UNIT - IV

UNIT - IV: Web Hacking & Attacking Applications

Web Servers, Web Application Vulnerabilities, and Web-Based Password Cracking Techniques, Denial of Service and Session Hijacking, Web Hacking, Attacking Applications: SQL Injection and Buffer Overflows

Web Servers

Web servers and web applications have a very high potential to be compromised. The primary reason for this is that the systems that run web server software must be publicly available on the Internet. The web server cannot be completely isolated and to some degree must be available to legitimate users. Once a web server has been compromised, the system can provide hackers with another door into the network. Not only the web server software but also applications that run on the web server are open to attack and can be exploited. Due to their function, web servers are more accessible than other systems and less protected, so they're easier to exploit.

The target information on a web server usually resides in a database on the web server; this database is accessed via a web application. For this reason, web servers and web applications go hand in hand. Compromising the web server is usually done to gain access to the underlying data in the web application.

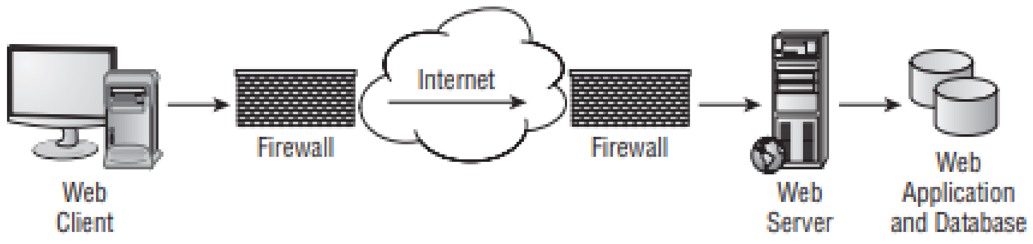
How Web Servers Work

Web servers use Hypertext Transfer Protocol (HTTP) and Hypertext Transfer Protocol Secure (HTTPS) to allow web-based clients to connect to them and view and download files. HTTP is an Application-layer protocol in the TCP/IP stack. HTTP and

HTTPS are the primary protocols used by web clients accessing web pages residing on web servers on the Internet. Hypertext Markup Language (HTML) is the language used to create web pages and allows those pages to be rendered in web browser software on web clients.

The HTTP protocol operates as shown in Figure 8.1.

FIGURE 8.1 HTTP protocol components



1. The web client initially opens a connection to the web server IP address using TCP port 80.
2. The web server waits for a GET request from the client requesting the home page for the website.
3. The web server responds with the HTML code for the web server home page.
4. The client processes the HTML code and the web client's browser software renders the page on the client device.

Understanding how web servers work and consequently how they are hacked is an important part of your job as a CEH. This includes knowing their vulnerabilities, as well as understanding the types of attacks a hacker may use. In addition, you should know when to use patch management techniques and understand the methods used to harden web servers.

Types of Web Server Vulnerabilities

Web servers, like other systems, can be compromised by a hacker. The following vulnerabilities are most commonly exploited in web servers:

Misconfiguration of the Web Server Software A common issue with using Microsoft's Internet Information Server (IIS) as a web server is the use of the default website. The permissions on the default website are open, meaning the default settings leave the site open to attack. For example, all users in the everyone group have full control to all the files in the default website directory. It is critical to edit and restrict permissions once IIS is installed on the server as the default system user, IUSR_COMPUTERNAME, is a member of the everyone group. Consequently, anyone accessing the default website will be able to access all files in the default website folder and will have dangerous permissions such as Execute and Full Control to the files.

Operating System or Application Bugs, or Flaws in Programming Code All programs, including the OS and web server applications, should be patched or updated on a regular basis. For Windows systems, this includes security patches, hot fixes, and Windows Updates. All of these patches can be automated or manually applied to the systems once they have been tested.

Vulnerable Default Installation Operating system and web server software settings should not be left at their defaults when installed, and should be updated on a continuous basis.

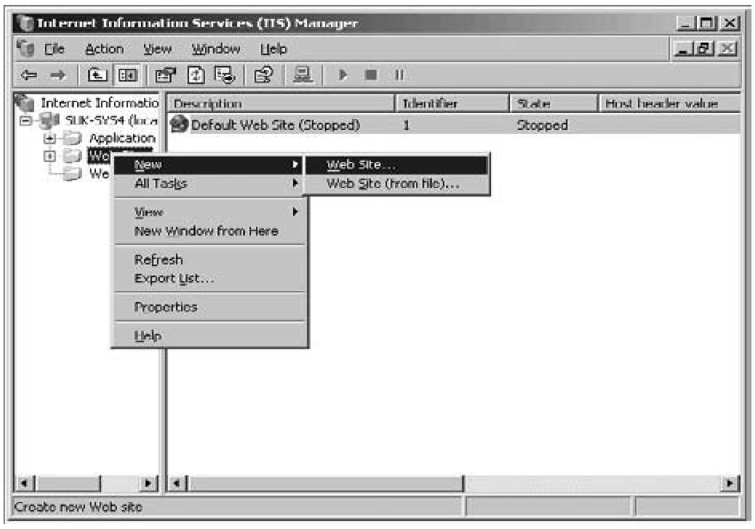
Hackers exploit these vulnerabilities to gain access to the web server. Because web servers are usually located in a demilitarized zone (DMZ) which is a publicly accessible area between two packet filtering devices and can be easily accessed by the organization’s client systems an exploit of a web server offers a hacker easier access to internal systems or databases.

Exercise 8.1

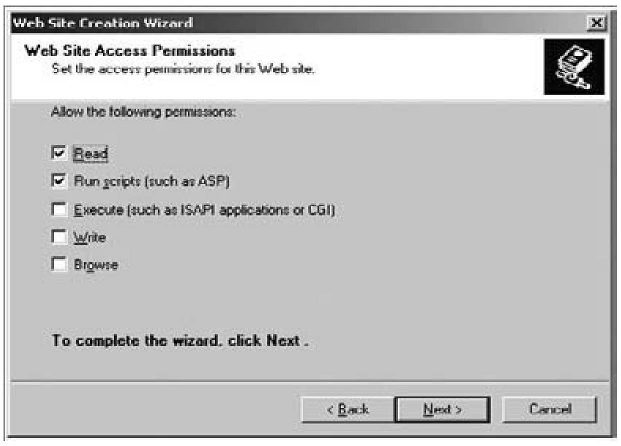
Disabling the Default Website in Internet Information Server

To disable the default website in IIS and add a new site, follow these steps:

1. Open IIS on your Windows Server or virtual machine (VM).
2. Select Web Sites in the left pane.
3. Right-click the default website in the right pane and select Stop from the context menu. The default website is now stopped.
4. To create a new site, right-click Web Sites in the left pane and select New Web Site.



5. The Web Site Creation Wizard launches. Within the wizard will be a screen to change permission on the website directory.


Web Site Creation Wizard

Web Site Access Permissions

Set the access permissions for this Web site.

Allow the following permissions:

- [x] Read
- [x] Run scripts (such as ASP)
- [ ] Execute (such as ISAPI applications or CGI)
- [ ] Write
- [ ] Browse

To complete the wizard, click Next .

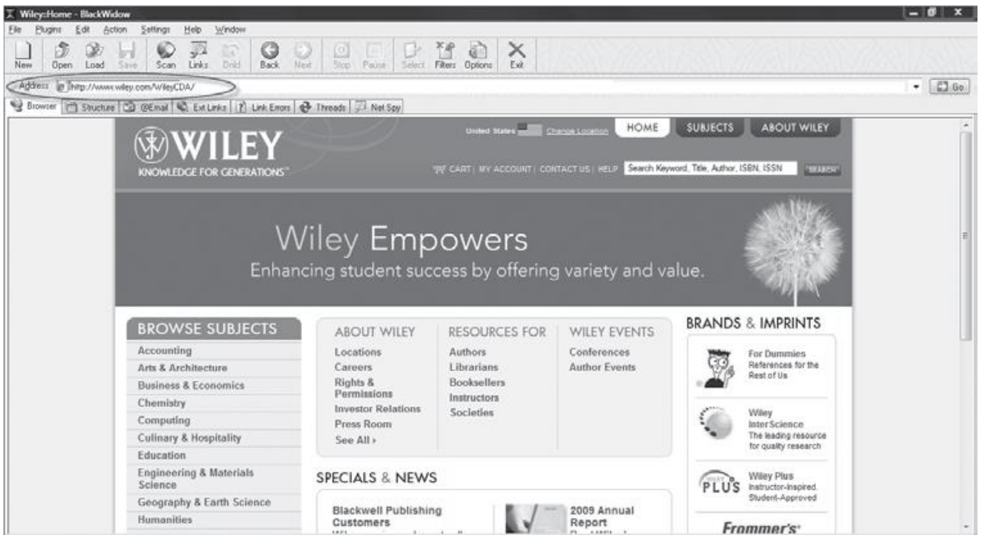
NOTE Website cloaking is the ability of a web server to display different types of web pages based on the user's IP address.

