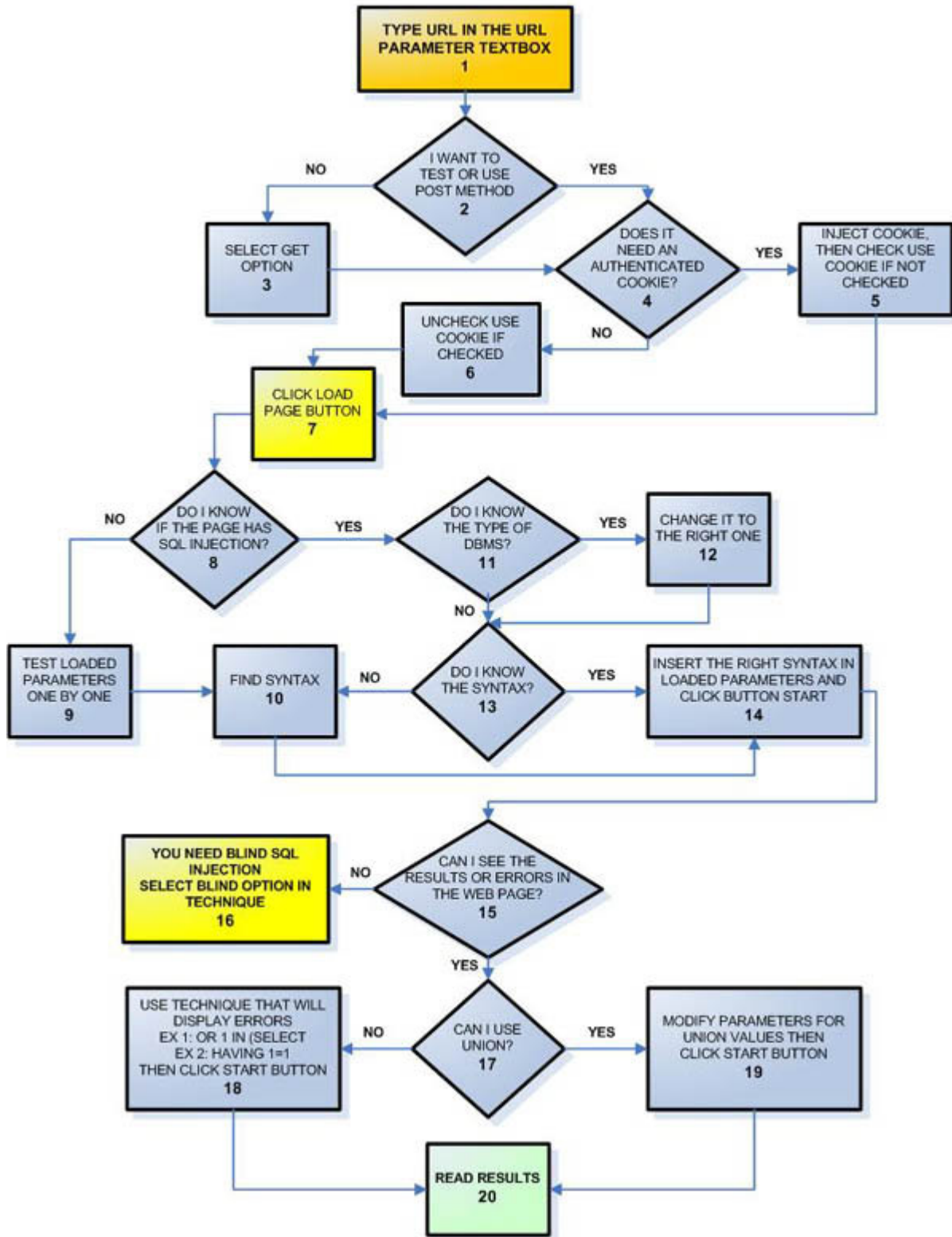
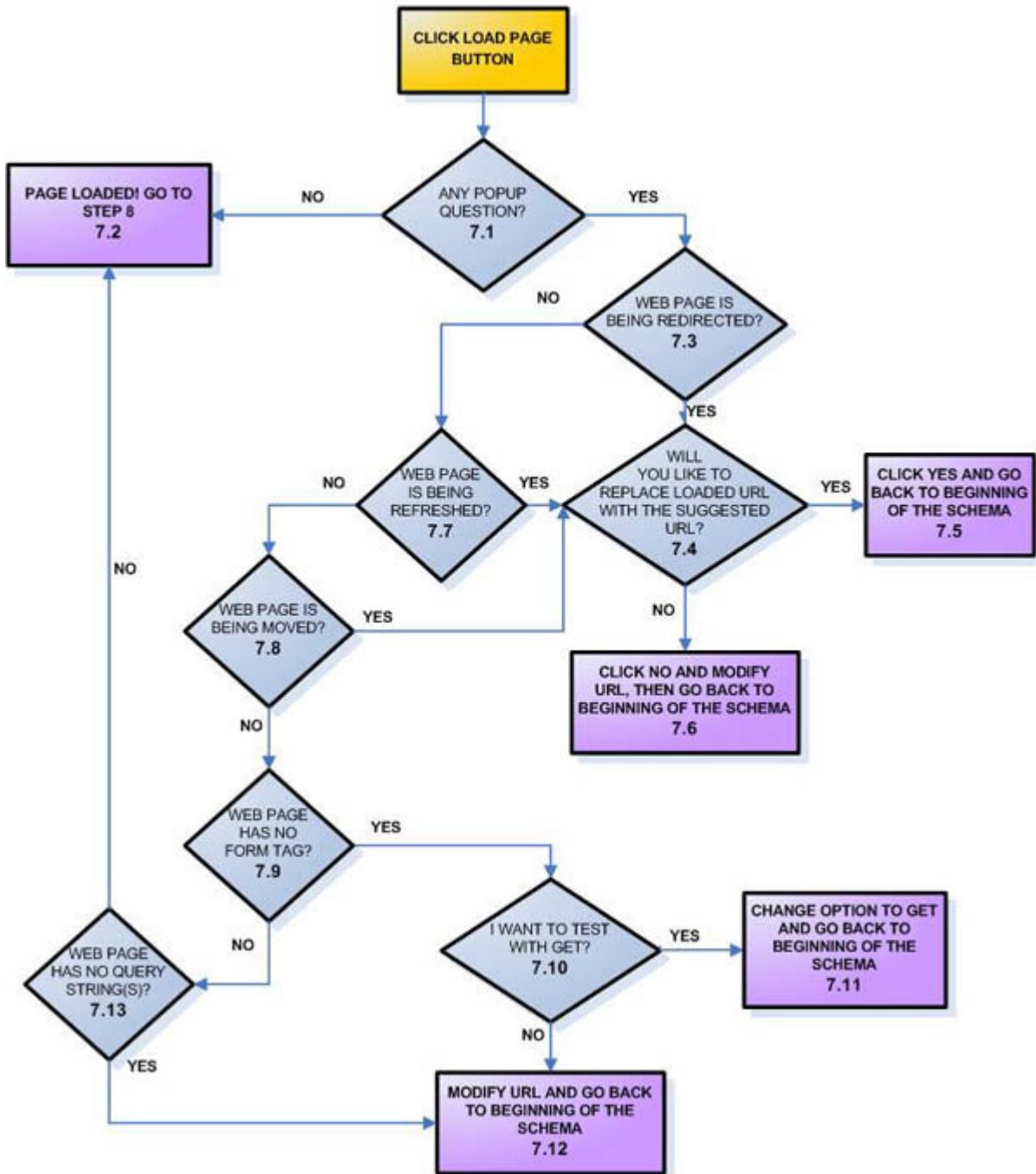


