labs 下载

Sql-labs 是一个印度程序员写的, 用来学习 sql 注入的一个游戏教程。博客地址为:

<http://dummy2dummies.blogspot.hk/>, 博客当中有一些示例, 国内很多博客内容都是从该作者的博客翻译过来的。同时该作者也发了一套相关的视频, 在 youtube 上可以查看。ps: 印度人讲英语口音太重了。。。。。凑合着听懂点。

此处考虑到有些朋友不会翻墙, 遂分享到国内地址。

<http://pan.baidu.com/s/1bo2L1JT>

Ps: 不想看视频的可以直接忽略视频, 口音实在脑门疼, 此处本来想自己录视频的, 但现在来看, 时间比较有限

Sql-labs 项目地址---Github 获取: <https://github.com/Audi-1/sql-labs>

(考虑到安全性问题, 就不搬运这个了)

Sql-labs 安装

需要安装以下环境

(1) apache+mysql+php

(2) Tomcat+mysql+java (部分关卡需要)

如果可以的话, 推荐在 windows 和 linux 下分别安装:

Windows 下可以用 wamp、phpstudy、apmserv 等直接安装, linux 下可在网上搜索教程进行安装。例如 ubuntu 下, 新手基本靠软件中心和 apt-get 进行安装。这里就不赘述环境的安装了。

我的测试环境是 windows 下用 wamp 直接搭建的, linux 平台用 ubuntu14.04 , apache+mysql+php

同时，在后面的几个关卡中，需要用到 tomcat+java+mysql 的服务器，此处因已经安装 apache+mysql+php，所以我们需要安装 tomcat+jre+java 连接 mysql 的 jar，具体过程不详细讲解。

Sql-labs 安装

将之前下载的源码解压到 web 目录下，linux 的 apache 为 /var/www/html 下，windows 下的 wamp 解压在 www 目录下。

修改 sql-connections/db-creds.inc 文件当中的 mysql 账号密码

```
<?php  
  
//give your mysql connection username n password  
$dbuser = 'root';  
$dbpass =  
$dbname = "security";  
$host = 'localhost';  
$dbname1 = "challenges";  
  
?>
```

将 user 和 pass 修改你的 mysql 的账号和密码，访问 127.0.0.1 的页面，点击

[Setup/reset Database for labs](#)

进行安装数据库的创建，至此，安装结束。我们就可以开始游戏了。

第一部分/page-1 Basic Challenges

Background-1 基础知识

此处介绍一些 mysql 注入的一些基础知识。

(1) 注入的分类--仁者见仁，智者见智。

下面这个是阿德玛表哥的一段话，个人认为分类已经是够全面了。理解不了跳过，当你完全看完整个学习过程后再回头看这段。能完全理解下面的这些每个分类，对每个分类有属于你的认知和了解的时候，你就算是小有成就了，当然仅仅是 sql 注入上。

基于从服务器接收到的响应

▲基于错误的 SQL 注入

▲联合查询的类型

▲堆查询注射

▲SQL 盲注

•基于布尔 SQL 盲注

- 基于时间的 SQL 盲注
- 基于报错的 SQL 盲注

基于如何处理输入的 SQL 查询 (数据类型)

- 基于字符串
- 数字或整数为基础的

基于程度和顺序的注入(哪里发生了影响)

- ★ 一阶注射
- ★ 二阶注射

一阶注射是指输入的注射语句对 WEB 直接产生了影响，出现了结果；二阶注入类似存储型 XSS，是指输入提交的语句，无法直接对 WEB 应用程序产生影响，通过其它的辅助间接的对 WEB 产生危害，这样的就被称为是二阶注入。

基于注入点的位置上的

- ▲ 通过用户输入的表单域的注射。
- ▲ 通过 cookie 注射。
- ▲ 通过服务器变量注射。 (基于头部信息的注射)

(2) 系统函数

介绍几个常用函数：

1. version()——MySQL 版本
2. user()——数据库用户名
3. database()——数据库名
4. @@datadir——数据库路径
5. @@version_compile_os——操作系统版本

(3) 字符串连接函数

函数具体介绍 <http://www.cnblogs.com/lcamry/p/5715634.html>

1. concat(str1, str2, ...)——没有分隔符地连接字符串
 2. concat_ws(separator, str1, str2, ...)——含有分隔符地连接字符串
 3. group_concat(str1, str2, ...)——连接一个组的所有字符串，并以逗号分隔每一条数据
- 说的比较抽象，其实也并不需要详细了解，知道这三个函数能一次性查出所有信息就行了。

(4) 一般用于尝试的语句

Ps:--可以用#替换，url 提交过程中 Url 编码后的#为%23

```
or 1=1--+
' or 1=1--+
" or 1=1--+
) or 1=1--+
') or 1=1--+
") or 1=1--+
")) or 1=1--+
```

一般的代码为：

```
$id=$_GET['id'];
$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
```

