CHAPTER 11

Performing a Web Site Vulnerability and Security Assessment

SECURITY TESTING is an absolute requirement for Web servers and Web applications. When you are performing Web application security assessments, there are multiple components to consider to adequately identify and remedy risks. There are also many tools, both freeware and commercial, available to perform security and vulnerability testing. The most accurate method will involve using multiple programs and manual techniques. In addition to selecting and using the right tools, it's equally important to plan the security assessment properly.

Some of the techniques and utilities mentioned in this chapter may be interpreted by systems administrators or security monitoring systems as intrusive or hostile. The techniques and utilities may also require administrator or root-level access to the system to successfully run or give the most accurate results. Be sure to have appropriate authority or permission, as well as the necessary access to the system, prior to performing any type of vulnerability or security assessment on a system.

Chapter 11 Topics

This chapter covers the following topics and concepts:

- What the difference is between software testing and Web site vulnerability security assessments
- How to perform an initial discovery on a targeted Web site
- How to perform a vulnerability and security assessment
- How to use planned attacks to identify vulnerabilities
- How to spot vulnerabilities in back-end systems and Structured Query Language (SQL) databases
- How to prepare a vulnerability and security assessment report
- What best practices for Web site vulnerability and security assessments are

Chapter 11 Goals

When you complete this chapter, you will be able to:

- Determine the difference between software testing versus Web site vulnerability and security assessments
- Perform initial discovery activities on a Web site
- Perform a vulnerability and security assessment on various Web site components
- Perform a planned attack to determine vulnerabilities of user input forms and screens on front-end systems
- Perform a planned attack on back-end systems and SQL databases
- Prepare a vulnerability and security assessment report
- Understand some of the best practices to use when performing Web site vulnerability and security assessments

Software Testing Versus Web Site Vulnerability and Security Assessments

Differentiating between software testing and security assessments can be difficult. Generally, software testing is a much broader set of procedures than Web site security assessments. In many cases, software testing includes an assessment of Web site security as a subset of the overall testing process.

Software testing often includes, at minimum, checks to ensure an application of any type:

- Meets the initial design requirements provided by the party requesting the application; referred to as verification and validation
- Operates as expected and without any errors
- Can be implemented so that it does not cause issues with other applications it may integrate with; referred to as compatibility

Depending on the software development life cycle methodology used, there can often be additional steps or phases in the testing process.

This chapter focuses on assessing the security of a specific type of application, a Web site, and its various parts and pieces. Web sites typically consist of four elements:

- Web server software, such as Microsoft's Internet Information Services or Apache HTTP Server
- A hardware server and operating system that the Web server runs on
- A software application that uses the Web server to collect or distribute information
- A database that stores the information being used by the application and/or Web server

A common implementation of these Web site elements involves three *tiers* or layers of hardware and software. These layers consist of a presentation tier with the Web server and its hardware, an application or logic tier with the software application and its hardware server, and a database tier which includes the database software and its hardware server. Although these three tiers are often implemented on physically separate hardware servers for optimal security, they represent logical parts of the overall Web site platform. In situations where software capability doesn't support three separate servers, or where budget might not allow for multiple servers, the tiers can be installed together on shared servers.

Performing an Initial Discovery on the Targeted Web Site

The first step in a Web site assessment is to identify the components that make up the Web site and that will be tested. In security terms, this discovery activity is also referred to as **fingerprinting** and **enumeration**—identifying and listing various components of a Web site platform that need to be tested or attacked. A variety of tools and techniques determine the following types of elements:

- Internet Protocol (IP) addresses associated with the Web site platform
- Services and/or applications that are running on the servers in the Web site platform, for example, Hypertext Transfer Protocol (HTTP), Domain Name System (DNS), File Transfer Protocol (FTP), Telnet, and Simple Mail Transfer Protocol (SMTP)
- The operating systems on all hardware servers supporting the Web site platform
- Any known (published) vulnerabilities with the services, applications, or operating systems

Both commercial and freeware programs are available to perform discovery activities. Commercial tools are generally more efficient and feature-rich than freeware tools. However, this chapter focuses on freeware tools that are sufficient for performing a vulnerability and security assessment.

Ping Sweep

Ping is a utility that was written for the IP protocol in 1983. Ping was designed to send a packet to an IP address to determine if it's active. If so, the utility measures the round-trip time of the packet that was sent and received. Because a ping serves much the same purpose as the sonar ships use, it was named after the characteristic "ping" noise that is made when sonar identifies an object in water. The ping command is typically executed against a single IP address; a **ping sweep** is the act of running the utility across a range of IP addresses.

NOTE

An active IP address is an address that's in use. An inactive IP address may mean the system is turned off or the IP address simply isn't assigned to a system.

Ping sweeps, sometimes called *host discovery*, are often the first step in a security assessment or attack because they can save time and effort by narrowing the number of IP addresses to assess/attack. Many other security testing tools are more intricate and exhaustive in the tests they run on each targeted IP address or system. To avoid wasting time trying to test or exploit an inactive IP address, ping sweeps are a valuable technique. The ping command is simple—it sends a packet to an IP address and waits a short time for a response. Scanning even an entire class C subnet, involving up to 254 hosts, can usually be done in less than 10 seconds depending on network speed.

Because the ping command was designed to communicate with a single IP address, it's necessary to use utilities specifically designed to ping sweep multiple IP addresses. Many freeware ping sweep utilities exist for nearly all major operating systems; here are some examples:

WARNING

Almost any intrusion detection system (IDS) will immediately detect a ping sweep and warn administrators, depending on its configuration. IDSs may take active measures to stop your scanning or prevent results from being relayed back to the scanner. Consider this if returned results are different than expected. Scanning should be done with approval by an organization's security administrators.