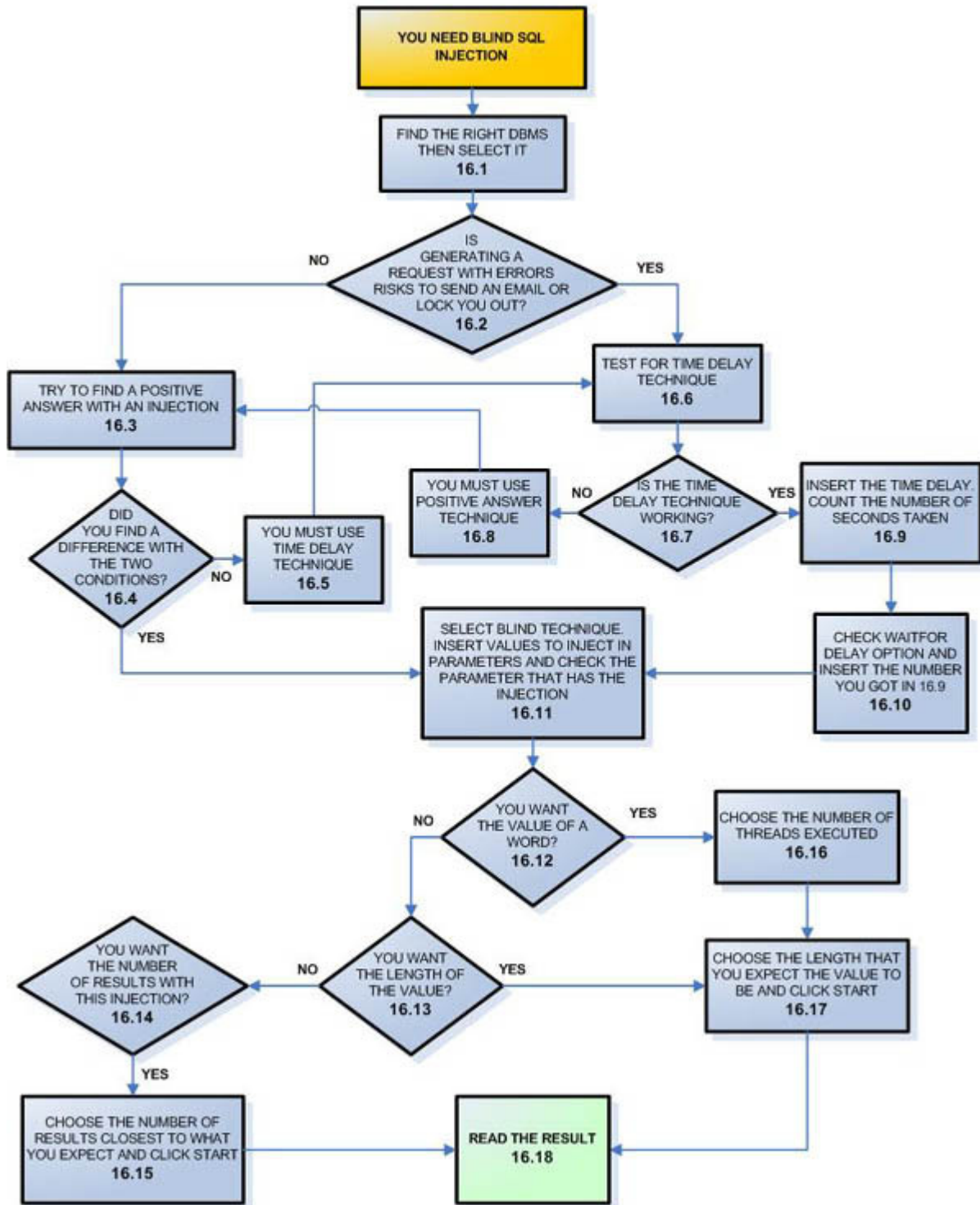
MAIN TUTORIAL SCHEMA



LOAD PAGE TUTORIAL SCHEMA



BLIND SQL INJECTION TUTORIAL SCHEMA



Please follow first the main schema and if you need more details go to their respective number detailed below. Moreover, in two circumstances you will get drill down schemas (7 and 16). There you can look at the sub-schema for more details and like the main schema go in their respective number for more details.

1. TYPE URL IN THE URL PARAMETER TEXTBOX

- Get the URL where you want to see if there are SQL injections, or that you already know there are.
- Copy paste it in that textbox



NOTE 1

You don't need to have the http:// it will automatically add it for you. However if the web site is in https you should add it yourself.

NOTE 2

It might happen that you already know there are SQL injections in a web site because you've got a positive answer from an automated tool such as Paros. I think it could be a good technique to use an automated tool to make the rough work of analyzing every page then to come back to exploit it with SQL Power Injector, unless of course you don't want to leave too much traces that those kind of applications leave.

GO TO **STEP 2**

2. I WANT TO TEST OR USE POST METHOD



That question could be quite easy to answer if your URL doesn't contain any query string variables. If it does and it's what you want to do then your answer is **NO – GO TO STEP 3**.

However sometimes you have query string variables but you still want to test it for POST so your answer is **YES – GO TO STEP 4**.

If you want to test for POST so your answer is **YES – GO TO STEP 4**.

NOTE

If you choose to use the POST method and there are no form tags in the loaded web page you will get a popup text stating that there is nothing. In that case, either you've got the wrong URL (a Frameset for example) or you should use GET method.

HINT

If you've got the message stating there are no Form tags in the requested URL, go back to the web page in the browser and right click close to the textbox you want to inject. There in the Properties copy paste the URL, it might be a different one containing your textboxes you want to test.

- IF YOUR ANSWER IS **NO** GO TO **STEP 3**
- IF YOUR ANSWER IS **YES** GO TO **STEP 4**

3. SELECT GET OPTION

- Select GET option

**NOTE**

Of course if you don't have any query string variables or they are malformed you will get a popup message stating that you don't have any and should select POST. In case they are malformed (by a bad copy paste for example) you just need to correct it.

GO TO [STEP 4](#)**4. DOES IT NEED A COOKIE?**

This case can occur when your SQL injection will be in a page where you need to be previously authenticated. Without this cookie you will never be able to load the right page and will always be kicked out.

If it's your case and you need to login first then your answer is **YES** – **GO TO [STEP 5](#)**.

If you don't need to login in then the answer is **NO** – **GO TO [STEP 6](#)**.

Another scenario is that you will need one to keep the context of the page with a session cookie even though there is no authentication at the beginning. A good example would be the .Net VIEWSTATE object that is passed from one page to the other with the help of the cookie session. If it doesn't exist, then you get an error message on the next page (in POST mode)

If it's your case and you need to keep the context then your answer is **YES** – **GO TO [STEP 5](#)**.

If you don't need to keep the context then the answer is **NO** – **GO TO [STEP 6](#)**.

NOTE 1

By default the option is set to *Use cookie*.

NOTE 2

In some rare occasion you won't want to keep the context of the cookie to be passed from one injection to the other (see blind injection technique **STEP 16**) and will want to create a fresh new cookie each time.

This case can occur when you are in a login page and succeed to have found your positive answer (special technique explained in **STEP 16.3**) by bypassing the login page (with the '**or 1=1--**' technique for example) but once in the web site there're no ways to get SQL injection. At that point, you'll need to execute your SQL injection with the positive answer technique, if you choose so and each time you have a true occurrence you'll be automatically sent to the other page when you request the page again (with the session cookie set the web application assumes that you're already authenticated). So no matter if your condition is true or not, it will redirect you there. Thus, you'll end up having only true answers... You should then consider that you don't need any cookie and go to **STEP 6**.

- IF YOUR ANSWER IS **NO** GO TO **STEP 6**
- IF YOUR ANSWER IS **YES** GO TO **STEP 5**

5. INJECT COOKIE, THEN CHECK USE COOKIE IF NOT CHECKED

In this step it's a bit trickier, since the current version has no login feature that will get the resulting authenticated session cookie. In order to achieve so, you will need to use a proxy application to trap the cookie generated or use a browser that has the feature to have the cookie information in its properties such as Mozilla.

NOTE

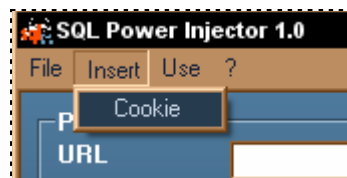
You'll get about the same information in the question "*I know I can SQL inject but only after I'm logged in the web application, what can I do?*" in the FAQ section of the web site.

Here's what you need to do to get the authenticated session cookie:

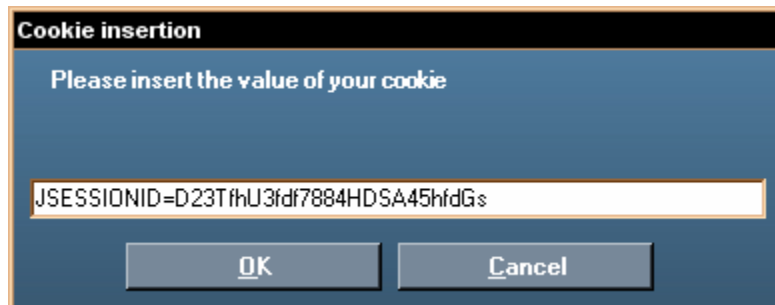
1. With a normal browser go on the web site and log on. Normally the web site will create a session with a session token in the cookie.
2. You need to get it either with the browser preferences (Mozilla, Netscape and the like) or with a proxy application such as Paros (<http://www.parosproxy.org/index.shtml>) or webscarab (<http://www.owasp.org/software/webscarab.html>) to name a few... I personally recommend using a proxy application because with the browsers you won't get the cookie information in the right format, that is to say:
MySessionID=AGDAFHAD3142324. Once you found it, just copy it.


```
GET http://localhost/VulnerableSQLSite/Search.asp HTTP/1.0
Accept: */*
Accept-Language: en-us
Pragma: no-cache
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.0.3705; .NET C
Host: localhost
Proxy-Connection: Keep-Alive
Cookie: ASPSESSIONIDSSRRTBBC=EBIPJBCBEDMHMDIPGOFLLBBCN
```

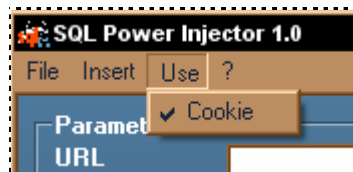
3. Now, with this information go in the application SQL Power Injector and in the menu choose the Insert and click Cookie.



