Foundstone®
Professional Services
A DIVISION OF McAFEE

Foundstone Hacme Books v2.0™ Strategic Secure Software Training Application

User and Solution Guide

Author: Roman Hustad, Foundstone Professional Services

May 30, 2006

Introduction

Foundstone Hacme Books™ is a learning platform for secure software development and is targeted at software developers, application penetration testers, software architects, and anyone with an interest in application security. As a full-featured J2EE application, Hacme Books is representative of real-world J2EE scenarios and demonstrates the security problems that can potentially arise in these applications.

In order to both raise general awareness and still challenge the more advanced developer, Hacme Books was built with several layers of vulnerabilities, from simple implementation issues to more complex design flaws. Hacme Books represents a test-driven, pattern-oriented design and implementation approach which demonstrates best practices and facilitates reuse and extensibility. Hacme Books is not a vanilla JSP and Servlet application of the type that are often used as platforms for demonstrating security techniques; Hacme Books has a multi-tiered MVC architecture and leverages many popular open-source Java projects: Struts, Spring, XDoclet, and Axis.

Application security often focuses on vulnerability classifications and checklists as sources for verification. This is not a wholly incorrect approach, but it can be misused by auditors and testers who simply review an application for a known set of issues. An alternate approach is to filter all inbound HTTP traffic through some sort of proxy that analyzes basic aspects of the protocol stream. This approach may seem cost-effective, but it has a severe penalty in accuracy due to a lack of context. Examples from the real-world show us that humans can identify what security issues are *relevant to their business* more effectively than an external component. Hacme Books offers a different approach. During the lab sessions of Foundstone's Writing Secure Code - Java course, Hacme Books is examined holistically and transformed by students into a secure application by leveraging the strength of architectural patterns.

Hacme Books is offered with the full source code under the Apache Software Foundation License Version 2.0. Being fully open-source allows Hacme Books to be subject to an unbiased peer review process, which will be used to constantly improve the quality and accuracy of the application. One important thing to note is that although Hacme Books uses best practices from a software engineering perspective, it intentionally includes mistakes that are representative of the issues that Foundstone sees daily through our consulting engagements in order to serve its purpose as a tool for learning.

Those who are particularly security conscious may notice that Hacme Books initiates some outbound traffic to the Internet. An examination of the network traffic will show that there is no attempt to violate your privacy. This traffic simply accesses a copy of the latest Foundstone class schedule, and also attempts to determine if you have Hacme Bank™ running on your local machine by opening a web services connection. Hacme Bank™ is a Foundstone Free Tool which simulates an insecure online banking application and is based on Microsoft's .NET Framework. Hacme Books uses Hacme Bank as a backend for payment processing.

Let's get started!

Overview of Hacme Books Distribution

Hacme Books is a pure Java web application that is distributed in 2 different formats:

- Windows Binary Installer
  - Complete installation is covered in this document
  - Download from http://www.foundstone.com/s3i
- Full Source Code
  - Not covered in this document
  - Requires installation of several development tools and intermediate Java skills
  - Download via anonymous CVS from http://sourceforge.net/projects/foundstone
  - See the HOWTO document in the HacmeBooks CVS folder at sourceforge for detailed instructions

Prerequisites

- The Windows Binary Installer only supports Microsoft Windows 2000/XP. Users of other operating systems can install Hacme Books by building the source code.

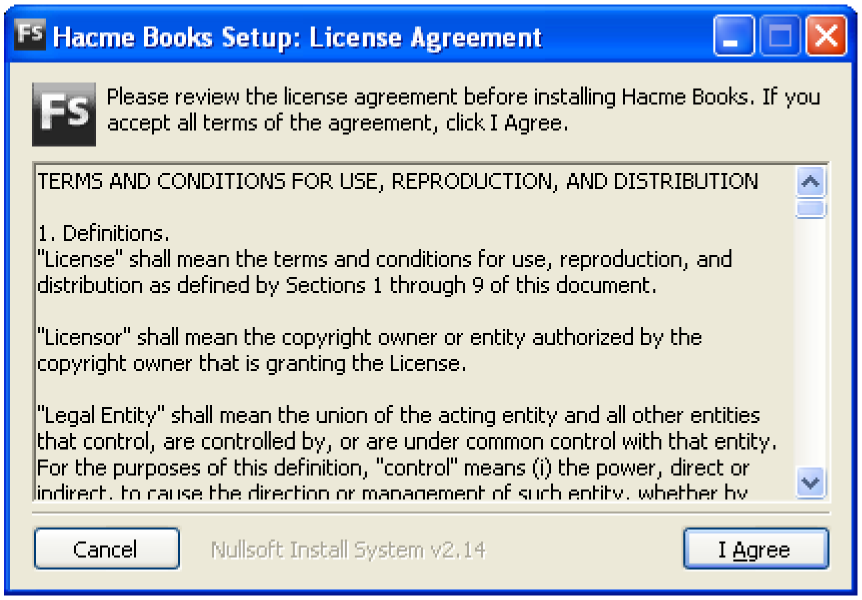
- Java Development Kit (JDK) 1.4.x or higher must be installed prior to the installation of Hacme Books. It will not run on 1.3.x. Please note that the full JDK is required; simply having a recent Java Runtime Engine (JRE) installed on your machine is not sufficient. To download the JDK, please visit http://java.sun.com/j2se/downloads. Restart after installation.

- The JAVA_HOME environment variable must point to the JDK installation directory. On Windows XP, you can set this variable manually:
  - Start → Settings → Control Panel → System → Advanced → Environment Variables
  - A typical value for JAVA_HOME would look something like this:
    C:\Program Files\Java\jdk1.5.0_06

- Apache's Tomcat application server is distributed with Hacme Books. It is possible that another instance of Tomcat is installed on your machine. If this is the case, refer to the Tomcat documentation for instructions on running multiple instances of Tomcat. Hacme Books is installed on port 8989 to avoid conflict with other common web server-based tools.

Windows Binary Installation

- Download the Hacme Books executable installer from http://www.foundstone.com/s3i.
  - Double-click on HacmeBooksSetup.exe.
  - **Figure 1** displays the license text, which is based on the Apache 2.0 License. You must agree to the terms of the license by clicking the ***I Agree*** button in order to continue installation.
  - **Figure 2** displays a security warning from Foundstone regarding Hacme Books. Click ***Next*** if you understand the warning and choose to proceed.
  - **Figure 3** displays the desired installation folder. You can edit the directory path or leave the default setting. Click ***Install*** to begin installation of Hacme Books.
  - **Figure 4** is a non-interactive screen you will see as the installer copies files to the target directory. If you do not have a recent JDK properly installed, the Hacme Books installer will detect this condition and notify you.
  - **Figure 5** is the last dialog. Just click ***Close***.
- Once installation is complete, you must start the Hacme Books application server (Tomcat):
  Start → Programs → Foundstone Free Tools → Hacme Books Server START
- To view Hacme Books in Internet Explorer (should look like **Figure 6**):
  Start → Programs → Foundstone Free Tools → Hacme Books
- Uninstalling Hacme Books is very easy: there is an uninstall program available from the Start Menu. Hacme Books may also be uninstalled using the Add/Remove Programs Control Panel.


Hacme Books Setup: License Agreement
Cancel
Nullsoft Install System v2.14
I Agree

Please review the license agreement before installing Hacme Books. If you accept all terms of the agreement, click I Agree.

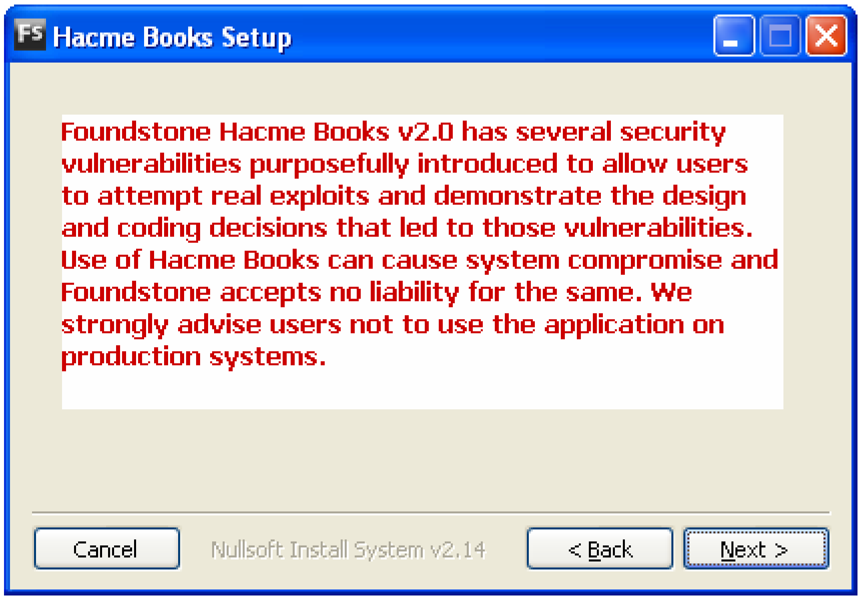
TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.
"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

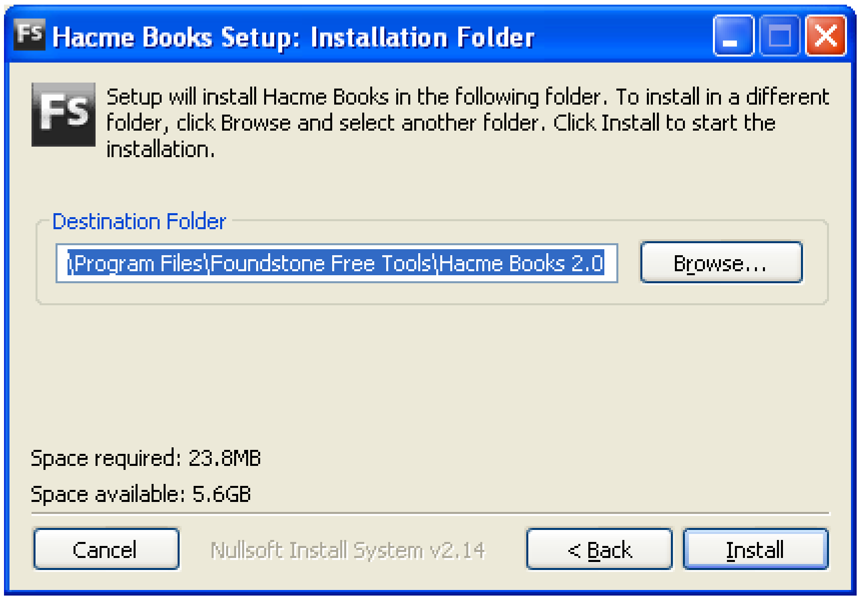
"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by

Figure 1



Foundstone Hacme Books v2.0 has several security vulnerabilities purposefully introduced to allow users to attempt real exploits and demonstrate the design and coding decisions that led to those vulnerabilities. Use of Hacme Books can cause system compromise and Foundstone accepts no liability for the same. We strongly advise users not to use the application on production systems.

Figure 2


Hacme Books Setup: Installation Folder

Setup will install Hacme Books in the following folder. To install in a different folder, click Browse and select another folder. Click Install to start the installation.