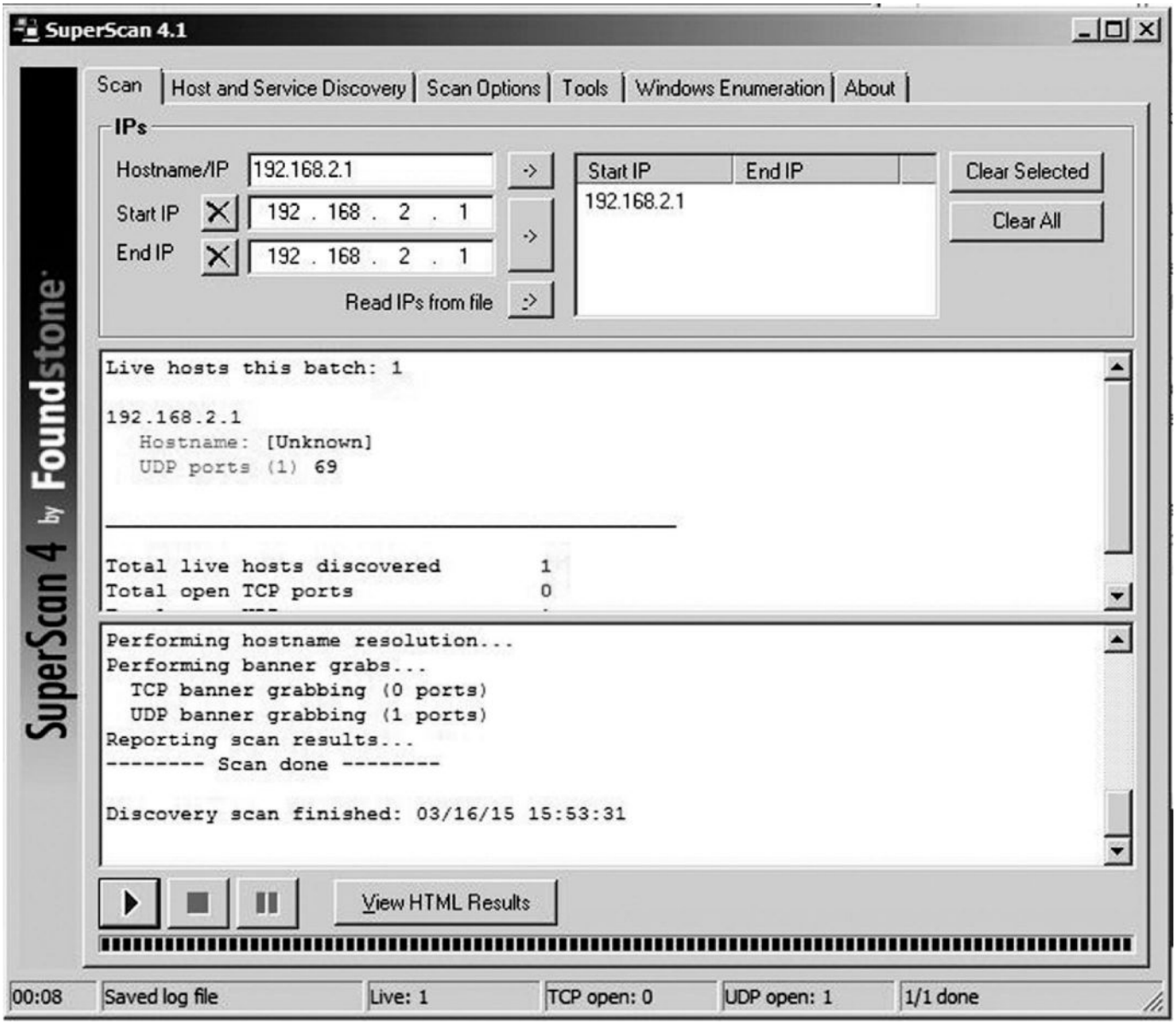
- **Windows**—Pinkie, IP Address Tracker by SolarWinds, SuperScan by McAfee, and Zenmap, a Windows version of Nmap with a graphical interface
- **UNIX/Linux**—Hping and Nmap

The utilities tend to offer easy-to-use, intuitive interfaces, such as the SuperScan interface shown in Figure 11-1.

Ping sweep capabilities are now included with many vulnerability scan and security assessment tools. Performing ping sweeps from within these utilities eliminates the need to reenter active IP addresses into the vulnerability programs.

The exact output received from a ping sweep utility depends on the version of the utility used. However, at a minimum, the output will include the IP address and an indication as to whether the IP address is considered "live." Most of the above-mentioned utilities allow you to print the output, or you can save it and then use Microsoft Word or Microsoft Excel to remove all non-responsive IP addresses.

FIGURE 11-1 The SuperScan ping sweep utility.



Nmap

Nmap, or Network Mapper, was developed in the late 1990s as a Linux port scanning tool to identify which IP ports, or services, were active on an IP address. Opinions are mixed as to whether the original intent of Nmap was to help systems administrators or hackers gain information about systems and networks. The result is that Nmap is one of the oldest and most widely used tools by security professionals and hackers alike. The tool identifies and assesses services and vulnerabilities on computer systems.

Nmap's core functionality consists of three features: ping sweeping, port scanning, and operating system (OS) detection. One of the most powerful and controversial features of Nmap, however, is its Nmap Scripting Engine. This scripting engine allows users to customize such things as how aggressively Nmap performs its scans and which IP port and services it probes. There are over 100 scripts included with the Nmap utility, and users familiar with the Lua programming language can create their own.

To use Nmap as a command-line utility, which is common for the UNIX and Linux versions, a user needs a strong fundamental knowledge of both Nmap's options as well as UNIX or Linux commands. For the more novice user, the Windows version of Nmap, known as Zenmap, has a graphical interface and is much easier to configure and execute. Although there are hundreds of scanning and probing options available in the utility, this section focuses on Nmap's abilities to discover and enumerate hosts. Additional Nmap capabilities will be discussed later in this chapter.