4. An input box will be displayed, just paste the cookie value there and click ok. The format should look like this for a JSP session for example:
JSESSIONID=D23TfhU3fdf7884HDSA45hfdGs



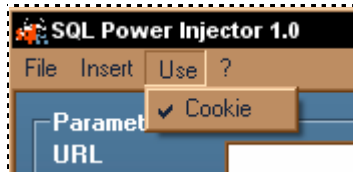
Once you're done, you will need to make sure that the application will use the cookie in the menu Use and then that Cookie is checked. If it's not checked, do so.



GO TO [STEP 7](#)

6. UNCHECK USE COOKIE IF CHECKED


As the title implies uncheck the use cookie option if it's checked. You'll find this option in the menu under Use and Cookie.



GO TO [STEP 7](#)

7. CLICK LOAD PAGE BUTTON

This step has been drilled down to sub-steps to facilitate the understanding as you can notice as well in the schemas. It was created like this to alleviate the complexity of one step in a different schema where it can be exploded in a many more understandable sub-steps. Here we assume that the user will logically follow those steps in the reading order.

- Click on Load Page button 

GO TO [STEP 7.1](#)

7.1 ANY POPUP QUESTION?

If there is no popup message after you have loaded the page then everything is all right and the answer of the question is **NO** – GO TO [STEP 7.2](#).

If you've got a popup message stating something went wrong then the answer is **YES** – GO TO [STEP 7.3](#)

NOTE

The next popup error messages discussed in the sub-steps include only the most common errors. There is no way to cover every possibility of error that an application could have, but at least the ones that are possible in the using context are explained.

- IF YOUR ANSWER IS **NO** GO TO [STEP 7.2](#)
- IF YOUR ANSWER IS **YES** GO TO [STEP 7.3](#)

7.2 PAGE LOADED! GO TO STEP 8

The page has been loaded successfully and you can get back to the main tutorial schema to continue the process.

You should get values in the String Parameter Datagrid. Those values will be from the form (INPUT tag of text, hidden and password type) if you have selected POST method or the values of the query strings in the URL if you have chosen GET method.

It should look something like that:

String Parameters				
	Name	Starting string	Varying string	Ending string
▶	<input type="checkbox"/> login			
	<input type="checkbox"/> pass			

NOTE

Of course the variables could be possibly different than this example since it depends entirely of each web page loaded.

GO TO **STEP 8** ON MAIN TUTORIAL SCHEMA

7.3 WEB PAGE IS BEING REDIRECTED?

The URL you have loaded has to be redirected, which is why you might want to follow the redirection to get to the real web page.

If you get the redirection popup message then your answer is **YES – GO TO STEP 7.4.**

If you don't get any popup message stating that there is a redirection then your answer is **NO – GO TO STEP 7.7.**

The question will look something like that:



- IF YOUR ANSWER IS **NO** GO TO **STEP 7.7**
- IF YOUR ANSWER IS **YES** GO TO **STEP 7.4**

7.4 WILL YOU LIKE TO REPLACE LOADED URL WITH THE SUGGESTED URL?

At that point the suggested URL might point to somewhere you don't want to go or the parsing of the URL has not been successful created thus displaying a weird looking URL. It gives you the choice to replace it automatically in the URL textbox parameter so the answer is **YES – GO TO STEP 7.5** or to keep the old one so the answer is **NO – GO TO STEP 7.6**.

NOTE

As mentioned before, the suggested URL might have been badly parsed to various reasons. I will try to fix that problem to handle all possibilities in a near future.

HINT 1

It could be a good idea to replace the URL with the new one even though it was badly parsed. You can always modify and fix it before to reload the new web page.

HINT 2

If you have really hard time to figure out what is the suggested URL just click No and go read the View Source (click on the View Source tab) to see the exact syntax.

- IF YOUR ANSWER IS **NO** GO TO **STEP 7.6**
- IF YOUR ANSWER IS **YES** GO TO **STEP 7.5**

7.5 CLICK YES AND GO BACK TO BEGINNING OF THE SCHEMA

Answer yes and automatically your URL will be the one suggested in the URL parameter textbox. You'll have to click again Click Load Page button.

After, you will need to go back to beginning of the "Load page" schema to continue the process (**STEP 7**).

GO TO **STEP 7 AT THE BEGINNING OF THE LOAD PAGE SCHEMA**

7.6 CLICK NO AND MODIFY URL, THEN GO BACK TO BEGINNING OF THE SCHEMA

- Click on yes
- If you need to modify or change the URL that has been loaded, do so
- Click Load Page button again

Once done, you will need to go back to beginning of the “Load page” schema to continue the process (**STEP 7**).

GO TO **STEP 7** AT THE BEGINNING OF THE LOAD PAGE SCHEMA

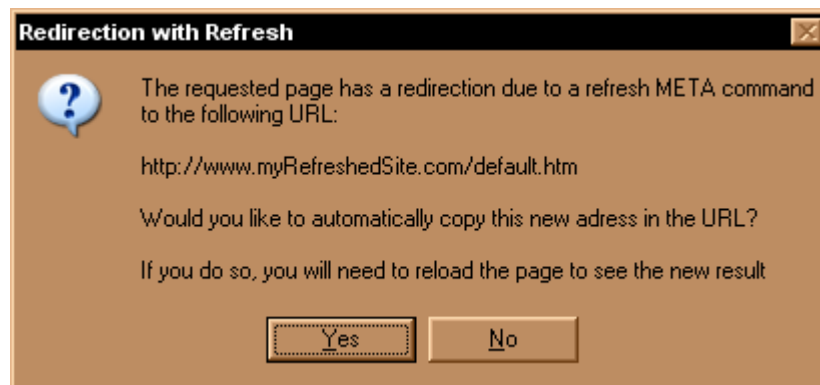
7.7 WEB PAGE IS BEING REFRESHED?

The URL you have loaded have to be refreshed, which is why you might want to follow the redirection to get to the real web page.

If you get the redirection with refresh popup message then your answer is **YES – GO TO STEP 7.4**.

If you don't get any popup message stating that there is a redirection due to a META Refresh tag then your answer is **NO – GO TO STEP 7.8**.

The question will look something like that:



- IF YOUR ANSWER IS **NO** GO TO **STEP 7.8**
- IF YOUR ANSWER IS **YES** GO TO **STEP 7.4**

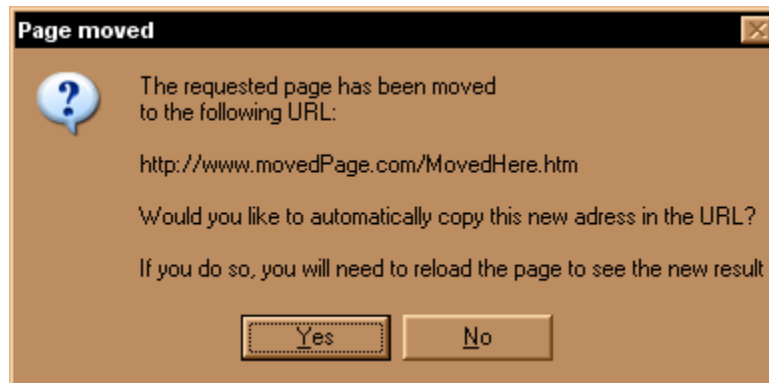
7.8 WEB PAGE IS BEING MOVED?

The URL you have loaded has moved, which is why you might want to follow the redirection to get to the real web page.

If you get the Page moved popup message then your answer is **YES – GO TO STEP 7.4**.

If you don't get any popup message stating that it has been moved then your answer is **NO – GO TO STEP 7.9**.

The question will look something like that:



- IF YOUR ANSWER IS **NO** GO TO **STEP 7.9**
- IF YOUR ANSWER IS **YES** GO TO **STEP 7.4**

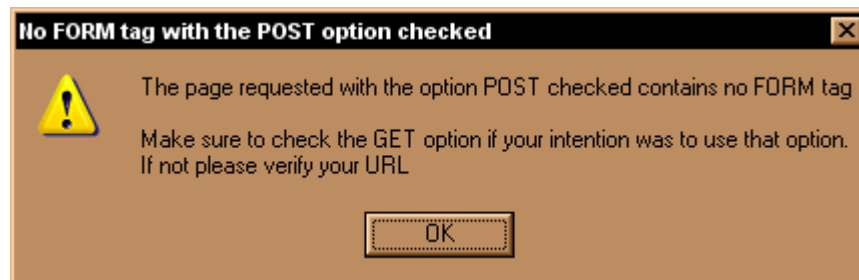
7.9 WEB PAGE HAS NO FORM TAG?

The URL you have loaded has no form tag and you selected the option POST.

If you get this message then your answer is **YES** – GO TO **STEP 7.10**.

If you don't get any popup message stating that no form tag has been found then your answer is **NO** – GO TO **STEP 7.13**.

The popup message will look something like that:



- IF YOUR ANSWER IS **NO** GO TO **STEP 7.13**
- IF YOUR ANSWER IS **YES** GO TO **STEP 7.10**

7.10 I WANT TO TEST WITH GET?

As the popup message stated in the **STEP 7.9** you might have wanted to load the page with GET method but you forgot to select it, if it is your case then your answer is **YES** – GO TO **STEP 7.11**.

If it was really the POST option you wanted perhaps you have made a mistake in the URL and should change it, in that case your answer is **NO – GO TO STEP 7.12**.