Destination Folder

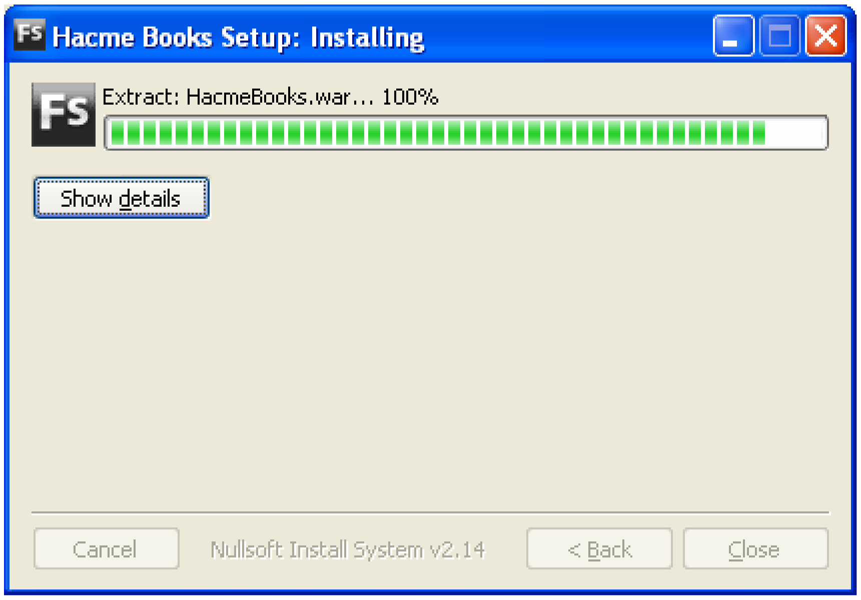
\Program Files\Foundstone Free Tools\Hacme Books 2.0

Space required: 23.8MB

Space available: 5.6GB

Nullsoft Install System v2.14

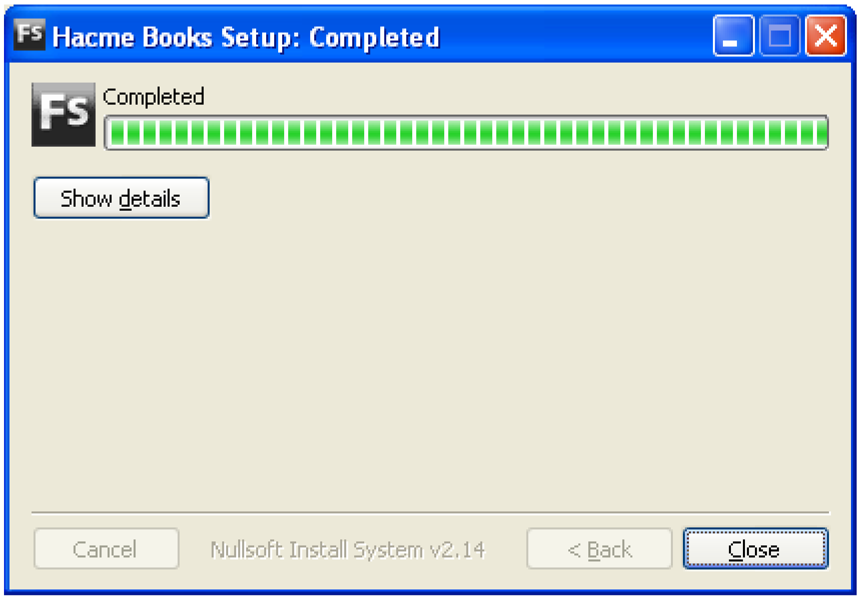
Figure 3


Hacme Books Setup: Installing

Extract: HacmeBooks.war... 100%

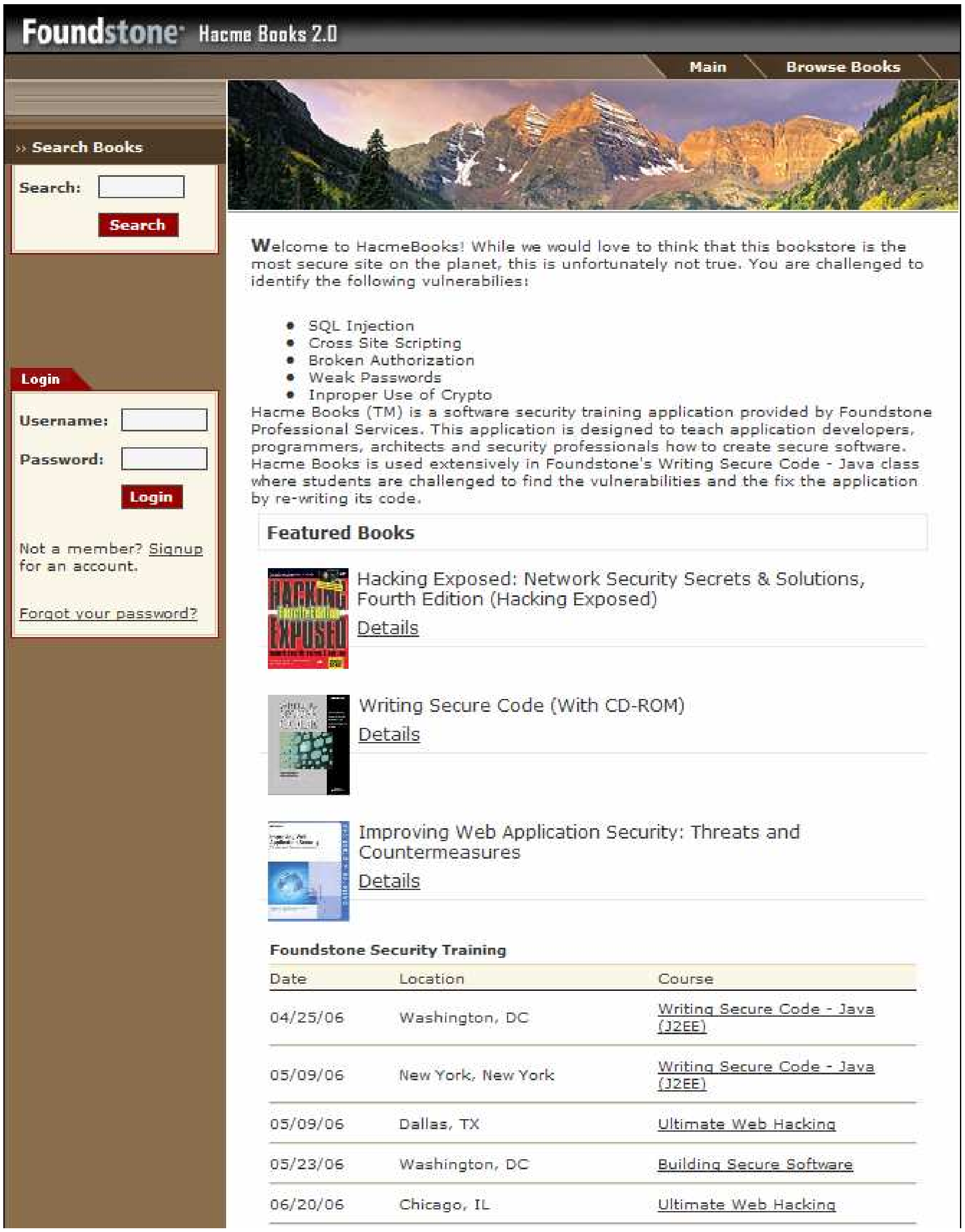
Nullsoft Install System v2.14

Figure 4



Hacme Books Setup: Completed
Completed
Show details
Cancel
Nullsoft Install System v2.14
< Back
Close

Figure 5



Not a member? Signup for an account.

Forgot your password?

Welcome to HacmeBooks! While we would love to think that this bookstore is the most secure site on the planet, this is unfortunately not true. You are challenged to identify the following vulnerabilies:

- SQL Injection
- Cross Site Scripting
- Broken Authorization
- Weak Passwords
- Inproper Use of Crypto

Hacme Books (TM) is a software security training application provided by Foundstone Professional Services. This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Books is used extensively in Foundstone's Writing Secure Code - Java class where students are challenged to find the vulnerabilities and the fix the application by re-writing its code.

Featured Books

Hacking Exposed: Network Security Secrets & Solutions, Fourth Edition (Hacking Exposed)
Details

Writing Secure Code (With CD-ROM)
Details

Improving Web Application Security: Threats and Countermeasures
Details

Foundstone Security Training

Date	Location	Course
04/25/06	Washington, DC	Writing Secure Code - Java (J2EE)
05/09/06	New York, New York	Writing Secure Code - Java (J2EE)
05/09/06	Dallas, TX	Ultimate Web Hacking
05/23/06	Washington, DC	Building Secure Software
06/20/06	Chicago, IL	Ultimate Web Hacking

Figure 6

WARNING! This application was designed as a training tool for Foundstone's Writing Secure Code – Java (J2EE) class. As such, we recommend that you attempt to identify the vulnerabilities that exist in this application before you read further.

Learning Guide

There are two fundamental approaches to web application security testing:

- Whitebox testing (AKA Code Review)
  - The tester has access to source code, configuration files, and the actual deployed application

- Blackbox testing (AKA Penetration Test or Pen-test)
  - The tester has access to the application's end-user interface only

Whitebox testing is always going to produce a more accurate result based on the fact that the source code is available. Testers are able to review data flows through the application from the presentation tier all the way through to the data access tier. Therefore, the results yielded from whitebox testing are going to be far more precise than the results gathered from blackbox testing.

For example, if there is a SQL injection vulnerability discovered in 50 different areas of a web application, a blackbox pen-tester will identify 50 vulnerabilities. However, there may be a single library that makes the database calls, which a whitebox tester can identify as one vulnerability. In addition, a whitebox review can reveal vulnerabilities in configuration and integration points. For instance, Hacme Books communicates with Hacme Bank (a similar application written in ASP.NET) to actually debit a book purchaser's bank account. A review of Hacme Books' configuration files may uncover the location of the Hacme Bank web services endpoint, which you might explore for additional vulnerabilities.

Foundstone suggests that anyone with a development background should perform a code review first, and then perform the blackbox penetration test that is described in the rest of this document. This will validate the earlier review. The source code for Hacme Books is available via anonymous CVS at sourceforge.net/projects/foundstone.

For non-developers, the blackbox test is most appropriate.

For this guide we will focus on the blackbox, or pen-test, approach. The code review methodology is a much more intensive activity that we tackle in Foundstone's *Writing Secure Code* classes. For more information about Foundstone's *Writing Secure Code* classes, go to www.foundstone.com/education.