In many cases, it is useful to gather all or a portion of the files that make up a website. One option is to right-click any web page and select View Source from the context menu. This command will open up a new window with the source code for the page. You can then save the text file as a document on the local machine. This approach works, but it isn't a practical way of copying all the files for a target website. An easy-to-use program called BlackWidow can make the process of copying website files much easier. Exercise 8.2 shows you how to use the BlackWidow program to copy an entire website or a portion of the site.

ExErCiSE 8.2

using BlackWidow to Copy a Website

1. Download and install the BlackWidow application from www.softbytelabs.com.
2. Open the BlackWidow program.
3. Enter a target website address in the BlackWidow address bar:

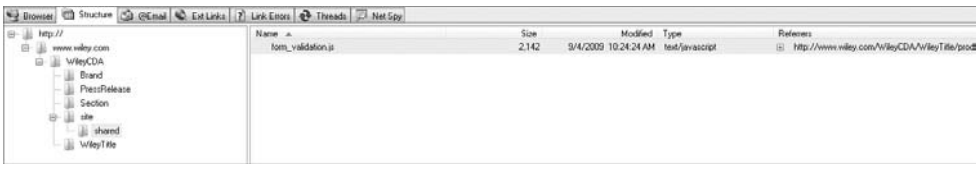

Wiley:Home - BlackWidow
File Plugins Edit Action Settings Help Window
New Open Load Save Scan Links Dnld Back Next Stop Pause Select Filters Options Exit
Address http://www.wiley.com/WileyCDA/
Go
Browser Structure @Email Ext Links Link Errors Threads Net Spy
WILEY
KNOWLEDGE FOR GENERATIONS
HOME SUBJECTS ABOUT WILEY
CART | MY ACCOUNT | CONTACT US | HELP
Search Keyword, Title, Author, ISBN, ISSN
Wiley Empowers
Enhancing student success by offering variety and value.
BROWSE SUBJECTS
Accounting
Arts & Architecture
Business & Economics
Chemistry
Computing
Culinary & Hospitality
Education
Engineering & Materials Science
Geography & Earth Science
Humanities
ABOUT WILEY
Locations
Careers
Rights & Permissions
Investor Relations
Press Room
See All ›
RESOURCES FOR
Authors
Librarians
Booksellers
Instructors
Societies
WILEY EVENTS
Conferences
Author Events
SPECIALS & NEWS
Blackwell Publishing Customers
2009 Annual Report
BRANDS & IMPRINTS
For Dummies
References for the Rest of Us
Wiley InterScience
The leading resource for quality research
Wiley Plus
Instructor-Inspired. Student-Approved
Frommer's

4. Click the Scan button on the BlackWidow toolbar.

5. Click the Structure tab.

ExErCiSE 8.2 *(continued)*

6. Browse the website folder structure. Right-click a file or folder and choose Copy Selected Files to copy the website files to your computer.



```
http://
  www.wiley.com
    WileyCDA
      Brand
      PressRelease
      Section
      site
        shared
      WileyTitle
```

Name	Size	Modified	Type	Referrers
form_validation.js	2,142	9/4/2009 10:24:24 AM	text/javascript	http://www.wiley.com/WileyCDA/WileyTitle/prod

Attacking a WebServer

Web servers typically listen on TCP port 80 (HTTP) and TCP port 443 (HTTPS). Because those ports must be open and available to web clients, any firewalls or packet filtering devices between the web client and web server must pass traffic destined for those ports. Web application software sits on top of the web server software and allows access to additional ports.

One of the initial information-gathering steps targeting web servers is *banner grabbing*. Banner grabbing is an attempt to gather information about a web server such as the OS and web server software and version. Exercise 8.3 shows you how to use banner grabbing.

ExErCiSE 8.3

Banner Grabbing

1. At the command prompt on your Windows PC, type

```
telnet <IP address> 80
```

   The IP address is the address of the web server target. Also, the URL can be used instead of the IP address.

2. Next, in the telnet window type

```
HEAD/HTTP/1.0
```

   Then press Enter.

The web server banner will then be returned. The banner will look something like the fol- lowing:

```
Server:  Microsoft-IIS/5.0
Date: Fri, 14 Aug 2009 1:14:42 GMT
Content-Length:340
Content-Type:
text/html
```

The banner grabbing result will usually identify the web server type and version. This information is important because exploits against this web server type and version can be identified. The next step after banner grabbing would be to attack the web server or attack a web application and gain access to data on the server.

A benign but visible type of attack against web servers is defacement. Hackers deface websites for sheer joy and an opportunity to enhance their reputations rather than gathering any useful data. *Defacing* a website means the hacker exploits a vulnerability in the OS or web server software and then alters the website files to show that the site has been hacked. Often the hacker displays their hacker name on the website's home page.

Common website attacks that enable a hacker to deface a website include the following:

Capturing administrator credentials through man-in-the-middle attacks

Revealing an administrator password through a brute-force attack

Using a DNS attack to redirect users to a different web server

Compromising an FTP or email server

Exploiting web application bugs that result in a vulnerability

Misconfiguring web shares

Taking advantage of weak permissions

Rerouting a client after a firewall or router attack

Using SQL injection attacks (if the SQL server and web server are the same system)

Using telnet or Secure Shell (SSH) intrusion

Carrying out URL poisoning, which redirects the user to a different URL

Using web server extension or remote service intrusion

Intercepting the communication between the client and the server and changing the cookie to make the server believe that there is a user with higher privileges (applies to cookie-enabled security)

Exercise 8.4 walks you through using the Metasploit Framework to exploit a web server vulnerability.

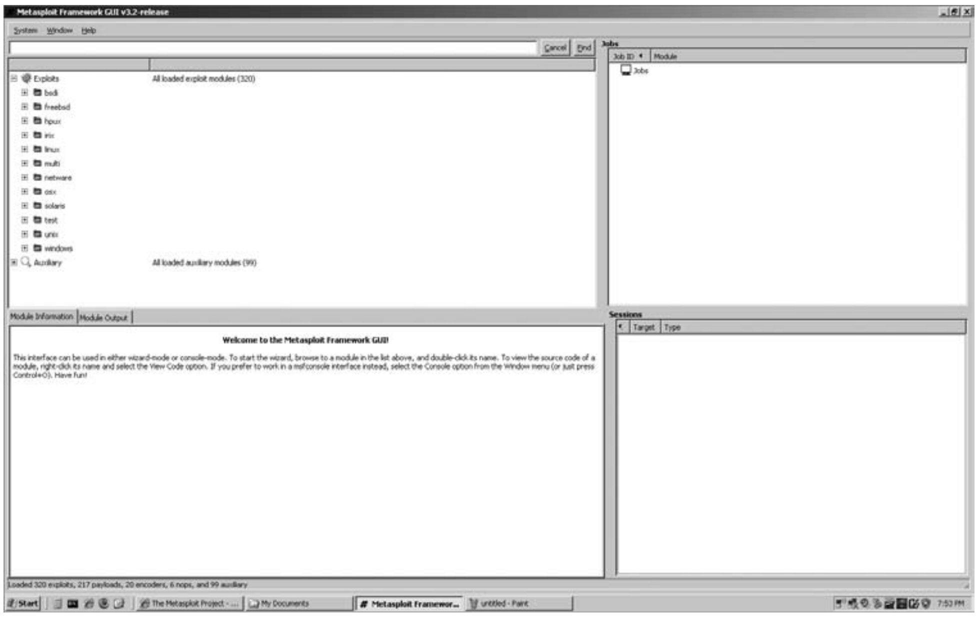
WARNING

It is important that the machine or VM have all antivirus and firewall pro- grams completely shut down prior to installing Metasploit. Otherwise, the antivirus or firewall can block some components of Metasploit, causing it not to function or open properly. As we mentioned in the lab setup guide in the Introduction to this book, you should never install Metasploit on a production machine. Use either a VM or lab test machine to run this software.

ExErCiSE 8.4

using metasploit to Exploit a Web Server Vulnerability

1. Download and install Metasploit 3.2 on your Windows XP or Vista computer or VM (www.metasploit.com).
2. Choose all the default options when installing Metasploit.
3. Select the Online Update option in the Metasploit 3 folder under Programs.
4. After the online update has completed, open the Metasploit GUI file in the Metasploit 3 folder.



```
Exploits        All loaded exploit modules (320)
  bsdi
  freebsd
  hpux
  irix
  linux
  multi
  netware
  osx
  solaris
  test
  unix
  windows
Auxiliary       All loaded auxiliary modules (99)
```

Welcome to the Metasploit Framework GUI!