- **IF YOUR ANSWER IS NO GO TO STEP 7.12**
- **IF YOUR ANSWER IS YES GO TO STEP 7.11**

7.11 CHANGE OPTION TO GET AND GO BACK TO BEGINNING OF THE SCHEMA

You obviously forgot to select the GET option, just do so. After you're done you'll have to click again Click Load Page button.



After, you will need to go back to beginning of the “Load page” schema to continue the process (**STEP 7**).

GO TO STEP 7 AT THE BEGINNING OF THE LOAD PAGE SCHEMA

7.12 MODIFY URL AND GO BACK TO BEGINNING OF THE SCHEMA

Your URL might have some mistake in it, verify if it's the right one or need to be modified and if you need to modify it, do so. After you're done you'll have to click again Click Load Page button.

NOTE

If you're in GET method and get the popup message that there are no query string(s) it might mean that you have a malformed query pair name value. Verify that part and make sure it hasn't been cut in the middle of it.

It can happen when you copy paste a URL from an email.

After, you will need to go back to the beginning of the “Load page” schema to continue the process (**STEP 7**).

GO TO STEP 7 AT THE BEGINNING OF THE LOAD PAGE SCHEMA

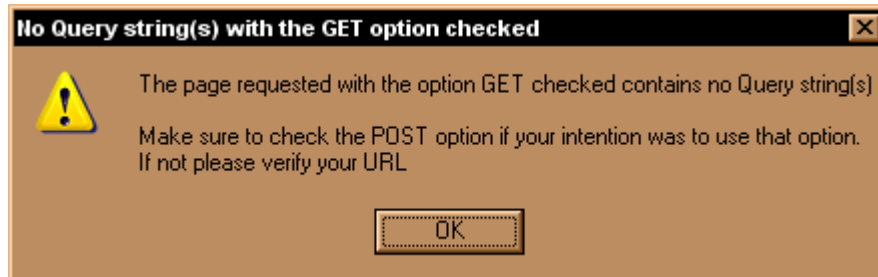
7.13 WEB PAGE HAS NO QUERY STRING(S)?

The URL you have loaded has no query string(s) and you selected the option GET.

If you get this message then your answer is **YES – GO TO STEP 7.12**.

If you don't get any popup message stating that no query string(s) have been found then your answer is **NO** – **GO TO STEP 7.2**.

The popup message will look something like that:



- IF YOUR ANSWER IS **NO** GO TO **STEP 7.12**
- IF YOUR ANSWER IS **YES** GO TO **STEP 7.2**

8. DO I KNOW IF THE PAGE HAS SQL INJECTION?

If you already know that there are SQL injections in that page then your answer is **YES** – **GO TO STEP 11**.

If you don't know then your answer is **NO** – **GO TO STEP 9**.

HINT

If you don't mind to be detected or blocked you can always use an automated tool to scan the whole web application to see if there are easy to find SQL injections. Usually, those scanners of vulnerabilities are not extremely good to find subtle SQL injections, but at least they can search several hundreds of page fairly easily for you. Where you might give up after ten or so, that application will do all of them.

However, I always rely on myself and it's where SQL Power Injector comes handy.

- IF YOUR ANSWER IS **NO** GO TO **STEP 9**
- IF YOUR ANSWER IS **YES** GO TO **STEP 11**

9. TEST LOADED PARAMETERS ONE BY ONE

This step is one of the most important given that it's the one where you will actually search for the existence of SQL injection vulnerabilities.

There are several ways to find them and I'm not going to list all of them since the aim of this tutorial is not have a crash course or white paper on SQL injection but more on how to use it in order to optimize the search. Nonetheless, I will give some syntax examples in the hint boxes that could possibly help.

Here is what you need to know for how you can find them with the application. We stated that once the page has been loaded it will discover all the inputs that you can inject depending of the method (GET or POST) and fill up a Datagrid.

NOTE

You don't need to bother with the encoding of the strings if it's in the URL (for GET method) or in the input objects. SQL Power Injector will automatically encode them in the right format to be understood on the web server. Sometimes this encoding will even at the same time bypass IDS, Reverse Proxies or any filtering mechanisms!

In this example we have two parameters: login and pass. We can start to test for injection right now. What we need to do is to insert the value we want to test inside the Starting string (shown up by the red box in the next figure).

String Parameters				
	Name	Starting string	Varying string	Ending string
▶	<input type="checkbox"/> login			
	<input type="checkbox"/> pass			

In the search of SQL injection we don't need to bother with the two other columns (Varying String and Ending String) for now, those will be used only for the blind SQL injection technique (it will be fully explained in the **STEP** and **SUB-STEP 16**). The Starting String represents the same thing than a normal textbox. All you will type there will be sent as if you modified the parameter in the URL or INPUTS fields.

In this case, we will use the Normal technique. Make sure that it's selected before to go on.

Technique

Normal
 Blind

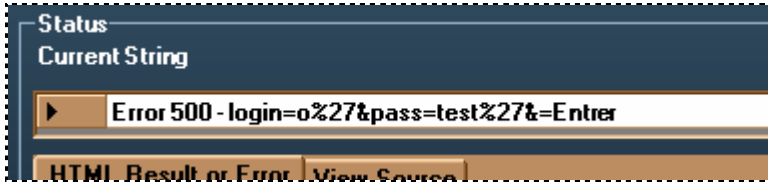
To test the values it's as easy as to fill them one by one, some of them or all of them and look at the result in status section both in the current string bar and the mini-browser in the bottom section. Once the parameter(s) that you want to test are filled, just click on the Start button.

Start

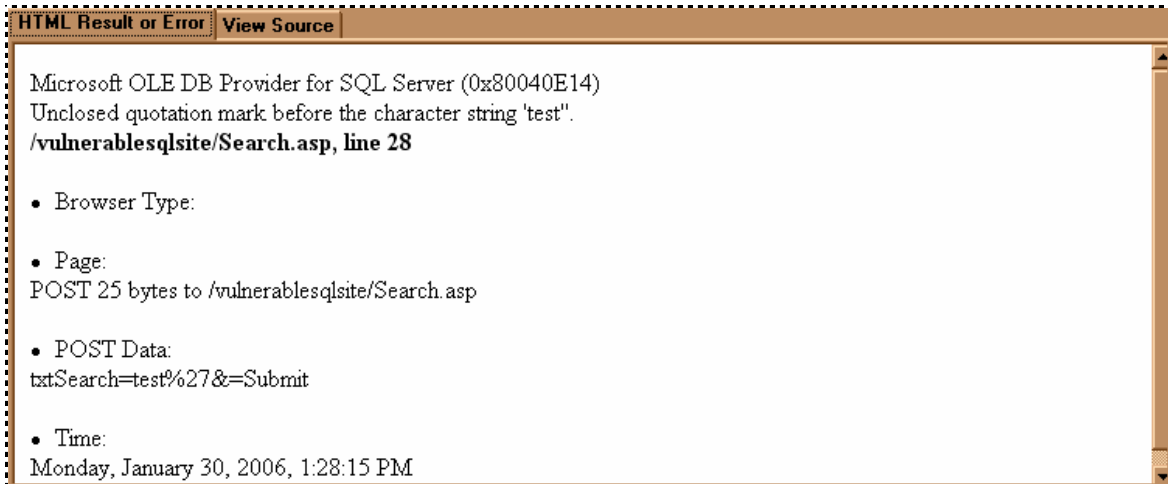
Here we tested both the values on the web site that we know that there are SQL injection vulnerabilities with quotes o' and test' and we've got interesting results:

String Parameters				
	Name	Starting string	Varying string	Ending string
	<input type="checkbox"/> login	o'		
▶	<input type="checkbox"/> pass	test'		

First, in the current string result we can see there is an error 500. It's a good sign that there is something amiss and that SQL injection could be the cause. Notice that you have as well the exact syntax that is sent to the web server, it can be useful to know.



Then the explicit result in the mini-browser:



We can plainly see that there is a SQL injection problem. That is one of the easiest ways to find out the existence of SQL injection since we have the explicit result in the response of the page. But it's not always like that...

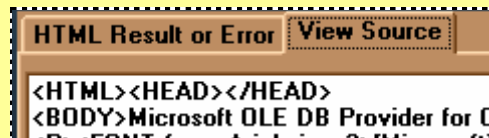
Sometimes you will get a nice page having a generic error message, such on the SQL Power Injector web site. (www.sqlpowerinjector.com)



In that case you will need to do extra work and sometimes for nothing since the error could have been generated by a bad conversion of a string to an integer or because the modified value generates an application error. In some context it could be interesting for other kind of attacks, but here we cover the SQL injection.

HINT 1

Look also at the returned actual source text of the page in the View Source tab. Sometimes you will see information that will lead you to see that you succeed your injection.



In one of my pen test nothing was displayed in the returned page, but in the source code the ODBC error message was there between comment tags...

Other times the site might be vulnerable to SQL injections but some filtering mechanisms or any other kind of software or hardware will block them (IDS, Reverse Proxies, etc...) In those occasions you just need to be more clever and furtive.