Lesson 1 - Error Generation

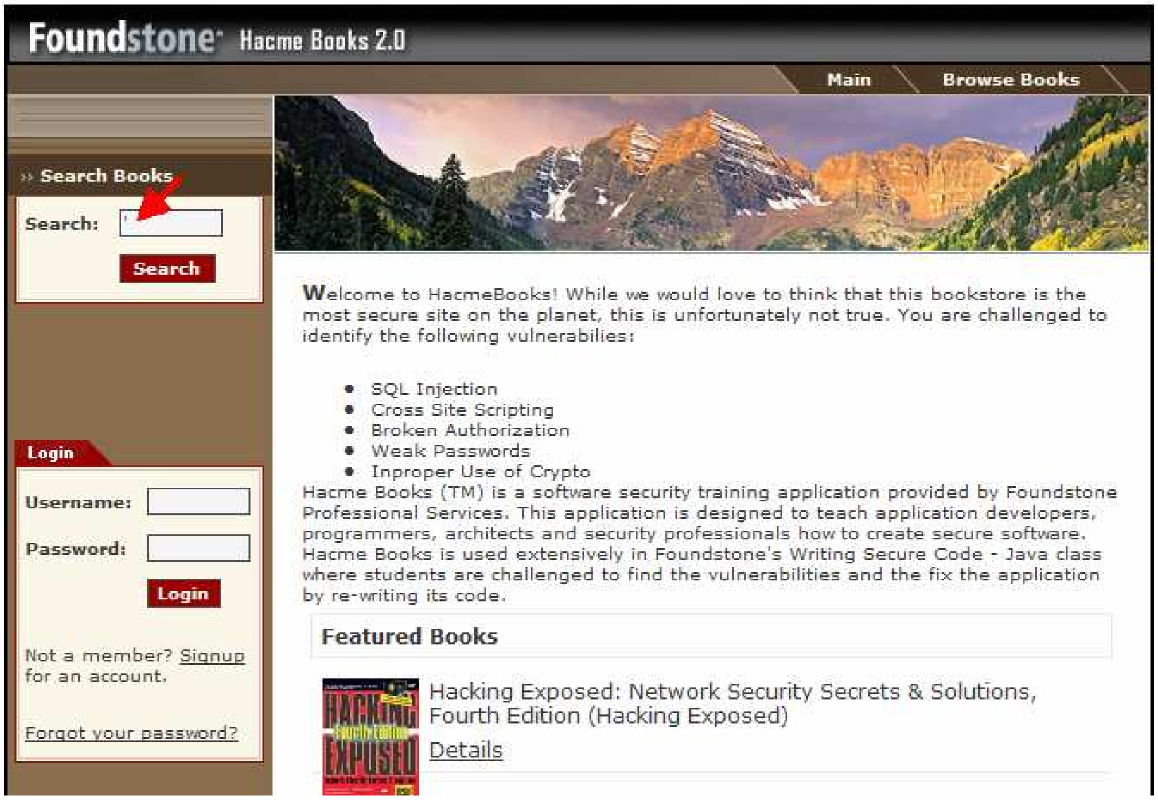
Lesson#		**1**
Vulnerability Exploited		Error Generation
Exploit Result		Detailed error message
Input Field(s)		Search
Steps	**1**	Enter a single apostrophe character in the Search field and press Enter.
Corresponding Figures(s)		Figure 7 Figure 8

When a malicious user is trying to gain access to a specific system, he is going to be able to launch the most effective attacks when he has the most information. Knowledge is power from an attacker's standpoint, since identifying and exploiting vulnerabilities requires understanding the system at some level.

One of the best things you can do as a developer is to limit the amount of system information that a user can gather from the application during fault conditions. Unfortunately, the developers of Hacme Books have decided that the most verbose error reporting is the best error reporting. Because of this, any faults that the attacker generates in the system are reported back to them with full Java stack traces.

This vulnerability can be demonstrated as follows:

1. From the home page. enter some malicious and/or inappropriate input into the search box. One good test to determine if an application is performing any input validation is to enter a single apostrophe into the search input field:



Welcome to HacmeBooks! While we would love to think that this bookstore is the most secure site on the planet, this is unfortunately not true. You are challenged to identify the following vulnerabilies:

- SQL Injection
- Cross Site Scripting
- Broken Authorization
- Weak Passwords
- Inproper Use of Crypto

Hacme Books (TM) is a software security training application provided by Foundstone Professional Services. This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Books is used extensively in Foundstone's Writing Secure Code - Java class where students are challenged to find the vulnerabilities and the fix the application by re-writing its code.

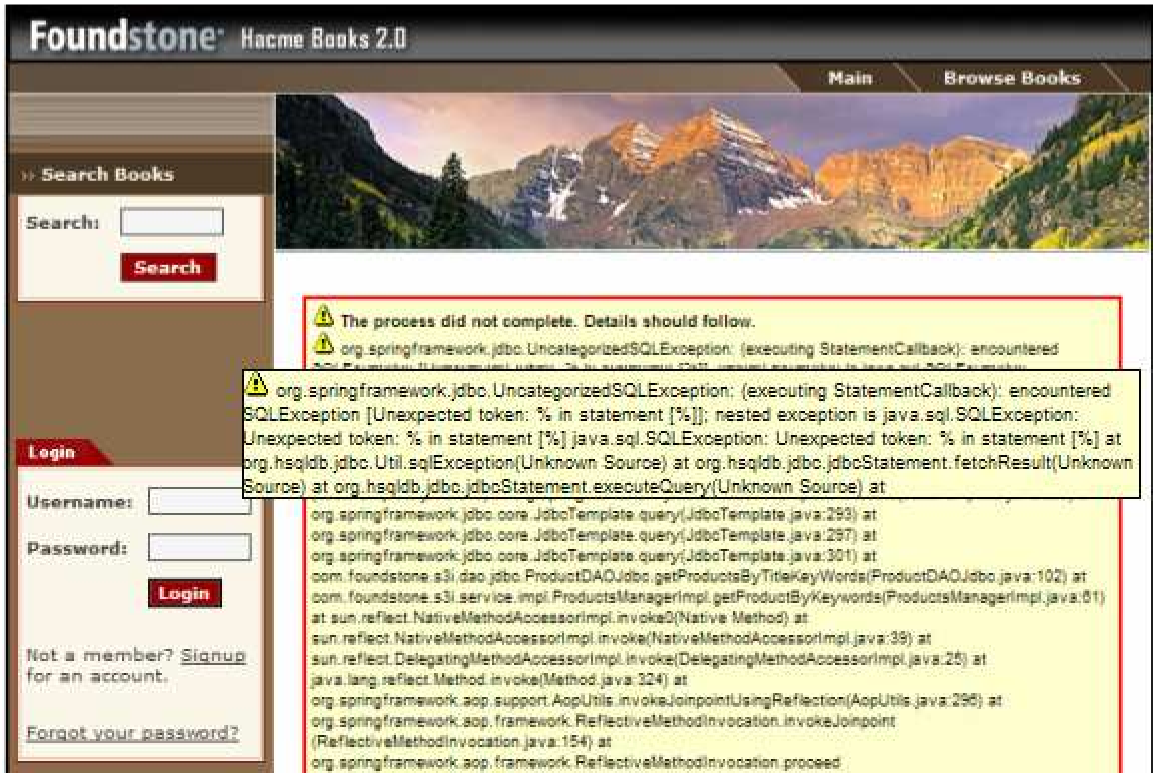
Featured Books

Hacking Exposed: Network Security Secrets & Solutions, Fourth Edition (Hacking Exposed)
Details

Not a member? Signup for an account.

Forgot your password?

Figure 7



```
The process did not complete. Details should follow.
org.springframework.jdbc.UncategorizedSQLException: (executing StatementCallback): encountered
SQLException [Unexpected token: % in statement [%]]; nested exception is java.sql.SQLException:
Unexpected token: % in statement [%] java.sql.SQLException: Unexpected token: % in statement [%] at
org.hsqldb.jdbc.Util.sqlException(Unknown Source) at org.hsqldb.jdbc.jdbcStatement.fetchResult(Unknown
Source) at org.hsqldb.jdbc.jdbcStatement.executeQuery(Unknown Source) at
org.springframework.jdbc.core.JdbcTemplate.query(JdbcTemplate.java:293) at
org.springframework.jdbc.core.JdbcTemplate.query(JdbcTemplate.java:297) at
org.springframework.jdbc.core.JdbcTemplate.query(JdbcTemplate.java:301) at
com.foundstone.s3i.dao.jdbc.ProductDAOJdbc.getProductsByTitleKeyWords(ProductDAOJdbc.java:102) at
com.foundstone.s3i.service.impl.ProductsManagerImpl.getProductByKeywords(ProductsManagerImpl.java:81)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method) at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39) at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25) at
java.lang.reflect.Method.invoke(Method.java:324) at
org.springframework.aop.support.AopUtils.invokeJoinpointUsingReflection(AopUtils.java:296) at
org.springframework.aop.framework.ReflectiveMethodInvocation.invokeJoinpoint
(ReflectiveMethodInvocation.java:154) at
org.springframework.aop.framework.ReflectiveMethodInvocation.proceed
```

Figure 8

2. After entering the apostrophe in the search box, you can see that there is quite a bit of information being returned. From this single fault condition, you now have the following information:

- Database Type – Hypersonic SQL (just Google org.hsqldb !)
- Application Server Type – Apache Tomcat
- Use of the Spring framework
- Other filters that are running in the Servlet stack
- The fact that this is a J2EE application – Identified by the Java namespaces

3. This stack trace contains enough information to begin launching your first attacks.

Lesson 2 - SQL Injection

General SQL Injection Overview

If a malicious user is trying to subvert your application and thinks you may be using a relational database, he or she may try to perform SQL Injection. SQL Injection is the process of placing special SQL characters in an input field, such that when that input is used as part of a database query it modifies the structure and effect of the query statement. In web applications, this user input typically arrives via HTTP request parameters.

A comprehensive overview of SQL injection can be found at the following URL: http://www.nextgenss.com/papers/advanced_sql_injection.pdf.

Based on the previous lesson, you now know the database server vendor and can analyze the various SQL constructs that are supported. This will provide you with the 'arsenal' necessary to successfully attack the system.

Lesson 2A (Denial of Service via SQL Injection):

Lesson#		**2A**
Vulnerability Exploited		SQL Injection
Exploit Result		Denial of Service (remote database shutdown)
Input Field(s)		Search
Input Data	**Step 1**	'; SHUTDOWN;--
	Step 2	';+SHUTDOWN;--
Corresponding Figures(s)		Figure 9 Figure 10 Figure 11

Security can be broken down into three attributes: Confidentiality, Integrity and Availability (CIA). A successful attack will disrupt one or more of these attributes. In this lesson, the attacker will impact the availability of the Hacme Books application. This attack pattern is referred to as Denial of Service (DoS). You will use SQL injection as the attack technique in this example.

If you review the HSQLDB documentation, you will find that there is a command to halt the database server called *SHUTDOWN* which would certainly cause DoS if executed successfully.

Most SQL injection vulnerabilities are a result of concatenating SQL statements with input that is collected directly from the user as shown in the code sample below:

```
String query = "select * from products where "
                        + createCriteria(inputKeywords);
```

This particular code snippet appends input into SQL criteria via a method that iterates over the tokenized user input. This allows a user to maliciously insert extra SQL statements. In this particular case the attacker will insert a SHUTDOWN command.