此处考虑两个点，一个是闭合前面你的 ‘ 另一个是处理后面的 ’ ，一般采用两种思路，闭合后面的引号或者注释掉，注释掉采用--+ 或者 # (%23)

(5) union 操作符的介绍

UNION 操作符用于合并两个或多个 SELECT 语句的结果集。请注意，UNION 内部的 SELECT 语句必须拥有相同数量的列。列也必须拥有相似的数据类型。同时，每条 SELECT 语句中的列的顺序必须相同。

SQL UNION 语法

```
SELECT column_name(s) FROM table_name1  
UNION
```

```
SELECT column_name(s) FROM table_name2
```

注释：默认地，UNION 操作符选取不同的值。如果允许重复的值，请使用 UNION ALL。

SQL UNION ALL 语法

```
SELECT column_name(s) FROM table_name1  
UNION ALL
```

```
SELECT column_name(s) FROM table_name2
```

另外，UNION 结果集中的列名总是等于 UNION 中第一个 SELECT 语句中的列名。

(6) sql 中的逻辑运算

这里我想说下逻辑运算的问题。

提出一个问题 Select * from users where id=1 and 1=1; 这条语句为什么能够选择出 id=1 的内容，and 1=1 到底起作用了没有？这里就要清楚 sql 语句执行顺序了。

同时这个问题我们在使用万能密码的时候会用到。

```
Select * from admin where username='admin' and password='admin'
```

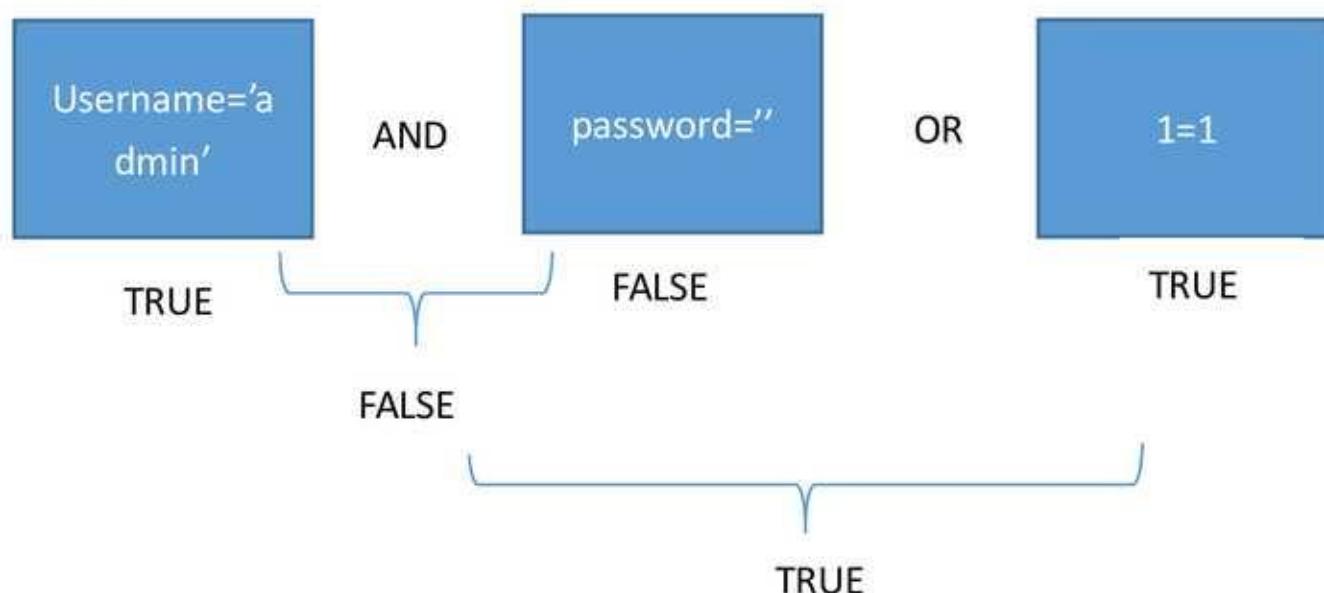
我们可以用 '' or 1=1# 作为密码输入。原因是为什么？

这里涉及到一个逻辑运算，当使用上述所谓的万能密码后，构成的 sql 语句为：

```
Select * from admin where username='admin' and password=''' or 1=1#'
```

Explain: 上面的这个语句执行后，我们在不知道密码的情况下就登录到了 admin 用户了。

原因是在 where 子句后，我们可以看到三个条件语句 `username='admin'` 和 `password=''` 或 `1=1`。三个条件用 and 和 or 进行连接。在 sql 中，我们 and 的运算优先级大于 or 的运算优先级。因此可以看到 第一个条件（用 a 表示）是真的，第二个条件（用 b 表示）是假的，`a and b = false`, 第一个条件和第二个条件执行 and 后是假，再与第三个条件 or 运算，因为第三个条件 `1=1` 是恒成立的，所以结果自然就为真了。因此上述的语句就是恒真了。