HINT 2

Many web sites will rely their security on doubling the quote (') thus preventing the attack. It's pretty common in PHP since before the version 5 there were no ways to use the equivalent of the "stored procedure" or "prepared statement" in that technology.

The problem with that technique is that if you use a numerical parameter you don't even need to use the quote to succeed the injection! But even if we need one there are so many ways to bypass that problem, many great papers explain or at least hint at them.

Just a nice trick if you can't use quotes in the injection when comparing a string is to use its ASCII version for example:

```
SELECT Count(*) FROM MyUserTable WHERE login LIKE '%admin%'
```

Is the same thing than

```
SELECT Count(*) FROM MyUserTable WHERE login LIKE  
Char(37)+Char(97)+Char(100)+ Char(109)+Char(105)+Char(110)+Char(37)
```

Notice that there are no quotes anymore.

HINT 3

Some web sites will rely their security on really lousy rules such as to find some "dangerous" SQL command words and raise an error. (Union, Select, or 1=1, etc...)

Not to mention that some valid requests can be invalidated, considering that many of those words are usual English words, it's pretty easy to bypass those mechanisms when

the attacker is aware of that feature.

The use of encoding or using comments `/* */` between the words or even putting more than one space can work pretty well. That's why there is an option to automatically put comments in the injection in the application.



HINT 4

Another nice trick is with a numerical value and add 1 to it.

Let's say we have three parameters: Country, Language and NewsId. When we change the NewsId for 99 we get a different one. But what happen if we actually send to the server 99+1?

String Parameters		
	Name	Starting string
<input type="checkbox"/>	Country	England
<input type="checkbox"/>	Language	en-us
<input type="checkbox"/>	NewsId	99+1

If we get the same news than with the one with the NewsId=100 it means we found a SQL injection! Notice again that there is no use of quotes here...

Note: normally you would need to encode the + sign by %2B but the application as explained earlier encodes it for you.

HINT 5

A last trick is to use the **WAITFOR DELAY** technique. This technique will only work for SQL Server. So even if you're not sure at this point which DBMS it is you can always try it, it cannot harm and at the same deduce the server from it.

String Parameters		
	Name	Starting string
<input type="checkbox"/>	Country	England';WAITFOR DELAY '0:0:10'-
<input type="checkbox"/>	Language	en-us
<input type="checkbox"/>	NewsId	100

As soon as you clicked on Start button count the number of seconds elapsed and see if it's about the same than the one you inserted (10 seconds in this example).

If the web server respond after about the time set, voila! You found SQL injection!

Note: this technique will be further explained in the blind SQL injection section **STEP 16.12**

GO TO [STEP 10](#)

10. FIND SYNTAX

Once you have found the existence of SQL injection you can now work on the right syntax to get valuable information. However it is entirely possible you've got the right syntax in the previous **STEP 9**. A lot of examples can be found there in the hint boxes.

Depending of if you see the results in the returned page or not, the syntax will be completely different. Those conditions will be explained later and some hints of syntax will be explained accordingly to the situation.

GO TO [STEP 14](#)

11. DO I KNOW THE TYPE OF DBMS?

For now the application supports three DBMS: SQL Server, Oracle and MySQL. I would assume that it supports Sybase as well given that it's about the same thing than SQL Server but it hasn't been tested officially so I can't tell for sure.

Although that it supports three DBMS, with the normal technique it's not really important. Because the values you modified in the Datagrid are exactly the same sent (except for the encoding part of course) and nothing is added to make it dependent on a DBMS language. In other hand, with blind SQL technique we add some functions (Count, length, ASCII, substrng functions) that are dependent to a DBMS language.

However, it's still good to know at this point which DBMS it is to be able to use the special commands or functions owned by each DBMS in your injection.

If you already know which DBMS it is then your answer is **YES – GO TO [STEP 12](#)**.

If you don't know then your answer is **NO – GO TO [STEP 13](#)**.

HINT

Here is a pretty efficient trick to find out which DBMS it is. We assume that you found already a way to inject in the web page.

The **user** command: this command displays the current database user using the connection string

Try this: Select user in your injection (in a '**UNION SELECT user** or '**or 1 in (SELECT user)** for example)

If you don't get any error it means it's a SQL Server.

To see if it's MySQL use the same syntax but user with () after that is to say **user()** . If

you don't get any errors then it's a MySQL server.

Finally to see if it's Oracle add **FROM DUAL** after your **SELECT USER** (without the parenthesis this time), if it works then it's an Oracle database. Oracle needs to always have the **FROM** statement in its **SELECT** statement.

- IF YOUR ANSWER IS **NO** GO TO **STEP 13**
- IF YOUR ANSWER IS **YES** GO TO **STEP 12**

12. CHANGE IT TO THE RIGHT ONE

This step is simply to change to the right one if you need so.



NOTE

By default the application is set SQL Server

GO TO **STEP 13**


13. DO I KNOW THE SYNTAX?

If you already know what is the syntax to inject then your answer is **YES** – GO TO **STEP 14**.

If you don't know then your answer is **NO** – GO TO **STEP 10**.

- IF YOUR ANSWER IS **NO** GO TO **STEP 10**
- IF YOUR ANSWER IS **YES** GO TO **STEP 14**

14. INSERT THE RIGHT SYNTAX IN LOADED PARAMETERS AND CLICK BUTTON START

- Insert the right syntax in the associated loaded parameters
- Click on Start button 

GO TO **STEP 15**

15. CAN I SEE THE RESULTS OR ERRORS IN THE WEB PAGE?

After you have clicked on the Start button, look at the resulting web page in the mini-browser. You might see a 500 error message like in the **STEP 9**. If it's the case then your answer is **YES – GO TO STEP 17**. Also see Hint 1 in the **STEP 9** to see if it's the case, if so then your answer is **YES – GO TO STEP 17**. Finally, the results can be seen in a **UNION** query, so that would make the answer **YES – GO TO STEP 17**.

If you don't see anything that could give you any clue on the data injected then your answer is **NO – GO TO STEP 16**.

NOTE

Perhaps the goal is not to see any data from the database and if it's the case you don't need to go forward. This scenario might happen if what you needed to do was to bypass the login page with a '**or 1=1--**' command.

But it's always interesting to see if there are hidden databases that could hold really sensitive data that are not linked to the current web application.

- IF YOUR ANSWER IS **NO** GO TO **STEP 16**
- IF YOUR ANSWER IS **YES** GO TO **STEP 17**

16. YOU NEED BLIND SQL INJECTION

This step has been drilled down to sub-steps to facilitate the understanding as you can notice as well in the schemas. It was created like this to alleviate the complexity of one step in a different schema where it can be exploded in a many more understandable sub-steps. Here we assume that the user will logically follow those steps in the reading order.

Unfortunately for you there are no results or errors that can give you any hint on the data injected. It means you'll have to do it in the hard way. It can be really time consuming and so many time spent to get nothing really valuable that many would abandon during the process. But now with SQL Power Injector the tedious repetitive tasks are automated.

As stated in good software design architecture, the menial tasks are done for you and you can concentrate on the business logic, in our case the SQL injection.

NOTE

Once you selected the Blind technique you notice that most of the application changes with new textboxes and options. That's normal, it's just that I chose to show accordingly to the context what is used or not.

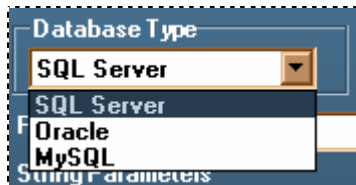
GO TO STEP 16.1

16.1 FIND THE RIGHT DBMS THEN SELECT IT

If you don't know which one it is you will need to find out. If in the **STEP 11** it wasn't that important to know here it is. Because the application will add all the SQL commands necessary to automate the injection with syntax that is proprietary to the DBMS.

You can use the same trick given in the Hint of **STEP 11**.

If you already know or just found out, change to the right one if you need to.



GO TO **STEP 16.2**

16.2 IS GENERATING A REQUEST WITH ERRORS RISKS TO SEND AN EMAIL OR LOCK YOU OUT?

Some web sites after a while will detect the attack attempts caused by the error and will block you out or slow you down. Or sometimes an Email will be sent out every time an error is generated. This occurrence will likely to happen when there is a nice web page stating that there is an error. (See my error page on **STEP 9**). But you never know, it could be any security software sending them. And of course, if an Email is sent every time you make an attempt you are more likely to have someone to react and block or slow you down after each time.

For any reasons, if you think it's risky to generate error on the web site your answer is **YES – GO TO STEP 16.6**.

If you don't see any risks then your answer is **NO – GO TO STEP 16.3**.

IMPORTANT NOTE

Even if your answer is **YES** it doesn't mean using the Positive answer technique will end up generating errors every time. The odds are fairly high but if you can find a situation where the negative answer doesn't generate an error, well you can use that technique without fear, it is actually much faster than the Time Delay technique (it's what happens when you go to **STEP 16.6**). Those two techniques will be described more in details later.

- IF YOUR ANSWER IS **NO** GO TO **STEP 16.3**
- IF YOUR ANSWER IS **YES** GO TO **STEP 16.6**

16.3 TRY TO FIND A POSITIVE ANSWER WITH AN INJECTION

I will explain what I mean by positive answer. It's a special technique that I'm not quite sure it has been documented before but in the context of SQL Power Injector it comes really handy to master.