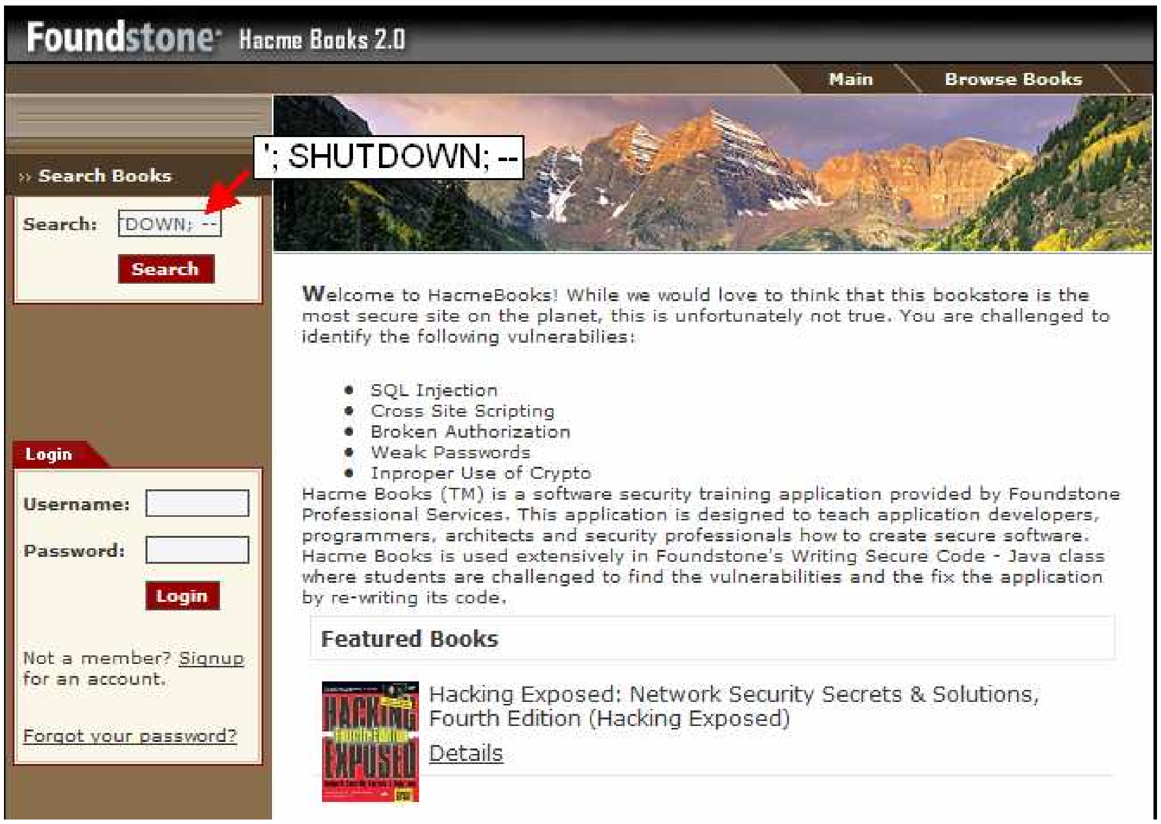
What the original developer intended is a SQL statement that looks like this (user input is highlighted in red):

```
select * from products where title like '%someuserkeyword%' and like
'%someotheruserinputtedkeyword%'
```

Once you understand how typical SQL statements are composed in vulnerable applications, you can plan a SQL Injection attack. The input in red is the data derived from user input. As an attacker, you know that an effective SHUTDOWN should look like this:

```
select * from products where title like '%'; SHUTDOWN; --%' and like
'%someotheruserinputtedkeyword%'
```

Because the double dash character sequence (--) indicates a comment in SQL, the rest of the SQL statement (in blue) is ignored by the SQL parser and if the attack is successful, the database will be shutdown. Here is a demo of the attack:



Welcome to HacmeBooks! While we would love to think that this bookstore is the most secure site on the planet, this is unfortunately not true. You are challenged to identify the following vulnerabilities:

- SQL Injection
- Cross Site Scripting
- Broken Authorization
- Weak Passwords
- Inproper Use of Crypto

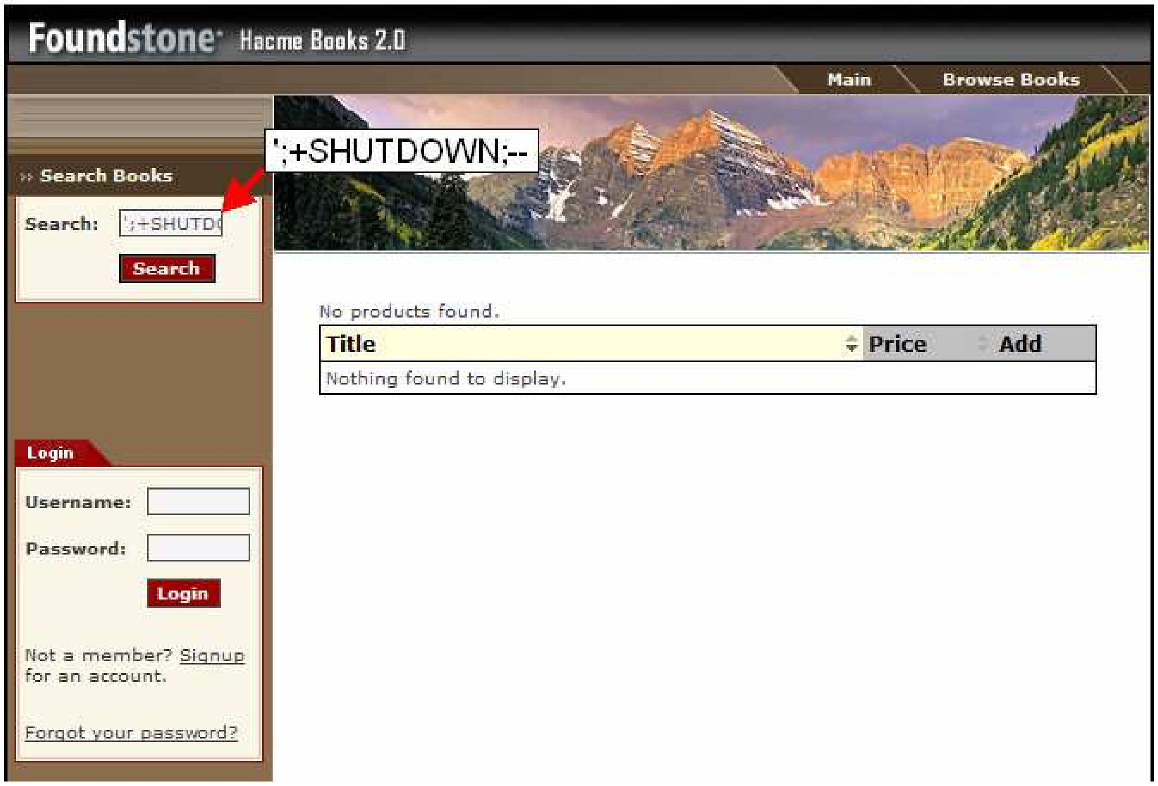
Hacme Books (TM) is a software security training application provided by Foundstone Professional Services. This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Books is used extensively in Foundstone's Writing Secure Code - Java class where students are challenged to find the vulnerabilities and the fix the application by re-writing its code.

Hacking Exposed: Network Security Secrets & Solutions, Fourth Edition (Hacking Exposed)
Details

Figure 9

However, this attack fails and does not cause a DoS. But a clever attacker can recognize and get around the problem.

Search engines generally tokenize input into separate pieces to process them as individual criteria. You may know that the '+' character typically forces a search engine to treat the input as one keyword, so just tweak the attack:

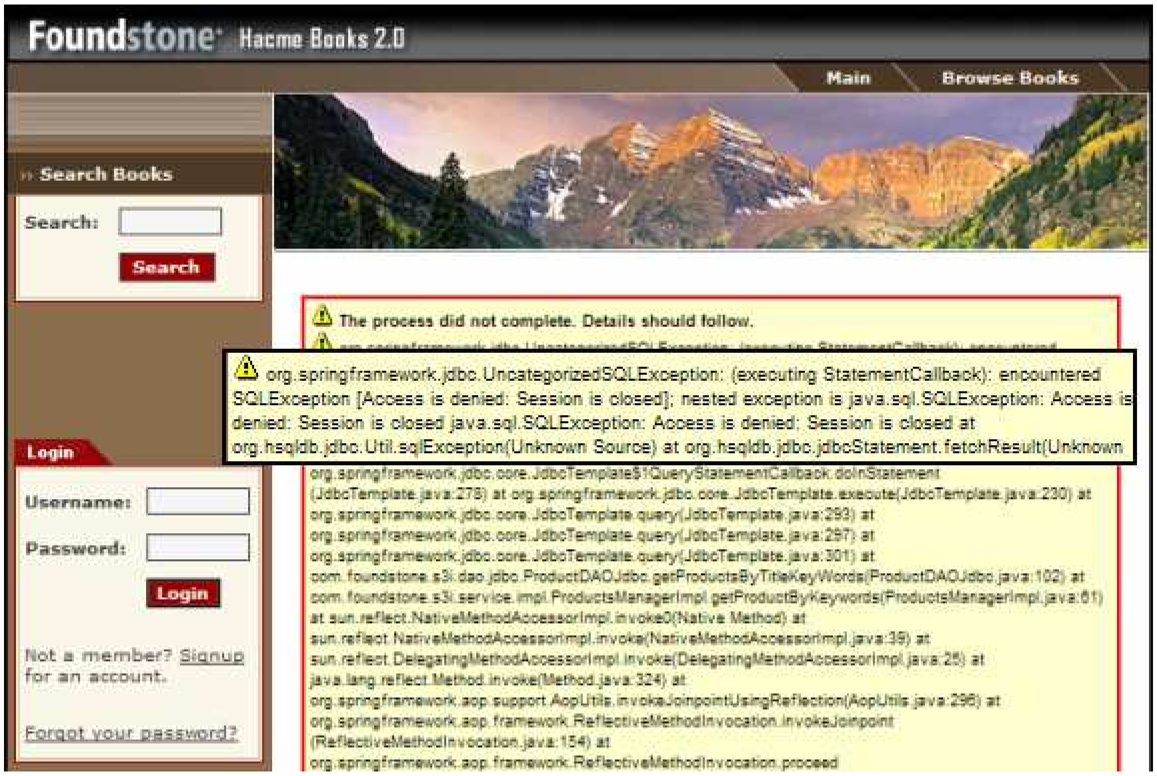


No products found.

Title	Price	Add
Nothing found to display.		

Figure 10

By utilizing the '+' functionality of the search engine, the shutdown attack succeeded as shown in Figure 11. This illustrates that SQL Injection, like many application attacks, does not always work on the first attempt but a persistent attacker can often find a way to exploit a hole even when it is difficult to do so.



```
⚠ The process did not complete. Details should follow.
⚠ org.springframework.jdbc.UncategorizedSQLException: (executing StatementCallback): encountered
SQLException [Access is denied: Session is closed]; nested exception is java.sql.SQLException: Access is
denied: Session is closed java.sql.SQLException: Access is denied: Session is closed at
org.hsqldb.jdbc.Util.sqlException(Unknown Source) at org.hsqldb.jdbc.jdbcStatement.fetchResult(Unknown
org.springframework.jdbc.core.JdbcTemplate$1QueryStatementCallback.doInStatement
(JdbcTemplate.java:278) at org.springframework.jdbc.core.JdbcTemplate.execute(JdbcTemplate.java:230) at
org.springframework.jdbc.core.JdbcTemplate.query(JdbcTemplate.java:293) at
org.springframework.jdbc.core.JdbcTemplate.query(JdbcTemplate.java:297) at
org.springframework.jdbc.core.JdbcTemplate.query(JdbcTemplate.java:301) at
com.foundstone.s3i.dao.jdbc.ProductDAOJdbc.getProductsByTitleKeyWords(ProductDAOJdbc.java:102) at
com.foundstone.s3i.service.impl.ProductsManagerImpl.getProductByKeywords(ProductsManagerImpl.java:61)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method) at
sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39) at
sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25) at
java.lang.reflect.Method.invoke(Method.java:324) at
org.springframework.aop.support.AopUtils.invokeJoinpointUsingReflection(AopUtils.java:296) at
org.springframework.aop.framework.ReflectiveMethodInvocation.invokeJoinpoint
(ReflectiveMethodInvocation.java:154) at
org.springframework.aop.framework.ReflectiveMethodInvocation.proceed
```

Figure 11

Lesson 2B (Data Tampering via SQL Injection):