The "standard" Nmap scan performs non-intrusive activities—those unlikely to impact the server or appear to security systems as an attack, such as:

TIP

For the best results from Nmap, run the utility from a computer on the same network segment as the Web site server(s) being assessed. Nmap has a few options that either don't execute well or don't report properly if Nmap is run from the same machine it's trying to scan.

- Basic ping operations, such as whether the IP address is alive/up or down/non-responsive or total time for the ping packet to travel round-trip
- The Media Access Control (MAC) address of the network card using the IP address that was scanned
- Open/active or closed/inactive status of the 1,000 most commonly used Transmission Control Protocol (TCP) ports

Although Nmap was originally written for Linux, the Zenmap Windows version of Nmap is easy to use and has some advantages. For those who don't use Nmap frequently enough to memorize all the command-line options, Zenmap allows users to click on desired scanning options. However, it also has a field that displays the command line that corresponds to the options that have been selected, as shown in Figure 11-2. The command-line feature makes Zenmap useful for learning Nmap.

OS Fingerprint

Another important part of finding Web site vulnerabilities is identifying the operating system running on the hardware systems that support the Web server, application, and database components. This activity is known as *OS fingerprinting*. Operating systems are designed differently and have unique vulnerabilities. Identifying the underlying operating system of a Web site can help determine which tools will be needed to further assess the server, the skills required for manual assessment techniques, and even the types of attacks that should be anticipated. As with ping sweeps and basic discovery, numerous tools provide OS fingerprinting. Many of those tools perform other security assessment functions in addition to OS fingerprinting, such as ping sweeping, port scanning, and vulnerability testing. Two of the most popular OS fingerprinting tools are Nmap and Nessus.

FIGURE 11-2 The Zenmap interface.

Nmap returns the OS fingerprint as part of a "regular scan" profile selected in Zenmap, or it can be specifically requested by executing the following command:

```
nmap -O ip address
```

The result will be a line in the output that resembles the following:

```
OS details: Linux 2.6.19 - 2.6.31
```

Nessus Vulnerability and Port Scan

Like Nmap, Nessus began as a free, open source application. Because it was created in the late 1990s, around the same time that network security was gaining in importance, it quickly became one of the most popular

vulnerability scanning tools for security professionals. Although there are a few feature overlaps between Nessus and Nmap, they are largely considered complementary tools. Where Nmap's strengths lie in its ability to probe and identify systems and their services, Nessus is equally proficient at testing those systems and services for vulnerabilities. Nessus is now a commercial product for use in corporate environments, but it still is offered free to home users scanning personally owned systems.

Nessus is primarily comprised of two components: a scanning engine and thousands of plug-ins that associate vulnerabilities with items such as services, operating systems, applications, and so on. To scan a system, a user first creates a policy, which means choosing a set of options and plug-ins to use during the scan. After creating a policy, the user initiates the scan by entering information about the host system to be scanned and then choosing the policy against which to assess that host or hosts. Figure 11-3 shows the summary screen of a normal Nessus vulnerability scan.

NOTE

Some of the features that the commercial version of Nessus offers over the free home edition are customized scans for certain compliance acts like Payment Card Industry (PCI), "policy" testing such as password strength, and analysis of data for sensitive elements such as Social Security numbers.

FIGURE 11-3 Summary page from a Nessus vulnerability scan.

Nessus also has the ability to perform both authenticated and unauthenticated scans; unauthenticated is the default. The difference between the two methods of scanning can be very important depending on how accurate the results need to be. Unauthenticated scanning means Nessus does not have permission to log onto the target system while scanning it. Nessus determines which vulnerabilities to report largely by first identifying the services present by the IP ports open and then searching the plug-ins to see which vulnerabilities are associated with those services and ports. In some cases, this leaves room for error because Nessus cannot confirm the exact configuration of the application or port and whether it already addresses its vulnerabilities. By configuring Nessus with the appropriate logon credentials, it can attempt to validate some of its findings by running additional tests that are only possible while being logged on.

Performing a Vulnerability and Security Assessment

To assess the vulnerabilities and security of a Web site application, you must first identify the components of the Web site environment. A typical single-server Web site consists of:

- **A Web server OS**—The operating system of the hardware server that the components reside on
- **A Web server application**—The actual application that is collecting, using, and/or providing data

- **A Web server front end**—The Web server software that presents the application to users in the form of HTTP pages
- **Web site forms**—The input fields, or forms, that are used to gather data from users

> **WARNING**
>
> It's worth mentioning again that the scan and assessment applications covered in this chapter, as well as many other security tools and commands, are very powerful and can harm systems, depending on how they are configured and used. Users should not attempt this type of testing without proper authorization and planning. If possible, testing should first be performed on mirrored, backup, or test systems to determine the likely impact.

Because each of these components is unusual in purpose and design, it's important to assess them separately, even if common tools or techniques are used. This is especially true for Web sites where the components reside on separate, dedicated servers.

The following sections discuss tools and techniques that are commonly used to assess each component for vulnerabilities and other security concerns. Because this chapter cannot cover every security program, or their configuration options, it describes the desired activities and results of a security assessment. The tools mentioned above, as well as other commands or techniques, may be referred to illustrate the process.

Web Server OS

Because the server operating system is like the foundation of a house, supporting all the other components, it's important to identify and assess the OS version and its running services accurately. At minimum, an assessment of the Web server OS should involve:

- Identifying the OS type and version
- Identifying major service packs and patches that have been installed
- Identifying the active services or ports being supported by the OS
- Identifying any known vulnerabilities associated with the components above

An unauthenticated scan is usually acceptable for OS assessment, but if privileged credentials are available, it's recommended that both an unauthenticated and an authenticated scan be performed.