①Select * from users where id=1 and 1=1;

②Select * from users where id=1 && 1=1;

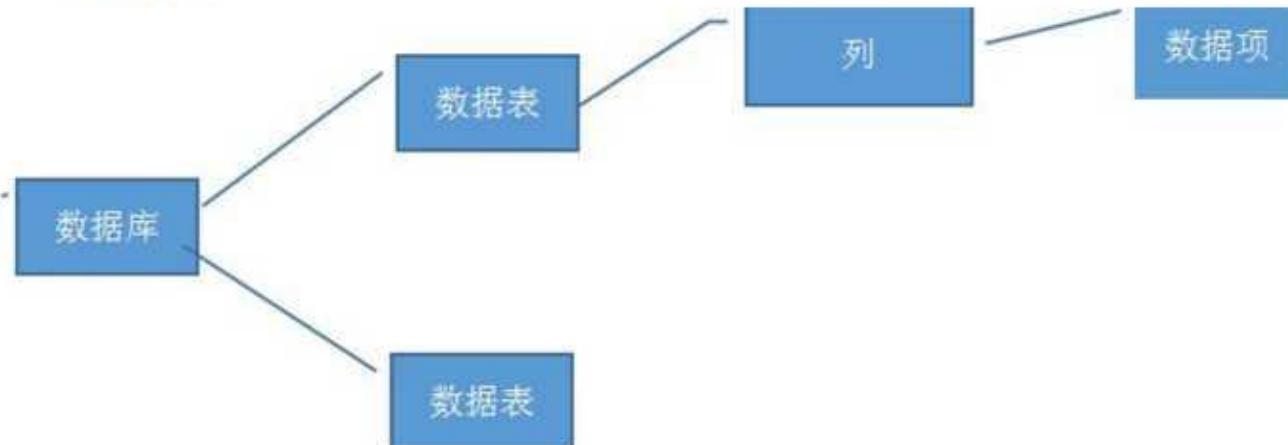
③Select * from users where id=1 & 1=1;

上述三者有什么区别？①和②是一样的，表达的意思是 id=1 条件和 1=1 条件进行与运算。

③的意思是 id=1 条件与 1 进行&位操作，id=1 被当作 true，与 1 进行 & 运算 结果还是 1，再进行=操作，1=1，还是 1 (ps: &的优先级大于=)

Ps: 此处进行的位运算。我们可以将数转换为二进制再进行与、或、非、异或等运算。必要的时候可以利用该方法进行注入结果。例如将某一字符转换为 ascii 码后，可以分别与 1, 2, 4, 8, 16, 32...。进行与运算，可以得到每一位的值，拼接起来就是 ascii 码值。再从 ascii 值反推回字符。（运用较少）

(7) 注入流程



我们的数据库存储的数据按照上图的形式，一个数据库当中有很多的数据表，数据表当中有很多的列，每一列当中存储着数据。我们注入的过程就是先拿到数据库名，在获取到当前数据库名下的数据表，再获取当前数据表下的列，最后获取数据。

现在做一些 mysql 的基本操作。启动 mysql，然后通过查询检查下数据库：

```
show databases;
```

A terminal window titled 'root@bt: /home/dhakkan' is shown. The user has typed 'show databases;' and is awaiting a response. A large white arrow points from the text '这个实验用到的数据库名为 security, 所以我们选择 security 来执行命令。' back to the terminal window, indicating the next step in the process.

```
root@bt: /home/dhakkan
File Edit View Terminal Help
Server version: 5.1.62-0ubuntu0.10.04.1 (Ubuntu)

Copyright (c) 2000, 2011, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| openx |
| owasp10 |
| security |
| test1 |
+-----+
6 rows in set (0.05 sec)
```

这个实验用到的数据库名为 security, 所以我们选择 security 来执行命令。

```
use security;
```

Mysql注入---sqlilabs---lcamry

```
root@bt: /home/dhakkan
File Edit View Terminal Help
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| openx |
| owaspie |
| security |
| test1 |
+-----+
6 rows in set (0.05 sec)

mysql> use security;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> sho
```

我们可以查看下这个数据库中有哪些表

```
show tables;
```

```
root@bt: /home/dhakkan
File Edit View Terminal Help
| openx |
| owaspie |
| security |
| test1 |
+-----+
6 rows in set (0.05 sec)

mysql> use security;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables in security |
+-----+
| emails |
| referers |
| uagents |
| users |
+-----+
4 rows in set (0.00 sec)
```

现在我们可以看到这里有四张表，然后我们来看下这张表的结构。

```
desc emails;
```

欢迎访问：电子书学习和下载网站 (<https://www.shgis.com>)
文档名称：《MySQL 注入天书 - sqlilabs 注入手册》Made by Lcamry.pdf
请登录 <https://shgis.com/post/4081.html> 下载完整文档。
手机端请扫码查看：