This interface can be used in either wizard-mode or console-mode. To start the wizard, browse to a module in the list above, and double-click its name. To view the source code of a module, right-click its name and select the View Code option. If you prefer to work in a msfconsole interface instead, select the Console option from the Window menu (or just press Control+O). Have fun!

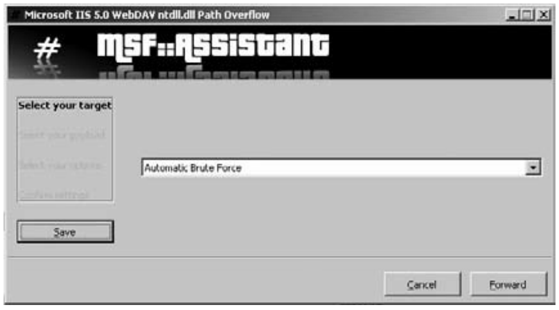
Loaded 320 exploits, 217 payloads, 20 encoders, 6 nops, and 99 auxiliary

5. Expand the Windows folder under Exploits and then expand the IIS folder.

ExErCiSE 8.4 ***(continued)***

6. Double-click the ms03_007_ntdll_webdav exploit. The

MSF Assistant Wizard launches.


Microsoft IIS 5.0 WebDAV ntdll.dll Path Overflow

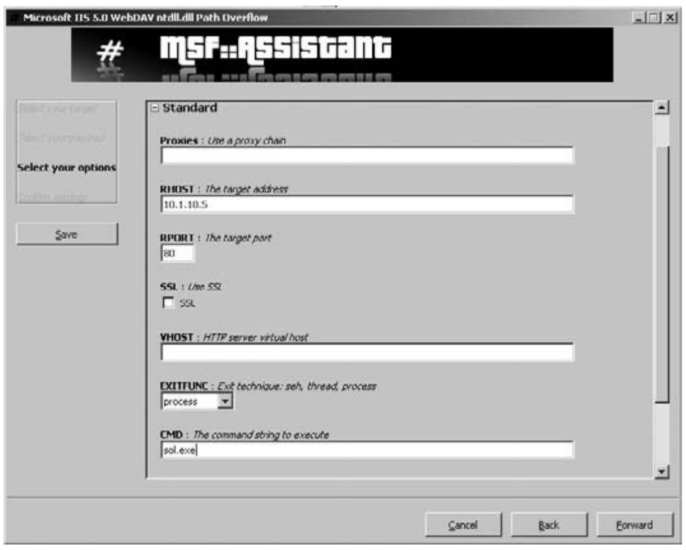
Automatic Brute Force

7. Click the Forward button to move to the next screen of the wizard.

8. Select Windows/Exec from the Payload drop-down list, and then click Forward.

9. Type the IP address of the target IIS web server in the RHOST field. *This server should be an unpatched version of Windows 2000 for this particular payload to work. If that is not the case, choose a different payload to which the server is vulnerable.*

10. Type **sol.exe** in the CMD field. This is the executable that will be run on the remote target host. sol.exe is the solitaire game, which should be on all Windows operating systems. The payload is what will be delivered to the target system. In this case, it
is similar to typing **sol.exe** at the command prompt of the IIS server. Obviously this executable is benign, but this exercise illustrates how a more dangerous executable, such as a virus or Trojan, could be run on a target system.


Microsoft IIS 5.0 WebDAV ntdll.dll Path Overflow

Standard

Proxies : *Use a proxy chain*

RHOST : *The target address*
10.1.10.5

RPORT : *The target port*
80

SSL : *Use SSL*
☐ SSL

VHOST : *HTTP server virtual host*

EXITFUNC : *Exit technique: seh, thread, process*
process

CMD : *The command string to execute*
sol.exe

ExErCiSE 8.4 ***(continued)***

11. Click the Forward button to move to the next screen of the wizard.
12. Click the Apply button. The exploit will appear under Jobs until it is delivered to the target system.
13. Confirm in the Windows IIS Server VM or on the IIS PC that the Solitaire program is running. If the program is not running, confirm that Solitaire is installed on the IIS server and try the Metasploit exploit again.

Hacking Internet Information Server

Windows IIS is one of the most popular web server software products. Because of the popularity and number of web servers running IIS, many attacks can be launched against IIS servers. The three most common attacks against IIS are as follows:

Directory traversal

Source disclosure

Buffer overflow

A *directory-traversal attack* is based on the premise that web clients are limited to specific directories within the Windows files system. The initial directory access by web clients is known as the *root directory* on a web server. This root directory typically stores the home page usually known as Default or Index, as well as other HTML documents for the web server. Sub directories of the root directory contain other types of files; for example, scripts may contain dynamic scripting files for the web server. The web server should allow users to access only these specific directories and sub directories of root. However, a directory traversal attack permits access to other directories within the file system.

Windows 2000 systems running IIS are susceptible to a directory traversal attack, also known as the Unicode exploit. The vulnerability in IIS that allows for the directory traversal/Unicode exploit occurs only in un patched Windows 2000 systems and affects CGI scripts and Internet Server Application Programming Interface (ISAPI) extensions such as
.asp. The vulnerability exists because the IIS parser was not properly interpreting Unicode, thus giving hackers system-level access.

Essentially, Unicode converts characters of any language to a universal hex code specification. However, the Unicode is interpreted twice, and the parser only scans the resulting request once (following the first interpretation). Hackers could therefore sneak file requests through IIS. For example, utilizing %c0% af instead of a slash in a relative path name exploits the IIS vulnerability. In some cases, the request lets the hacker gain access to files that they otherwise shouldn't be able to see. The Unicode directory traversal vulnerability allows a hacker to add, change, or delete files, or upload and run code on the server. The ability to add or run files on the system enables a hacker to install a Trojan or backdoor on the system.

NOTE The IIS Unicode exploit is an outdated vulnerability and is presented in this text as a proof of concept—that is, proof that the vulnerability exists and can be exploited.

Buffer overflow attacks are not unique to web servers and can also be launched against other types of systems. A buffer overflow involves sending more data, usually in the form of a text string, than the web server is capable of handling. The primary entry point for buffer overflows is a web form on the web server.

Source disclosure attacks occur when the source code of a server application can be gathered. Source disclosure attacks can lead to a hacker identifying the application type, programming language, and other application-specific information. All this information can allow a potential hacker to identify security holes and potential exploits that can be delivered to the web server. Again, most of a hacker’s time is spent gathering information about a target in order to identify the best point of entry for an exploit.

Putting it All Together using Source disclosure Attacks

An example of performing a source disclosure attack would be to run Black Widow against a web server and copy all the files to a local directory. In reviewing the source files from Black Widow, you can obtain the name of the server, the IP address, and the version. Additional information-gathering tools such as Netcraft can aid in the discovery of the OS, web server software type, and version. Additional information may be gathered regarding the JavaScript (.js files) or Active Server Pages (.asp files) that reside on the server. Based on the web server applications and vulnerabilities, Metasploit can be used to deliver a payload to the server. Depending on the patch level and vulnerability, the pay load can be fairly benign or serious enough to cause the hacker to gain access to valuable data. The best countermeasure to the source disclosure attack and other types of attacks is to patch the OS, web server, and all server applications to the most current level and maintain an active patch management program.

A CEH must be aware of all the information-gathering techniques to identify potential vulnerabilities in web servers and web applications. The reason this knowledge is so important for the CEH is so that they can defend against the same attacks and implement counter measures to prevent attacks.

Patch Management Techniques

Patch management plays a critical role in preventing and mitigating the risk of attack against web servers and web applications. *Patch management* is the process of updating appropriate patches and hot fixes required by a system vendor. Proper patch management involves choosing how patches are to be installed and verified, and testing those patches on a non-production network prior to installation.

You should maintain a log of all patches applied to each system. To make patch installation easier, you can use automated patch-management systems provided by Patch Link, St. Bernard Software, Microsoft, and other software vendors to assess your systems and decide which patches to deploy.

Real World Scenario

First Week on the job as a Web Administrator

As a newly hired network administrator for a small company of 40 employees, it was my responsibility to review the configuration and patches for a small network with two servers. The company used IIS 5.0 on a Windows 2000 server that had been serving the corpo- rate website to clients for three years. The servers had been installed and configured by a consulting company three years prior to my joining the staff.

The website content was updated regularly by the marketing assistant, but no other update had been made to the server.

So, I embarked upon updating and performing patch management on the web server. The company had no firewall protecting the Internet connection, and the Windows Server OS had not had any patches or hot fixes applied to it since installation. The IIS web server software was also out of date. All of this presented a huge security risk to the organization, and patch management was the highest priority to protect the web server and applications running on it.