A typical enumeration and assessment of the Web server OS might be as follows:

1. If the IP address of the Web server is unknown, or if multiple Web servers are suspected present on the network, run an Nmap scan with the "Quick" profile selected against the entire IP address range. Any IP addresses reported as having ports 80 and/or 443 open should indicate they are Web servers.
2. Once the Web server IP addresses are known, run a vulnerability scanner against those IP addresses. If using Nessus, a scan with the default plug-ins enabled would suffice. This will produce a report that includes information about the OS and any vulnerabilities associated with the OS and other running services.
3. If privileged credentials are available for the OS, such as a Windows administrator-equivalent logon or a UNIX root-equivalent ID, it's recommended to run an authenticated vulnerability scan. Depending on the scanner being used, this could help identify vulnerabilities with weak password policies, domain relationships, and so on.
4. Manually review the available services that are enabled at the Web server OS level and determine if they are necessary. Unnecessary services, even if found to have no vulnerabilities, can present a future risk to the system.

Arguably, the most valuable information gained from a Web server OS scan is whether the operating system is missing any critical security patches or running any insecure services. Patches are typically easy to address by applying patches that are missing and recommended by the scan program. Insecure services can be more difficult to remedy if they are truly insecure by design but are needed for business purposes. An example is Telnet services that allow users to remotely connect or logon to the Web server. Often, this is reserved for administrators who need Telnet services to manage the Web server's operating system. However, Telnet is insecure in that it does not require authentication and transmits data in plaintext. This includes transmitting logon IDs and passwords in

plaintext even when Telnet is configured with authentication enabled. Some common remediations include limiting Telnet capabilities to certain workstations or implementing a more secure equivalent, Secure Shell (SSH).

Additional items to look for when scanning the Web server OS include:

- Insecure Simple Network Management Protocol (SNMP) configurations that use default names or strings and may allow an attacker to intercept server monitoring traffic.
- Weak passwords or password policies. Testing actual passwords typically requires using a separate password auditing/hacking utility.
- Services that allow remote connectivity to the Web server OS such as Telnet, SSH, rlogin, and so on. If present, use SANS Institute (*http://www.sans.org*) or the Carnegie Mellon University Computer Emergency Readiness Team (*http://www.cert.org*) Web sites to research vulnerabilities and attacks related to those services.

Web Server Application

Assessing Web server applications can be a complex task because applications are unique and offer a variety of services. A Web server application can be anything from a grouping of scripts to a fully custom-coded program. In many cases, assessing Web server applications consists of scanning the application for known vulnerabilities as well as performing more specific testing of the code itself.

Vulnerability scanning and security assessment of Web server applications are much like assessments of the Web server OS and front end. Depending on the type of Web application on the server, tools like Nessus or Metasploit may be useful in finding and testing vulnerabilities in the application. However, for traditional custom-coded applications, it's recommended that the code be reviewed by a commercial source code assessment tool such as WebInspect or IBM Rational Appscan. These source code tools examine various programming languages and find poorly designed or insecure coding techniques that could present risks, such as:

- The ability to circumvent the application's authentication process
- Code or command injection that forces the application to perform in a certain manner
- Manipulating uniform resource locators (URLs) and/or data input fields to traverse data directories or the application itself

As with the assessment of the Web server OS, it's recommended that both an authenticated and unauthenticated test of the Web server application be performed. This will identify potential risks that might be exploited by either an outside attacker or an authorized user. The following steps are typical of a basic Web application security assessment:

1. Identify the type of application that is running on the Web server, such as Microsoft SharePoint, an e-mail system, or a form-based front end for a database.
2. Research the application to determine the types of code or scripting languages being used, such as C++ or Java.
3. Select an appropriate utility to run against the Web application or scripts, such as N-Stalker. (See *http://www.nstalker.com*.)

Web Site Front End

Assessing Web site front-end software, the program that serves HTTP pages to users, is very similar to assessing OS software. Like an OS, Web site software typically consists of a core program combined with add-on services to support specific capabilities. In many cases, the vulnerability scan of the Web server OS will identify Web site front-end software and its vulnerabilities as part of its overall enumeration. However, depending on the scan tools used, there are typically some Web site-specific options that can help the scan application perform a more thorough assessment of Web site front-end vulnerabilities.

Because the Web site front-end software often integrates or supports the Web server application, many of the tools used to assess the application also can be used to assess the front end. Often, the Web site application consists of Web pages or forms that are designed in Java or PHP. Web site front-end software is often the platform for more popular but sometimes more exploitable technologies such as Flash, Shockwave, and others. These technologies are often targeted by exploits because they are used so heavily in Web sites and applications.

Some common activities performed when assessing Web site front-end software are:

1. Identify the type of Web site front-end software in use, such as Microsoft Internet Information Services (IIS) or Apache.
2. Determine the functions that the front-end software is supposed to provide—simple presentation of Hypertext Markup Language (HTML) pages, data input/output, access to sensitive files/data, and so on.
3. Run a Web server security utility such as Nessus, N-Stalker, or Acunetix Web Vulnerability Scanner. In addition to looking for basic vulnerabilities, the scanners should be configured to test for cross-site scripting.
4. Use a utility such as HTTrack Website Copier to crawl, or scan, the Web site's pages for hidden fields. Hidden fields are sometimes used to track the activities or actions of the Web site user.
5. HTTrack can also provide information about the Web site's directory structure. This can be useful in performing a directory traversal attack in which the user attempts to access hidden directories by simply changing the URL in the Web browser address bar.
6. Use resources such as Google's advanced search commands to search the Web site for potentially sensitive files or information. Some examples of useful Google search commands that can be entered directly into the Google search field include:
   - **ssn site:acme.com**—Searches the site Acme.com for any Web pages that include the text "ssn"
   - **acme.com -filetype:doc**—Searches the Acme.com site and returns all files with .doc extensions stored on the Web site

Web Site Forms and User Inputs

Web site forms allow Web site users to input data either into the Web site front end or the Web site application. Although data input forms and fields are useful, they present a significant security risk to a Web site if not implemented properly. When a user enters data into a Web site field or form, there are usually scripts and programs that process that data. Depending on how well the Web site front end or applications have been configured, or are designed, they may have problems if a user enters bad or improperly formatted text.