Lesson#	**2B**
Vulnerability Exploited	SQL Injection
Exploit Result	Malicious product data inserted
Input Field(s)	Feedback
Input Data	*my feedback', 735); insert into products (title, description, popularity, price, vendor, category, publisher, isbn, author, imgurl, quantity) values ('Eat my shorts you pointy haired boss','A great book',4,29.95,'Amazon','Technical','Addison Wesley','1234567890123','Disgruntled Employee','http://',1); --*
Corresponding Figures(s)	Figure 12 Figure 13

The malicious user will now attack the 'I' in CIA; Integrity. Let's say the attacker is a bitter Hacme Books employee that wants to get back at his/her employer for unfair treatment.
In this lesson, the attacker wants to add a book to the company's database that will be embarrassing. They know that any user who has successfully logged into Hacme Books can

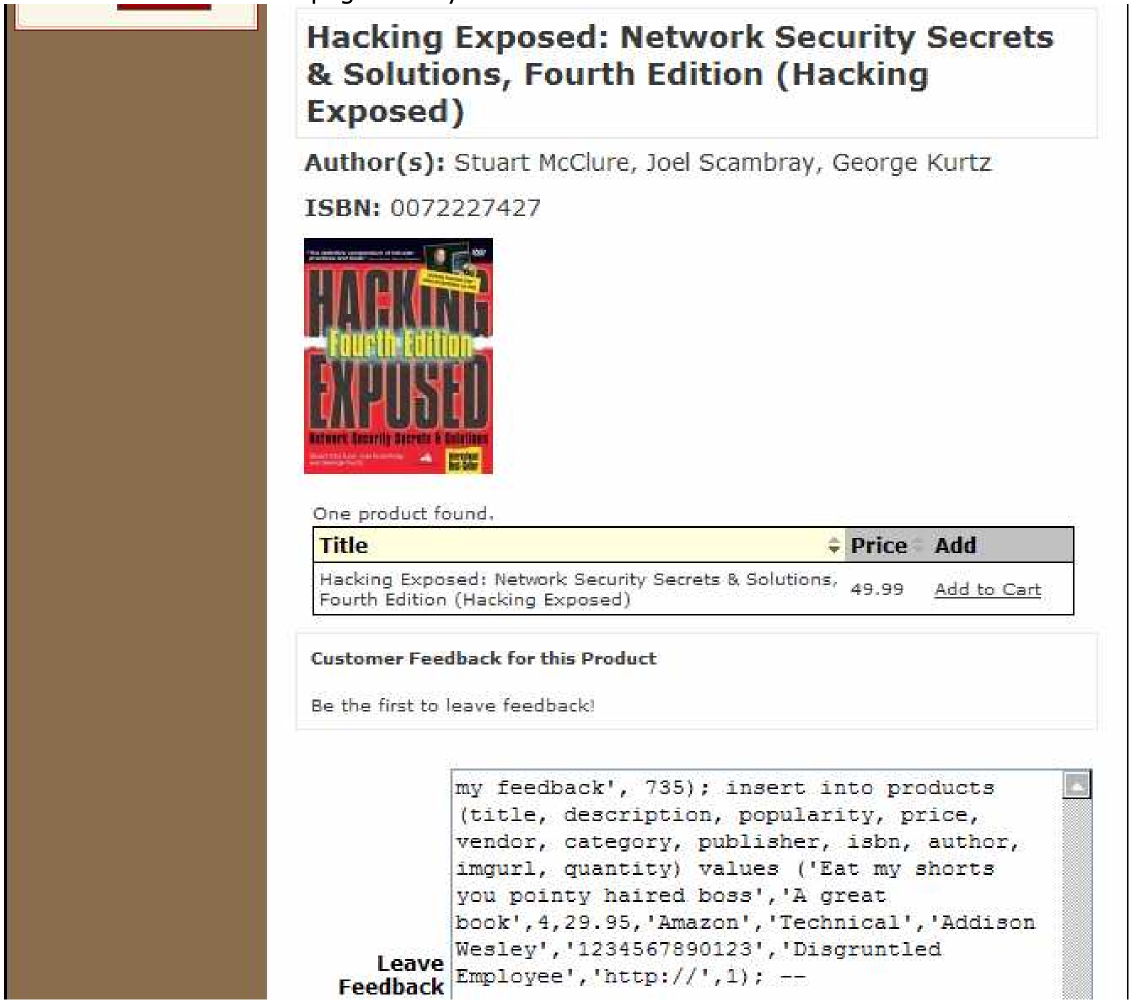
leave feedback for a book; any data in the feedback is allowed because there is no input validation.

There are ways to determine the schema of the target database using SQL Injection. These techniques are covered in the aforementioned SQL injection whitepaper, and are demonstrated in the whitepaper for another Foundstone Free Tool: Hacme Bank™ 2.0. For this scenario, it is assumed that the attacker has already learned the data schema and can therefore craft a valid SQL INSERT statement.

First you should stop and restart the Hacme Books server (i.e. Tomcat), which will recover the database from the shutdown attack.

Then you need to log in to the application. Either create a new account or try the username 'testuser' with the password 'password'. Among its many design flaws, Hacme Books does not require users to create passwords that are difficult to guess.

Then browse to the detail page for any book title and leave some "feedback":



Hacking Exposed: Network Security Secrets & Solutions, Fourth Edition (Hacking Exposed)

Author(s): Stuart McClure, Joel Scambray, George Kurtz

ISBN: 0072227427

One product found.

Title	Price	Add
Hacking Exposed: Network Security Secrets & Solutions, Fourth Edition (Hacking Exposed)	49.99	Add to Cart

Customer Feedback for this Product

Be the first to leave feedback!

Leave Feedback

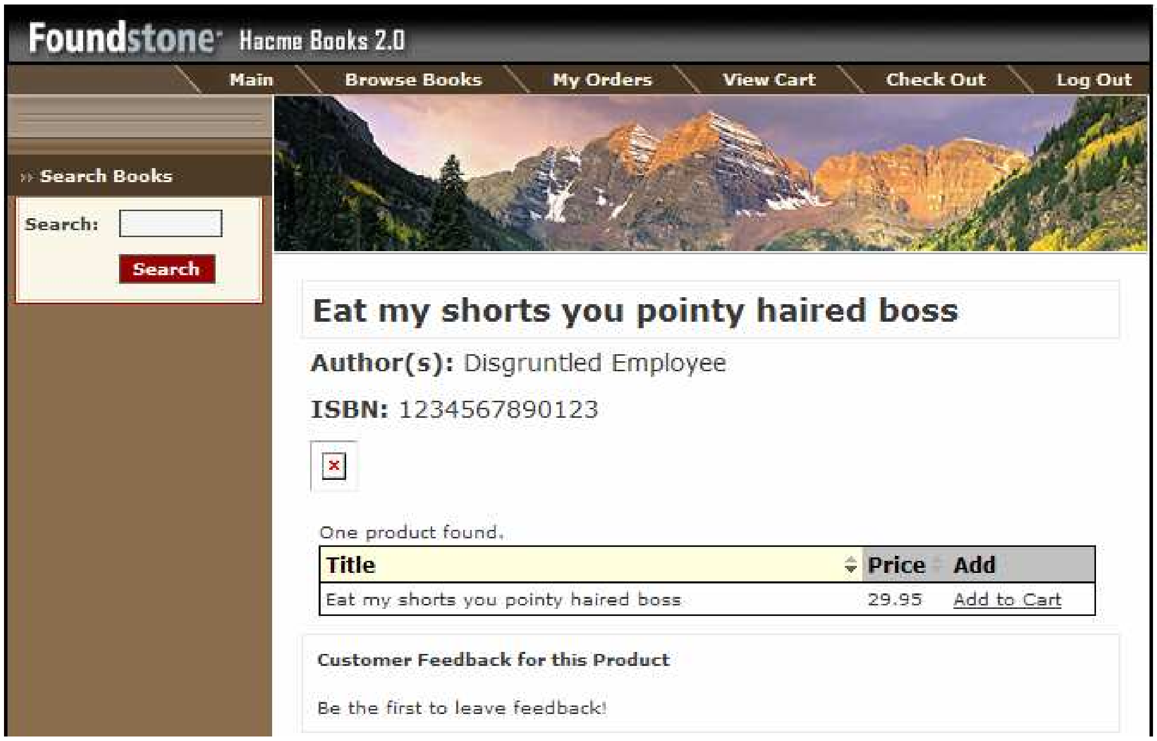
```
my feedback', 735); insert into products
(title, description, popularity, price,
vendor, category, publisher, isbn, author,
imgurl, quantity) values ('Eat my shorts
you pointy haired boss','A great
book',4,29.95,'Amazon','Technical','Addison
Wesley','1234567890123','Disgruntled
Employee','http://',1); --
```

Figure 12

Entering the following SQL Injection string will leave feedback, but will also create a new book title that would embarrass the company when viewed by the public. Note that the number 735 is specific to the "Hacking Exposed" book title; you have probably already deduced that it is the primary key of this item in the “products” table. If you are using a different book title, you can conveniently find the primary key of the book in the address bar of your browser.

```
my feedback', 735); insert into products (title, description,
popularity, price, vendor, category, publisher, isbn, author, imgurl,
quantity) values ('Eat my shorts you pointy haired boss','A great
book',4,29.95,'Amazon','Technical','Addison
Wesley','1234567890123','Disgruntled Employee','http://',1); --
```

To confirm the successful attack, perform a search on the newly created title. You should now find it in the database. Figure 13 shows the details page for the example attack:



Eat my shorts you pointy haired boss

Author(s): Disgruntled Employee

ISBN: 1234567890123

One product found.

Title	Price	Add
Eat my shorts you pointy haired boss	29.95	Add to Cart

Customer Feedback for this Product

Be the first to leave feedback!

Figure 13

Lesson 3A - XSS – Cross Site Scripting

Lesson#	**3A**
Vulnerability Exploited	Cross-site Scripting
Exploit Result	Product is forcibly added to cart
Input Field(s)	Feedback
Input Data	*You should buy our latest bestseller instead of this book.* *<script>* *location="http://localhost:8989/HacmeBooks/addShoppingCart.html?productId=1470"* *</script>*
Corresponding Figures(s)	Figure 13 Figure 14 Figure 15

Cross site scripting (XSS) is one of the most common application attacks seen on the public internet. It is often used for defacing sites, phishing, and stealing session identifiers. Although there are variations on this technique, it is usually performed by entering a JavaScript function into an input field whose value is redisplayed by the application.

Web applications that usually have this kind of input field include bulletin boards, auction sites, blogs with user comments, and e-commerce sites that allow product reviews. Hacme Books falls into the last category because it allows users to leave feedback on a particular book title, which is then displayed to other users of the application. This allows an attacker to actually specify a portion of the HTML and JavaScript that is sent to these other users, who we shall now refer to as "victims."