As I applied security patches and hot fixes, to first the OS and then IIS, I found that malware, such as the Code Red worm and numerous viruses, had already attacked the system. It took several days of applying patches and hot fixes and updating virus definitions before the web server was brought up-to-date. Luckily for the small company, I was able to bring the OS and web server software up-to-date and implement a system for patch management before the network was damaged or a serious security breach occurred.

Hacking Tools

N-Stalker Web Application Security Scanner allows you to assess a web application for a large number of vulnerabilities, including cross-site scripting, SQL injection, buffer over- flow, and parameter-tampering attacks.

The Metasploit Framework is a freeware tool used to test or hack operating systems or web server software. Exploits can be used as plug-ins, and testing can be performed from a Windows or Unix platform. Metasploit was originally a command-line utility, but it now has a web browser interface. Using Metasploit, hackers can write their own exploits as well as utilize standard exploits.

CORE IMPACT and SAINT Vulnerability Scanner are commercial exploit tools used to test and compromise operating systems and web server software.

Web Server Hardening Methods

A web server administrator can do many things to *harden* a server (increase its security). The following are ways to increase the security of the web server:

Rename the administrator account, and use a strong password. To rename the administrator account in Windows, open the User Manager, right-click the Administrator account, and select Rename.

Disable default websites and FTP sites. The process to disable default websites was described earlier in this chapter: right-click the default website in IIS Manager and choose Stop. The same process works for the default FTP site.

Remove unused applications from the server, such as Web DAV. Unnecessary applications can be removed on a server by using Add/Remove Programs in the Windows Control Panel.

Disable directory browsing in the web server's configuration settings.

Add a legal notice to the site to make potential attackers aware of the implications of hacking the site.

Apply the most current patches, hot fixes, and service packs to the operating system and web server software.

Perform bounds checking on input for web forms and query strings to prevent buffer overflow or malicious input attacks.

Disable remote administration.

Use a script to map unused file extensions to a 404 ("File not found") error message.

Enable auditing and logging.

Use a firewall between the web server and the Internet and allow only necessary ports (such as 80 and 443) through the firewall.

Replace the GET method with the POST method when sending data to a web server.

Web Application Vulnerabilities

In addition to understanding how a hacker can exploit a web server, it's important for a CEH to be familiar with web application vulnerabilities. In this section, we'll discuss how web applications work, as well as the objectives of web application hacking. We'll also examine the anatomy of a web application attack and some actual web application threats. Finally, we'll look at Google hacking and countermeasures you should be familiar with.

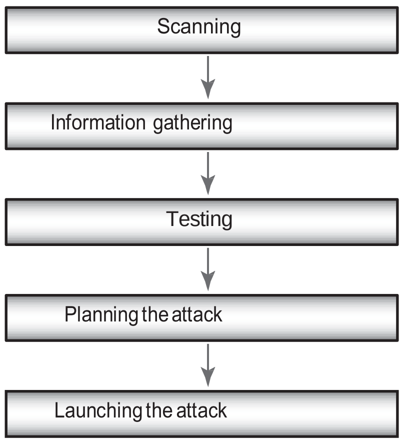
Web applications are programs that reside on a web server to give the user functionality beyond just a website. Database queries, web mail, discussion groups, and blogs are all examples of web applications.

A web application uses a client/server architecture, with a web browser as the client and the web server acting as the application server. JavaScript is a popular way to implement web applications. Because web applications are widely implemented, any user with a web browser can interact with most site utilities.

The purpose of hacking a web application is to gain confidential data. Web applications are critical to the security of a system because they usually connect to a database that contains information such as identities with credit card numbers and passwords. Web application vulnerabilities increase the threat that hackers will exploit the operating system and web server or web application software. Web applications are essentially another door into a system and can be exploited to compromise the system.

Hacking web applications is similar to hacking other systems. Hackers follow a five-step process: they scan a network, gather information, test different attack scenarios, and finally plan and launch an attack. The steps are listed in Figure 8.2.

Figure 8.2 The stages of a web application attack



Web Application Threats and Countermeasures

Many web application threats exist on a web server. The following are the most common threats and their countermeasures:

Cross-Site Scripting A parameter entered into a web form is processed by the web application. The correct combination of variables can result in arbitrary command execution. Countermeasure: Validate cookies, query strings, form fields, and hidden fields.

A countermeasure to cross-site scripting is to replace left and right angle bracket characters (< and >) with &lt; and &gt; using server scripts. A countermeasure to SSL attacks is to install a proxy server and terminate SSL at the proxy or install a hardware SSL accelerator and terminate SSL at this layer.

SQL Injection Inserting SQL commands into the URL gets the database server to dump, alter, delete, or create information in the database. SQL injection is covered in detail in Chapter 9, “Attacking Applications: SQL Injection and Buffer Overflows.” Countermeasure: Validate user variables.

Command Injection The hacker inserts programming commands into a web form. Countermeasure: Use language-specific libraries for the programming language.

Cookie Poisoning and Snooping The hacker corrupts or steals cookies. Countermeasures: Don’t store passwords in a cookie; implement cookie timeouts; and authenticate cookies.

Buffer Overflow Huge amounts of data are sent to a web application through a web form to execute commands. Buffer overflows is covered in detail in Chapter 9. Countermeasures: Validate user input length; perform bounds checking.

Authentication Hijacking The hacker steals a session once a user has authenticated. Countermeasure: Use SSL to encrypt traffic.

Directory Traversal/Unicode The hacker browses through the folders on a system via a web browser or Windows Explorer. Countermeasures: Define access rights to private folders on the web server; apply patches and hot fixes.

Hacking Tools

Instant Source allows a hacker to see and edit HTML source code. It can be used directly from within the web browser.

Wget is a command-line tool that a hacker can use to download an entire website, complete with all the files. The hacker can view the source code offline and test certain attacks prior to launching them against the real web server.

WebSleuth uses spidering technology to index an entire website. For example, WebSleuth can pull all the email addresses from different pages of a website.

BlackWidow can scan and map all the pages of a website to create a profile of the site.

SiteScope maps out the connections within a web application and aids in the deconstruction of the program.

WSDigger is a web services testing tool that contains sample attack plug-ins for SQL injection, cross-site scripting, and other web attacks.

Burp is a Windows-based automated attack tool for web applications. It can also be used to guess passwords on web applications and perform man-in-the-middle attacks.

Google Hacking

Google hacking refers to using Google's powerful search engine to locate high-value targets or to search for valuable information such as passwords.

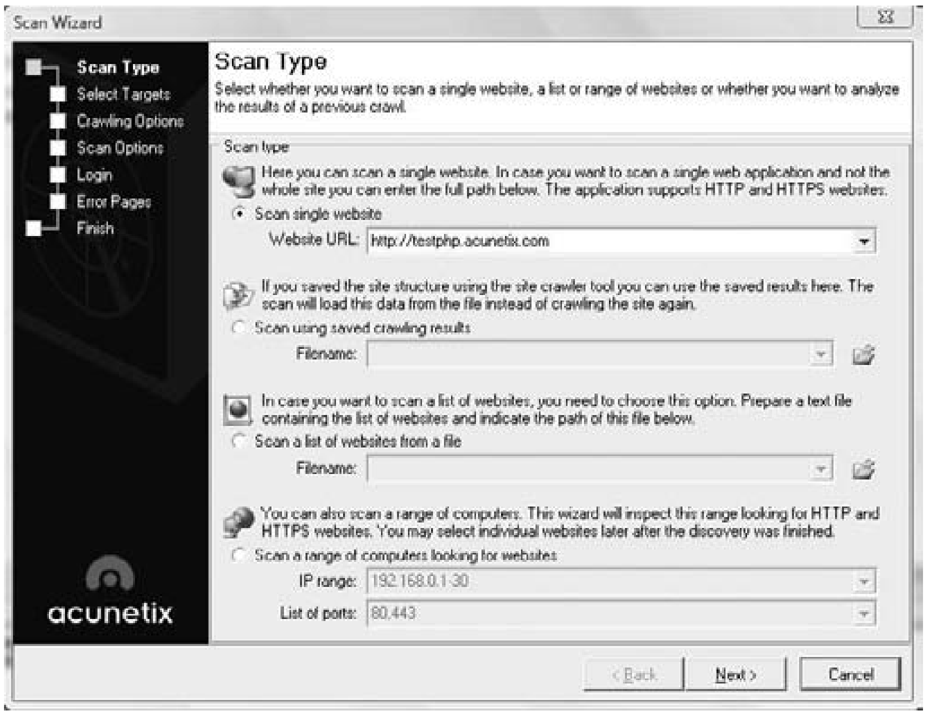
Many tools, such as http://johnny.ihackstuff.com and Acunetix Web Vulnerability Scanner, contain a list of Google hacking terms organized in a database, to make searching easier (see Exercise 8.5). For example, you can enter the term *password* or *medical records* in the Google search engine and see what information is available. Many times, Google can pull information directly out of private databases or documents.