Your goal is to make the response of your injection to be different when a condition is true and when a condition is false. The difference can be in the text or can be in the cookie, it doesn't matter.

Let's use an example. We have a search page that displays an author list (from the Pubs SQL Server database) depending on the search criteria. We found that there is an SQL injection error but unfortunately there is a nice page Error Page with no information whatsoever. And worse, there is no way to use the **UNION** command.

1. We try to find a way to SQL inject the condition without raising an error. We succeed with '**and 1=1--**

String Parameters		
	Name	Starting string
▶	<input type="checkbox"/> txtSearch	' and 1=1--

And we get this information in the mini-browser:

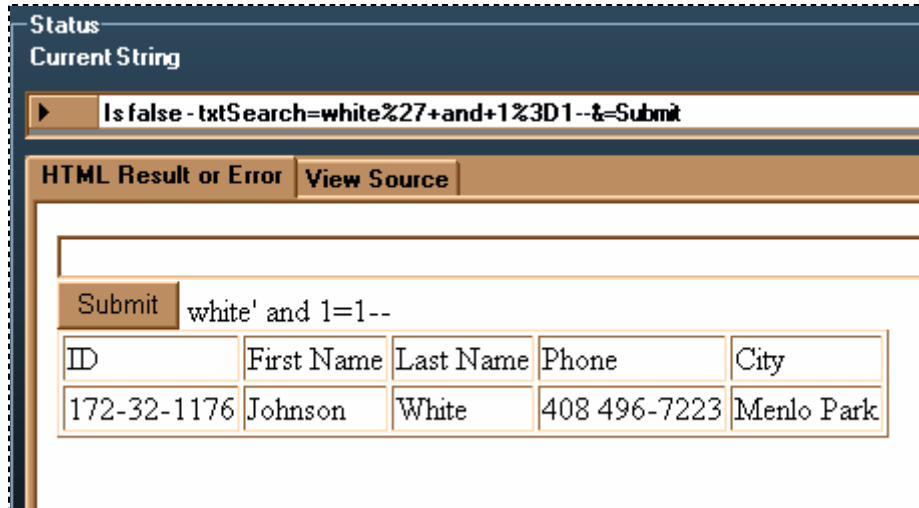
HTML Result or Error	View Source			
<input type="text"/>				
Submit ' and 1=1--				
No result found!				
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
ID	First Name	Last Name	Phone	City

2. We try to see what happen with '**and 1=2--** and search for the difference. We get this information:

HTML Result or Error	View Source			
<input type="text"/>				
Submit ' and 1=2--				
No result found!				
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
ID	First Name	Last Name	Phone	City

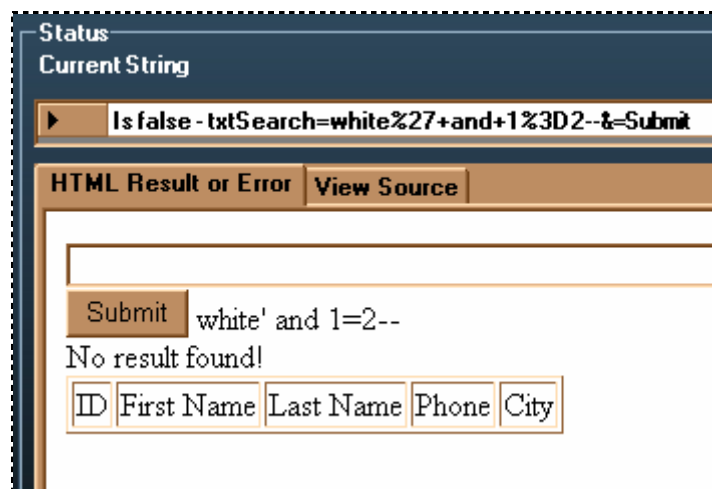
The same information! Even in the View Source there are no differences.

- Let's try to inject it with a real value, let's use **white' and 1=1--**.



We have two interesting information: the Current String states it is at "Is false" and we get a result with this injection.

- Now we want to try with the negative condition **white' and 1=2--**. Let's see what we get:



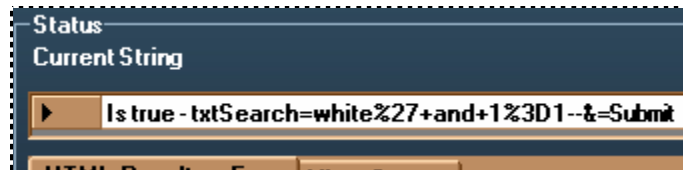
This time we have most definitely something different! Again notice that the Current String value states it is at "Is false"

- We compare the two results and try to find text in the positive condition that you are sure that won't be on the negative one. We decided to use Johnson after we searched that word in the View Source section of the negative condition and found none.

6. We insert it in the Positive Answer textbox.



7. We click on Start again and now we can see that in the Current String we have “Is true”



You're done! You found one.

NOTE

There are various ways to find a positive condition and the ' **and 1=1--** is one of them. Sometimes I succeed with ' **or 1=1--** as well. Also you might have to add some parenthesis or extra special characters in order to succeed it. Every situation has its own syntax.

GO TO [STEP 16.4](#)

16.4 DID YOU FIND A DIFFERENCE WITH THE TWO CONDITIONS?

If you found your Positive Answer value then your answer is **YES** – GO TO [STEP 16.11](#).

If you didn't find any difference in the text or cookie then your answer is **NO** – GO TO [STEP 16.5](#).

- IF YOUR ANSWER IS **NO** GO TO [STEP 16.5](#)
- IF YOUR ANSWER IS **YES** GO TO [STEP 16.11](#)

16.5 YOU MUST USE TIME DELAY TECHNIQUE

You are here because you haven't found any Positive Answer in the **STEP 16.3**, but all is not lost because you can switch to Time Delay technique. The only thing is that it's much slower.

GO TO [STEP 16.6](#)

16.6 TEST FOR TIME DELAY TECHNIQUE

This technique as the name implies is to see if there are ways to inject time delay at the DBMS server side. It's a special technique that makes the current connection to the DBMS server to hang to the time set. You can see that you succeeded when the web page is displayed after the time you set as the delay.

NOTE

Most of the time it's roughly one second more than the time set because the page needs time for the round trip and the execution on the server side.

This technique is really useful when you don't have result back in the return web page or when you want to make sure that you won't create any error on the Server side while in the automatic mode.

It can be really hard to exploit if you are not using SQL Server. Because in Oracle you need to be inside a BEGIN and END clause in order to inject the semi-colon (;) and then the sleep(sec) function. It happens really rarely.

MySQL is much subtler than Oracle and SQL Server, a method has been found with the function BENCHMARK that can create a delay depending on the number set. It is unfortunately unreliable in SQL Power Injector, although it does work.

HINT

I will show an example for each DBMS for the time delay with a delay of 5 seconds.

SQL Server:

InjectedValue'; WAITFOR DELAY '0:0:5'--

ORACLE:

InjectedValue'; BEGIN DBMS_LOCK.SLEEP(5); END;

MySQL:

InjectedValue OR IF (1=1, BENCHMARK(500000, MD5(CHAR(1))), null)

GO TO STEP 16.7

16.7 IS THE TIME DELAY TECHNIQUE WORKING?

If you succeed to make it work then your answer is **YES** – **GO TO STEP 16.9**.

If you didn't make it work then your answer is **NO** – **GO TO STEP 16.8**.

- IF YOUR ANSWER IS **NO** GO TO **STEP 16.8**
- IF YOUR ANSWER IS **YES** GO TO **STEP 16.9**

16.8 YOU MUST USE POSITIVE ANSWER TECHNIQUE

You are here because you didn't make the time delay work in the **STEP 16.6**, but all is not lost because you can switch to Positive Answer technique. The good news is that it's much faster. However it is possible that you might end up generating errors.

GO TO [STEP 16.3](#)

16.9 INSERT THE TIME DELAY, THEN COUNT THE NUMBER OF SECONDS TAKEN

Chances are that you have already inserted the time delay in the Starting String to test it out. If you didn't it's time to do so.

Now right after you clicked the Start button count the number of seconds taken, yes it might seem silly since we are the one to set the delay but in some rare occasions the injection you're doing might be concatenated in several places. So, knowing that possibility we need to count and when we have the number we have to do a simple math calculation.

If the delay set is 5 seconds and it took around 15 seconds, how many concatenations occurred?

5 seconds for normally 1
15 seconds for ?

(15 * 1) / 5 = 3 places

That information could be useful to know, but in our case only the number of seconds was enough. It's the number you need for the parameter "Delay in seconds".

GO TO [STEP 16.10](#)

16.10 CHECK WAITFOR DELAY OPTION AND INSERT THE NUMBER YOU GOT IN 16.9

- Check WAITFOR DELAY option



- Set the number of seconds counted (can be different of the one you set)

**HINT**

You can fine tune the speed of the results by lowering the number in “Delay in seconds” by 1 each time and the one you set in the Ending String until you get a lot of false positive. Generally 3 seconds is the best.