There are a few ways that input can be incorrectly entered, causing Web site problems. The most common involves entering a very large number of characters into a field that might expect only a few. For example, an attacker tries to enter 100 characters into a phone number field. A properly configured, or well-written, Web site application would either require the correct number of characters before processing the data or check for incorrect data and not process it.

Another form of improperly formatted data involves entering special commands into fields. This is often known as code injection or cross-site scripting (XSS). Cross-site scripting has long been one of the most widely exploited vulnerabilities. It's popular because it takes advantage of two common features of Web sites: JavaScript and data input fields/forms. The term stems from the nature of the technique: An attacker accesses a Web site and provides data in the site's fields that cause issues with how JavaScript interprets the data and/or executes it. This produces results ranging from wrong characters or data being displayed to compromising the security of Web browser sessions that interact with the Web site.

Most vulnerability scanners, commercial or free, can perform at least a basic scan for cross-site scripting and code injection vulnerabilities. Because there are so many types of cross-site scripting and code injection vulnerabilities, it's recommended that multiple scan programs be run against a Web site.

Incorporate PCI DSS for E-commerce Web Sites

One of the most common uses for Web sites is e-commerce. Web sites that store, transmit, or process credit card numbers must comply with security standards that have been set by the payment brands—Visa, MasterCard, American Express, Discover, and JCB. These standards are known as the Payment Card Industry Data Security Standards (PCI DSS). Although this chapter won't cover PCI DSS in detail, it's worth noting the sections within PCI DSS that apply to Web site security testing. Failure to comply with PCI DSS can lead to a Web site's losing the right to process credit card transactions as well as to fines for the Web site's owner by the payment brands.

PCI DSS consists of 12 major security requirement sections, each with numerous specific controls. These major requirement sections are:

1. Install and maintain a firewall configuration to protect cardholder data.
2. Don't use vendor-supplied defaults for system passwords and other security parameters.
3. Protect stored cardholder data.
4. Encrypt transmission of cardholder data across open, public networks.

5. Use and regularly update antivirus software.
6. Develop and maintain secure systems and applications.
7. Restrict access to cardholder data by business need-to-know.
8. Assign a unique ID to each person with computer access.
9. Restrict physical access to cardholder data.
10. Track and monitor all access to network resources and cardholder data.
11. Regularly test security systems and processes.
12. Maintain an information security policy for employees and contractors.

For Web sites that handle credit cards, all 12 requirements will apply. The requirements most related to this section are:

- **Requirement 2, vendor-supplied defaults**—Although changing vendor-supplied default passwords and account names is largely a manual process, some advanced vulnerability scan tools can check for well-known vendor default values. If a Web site must be PCI compliant, then it's recommended that a security assessment tool be used that can check for vendor default accounts and passwords.
- **Requirement 6, secure systems and applications**—Some controls in this requirement are most relevant to this chapter in that they require tests for specific types of issues, such as input validation, cross-site scripting, data injection flaws, malicious code execution, and error handling.
- **Requirement 11, testing systems and processes**—This requirement contains the mandate that Web applications and systems be regularly scanned for vulnerabilities, from both inside the network where the server sits and from the Internet. Internet-based scans must be performed by an approved third-party scanning vendor whose scanning product tests for specific vulnerabilities determined by the payment brands. For a Web system to be eligible for PCI compliance, both the internal and third party/external scans must be performed and successfully passed quarterly, meaning no vulnerabilities are found. To be deemed compliant, the system and environment also must be assessed and meet all the other controls in the PCI DSS standard.

Using Planned Attacks to Identify Vulnerabilities

One of the most effective ways to complement vulnerability scanning and comprehensively assess the security of a Web system or application is to perform a planned attack. Many security professionals refer to this as **penetration testing**, or *pen testing* for short.

The specific activities performed during planned attacks vary depending on the skill of the tester, the type of system, and the desired outcome. However, the commonly used process consists of three steps:

1. Developing a plan of attack
2. Identifying the security gaps and holes
3. Attempting to escalate privilege

A good site to reference when planning attacks on applications and Web technologies is the Open Web Application Security Project (OWASP) at *http://www.owasp.org*. OWASP has been helping set application security standards and practices for years. Similar to SANS and CERT, OWASP tracks data on evolving attack methods, common security coding issues, and so on. By researching how application security should work, per OWASP, an assessor can develop ideas and techniques for how to pen test a specific application or system.

Develop an Attack Plan

The attack plan for Web front-end systems often centers on the application's user interface and how a Web site visitor, legitimate or not, could gain unauthorized access to data through the Web server or application. If system availability is an important issue or if the Web server is part of a larger group of systems, then the tester might also incorporate attacks such as denial of service (DoS) and/or deeper reconnaissance attacks. To keep the planned attack efficient and focused, it's important to establish the exact strategies that will be attempted as well as possible outcomes. A basic planned attack on a Web site front end might include:

- Strategy: Attacking input fields and forms in an attempt to gain unauthorized access

- Systems: Web site Acme.com
- Techniques: Input invalid data into fields, attack authentication form, and attempt buffer overflow

Identify Gaps and Holes

Earlier scans with vulnerability programs should have produced information on the OS version, type of Web software running, patch levels, and perhaps even configuration settings. Those pieces of information provide insight into possible security gaps.

technical TIP

While you are viewing a Web page, you can see the source code by selecting View, Page Source, or just Source, from the browser's menu bar. Performing a search on the text "type=hidden" should identify whether any hidden fields exist on the Web page. Depending on the nature of the field, it may be possible to modify files or gain access to resources by manipulating the hidden field values.