Exercise 8.5

using Acunetix Web Vulnerability Scanner

1. Download and install Acunetix Web Vulnerability Scanner from www.acunetix.com.

2. Open the web scanner and select File □New Scan to open the Scan Wizard:



Scan Type

Select whether you want to scan a single website, a list or range of websites or whether you want to analyze the results of a previous crawl.

Scan type

Here you can scan a single website. In case you want to scan a single web application and not the whole site you can enter the full path below. The application supports HTTP and HTTPS websites.

(●) Scan single website

Website URL: http://testphp.acunetix.com

If you saved the site structure using the site crawler tool you can use the saved results here. The scan will load this data from the file instead of crawling the site again.

( ) Scan using saved crawling results

Filename:

In case you want to scan a list of websites, you need to choose this option. Prepare a text file containing the list of websites and indicate the path of this file below.

( ) Scan a list of websites from a file

Filename:

You can also scan a range of computers. This wizard will inspect this range looking for HTTP and HTTPS websites. You may select individual websites later after the discovery was finished.

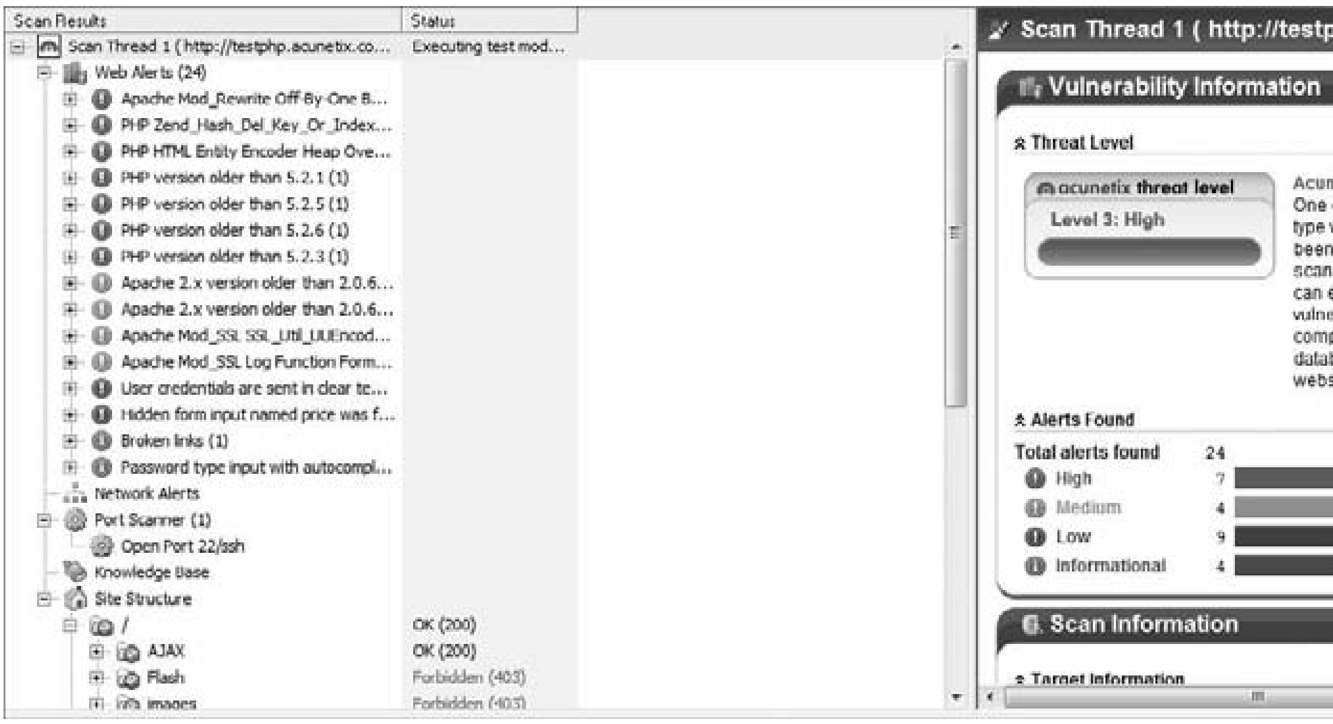
( ) Scan a range of computers looking for websites

IP range: 192.168.0.1-30

List of ports: 80,443

3. Follow the wizard prompts; accept the default values for the initial scan.

4. View the scan report once the scan is complete. Notice the web server and applica- tion vulnerabilities in the scan report.



```
Scan Results                                         Status
Scan Thread 1 ( http://testphp.acunetix.co...        Executing test mod...
  Web Alerts (24)
    Apache Mod_Rewrite Off-By-One B...
    PHP Zend_Hash_Del_Key_Or_Index...
    PHP HTML Entity Encoder Heap Ove...
    PHP version older than 5.2.1 (1)
    PHP version older than 5.2.5 (1)
    PHP version older than 5.2.6 (1)
    PHP version older than 5.2.3 (1)
    Apache 2.x version older than 2.0.6...
    Apache 2.x version older than 2.0.6...
    Apache Mod_SSL SSL_Util_UUEncod...
    Apache Mod_SSL Log Function Form...
    User credentials are sent in clear te...
    Hidden form input named price was f...
    Broken links (1)
    Password type input with autocompl...
  Network Alerts
  Port Scanner (1)
    Open Port 22/ssh
  Knowledge Base
  Site Structure
    /                                                OK (200)
      AJAX                                           OK (200)
      Flash                                          Forbidden (403)
      images                                         Forbidden (403)
```

```
Scan Thread 1 ( http://testp
Vulnerability Information
Threat Level
acunetix threat level
Level 3: High
Alerts Found
Total alerts found   24
  High                7
  Medium              4
  Low                 9
  Informational       4
Scan Information
Target information
```

5. Create another scan using the wizard and target your lab web server or web server VM. View and analyze the scan report for your lab web server.

Web-Based Password Cracking Techniques

As a CEH, you need to be familiar with the techniques hackers use to crack web-based passwords. This includes being able to list the various authentication types, knowing what a password cracker is, identifying the classifications of password-cracking techniques, and knowing the available countermeasures. We'll look at each in the following sections.

Authentication Types

Web servers and web applications support multiple authentication types. The most common is HTTP authentication. There are two types of HTTP authentication: basic and digest. Basic HTTP authentication sends the username and password in clear text, whereas digest authentication hashes the credentials and uses a challenge-response model for authentication.

In addition, web servers and web applications support the following types of authentication:

NTLM Authentication This type uses Internet Explorer and IIS web servers, making NTLM more suitable for internal authentication on an intranet that uses Microsoft operating systems. Windows 2000 and 2003 servers utilize Kerberos authentication for a more secure option.

Certificate-Based Authentication This type uses an x.509 certificate for public/private key technology.

Token-Based Authentication A token, such as SecureID, is a hardware device that displays an authentication code for 60 seconds; a user uses this code to log into a network.

Biometric Authentication This type uses a physical characteristic such as fingerprint, eye iris, or handprint to authenticate the user.

Password Attacks and Password Cracking

The three types of password attacks are as follows:

Dictionary Uses passwords that can be found in a dictionary

Brute-Force Guesses complex passwords that use letters, numbers, and special characters

Hybrid Uses dictionary words with a number or special character as a substitute for a letter

A *password cracker* is a program designed to decrypt passwords or disable password protection. Password crackers rely on dictionary searches (attacks) or brute-force methods to crack passwords.