NOTE

There is a limitation for this technique to one thread in case that you have the option use cookie checked and you use more than one thread. The problem, as far as I understood it, is that the web server recognizes the requester using the same session cookie and will give the same thread on the server side, adding thus the total number of seconds that all the thread together take (ex: 5 threads with 3 seconds delay will take 15 seconds total for each request). It applies on IIS, I must verify on the other web servers if it's the case.

If it happens, you have to uncheck the use cookie option in the menu. If you absolutely need it, then just use one thread at the time, it's not perfect but it's better than nothing.

GO TO [STEP 16.11](#)**16.11 SELECT BLIND TECHNIQUE. INSERT VALUES TO INJECT IN PARAMETERS AND CHECK THE PARAMETER THAT HAS THE INJECTION**

You have to select the Blind technique before to continue.



Now we are in a major piece of SQL Power Injector that is more complex than the rest of the application but once understood it's pretty easy to use.

The main part to be understood in that technique is the concept of the three strings portion. These three text insertion values are the main core of what makes the application powerful.

So far you have found SQL injection and you know which field or row will be used to inject SQL. It's the one you need to check and at that moment the application will use the three text insertion values that you inserted.

Let's say we have five parameters: Country, txtSearch, Language, jscript and css. And we discovered that there is a SQL injection vulnerability with txtSearch. So we will check that parameter.

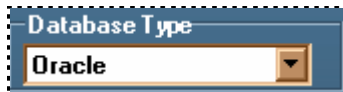
String Parameters				
	Name	Starting string	Varying string	Ending string
<input type="checkbox"/>	Country	Canada		
<input checked="" type="checkbox"/>	txtSearch	SQLInjectionFoundHere'		
<input type="checkbox"/>	Language	en-us		

Checking this parameter will add according to the type chosen (Word, Length or Count) some strings after the Starting String (where it gets its name from) and before the Varying String and finally after the Varying String and before the Ending String (where it gets its name from as well).

Let's use the same parameters to demonstrate it with an example. For example we want to get the IP address of the server hosting an Oracle DB with the Positive Answer technique:

Options:

- **Database type:** Oracle



- **Mode:** Blind



- **Type:** Word



- **Starting length:** 30



- **Number of threads:** 8



- **Positive answer textbox:** CLERK



String Parameters				
	Name	Starting string	Varying string	Ending string
<input type="checkbox"/>	Country	Canada		
<input checked="" type="checkbox"/>	txtSearch	SMITH' AND	SYS_CONTEXT('USERENV', 'IP_ADDRESS']	FROM DUAL)--
<input type="checkbox"/>	Language	en-us		

Resulting in this SQL statement:

SMITH' AND 30 < (SELECT ASCII(SUBSTR(CAST(SYS_CONTEXT('USERENV', 'IP_ADDRESS') AS VARCHAR(4000)), 1, 1)) FROM DUAL)--

Legend:

- Blue: Starting string
- Red and italic: Automatically added strings
- Purple and bold: Varying string
- Green: Ending string

As you can see many strings are added automatically and some value will change until it gets the full IP address value of the previous example.

You will have this result with all the requests sent:

Status	Current String
Thread 4: Istrue - Country=Canada&txtSearch=SMITH%27+AND+46%3D%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_AD	
Thread 3: Istrue - Country=Canada&txtSearch=SMITH%27+AND+57%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_ADDRE	
Thread 2: Isfalse - Country=Canada&txtSearch=SMITH%27+AND+49%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_ADDR	
Thread 1: Isfalse - Country=Canada&txtSearch=SMITH%27+AND+49%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_ADDR	
Thread 8: Isfalse - Country=Canada&txtSearch=SMITH%27+AND+45%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_ADDR	
Thread 3: Istrue - Country=Canada&txtSearch=SMITH%27+AND+55%3D%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_AD	
Thread 2: Istrue - Country=Canada&txtSearch=SMITH%27+AND+50%3D%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_AD	
Thread 1: Istrue - Country=Canada&txtSearch=SMITH%27+AND+49%3D%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_AD	
Thread 8: Istrue - Country=Canada&txtSearch=SMITH%27+AND+46%3D%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_AD	
Thread 8: Istrue - Country=Canada&txtSearch=SMITH%27+AND+255%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_ADDR	
Thread 8: Istrue - Country=Canada&txtSearch=SMITH%27+AND+127%28select+ASCII%28SUBSTR%28CAST%28SYS_CONTEXT%28%27USERENV%27,+%27IP_ADDR	

And now you want to get the current user name? Easy! You just need to change the Varying String to USER. As you can see once the syntax is found, the rest is just a matter to change some value.

Subsequently you can construct your SQL string to inject with the specific commands you want and it will fill in the information that is repetitive if you are in blind mode.

I discovered with time that when I was building my SQL strings to inject that the syntax could be broken in three main portions. And if some value can change as you have seen in the example, the one that always revolve around is the Varying String.

HINT

You can use your mouse to hover a row to get the real value that will be sent to the web server. It is really useful when you are confused with what is automatically injected. Also, it's a good way to see in one glimpse your SQL statement.

String Parameters				
	Name	Starting string	Varying string	Ending string
<input type="checkbox"/>	Country	Canada		
<input checked="" type="checkbox"/>	txtSearch	SMITH'AND	SYS_CONTEXT('USERENV','IP_ADDRESS')	FROM DUAL)--
<input type="checkbox"/>	Language	en-us		

txtSearch=SMITH'AND 100 < (select ASCII(SUBSTRING(CAST(SYS_CONTEXT('USERENV','IP_ADDRESS') AS CHAR(4000)),1,1)) FROM DUAL)--

GO TO [STEP 16.12](#)

16.12 YOU WANT THE VALUE OF A WORD?

By word it means that it's the value in the Varying String you want to get. If it's a user then it will be one word, if it is a stored procedure then it will be as many words it has, and so on.

You will need to select the type Word if it's not already selected.

Type

Word Length Count

NOTE 1

Word is selected by default.

As an example in the next figure, you want to get the database user used by the connection string.

String Parameters				
	Name	Starting string	Varying string	Ending string
<input type="checkbox"/>	login	s		
<input checked="" type="checkbox"/>	pass	s') or	user)--

You will get the result displayed like the next figure:

Results

Current Char **W** Length **6** Word **WEBACS** Time taken **9 s 218 ms**

Note: the Word part has been cut to make it possible to fit in the Tutorial

In the last figure, the Current Char as the name implies is the current character being sought, the Length is the length of the Varying String value, Word is the actual value of

the Varying String, in this case the user of a the database, and finally, Time taken is the time that actually took since it started its search for the word.

NOTE 2

In order to find the value of the Varying String the application needs to find out the length first. It will use it to go fetch character by character until the size is found.

Besides it's always good to know in advance to have a general idea of how long it will take.

If you want to see the value of a word then your answer is **YES – GO TO STEP 16.16**.

If you don't want to see the value of a word then your answer is **NO – GO TO STEP 16.13**.

IMPORTANT NOTE

You absolutely need to have only one value count if not you will end up to have an error message. To see if you have more than one value, switch the type to Count option and look if you get a count of 1.



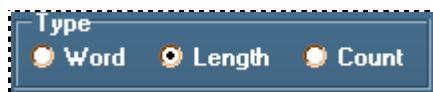
If you don't get a count of 1 try to use **WHERE** clause to make it go down to 1.

- IF YOUR ANSWER IS **NO** GO TO **STEP 16.13**
- IF YOUR ANSWER IS **YES** GO TO **STEP 16.16**

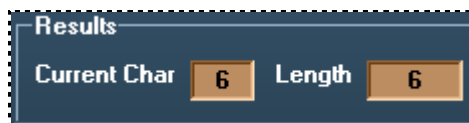
16.13 YOU WANT THE LENGTH OF THE VALUE?

This time it means that it's length of the value in the Varying String you want to get.

You will need to select the type Length if it's not already selected.



This time you will get the result displayed like the next figure:



In the last figure, the Current Char as the name implies is the current character being sought and the Length is the length of the Varying String value.

If you want to see the length of the Varying String then your answer is **YES – GO TO STEP 16.17.**

If you don't want to see the length of the Varying String then your answer is **NO – GO TO STEP 16.14.**

- **IF YOUR ANSWER IS NO GO TO STEP 16.14**
- **IF YOUR ANSWER IS YES GO TO STEP 16.17**

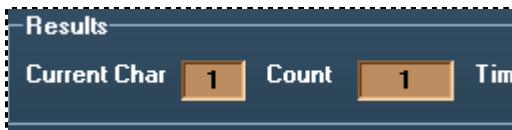
16.14 YOU WANT THE NUMBER OF RESULTS WITH THIS INJECTION?

Now it means it's the number of occurrence that the Varying String with the conditions if any (**WHERE CLAUSE**) will have.

You will need to select the type Count if it's not already selected.



This time you will get the result displayed like the next figure:



In the last figure, the Current Char as the name implies is the current character being sought and the Count is the number of results you've got with the injection.

If you want to see the number of results you've got with the injection then your answer is **YES – GO TO STEP 16.15.**

NOTE

In this case there is no **“NO”** answer because you should be there after answering no to Word and to Length leaving you with no other possible option.