An easy way to determine whether a site is vulnerable to XSS is to attempt execution of the JavaScript "alert" function from an input field using this simple script:

```
<script>alert("XSS")</script>
```

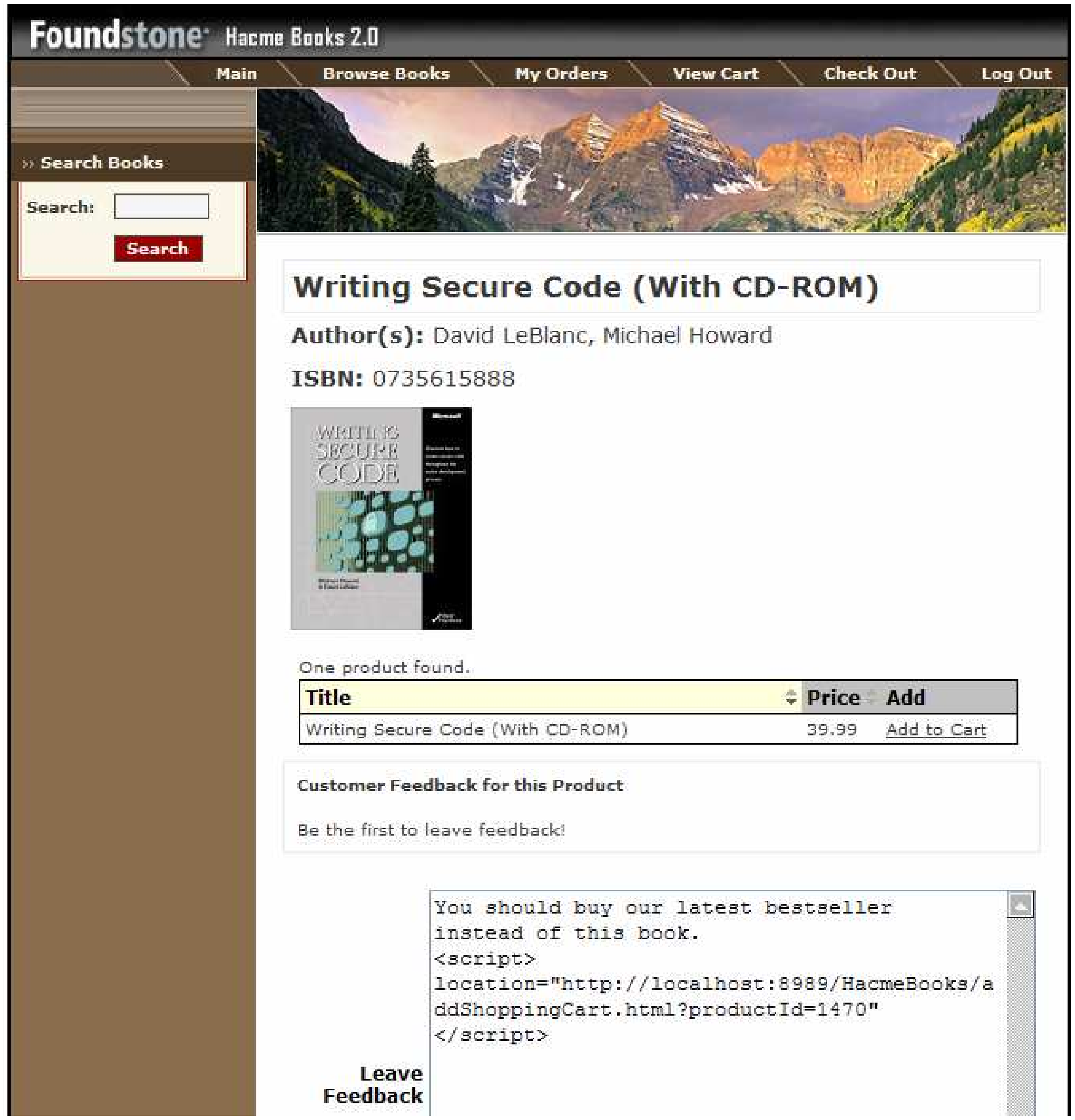
If an alert window pops up when you redisplay the page then you have found an XSS vulnerability, assuming that this input will also be displayed to other users (since it would not be very interesting to attack yourself). However, this alert window is just a test, so let's figure out a way to do something more malicious with JavaScript and XSS.

In the previous lesson, a bitter ex-employee of Hacme Books successfully created an embarrassing book using SQL Injection. However, they are upset because customers are not searching for the book. To solve this problem, the attacker adds a JavaScript redirect to the embarrassing book on the product details page for several other books. After some thought, the attacker decides it would be even better to redirect victims to the "Add to cart" link!

Navigate to the product details page for the embarrassing book (Figure 13), then right-click on the "Add to cart" link and select "Copy Link Location" (with Firefox; if using Internet Explorer select "Copy Shortcut" for the same effect). This is the link where you want to redirect victims.

Then navigate to the product details page for a different book and enter the following feedback (pasting the correct URL from your clipboard), as shown in Figure 14:

```
You should buy our latest bestseller instead of this book.
<script>
location="http://localhost:8989/HacmeBooks/addShoppingCart.html?productId=1470"
</script>
```



Writing Secure Code (With CD-ROM)

Author(s): David LeBlanc, Michael Howard

ISBN: 0735615888

One product found.

Title	Price	Add
Writing Secure Code (With CD-ROM)	39.99	Add to Cart

Customer Feedback for this Product

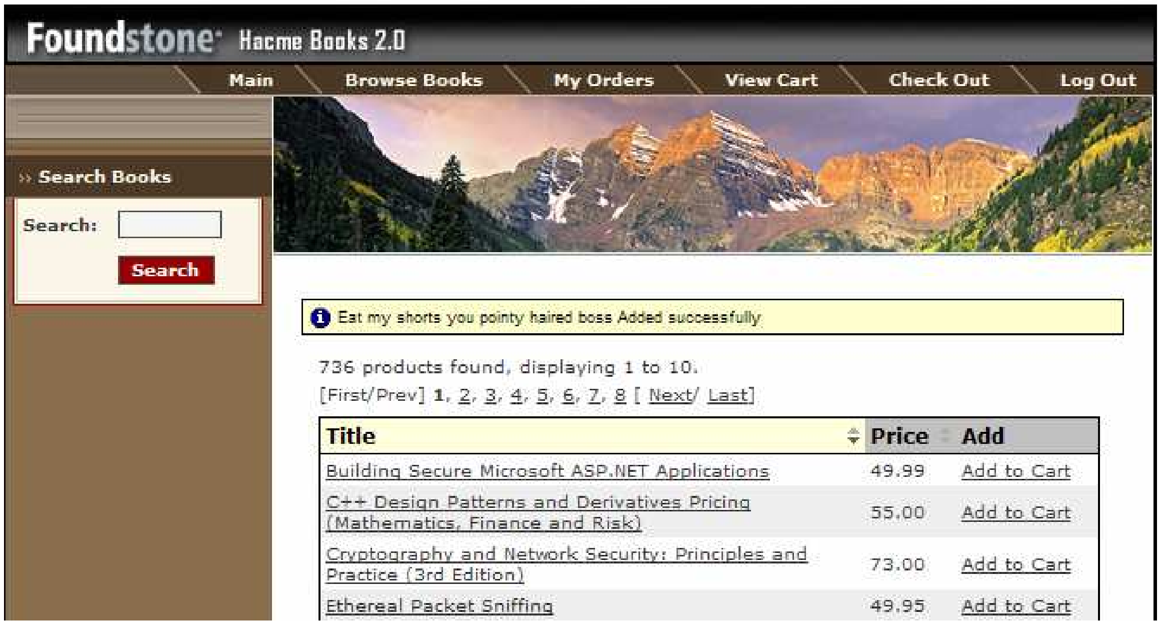
Be the first to leave feedback!

Leave Feedback

```
You should buy our latest bestseller
instead of this book.
<script>
location="http://localhost:8989/HacmeBooks/a
ddShoppingCart.html?productId=1470"
</script>
```

Figure 14

After you submit the "feedback" as shown in Figure 14, navigate back to the detail page for the same book title.



Eat my shorts you pointy haired boss Added successfully

736 products found, displaying 1 to 10.
[First/Prev] 1, 2, 3, 4, 5, 6, 7, 8 [Next/ Last]

Title	Price	Add
Building Secure Microsoft ASP.NET Applications	49.99	Add to Cart
C++ Design Patterns and Derivatives Pricing (Mathematics, Finance and Risk)	55.00	Add to Cart
Cryptography and Network Security: Principles and Practice (3rd Edition)	73.00	Add to Cart
Ethereal Packet Sniffing	49.95	Add to Cart

Figure 15

The attacker successfully forced the embarrassing book into the victim's cart when they visited the "Writing Secure Code" product details page.

While this example is relatively benign, an attacker could execute any arbitrary JavaScript in the victim's browser. This could include sending the victim's session identifier to the attacker's website, allowing a session hijack. The potential for abuse is limited only by the attacker's imagination and the functionality of JavaScript. Some forms of Cross-Site Scripting could be considered social engineering attacks because they require the victim to visit a vulnerable website, which is easy to achieve using techniques such as phishing.

Note that the various browsers handle the nuances of JavaScript somewhat differently so you may need to try several variations to get these types of attacks to work.

Lesson 3B (Cross-site Request Forgery):

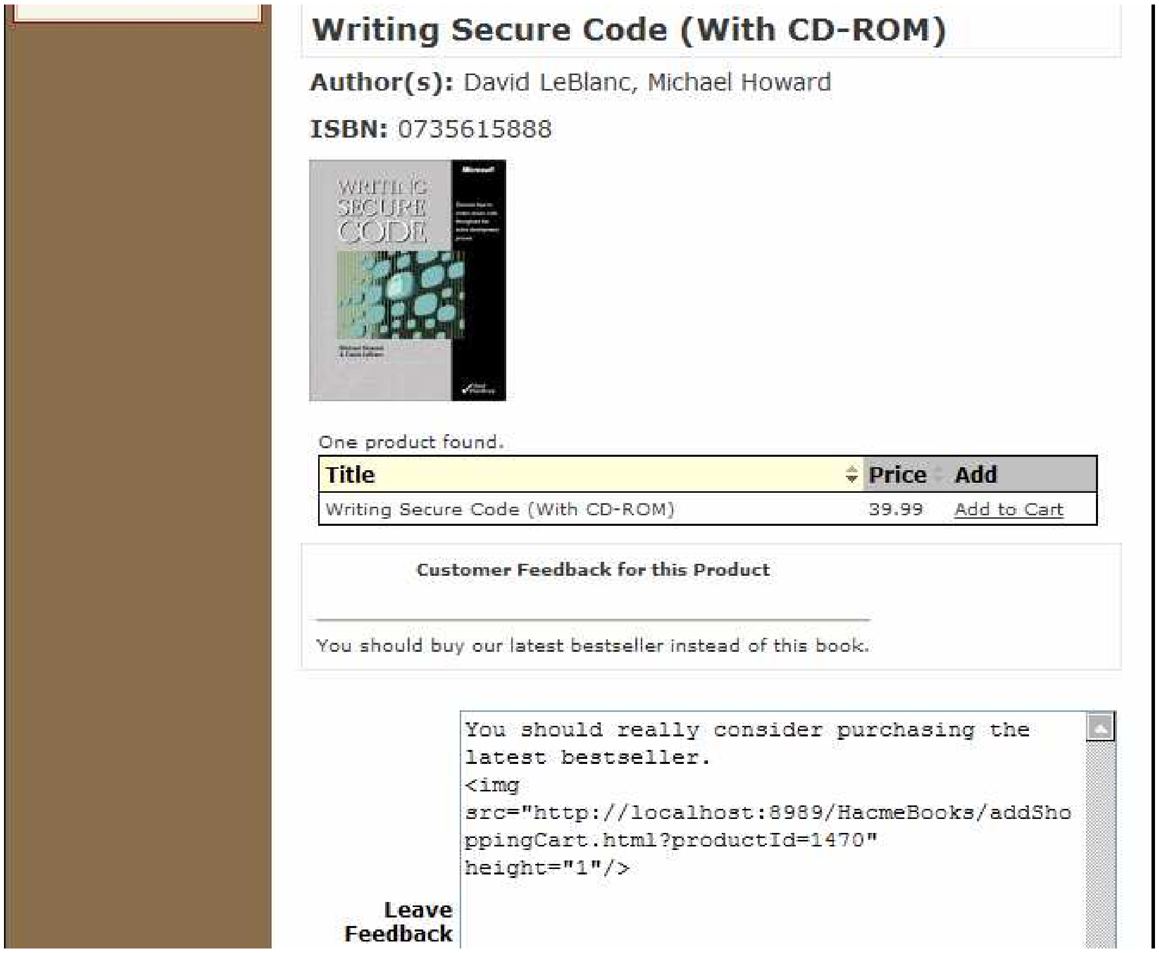
Lesson#	**3B**
Vulnerability Exploited	Cross-site Request Forgery
Exploit Result	Product is forcibly added to cart
Input Field(s)	Feedback
Input Data	*You should really consider purchasing the latest bestseller. <img src="http://localhost:8989/HacmeBooks/addShoppingCart.html?productId=1470" height="1"/>*
Corresponding Figures(s)	Figure 16

After some frustration, the clever victim may find an easy way to foil the XSS attack by disabling JavaScript in their browser. (Firefox: Tools > Options > Content > Enable JavaScript; Internet Explorer: Tools > Internet Options > Security > Internet > Custom Level > Scripting > Active Scripting > Disable). However, the attacker has another trick available with Cross-site Request Forgery (CSRF).