The first step in a dictionary attack is to generate a list of potential passwords that can be found in a dictionary. The hacker usually creates this list with a dictionary generator program or dictionaries that can be downloaded from the Internet. Next, the list of dictionary words is hashed or encrypted. This hash list is compared against the hashed password the hacker is trying to crack. The hacker can get the hashed password by sniffing it from a wired or wireless network or directly from the Security Accounts

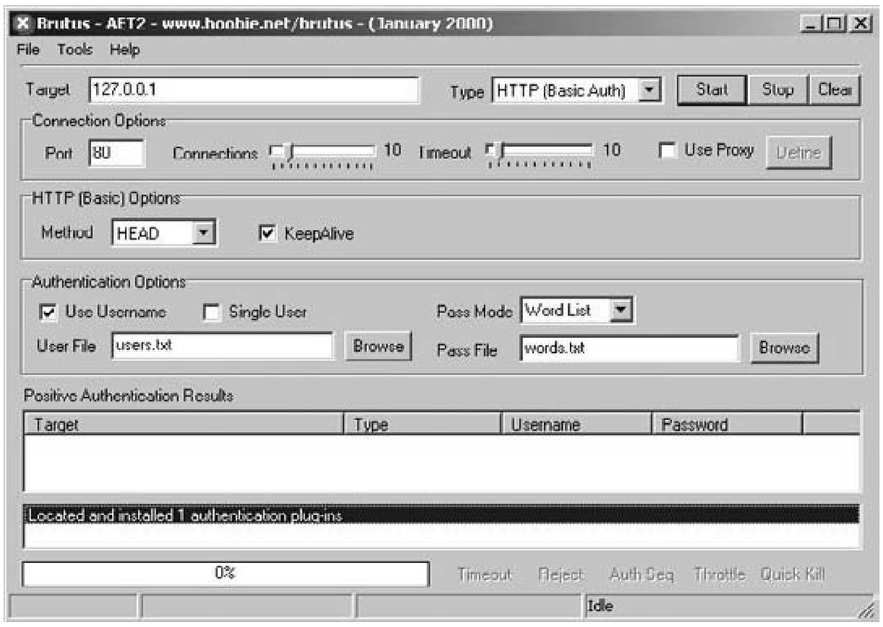
Manager (SAM) or shadow password files on the hard drive of a system. Finally, the program displays the un encrypted version of the password. Dictionary password crackers can only discover passwords that are dictionary words.

If the user has implemented a strong password, then brute-force password cracking can be implemented. Brute-force password crackers try every possible combination of letters, numbers, and special characters, which takes much longer than a dictionary attack because of the number of permutations. Exercise 8.6 walks you through using a password cracker called Brutus.

ExErCiSE 8.6

using a Password Cracker

1. Download and install Brutus from www.hoobie.net.
2. Open Brutus and type the web server address in the target field.



Brutus - AET2 - www.hoobie.net/brutus - (January 2000)

Target: 127.0.0.1
Type: HTTP (Basic Auth)

Connection Options
Port: 80
Connections: 10
Timeout: 10
☐ Use Proxy

HTTP (Basic) Options
Method: HEAD
☑ KeepAlive

Authentication Options
☑ Use Username
☐ Single User
Pass Mode: Word List
User File: users.txt
Pass File: words.txt

Positive Authentication Results

Target	Type	Username	Password

Located and installed 1 authentication plug-ins

0%

Idle

3. Click the Start button and view passwords in the positive

authentication results field at the bottom of the screen.

Hacking Tool

Webcracker is a tool that uses a word list to attempt to log on to a web server. It looks for the "HTTP 302 object moved" response to make guesses on the password. From this response, the tool can determine the authentication type in use and attempt to log on to the system.

The best password-cracking countermeasure is to implement strong passwords that are at least eight characters long (the old standard was six) and that include alphanumeric characters. User names and passwords should be different, because many user names are transmitted in clear text. Complex passwords that require uppercase, lowercase, and numbers or special characters are harder to crack. You should also implement a strong authentication mechanism such as Kerberos or tokens to protect passwords in transit.

Attacking Applications: SQL Injection and Buffer Overflows

SQL injection and buffer overflows are hacking techniques used to exploit weaknesses in applications. When programs are written, some parameters used in the creation of the application code can leave weaknesses in the program. SQL injection and buffer overflows are covered in the same chapter because they both are methods used to attack application and are generally caused by programming flaws. Generally, the purpose of SQL injection is to convince the application to run SQL code that was not intended.

SQL injection is a hacking method used to attack SQL databases, whereas buffer over flows can exist in many different types of applications. SQL injection and buffer overflows are similar exploits in that they’re both usually delivered via a user input field. The input field is where a user may enter a username and password on a website, add data to a URL, or perform a search for a keyword in another application. The SQL injection vulnerability is caused primarily by unverified or unsanitized user input via these fields.

Both SQL Server injection and buffer overflow vulnerabilities are caused by the same issue: invalid parameters that are not verified by the application. If programmers don’t take the time to validate the variables a user can enter into a variable field, the results can be serious and unpredictable. Sophisticated hackers can exploit this vulnerability, causing an execution fault and shutdown of the system or application, or a command shell to be executed for the hacker.

SQL injection and buffer overflow countermeasures are designed to utilize secure programming methods. By changing the variables used by the application code, weaknesses in applications can be greatly minimized. This chapter will detail how to perform a SQL injection and a buffer overflow attack and explore the best countermeasures to prevent the attack.

SQL Injection

As a CEH, it’s important for you to be able to define SQL injection and understand the steps a hacker takes to conduct a SQL injection attack. In addition, you should know SQL Server vulnerabilities, as well as countermeasures to SQL injection attacks.

SQL injection occurs when an application processes user-provided data to create a SQL statement without first validating the input. The user input is then submitted to a web application database server for execution. When successfully exploited, SQL injection can give an attacker access to database content or allow the hacker to remotely execute system commands. In the worst-case scenario, the hacker can take control of the server that is hosting the database. This exploit can give a hacker access to a remote shell into the server file system. The impact of a SQL injection attacks depends on where the vulnerability is in the code, how easy it is to exploit the vulnerability, and what level of access the application has to the database. Theoretically, SQL injection can occur in any type of application, but it is most commonly associated with web applications because they are most often attacked. As previously discussed in Chapter 8, “Web Hacking: GOOGLE, Web Servers, Web Application Vulnerabilities, and Web-Based Password Cracking Techniques,” web applications are easy targets because by their very nature they are open to being accessed from the Internet. You should have a basic understanding of how databases work and how SQL commands are used to access the information in the databases prior to attempting the CEH exam.

During a web application SQL injection attack, malicious code is inserted into a web form field or the website’s code to make a system execute a command shell or other arbitrary commands. Just as a legitimate user enters queries and additions to the SQL database via a web form, the hacker can insert commands to the SQL Server through the same web form field. For example, an arbitrary command from a hacker might open a command prompt or display a table from the database. A database table may contain personal information such as credit card numbers, social security numbers, or passwords. SQL Servers are very common database servers and used by many organizations to store confidential data. This makes a SQL Server a high-value target and therefore a system that is very attractive to hackers.

Real World Scenarlo

determining SQL Injection vulnerabilities

While performing a black-hat penetration test on a corporate network, a security tester, Tom, found a custom application on one of the publicly accessible web servers. Since this was a black hat test, Tom did not have access to the source code to see how the program had been created. But after performing some information gathering, he was able to determine that the server was running Microsoft Internet Information Server 6 along with ASP.NET, and this suggested that the database was Microsoft's SQL Server.

The login page of the web application had a username, a password field, and a forgotten password link, which ended up being the easiest way into the system. A forgotten password link works by looking in the user database for the user's email address and sending an email containing the password to that address.

So to determine if the forgotten password link was vulnerable to SQL injection, Tom entered a single quote as part of the data in the forgotten password field. The purpose was to see if the application would construct a SQL string literally without sanitizing the user input.
When submitting the form with a quote in the email address, he received a 500 error (server failure), and this suggested that the user input was being parsed literally.

The underlying SQL code of the form probably looked something like this:

```
SELECT fieldlist
  FROM table
  WHERE field = '$EMAIL';
```

Tom typed his email address followed by a single quote in the forgotten email link field. The SQL parser of the web application found the extra quote mark and aborted with a syntax error. When Tom received this error message, he was able to determine that the user input was not being sanitized properly and that the application could be exploited. In this case, he did not need to continue and exploit the application since the error message was proof enough that the application was vulnerable to a SQL injection attack. As a result of this penetration test, the client was able to fix the SQL Server vulnerability.

Finding a SQL Injection Vulnerability

Before launching a SQL injection attack, the hacker determines whether the configuration of the database and related tables and variables is vulnerable. The steps to determine the SQL Server's vulnerability are as follows:

1. Using your web browser, search for a website that uses a login page or other database input or query fields (such as an "I forgot my password" form). Look for web pages that display the POST or GET HTML commands by checking the site's source code.
2. Test the SQL Server using single quotes (''). Doing so indicates whether the user input variable is sanitized or interpreted literally by the server. If the server responds with an error message that says *use 'a'='a'* (or something similar), then it's most likely susceptible to a SQL injection attack.
3. Use the SELECT command to retrieve data from the database

or the INSERT command to add information to the database.

Here are some examples of variable field text you can use on a web form to test for SQL vulnerabilities:

```
Blah' or 1=1--
Login:blah' or 1=1--
Password::blah' or 1=1--
http://search/index.asp?id=blah' or 1=1--
```

These commands and similar variations may allow a user to bypass a login depending on the structure of the database. When entered in a form field, the commands may return many rows in a table or even an entire database table because the SQL Server is interpreting the terms literally. The double dashes near the end of the command tell SQL to ignore the rest of the command as a comment.