Another technique to identify Web front-end security gaps is to review the HTML source code for hidden fields. A Web developer might want to calculate or store data on a Web page that the application needs but users shouldn't see. For example, an online sales order form might use hidden fields that contain price and quantity information about products being purchased. Upon checkout, the Web site might use those hidden fields to calculate the final order total and number of items purchased. If the Web site and application are not properly secured, an attacker could discover the hidden fields, manipulate the values, and place a fraudulent order with lower prices.

Escalate the Privilege Level

Privilege escalation involves exploiting a vulnerability or flaw in a system to gain access to resources not otherwise available to the attacker or tester. Privilege escalation applies not only to gaining a higher role in the system, known as *vertical privilege escalation*, but also to gaining access to files or data that normally are restricted to peer users. The latter scenario, known as *horizontal privilege escalation*, may be much easier for an attacker to accomplish than vertical privilege escalation yet provide a comparable benefit.

One scenario used to test for horizontally escalated privilege levels involves visiting Web server pages where documents, files, or other pieces of information are requested, perhaps a secure area where user-specific documents are stored. A sign of potential vulnerability can be a URL that looks like:

```
http://www.acme.com/showmydocuments.asp?DocID=13
```

A valid test would be simply to change the "13" in the URL to other numbers and see if documents are accessible that normally shouldn't be—perhaps documents belonging to other users. This test actually uses two pen testing techniques—URL manipulation and directory traversal.

Another, perhaps more popular, form of attempted privilege escalation is buffer overflow attempts. By sending improperly formed information to the Web site or application, it's sometimes possible to overflow the computer's memory and cause the application to crash. Most buffer overflow attacks result in the Web application and system simply shutting down or rebooting. However, with improperly secured Web systems, it's sometimes possible that the server will close the Web application but keep the applications logon session, and rights, active. This has led to occasions where the Web system, upon closing the application, returned to a command prompt that had full administrative rights still active. This meant that if attackers could crash the application simply by flooding the computer's memory, their level of privilege could be escalated once they were at the console prompt.

Spotting Vulnerabilities in Back-End Systems and SQL Databases

Back-end systems are also subject to risk if not properly secured. Because the back-end databases of a Web application solution typically don't offer the same variety of services that a front-end system does, planned attacks may be easier. The strategy for attempting to compromise back-end systems is relatively the same, though. The intention is to gain access to data, either through compromising a database or escalating a privilege level.

Develop an Attack Plan

Developing an attack plan for a back-end system or database is very similar to developing one for a Web front-end system. Many attack methods can be performed through the same Web application or forms.

The most likely difference in planning an attack on a database is that additional discovery tools need to be used to identify the database type. Most scanning tools identify database types based on open ports or services running on the system. However, if the database type cannot be determined by scan tools, then another method that sometimes works is field manipulation and forcing error messages. Inputting a wrong value in a field the database processes may cause an error. The resulting error message displayed on the screen may give information about the database.

Once the database type is identified, the general plan of attack on a database might attempt to:

1. Access or retrieve data by injecting data into fields or forms
2. Access or retrieve data by gaining privileged access
3. Crash the database to gain privileged access to other portions of the system

Identify Gaps and Holes

Probing database applications for gaps and holes is a common feature of the penetration testing software on the market. Metasploit, a freeware tool, is adept at exploit testing, as its name implies. Metasploit has modules, or preconfigured test scripts, for numerous database types and their vulnerabilities, but it isn't as user-friendly as some other tools mentioned in this chapter. Even the Windows version runs as a command-line utility.

Many of the Web application scanners also have scripts and tests for back-end databases. Acunetix is one of those programs and will attempt to connect to the database as well as run Structured Query Language (SQL) commands.

Escalate the Privilege Level

Attempting to escalate privileges for back-end databases is very similar to the strategy used for Web server applications. Once again, Metasploit is a good utility for attempting various strategies for escalating privileges. One item to be cautious about, however, is that many of the scripts and techniques used to gain escalated privilege level involve strategies to crash either the application or the database itself. If the database is crashed, then accessing data may be considerably more difficult, even if privileged access is gained.

Perform an SQL Injection for Data Extraction

Most scanning and pen test utilities now have built-in SQL injection testing capabilities. However, SQL injection is a fairly simple activity that an attacker can attempt manually by manipulating URLs in a browser.

SQL injection is the act of inserting various SQL commands into a URL, or sometimes into a form field, so that the command will be run against the back-end database. The steps below contain a basic SQL injection attempt:

1. Look for Web pages that contain data entry fields, for entering data, such as a username or password. Web pages for that purpose are typically not static HTML but written using a language like ASP or PHP. These Web pages will contain extensions such as .asp, .php, or .jsp somewhere in the string of characters. For example, `http://mydatabase.com/index.asp?user=` might correspond to a text box asking for a logon ID.
2. Upon typing the name Sam into the logon ID field on the Web page, the string would look like: `http://mydatabase.com/index.asp?user=Sam`. The database might then start a query to look for the name Sam in its tables, and the actual database query might look like: `SELECT*FROM customers WHERE User='Sam'`
3. By either manipulating the URL or entering data into the field, you can now try some characters that have special meaning in SQL queries such as a single quote ', or two dashes --. The single quote, for example, tells SQL to escape from the search criteria and back to the SQL statement. Knowing this, you can enter data to the Web page's field to be injected as part of the SQL statement. For example, entering the characters `' OR 1 = 1` tells the statement to return data if statement processes are true. Because 1 always equals 1, the statement is true and all data is returned.
4. From here, more advanced knowledge of SQL query formatting and/or SQL server commands are necessary to get creative with SQL injection. Basically, though, any type of SQL command can now be inserted into the text box or URL and, if properly formatted, cause the SQL server to execute the command as if the attacker was sitting at the SQL console. For example, the following text would cause SQL to stop its normal query and execute anything after the semicolon, such as a privileged SQL server exec command: '; `exec`....