Like the previous XSS attack, CSRF can be used to cause the victim's browser to make a request to a URL of the attacker's choosing. However, CSRF does not rely on JavaScript. Instead, it leverages the HTTP GET request that the victim's browser sends when requesting an image with the <img> HTML tag. A CSRF attack does not request an image but rather a URL that will change state in the victim's session (such as adding an item to the cart). Changing server-side state with a GET request is technically a violation of the HTTP specification but many developers are unaware of this and do it anyway (HTTP POST should be used instead).

As a bonus, CSRF happens in the background without any opportunity for the user to notice the request (the same way that images load). Normally a broken image icon would appear if the browser does not retrieve an image, but the attacker can make the image height only 1 pixel, which should not be visible to the victim.

The CSRF attack uses the same URL as the XSS attack but instead embeds it as the src attribute of an HTML <img> tag. You will probably need to enter the <img> tag all on one line, although it will appear spread over several lines due to wrapping in the text area.



Writing Secure Code (With CD-ROM)

Author(s): David LeBlanc, Michael Howard

ISBN: 0735615888

One product found.

Title	Price	Add
Writing Secure Code (With CD-ROM)	39.99	Add to Cart

Customer Feedback for this Product

You should buy our latest bestseller instead of this book.

Leave Feedback

```
You should really consider purchasing the
latest bestseller.
<img
src="http://localhost:8989/HacmeBooks/addSho
ppingCart.html?productId=1470"
height="1"/>
```

Figure 16

As before with the XSS attack, when you view this page after the "feedback" is left you will have the embarrassing book automatically added to your cart. With CSRF there is no obvious indication to the user that this occurred until they view their cart. There are many other variations of CSRF possible, including the ability to use HTTP POST. This example attack did not even leverage the "cross-site" nature of CSRF, which means that the victim would unknowingly make a request to a website of the attacker's choosing. It is actually possible to attack intranet applications (and devices with web interfaces) *from the internet* by sending a CSRF attack in an HTML e-mail to a victim who is connected to both networks. More information about advanced CSRF attacks can easily be found by searching the Internet.

Lesson 4 - Crypto Wannabe

Lesson#	**4**
Vulnerability Exploited	Weak Cryptography
Exploit Result	Discount coupon code decrypted and modified
Input Field(s)	Coupon Code
Input Data	*IEODBOBOOG*
Corresponding Figures(s)	Figure 17 Figure 18

Often, developers believe that obscuring data protects it. The wonderful developers of Hacme Books believe in this practice, as well as in the importance of generosity. In fact, because they are so generous, they frequently hand out coupons and advertise promotions on the radio. In this lesson, the attacker heard that Hacme Books is offering a 15% discount on all books for a limited time only. The hacker has written down several different codes from different times.

- 15 %
    - AEODBOBOOF
- 25%
    - BEAAABBOOF
    - BEOABDBOOF

The attacker now stacks all of those codes on top of each other and sees:

```
AEODBOBOOF
BEAAABBOOF
BEOABDBOOF
```

Notice that there some common text with just these 3 different examples. The attacker knows that 2 of them share the 25% discount, but there are other common characteristics as well:

```
BEAAABBOOF
BEOABDBOOF
```

The attacker might deduce that the discount code is based on some sort of substitution algorithm. Since the last four characters of the coupons are the same, and there are four digits in a year…they can deduce that the last 4 characters of the coupon should be the year.

```
BEAAAB2006
BEOABD2006
```

It is now clear that the letter 'B' is the representation of the number '2', which also appears elsewhere in the coupon code:

```
2EAAAB2006
2EOABD2006
```

The attacker concludes that the developer simply decided to substitute numbers with letters.

```
1504202006
2511122006
2501242006
```

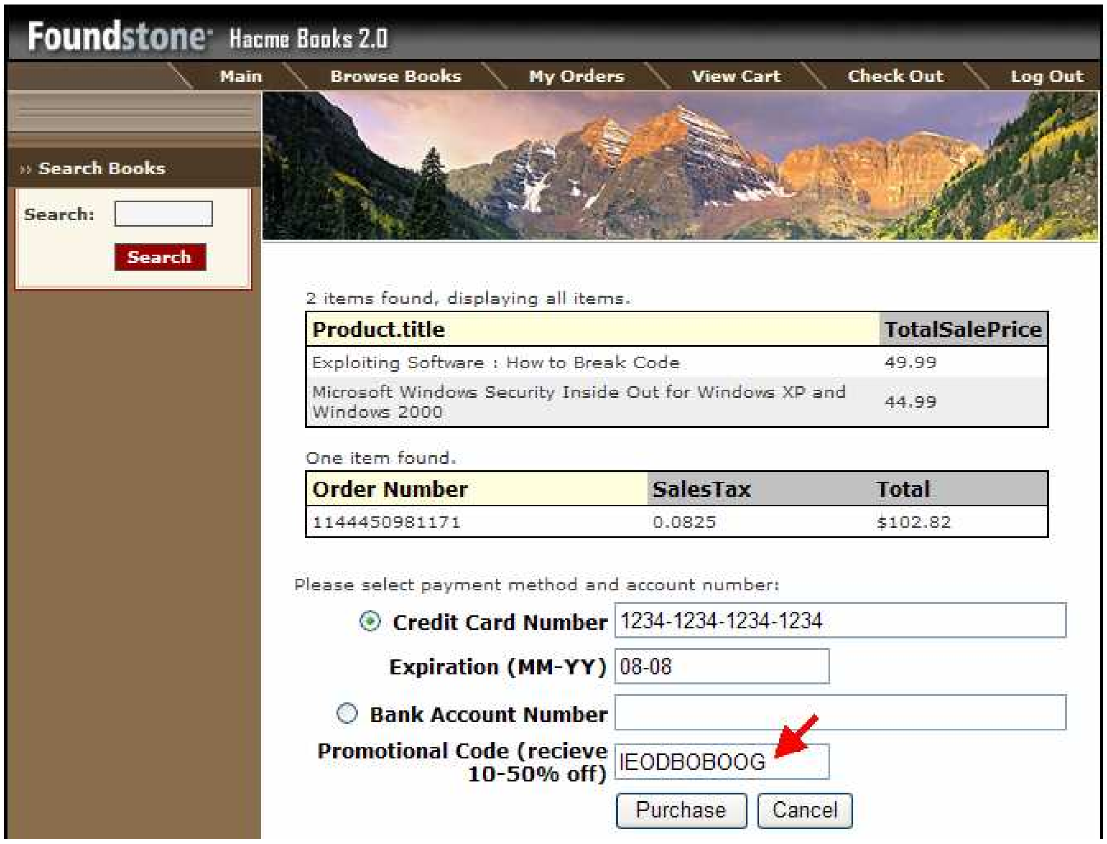
Further deduction reveals that this is a combination of discount percentage and expiration date!

Percentage	Month	Day	Year
15	04	20	2006
25	11	12	2006
25	01	24	2006

Now the attacker can test this theory. The attacker can create a new coupon code and try it with a purchase. How about a 95% discount for the next year?

95	04	20	2007
IE	OD	BO	BOOG

```
Test this coupon: IEODBOBOOG
```



2 items found, displaying all items.

Product.title	TotalSalePrice
Exploiting Software : How to Break Code	49.99
Microsoft Windows Security Inside Out for Windows XP and Windows 2000	44.99

One item found.

Order Number	SalesTax	Total
1144450981171	0.0825	$102.82

Please select payment method and account number:

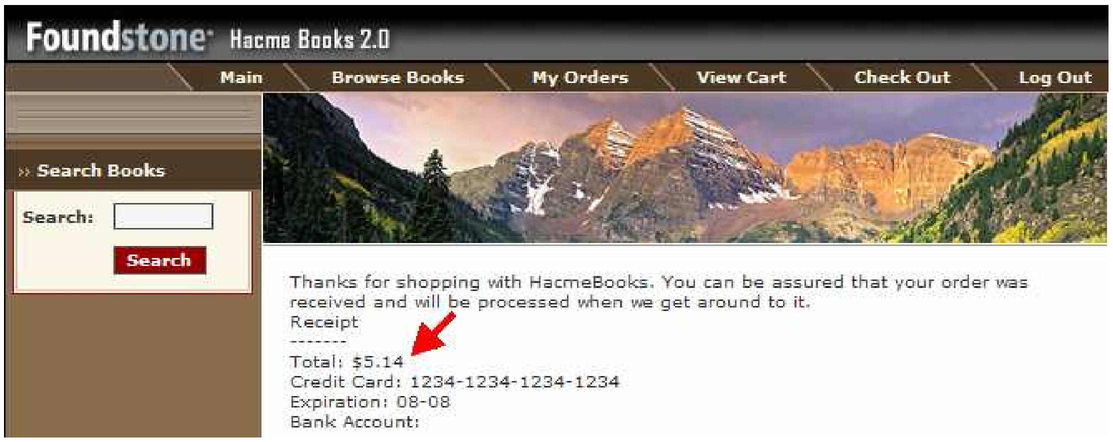
[✓ Credit Card Number] 1234-1234-1234-1234

Expiration (MM-YY) 08-08

Bank Account Number

Promotional Code (recieve 10-50% off) IEODBOBOOG

Figure 17



Thanks for shopping with HacmeBooks. You can be assured that your order was received and will be processed when we get around to it.
Receipt

Total: $5.14
Credit Card: 1234-1234-1234-1234
Expiration: 08-08
Bank Account:

Figure 18

While this kind of vulnerability may seem simplistic and contrived, it is based on reality. Foundstone security consultants regularly see problems of this nature when assessing real-world applications.