Here are some examples of how to use SQL commands to take control: To get a directory listing, type the following in a form field:

```
Blah';exec master..xp_cmdshell "dir c:\*.* /s >c:\directory.txt"--
```

To create a file, type the following in a form field:

```
Blah';exec master..xp_cmdshell "echo hacker was here > c:\hacker.txt"--
```

To ping an IP address, type the following in a form field:

```
Blah‘;exec master..xp_cmdshell “ping 192.168.1.1”--
```

The Purpose of SQL Injection

SQL injection attacks are used by hackers to achieve certain results. Some SQL exploits will produce valuable user data stored in the database, and some are just precursors to other attacks. The following are the most common purposes of a SQL injection attack:

Identifying SQL Injection Vulnerability The purpose is to probe a web application to discover which parameters and user input fields are vulnerable to SQL injection.

Performing Database Finger Printing The purpose is to discover the type and version of database that a web application is using and “fingerprint” the database. Knowing the type and version of the database used by a web application allows an attacker to craft database specific attacks.

Determining Database Schema To correctly extract data from a database, the attacker often needs to know database schema information, such as table names, column names, and column data types. This information can be used in a follow-on attack.

Extracting Data These types of attacks employ techniques that will extract data values from the database. Depending on the type of web application, this information could be sensitive and highly desirable to the attacker.

Adding or Modifying Data The purpose is to add or change information in a database.

Performing Denial of Service These attacks are performed to shut down access to a web application, thus denying service to other users. Attacks involving locking or dropping database tables also fall under this category.

Evading Detection This category refers to certain attack techniques that are employed to avoid auditing and detection.

Bypassing Authentication The purpose is to allow the attacker to bypass database and application authentication mechanisms. Bypassing such mechanisms could allow the attacker to assume the rights and privileges associated with another application user.

Executing Remote Commands These types of attacks attempt to execute arbitrary commands on the database. These commands can be stored procedures or functions available to database users.

Performing Privilege Escalation These attacks take advantage of implementation errors or logical flaws in the database in order to escalate the privileges of the attacker.

SQL Injection Using Dynamic Strings

Most SQL applications do a specific, predictable job. Many functions of a SQL database receive static user input where the only variable is the user input fields. Such statements do not change from execution to execution. They are commonly called static SQL statements.

However, some programs must build and process a variety of SQL statements at run time. In many cases the full text of the statement is unknown until application execution. Such statements can, and probably will, change from execution to execution. So, they are called dynamic SQL statements.

Dynamic SQL is an enhanced form of SQL that, unlike standard SQL, facilitates the automatic generation and execution of program statements. Dynamic SQL is a term used to mean SQL code that is generated by the web application before it is executed. Dynamic SQL is a flexible and powerful tool for creating SQL strings. It can be helpful when you find it necessary to write code that can adjust to varying databases, conditions, or servers. Dynamic SQL also makes it easier to automate tasks that are repeated many times in a web application.

A hacker can attack a web-based authentication form using SQL injection through the use of dynamic strings. For example, the underlying code for a web authentication form on a web server may look like the following:

```
SQLCommand = "SELECT Username FROM Users WHERE
Username = ‘‘‘  SQLCommand = SQLComand & strUsername
SQLCommand  =  SQLComand  &  ‘‘‘  AND
Password = ‘‘‘  SQLCommand = SQLComand
&  strPassword  SQLCommand = SQLComand
& ““““
strAuthCheck  =  GetQueryResult(SQLQuery)
```

A hacker can exploit the SQL injection vulnerability by entering a login and password in the web form that uses the following variables:

Username: kimberly

Password: graves’ OR ‘’=’

The SQL application would build a command string from this input as follows:

SELECT Username FROM Users
WHERE Username = ‘kimberly’
AND Password = ‘graves’ OR ‘’=’’

This is an example of SQL injection: this query will return all rows from the user’s database, regardless of whether kimberly is a real username in the database or graves is a legitimate pass- word. This is due to the OR statement appended to the WHERE clause. The comparison ‘’=’’ will always return a true result, making the overall WHERE clause evaluate to true for all rows in the table. This will enable the hacker to log in with any username and password.

In Exercise 9.1, you will use HP Scramlr to test for SQL injection vulnerabilities.

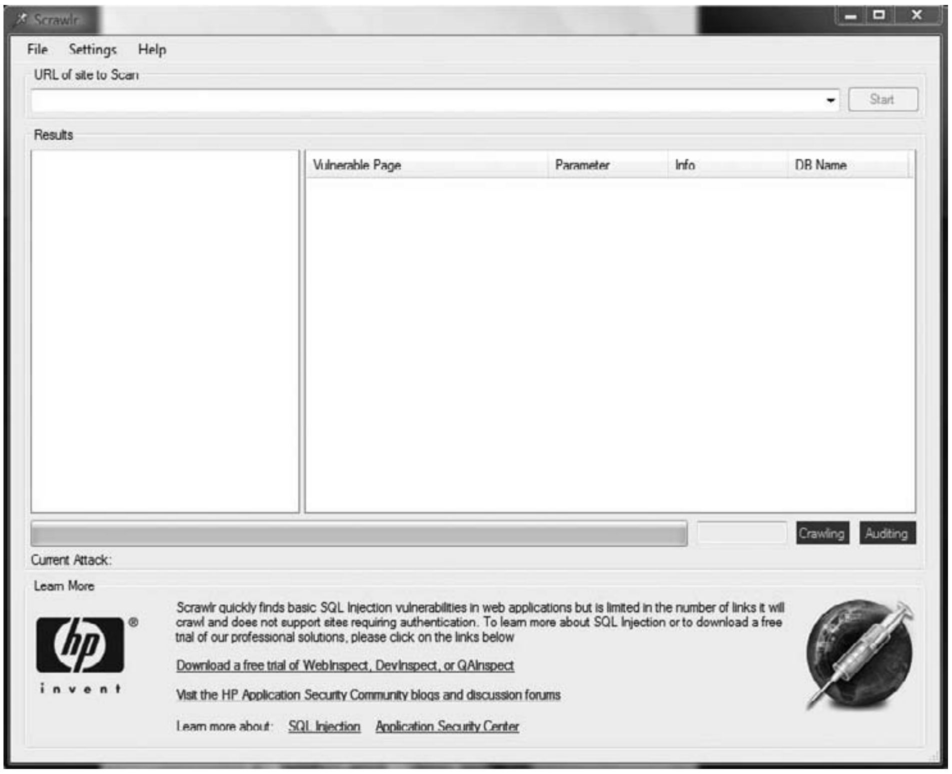
ExErCISE 9.1

Using Hp’s Scrawlr to test for SQL Injection vulnerabilities

1. Download Scrawlr from www.HP.com.
2. Install Scrawlr on your Windows lab PC.

3. Open the Scrawlr program.

4. Type a target web address in the URL Of Site To Scan field:



Scrawlr quickly finds basic SQL Injection vulnerabilities in web applications but is limited in the number of links it will crawl and does not support sites requiring authentication. To learn more about SQL Injection or to download a free trial of our professional solutions, please click on the links below

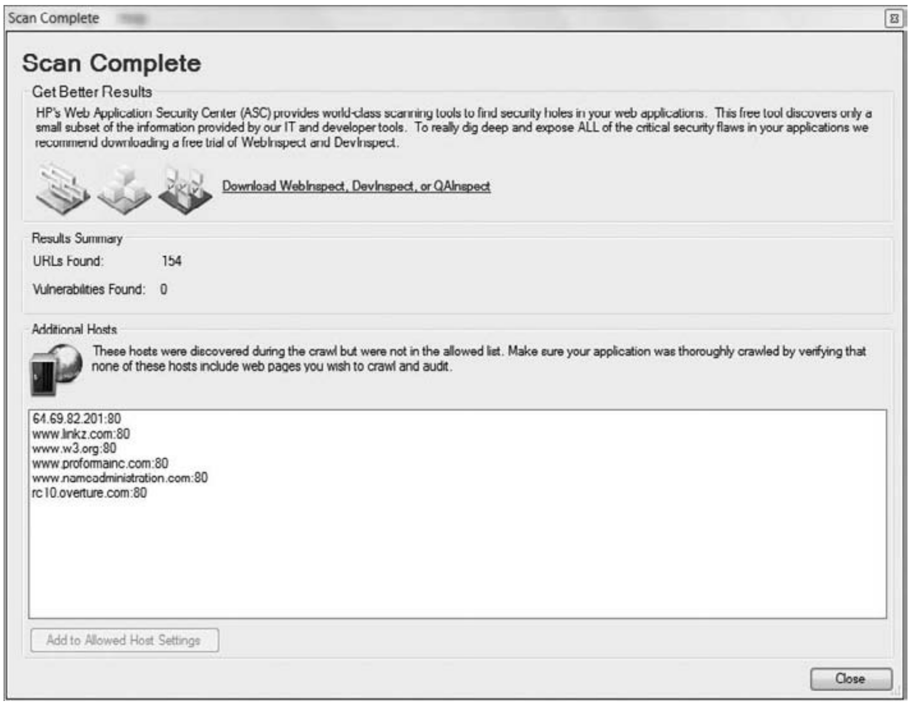
Download a free trial of WebInspect, DevInspect, or QAInspect

Visit the HP Application Security Community blogs and discussion forums

Learn more about: SQL Injection Application Security Center

5. Click the Start button to start the audit of the website for SQL injection vulnerabilities.

6. Once the SQL injection vulnerability scan is complete, Scrawlr will display additional hosts linked from the scanned site. It is a best practice to scan the linked sites as well as the main site to ensure no SQL injection vulnerabilities exist.