- **IF YOUR ANSWER IS YES GO TO STEP 16.15**

16.15 CHOOSE THE NUMBER OF RESULTS CLOSEST TO WHAT YOU EXPECT AND CLICK START

In this step you need to find the number of results closest to what you expect to have. Why? You can always leave it by default but you will get an error message if it is over it,

or if you expect a value of 2 it will make more call to the web site by default than if you set the starting Count value to 5 for example.

It's a really nice way to fine-tune the application and it can on the long run save you a lot of time.

NOTE 1

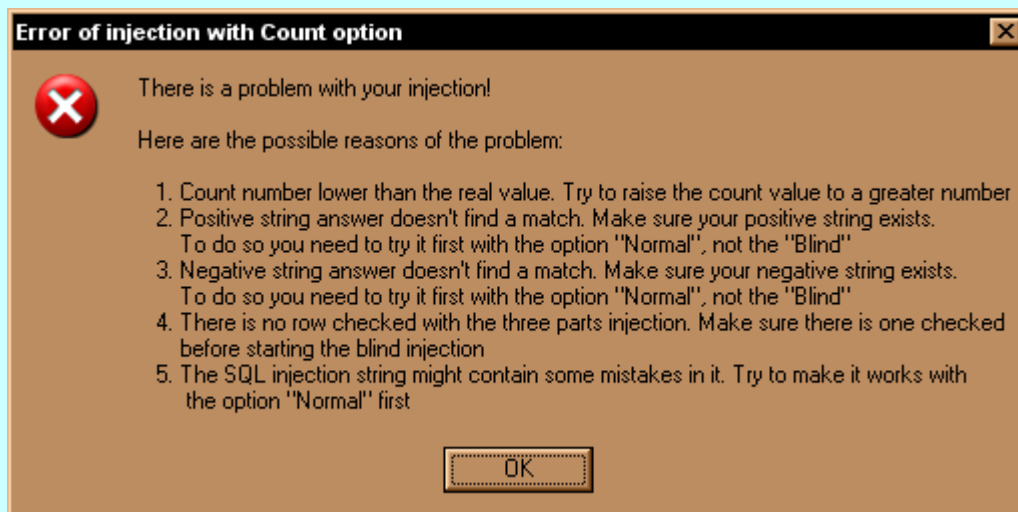
The value by default is 10.

NOTE 2

You should refer to the question "*Why can I change the number of the starting length and starting count?*" in the FAQ section of the web site to have a nice discussion about the choice of the Length and Count.

IMPORTANT NOTE

You will get a popup error message if your Count is set to a lower number than it is. If it happens you just need to raise the Count number to a level where you won't get popup error message anymore. The popup error message should look like this:



You can change the value in this textbox:



Once you are done, click Start button to launch the process.



GO TO [STEP 16.20](#)

16.16 CHOOSE THE NUMBER OF THREADS EXECUTED

Here you will need to decide the number of threads you want to use to find the value of the Varying String. It is one of the powerful features of the application since you can cut literally in half the time taken to get a word. In some cases it can save you half an hour to several hours on really long string (See the Statistics on the web Site).

NOTE

The value by default is 1.

You can change the value in this textbox:

A screenshot of a software interface showing a label 'Number of Threads' followed by a text input field containing the number '1' and a small spinner control.

HINT

If you know in advance the length of the value you should try to get the highest divisible value for the number of threads. In average all threads will finish at the same time so if they have the same number of characters to find it will all end at the same time. What happens sometimes it's that the number is not equal for each thread and one will have to finish after, rising thus the time taken.

Of course, the optimization is more complex than that, many things can affect the numbers, such as the CPU of the computer doing the injections, the capacity of the attacked web server to answer many requests at the same time to name a few.

GO TO [STEP 16.17](#)

16.17 CHOOSE THE LENGTH THAT YOU EXPECT THE VALUE TO BE AND CLICK START

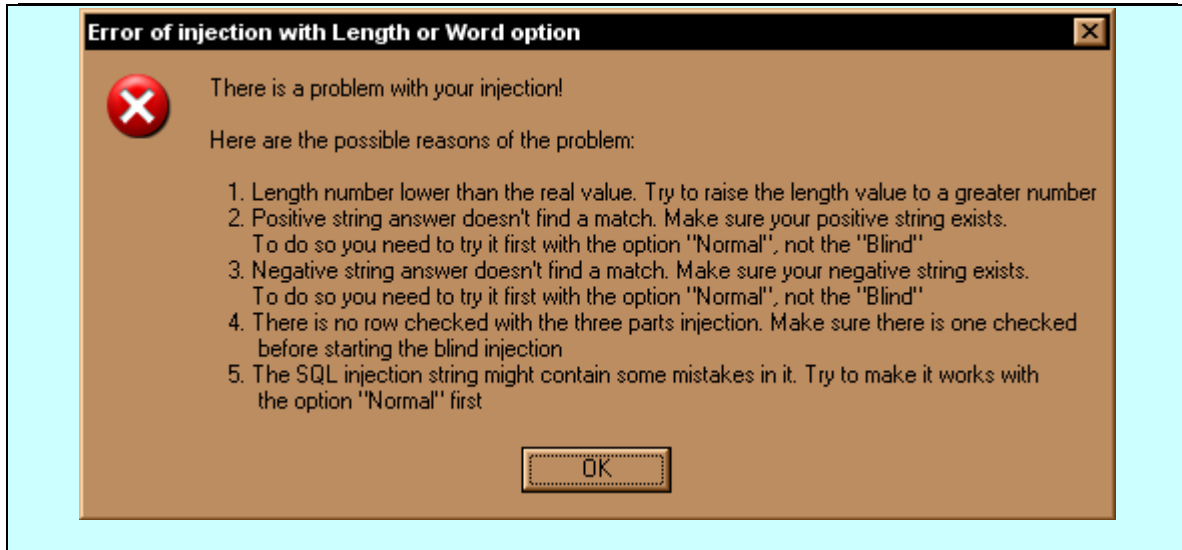
This time you will need to decide what will be the Length of the Varying String and set it up. Again it's a question of optimization and the reasons are the same than the **STEP 16.15** so please refer to that step to have more details.

NOTE 1

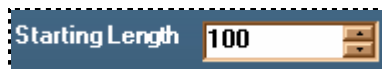
The value by default is 100.

IMPORTANT NOTE

You will get a popup error message if your Length is set to a lower number than the length of the value in Varying String is. If it happens you just need to raise the Length number to a level where you won't get popup error message anymore. The popup error message should look like this:



You can change the value in this textbox:



NOTE 2

You should refer to the question *“Why can I change the number of the starting length and starting count?”* in the FAQ section of the web site to have a nice discussion about the choice of the Length and Count.

GO TO [STEP 16.18](#)

16.18 READ THE RESULT

You are done! At that moment you should have the value you requested (Word, Length or Count)

CONGRATULATION YOU FINISHED THE BLIND SQL INJECTION TUTORIAL!!

17. CAN I USE UNION?

The UNION command is one of the most powerful commands you can inject when you only want to read data in the database. The sarcastic thing is that you will use the normal display feature of the web application to display your data. Normally it's going to be displayed in a table but be alert because many times I found it in the source code or even in the path of the **SRC** of an image!

If you can inject a UNION command then your answer is **YES – GO TO [STEP 19](#)**.

If you are not able to inject a UNION command then your answer is **NO** – GO TO **STEP 18**.

HINT

If when using the **UNION** you get a lot of undesirable data from the normal result you can easily get rid of them in inserting before the injection (most of the times a quote) weird data, like zxxzxxz. It needs of course to correspond to the data type of the **WHERE** clause.

This way, the normal **SELECT** part won't find anything and you will only get your data.

- IF YOUR ANSWER IS **NO** GO TO **STEP 18**
- IF YOUR ANSWER IS **YES** GO TO **STEP 19**

18. USE TECHNIQUE THAT WILL DISPLAY ERRORS THEN CLICK START BUTTON

There are many ways to get information from an error page and will most definitely not tell all of them, if such thing is possible at first place. They are just too many and I'm sure that I don't know half of them.

But I will try to give the basic ones.

- Or 1 in (select
- Having 1=1
- Putting a) or (
- Putting a ; or --
- Putting ' again somewhere else,
- Union select 1
- Etc

When you are done click Start button

A rectangular button with a dark grey background and the word 'Start' in white text.

GO TO **STEP 20**

19. MODIFY PARAMETERS FOR UNION VALUES THEN CLICK START BUTTON

We can assume that at this time you have already tested it to make it works so you have the real syntax, if not try to find the real syntax.

HINT

The main problem in **UNION** injection is to find the right number of columns that must correspond to the number of columns in the left **SELECT** (the one you use the **UNION** with).

A nice trick is to add a null value until you stop to get error, once done replace the first one with your value, let's say **@@servername**, and if you don't get any error the type is alright. If you do get an error replace it with null and try the next column and so on until it works.

Once your parameters are all set, click Start button

A rectangular button with a dark grey background and the word 'Start' in white text.**GO TO [STEP 20](#)****20. READ THE RESULTS**

You are done! At that moment you should have got the value you requested.

CONGRATULATION YOU FINISHED THE TUTORIAL!!