With some quick Google searching, an attacker could find information on the exec command and its options allowing for some very powerful capabilities. Exec can then be used to perform actions such as:

- `EXEC xp_cmdshell 'dir *.exe'` which returns a list of all .exe files on the server
- `EXEC @retstat...`, which runs a script, or collection of commands, from a remote server

Preparing a Vulnerability and Security Assessment Report

Reporting can be the most difficult part of performing a vulnerability or security assessment. Often, there are numerous audiences—with varying levels of technical knowledge—that need the results of the assessment. In some cases, there may be a need to present the data from different perspectives, such as a report focused on risk versus one focused on compliance. The general structure for an assessment report includes:

- Executive summary
- Summary of findings
- Details of the vulnerability assessment
- Details of the security assessment
- Recommended remediations

Many of the vulnerability and security assessment tools on the market, particularly the commercial programs, have the ability to generate data and graphs, if not an entire assessment report. However, it's often necessary to take the data from those tools and manually build an assessment report that will best suit the audience. The next few sections will discuss best practices for manually creating an assessment report.

Executive Summary

An executive summary is typically designed to do what its name says—provide executives, or other levels of management, the ability to understand the assessment's major points. Usually the first section or chapter of an assessment report, an executive summary should be relatively short and focus on the findings of most interest. This typically includes "critical" or "high" vulnerabilities and security gaps as well as any that might affect the system's compliance status—such as being PCI compliant/certified.

A good executive summary will include information that is specific and clear so it can be quickly consumed and understood. It should also include points management needs to know to make business decisions about risk, budgeting, resource planning, and so on. It's perfectly acceptable for an executive summary to include detailed information, or even advanced topics and terms, as long as it reads quickly and gets its points across clearly. Many executive summaries are no longer than a single page and contain graphs and charts to consolidate information and help focus the reader's attention.

technical TIP

In today's world of security and compliance, there is a big push for performing due diligence or due care—or asking external parties for proof of security assessments and remediation efforts. It's very common for companies to ask to see, or even insist on seeing, each other's security assessment and/or audit reports to ensure good security is in place. Keep this in mind when designing an assessment report and, specifically, the executive summary. Try to write the executive summary so that it can be used as a standalone report and is safe to distribute to external parties.

Some points an executive summary should cover include:

- A description of the systems or applications that were assessed.
- A brief explanation of why the systems or applications needed to be assessed, for example, for an annual compliance assessment or a post-incident assessment.
- A description of who performed the assessment and the general strategy or techniques used, for example: "Third-party forensics firm ScanCo performed the assessment and used a combination of automated discovery and vulnerability scans as well as manual penetration techniques."
- A high-level description of the most critical findings and the risks they present or how they affect compliance.
- A brief summary of any remedies committed to or underway.

Remember to keep the executive summary fairly high level. Avoid mentioning potentially sensitive information such as IP addresses, version numbers, and names of programs used that could be used by another party to compromise systems and applications. This keeps the executive summary clean of technical details and usable as a standalone document to show customers or other outside parties that an assessment was performed.

Summary of Findings

The summary of findings section should go a step farther than the executive summary and outline most or all the findings rather than just the most critical findings that are included in the executive summary. Many scan tools can generate summary graphs or tables straight from the collected data. The summary of findings is a good place to use graphs to show analysis or trends, such as:

- Percentage or quantity of vulnerabilities found by vulnerability type/category
- Percentage or quantity of vulnerabilities found by Web site component, for example, Web application versus Web front end versus OS and/or by IP address
- Percentage or quantity comparison of vulnerabilities by criticality, for example, high versus medium versus low
- Table or list of security gaps confirmed to be present on each Web site component or system

The summary of findings can also include text descriptions or observations about the percentages/quantities that are shown. For instance, if the vulnerability scans were performed only in a non-authenticated manner and did not log onto the systems being assessed, the assessor might want to mention that fact. The assessor might also want to explain briefly how that might generate false positives, or "suspected but not necessarily confirmed," vulnerabilities that could make the results look worse than they really are.

NOTE

A very useful but time-consuming feature to add to a summary of findings is summary data from any previous assessments. Historical assessment data can show trends which, in turn, can lead to identifying issues such as breakdowns in technology processes, poor security awareness, training, and so on.

Vulnerability Assessment

The vulnerability assessment section of the report is typically one of the easiest to generate, but it also contains the largest amount of data. Many vulnerability scanning tools are adept at generating detailed reports. The better tools can even export that data in various formats so that it can be easily imported and manipulated into a manual assessment report or presentation. Depending on the purpose for the vulnerability scan and report, it may be required to use the report provided by the scan tool, so that the findings cannot be manipulated or deleted. When selecting a vulnerability scan application, be sure to take into consideration its ability to report or export data in a convenient format.

The type, or complexity, of vulnerability scan usually determines the amount of data or findings that will be generated. It's not unusual for a single system or IP address to generate 50 to 100 findings. Considering that most vulnerability scan tools devote up to a half-page description for each vulnerability found, this can create a long report. Each write-up for a vulnerability finding typically includes the following information:

- The vulnerability name and description
- A Common Vulnerabilities and Exposures (CVE) number that uniquely identifies it across various security vendors and applications
- A Common Vulnerability Scoring System (CVSS) number that is often used to determine its criticality or importance
- Some type of criticality rating or designation, such as red, yellow, or green; high, medium, or low
- Information regarding any patches or configuration procedures to remedy the vulnerability

If the native report from the vulnerability scanning utility proves to be too long, try exporting the data into a format that allows for the fields to be manipulated, such as Excel. This will work, however, only in situations that

don't require a certified, or non-modifiable, report. In the case of PCI, where it's required that authorized third parties perform the scans, the reports must be generated and delivered in PDF format so that the results cannot be altered.