Scan Complete

Scan Complete

Get Better Results

HP's Web Application Security Center (ASC) provides world-class scanning tools to find security holes in your web applications. This free tool discovers only a small subset of the information provided by our IT and developer tools. To really dig deep and expose ALL of the critical security flaws in your applications we recommend downloading a free trial of WebInspect and DevInspect.

Download WebInspect, DevInspect, or QAInspect

Results Summary

URLs Found: 154

Vulnerabilities Found: 0

Additional Hosts

These hosts were discovered during the crawl but were not in the allowed list. Make sure your application was thoroughly crawled by verifying that none of these hosts include web pages you wish to crawl and audit.

```
64.69.82.201:80
www.linkz.com:80
www.w3.org:80
www.proformainc.com:80
www.namcadministration.com:80
rc10.overture.com:80
```

SQL Injection Countermeasures

The cause of SQL injection vulnerabilities is relatively simple and well understood: insufficient validation of user input. To address this problem, defensive coding practices, such as encoding user input and validation, can be used when programming applications. It is a laborious and time-consuming process to check all applications for SQL injection vulnerabilities.

When implementing SQL injection countermeasures, review source code for the following programming weaknesses:

Single quotes

Lack of input validation

The first countermeasures for preventing a SQL injection attack are minimizing the privileges of a user's connection to the database and enforcing strong passwords for SA and Administrator accounts. You should also disable verbose or explanatory error messages so no more information than necessary is sent to the hacker; such information could help them determine whether the SQL Server is vulnerable. Remember that one of the purposes of SQL injection is to gain additional information as to which parameters are susceptible to attack.

Another countermeasure for preventing SQL injection is checking user data input and validating the data prior to sending the input to the application for processing.

Some countermeasures to SQL injection are

Rejecting known bad input

Sanitizing and validating the input field

Buffer Overflows

As a CEH, you must be able to identify different types of buffer overflows. You should also know how to detect a buffer overflow vulnerability and understand the steps a hacker may use to perform a stack-based overflow attack. We'll look at these topics, as well as provide an overview of buffer-overflow mutation techniques, in the following sections.

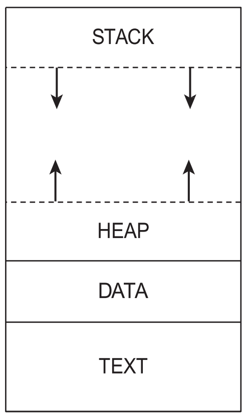
Types of Buffer Overflows and Methods of Detection

Buffer overflows are exploits that hackers use against an operating system or application; like SQL injection attacks, they're usually targeted at user input fields. A buffer overflow exploit causes a system to fail by overloading memory or executing a command shell or arbitrary code on the target system. A buffer overflow vulnerability is caused by a lack of bounds checking or a lack of input-validation sanitization in a variable field (such as on a web form). If the application doesn't check or validate the size or format of a variable before sending it to be stored in memory, an overflow vulnerability exits.

The two types of buffer overflows are stack based and heap based.

The *stack* and the *heap* are storage locations for user-supplied variables within a running program. Variables are stored in the stack or heap until the program needs them. Stacks are static locations of memory address space, whereas heaps are dynamic memory address spaces that occur while a program is running. A heap-based buffer overflow occurs in the lower part of the memory and overwrites other dynamic variables. See Figure 9.1.

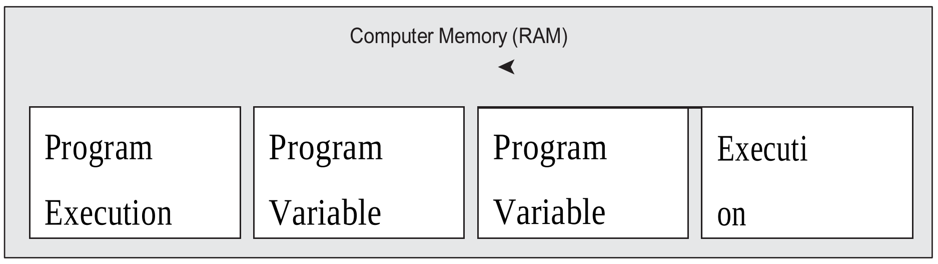
FIgUrE 9.1 Stack versus Heap Memory



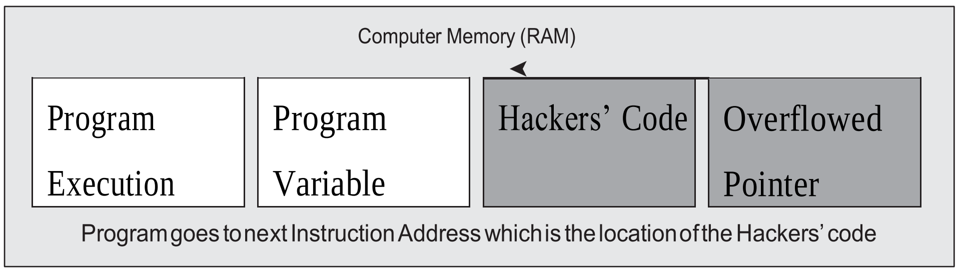
A call stack, or *stack*, is used to keep track of where in the programming code the execution pointer should return after each portion of the code is executed. A stack-based buffer overflow attack (Figure 9.2) occurs when the memory assigned to each execution routine is overflowed. As a consequence of both types of buffer overflows, a program can open a shell or command prompt or stop the execution of a program. The next section describes stack based buffer overflow attacks.

Figure 9.2 A stack-based buffer overflow attack

Normal Program Memory Stack



Buffer Overflow Memory Attack



To detect program buffer overflow vulnerabilities that result from poorly written source code, a hacker sends large amounts of data to the application via a form field and sees what the program does as a result.

The following are the steps a hacker uses to execute a stack-based buffer overflow:

1. Enter a variable into the buffer to exhaust the amount of memory in the stack.
2. Enter more data than the buffer has allocated in memory for that variable, which causes the memory to overflow or run into the memory space for the next process. Then, add another variable, and overwrite the return pointer that tells the program where to return to after executing the variable.
3. A program executes this malicious code variable and then uses the return pointer to get back to the next line of executable code. If the hacker successfully overwrites the pointer, the program executes the hacker's code instead of the program code.

Most hackers don't need to be this familiar with the details of buffer overflows. Prewritten exploits can be found on the Internet and are exchanged between hacker groups. Exercise 9.2 walks through using Metasploit to perform a Buffer Overflow attack.

NOTE The memory register that gets overwritten with the return address of the exploit code is known as the EIP.

ExErCISE 9.2

performing a Buffer Overflow Attack Using metasploit

1. Open the Metasploit Framework.
2. Start the test machine running Windows Server with IIS.
3. From Metasploit, run the IIS Buffer Overflow attack against the test machine running IIS.
4. Choose a payload to deliver to the IIS target system via the buffer overflow exploit.

Buffer Overflow Countermeasures

As you can see, hackers can graduate from standard buffer overflows to redirecting the return pointer to the code of their choosing. A hacker must know the exact memory address and the size of the stack in order to make the return pointer execute their code. A hacker can use a No Operation (NOP) instruction, which is just padding to move the instruction pointer and does not execute any code. The NOP instruction is added to a string before the malicious code to be executed.

If an Intrusion Detection System (IDS) is present on the network, it can thwart a hacker who sends a series of NOP instructions to forward to the instruction pointer. To bypass the IDS, the hacker can randomly replace some of the NOP instructions with equivalent pieces of code, such as x++,x-;?NOPNOP. This example of a mutated buffer overflow attack can bypass detection by an IDS.

Programmers should not use the built-in strcpy(), strcat(), and stradd() C/C++ functions because they are susceptible to buffer overflows. Alternatively, Java can be used as the programming language since Java is not susceptible to buffer overflows.