Lesson 5 - Broken Access Control

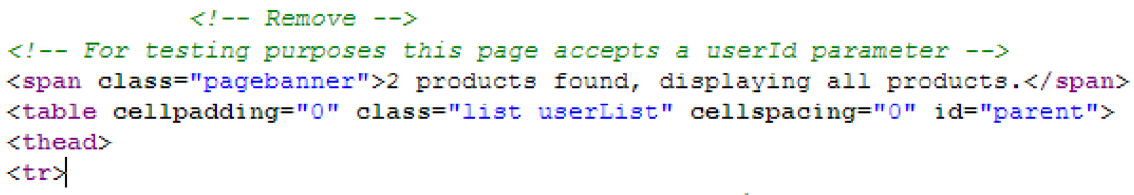
Lesson#	**5**
Vulnerability Exploited	Authorization Backdoor
Exploit Result	Horizontal Privilege Escalation
Input Field(s)	Browser Address Bar
Input Data	*http://localhost:8989/HacmeBooks/browseOrders.html?userId=testuser*
Corresponding Figures(s)	Figure 19 Figure 20

Developers tend to be very conscious of elevated privilege needs. Most developers effectively check for administrator privileges within escalated code blocks (called "vertical escalation").

But what happens when an application does not check if the user is crossing a *horizontal* privilege boundary by trying to impersonate a different user with similar permissions? It is often easy to overlook access control scenarios that are horizontal in nature.

In this lesson, the attacker will perform reconnaissance of the application by examining their own user profile and previous orders. The raw HTML that is delivered to the browser is often a good source of information for an attacker who is trying to learn how the site works. Plus, it is common for developers to leave comments in HTML code that are helpful to an attacker. They often confuse HTML comments for JSP comments and inadvertently reveal something about the underlying JSP code.

The attacker discovered the following in the HTML source code for the "My Orders" link:



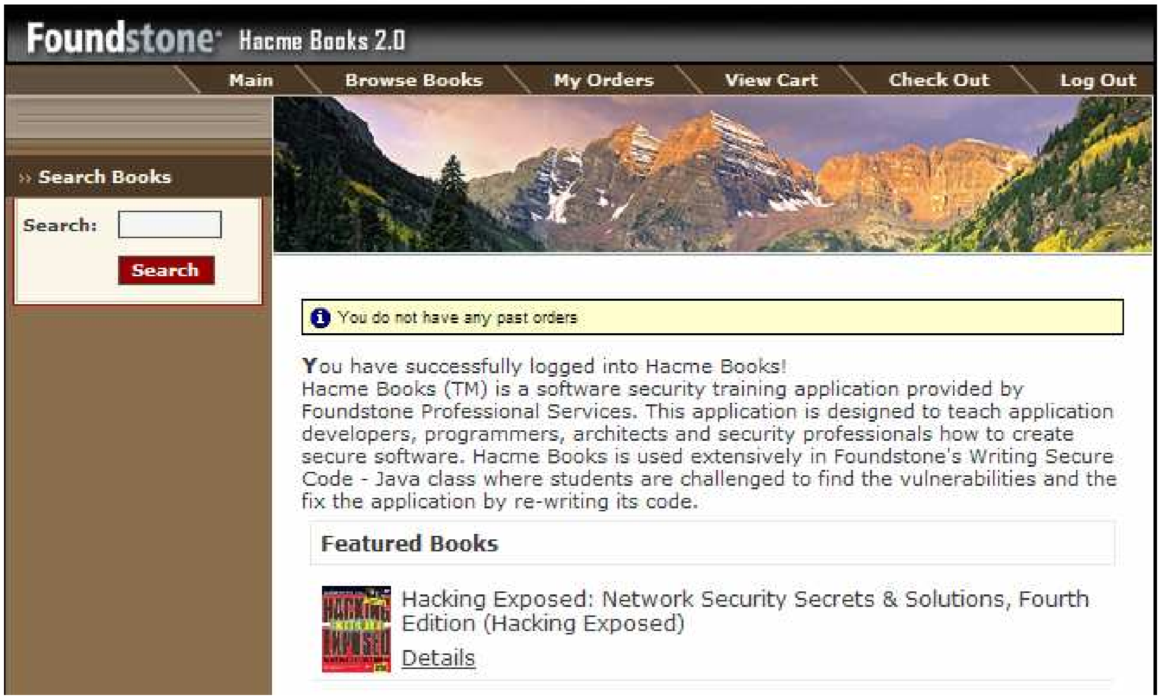
```
            <!-- Remove -->
<!-- For testing purposes this page accepts a userId parameter -->
<span class="pagebanner">2 products found, displaying all products.</span>
<table cellpadding="0" class="list userList" cellspacing="0" id="parent">
<thead>
<tr>
```

Hmmm… looks like a developer forgot to remove something from the functionality of the code!

The attacker will test this theory and see if they can exploit this backdoor in the application.

In order to prepare for this attack, the attacker needs to create some fake data (including at least 2 user accounts total) and complete some book purchase transactions. This will provide the seed data needed to launch the attack. Let's assume the attacker creates a new user with the account name of 'hacker'.

While logged in as 'hacker', they have never purchased any books. If they browse to 'My Orders' they will see the following:



You do not have any past orders

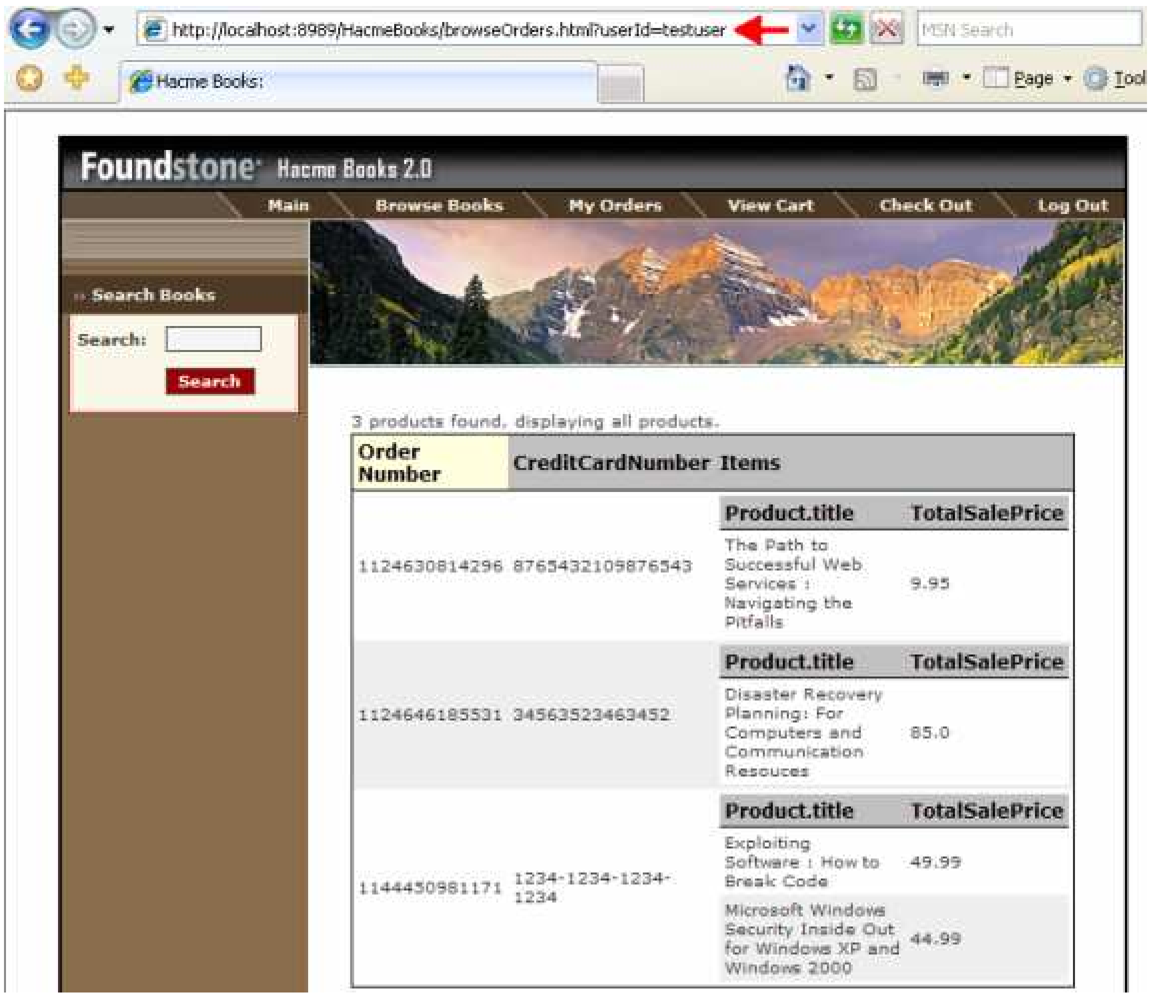
You have successfully logged into Hacme Books!
Hacme Books (TM) is a software security training application provided by Foundstone Professional Services. This application is designed to teach application developers, programmers, architects and security professionals how to create secure software. Hacme Books is used extensively in Foundstone's Writing Secure Code - Java class where students are challenged to find the vulnerabilities and the fix the application by re-writing its code.

Featured Books

Hacking Exposed: Network Security Secrets & Solutions, Fourth Edition (Hacking Exposed)
Details

Figure 19

In Figure 19, the application displays an information message that explains that the user has never bought anything. Based on the source code comment that was seen earlier, the attacker will try to access information from 'testuser' by manipulating the URL string in the address bar.



http://localhost:8989/HacmeBooks/browseOrders.html?userId=testuser
Hacme Books:
Foundstone Hacme Books 2.0
Main
Browse Books
My Orders
View Cart
Check Out
Log Out
Search Books
Search:
Search

3 products found, displaying all products.

Order Number	CreditCardNumber	Items	
		Product.title	**TotalSalePrice**
1124630814296	8765432109876543	The Path to Successful Web Services : Navigating the Pitfalls	9.95
		Product.title	**TotalSalePrice**
1124646185531	34563523463452	Disaster Recovery Planning: For Computers and Communication Resouces	85.0
		Product.title	**TotalSalePrice**
1144450981171	1234-1234-1234-1234	Exploiting Software : How to Break Code	49.99
		Microsoft Windows Security Inside Out for Windows XP and Windows 2000	44.99

Figure 20

The attacker's theory is correct. Now the attacker is free to launch future attacks on other user accounts.

This lesson points out two problems. First, the developer left comments in the source code that provided the attacker with the clues necessary to launch the attack. Second, the developer is not authorizing the action. There should be a horizontal privilege check to ensure that this information is only available to the user that made the purchases.

About Foundstone Professional Services

Foundstone Professional Services, a division of McAfee, offers a unique combination of services and education to help organizations continuously and measurably protect the most important assets from the most critical threats. Through a strategic approach to security, Foundstone identifies, recommends, and implements the right balance of technology, people, and process to manage digital risk and leverage security investments more effectively.

Foundstone's Strategic Secure Software Initiative (S3i™) services help organizations design and engineer secure software. By building in security throughout the Software Development Lifecycle, organizations can significantly reduce their risk of malicious attacks and minimize costly remediation efforts. Services include:

- Source Code Audits
- Software Design and Architecture Reviews
- Threat Modeling
- Web Application Penetration Testing
- Software Security Metrics and Measurement

For more information about Foundstone S3i services, go to www.foundstone.com/s3i.

Foundstone S3i training is designed to teach programmers and application developers how to build secure software and to write secure code. Classes include:

- Building Secure Software
- Writing Secure Code – Java (J2EE)
- Writing Secure Code – ASP.NET (C#)
- Ultimate Web Hacking

For the latest course schedule, go to www.foundstone.com/education.

Acknowledgements

Many individuals at Foundstone contributed to the development and testing of Hacme Books and the production of this whitepaper.   Shanit Gupta and Alex Smolen were primary contributors and their help is greatly appreciated. David Raphael was the original author of Hacme Books 1.0. In addition, the rest of the "con" group helped review the project and prepare it for release.