Security Assessment

The security assessment, or penetration test, section of the report is likely to be the least structured due to the subjective nature of security assessment testing and tools. Depending on the approach used, there are a few ways that security assessment information can be reported.

One effective way of organizing security assessment information is to group the findings by system or component and then sub-group the findings within each system by the type of attack performed or the functionality that was being tested. The following is an outline of a method for reporting security or penetration test results:

I. Web site front end
   a. Authentication attack findings
      i. Tools/techniques used
      ii. Result
   b. Field manipulation attack findings
      i. Tools/techniques used
      ii. Result
   c. Code/field injection attack findings
   d. Scripting attack findings

II. Web site database
   a. Authentication attack findings
   b. Field manipulation attack findings
   c. Code/field injection attack findings

Recommendations

The recommendations section, also referred to as the remediation section, of the report is arguably the most important section. Because vulnerability remediation will likely have been mentioned as part of the data from vulnerability scan tools, this section is primarily a summary of fixes. There may be a large number of findings to fix and complexity involved from one fix to another. Therefore, you should consider categorizing the recommendations by short term and long term.

Short Term

Short-term recommendations should include fixes for the most critical findings, or those that are most likely to have a significant impact on the system in the near future. Because short-term fixes are likely to get the most visibility and attention, it's a good idea to streamline and group these recommendations. If some of the remediations involve the same personnel, processes, or systems, consolidate the recommendation into a single statement or item, if possible. This will help organize the effort to fix the issue and ensure that fixes are consistent across systems or technologies.

NOTE

One effective technique is to display recommendations in a table with information that helps readers understand how best to implement the fix. That can include security information, such as the potential impact if the recommendation isn't followed or alternative solutions if the fix isn't possible now. It may also contain non-security information such as an estimate of resources that will be needed.

It's also necessary to define what's meant by short term. This may have to be defined per recommendation. Some vulnerabilities and security gaps that are considered extremely critical and easily exploited may need to be

addressed in hours or days. Other highly critical findings that are harder to exploit might be considered short term but can be addressed in weeks. This difference may also have to do with the time needed to test the recommended fix. It's best to define the short term in a range of time, say 0 to 30 days, and then associate a number of days or weeks with each recommendation. This also will allow IT and business personnel to prioritize the recommendations, if needed.

In addition to simply using the vulnerability criticality ratings, it's also highly recommended that the assessor refer to other sources of attack trend information, such as *http://www.sans.org* or *http://www.cert.org*, to determine if any recent attacks or trends should escalate findings into the higher-risk, short-term recommendation category.

Long Term

Long-term recommendations, perhaps better stated as longer-term recommendations, should include all the remediations that don't fall into the short-term category. As with short-term recommendations, it will be necessary to define the long term. This will help IT and business personnel keep fixes on schedule.

Long-term recommendations may also include non-technical suggestions, such as procedural improvements or technology evaluations. For example, if a large number of vulnerabilities or findings were associated with an aging operating system, the short-term recommendation could be to patch or secure the operating system and a long-term recommendation would be to evaluate an alternative OS or a more expedient patch process.

Best Practices for Web Site Vulnerability and Security Assessments

Web site security testing can involve a number of tools and techniques. However, some best practices may prove helpful when choosing or using the method for assessing a Web site application.

Choose the Right Tools

Although there are very few bad tools for performing security assessments and vulnerability scans, some are certainly better than others. Rarely will one tool find every vulnerability or security hole. Don't be afraid to evaluate different tools and use multiple tools, even for the same function, when performing security assessments. That certainly results in more effort and data to review, but the end goal isn't necessarily speed and convenience when testing application security. In most cases, attackers use a wide variety of tools—often the same tool run from different operating systems. Not only does that give the attackers a better chance of compromising a system or application, but it also helps them learn the various operating systems and application languages. A thorough assessor will follow the same practice. Some additional items to remember when selecting tools:

- Keep data export capabilities in mind for when it's time to manipulate and report the findings.
- Know the capabilities of the utilities being used and how they may affect production systems or data.
- Try to use both commercial and freeware programs. Commercial security tools are typically updated more frequently with the latest vulnerability and attack data.

Test Inside and Out

As mentioned earlier in this chapter, there often are differences in the results of non-authenticated scans and tests versus authenticated ones. It can be difficult or even impossible to get the credentials needed to perform authenticated tests—so many times authenticated testing is simply passed over. However, authenticated scans need to be performed in order to accurately evaluate an application's true security risks. If authenticated testing of production systems is simply too controversial or risky, then perform authenticated scans on a test system—which can help justify getting a test system in place if one isn't already present.

Think Outside the Box

Attackers are as familiar with security testing tools as security professionals are—probably more so. Although the standard tools and techniques are still necessary, be on the lookout for new techniques. Using Google advanced search techniques is one way of performing non-standard testing. Be on the lookout for new crawlers and other utilities or services that are designed to help advertise or improve search results for Web content. Many of these engines will find ways, albeit usually well-intentioned, to get to data within the Web server or Web application quicker than some attackers or security tools can.

Research, Research, Research

There are plenty of Web sites available that monitor attack trends, offer unique testing tools, and even support forums that attract assessors and attackers alike. It's not enough to simply run tools or perform a few manual techniques and consider an application tested. The tools and techniques are a start—but knowing things like the technology behind the attacks, how attacks are evolving, which industries the attacks are happening in, and so on will prove invaluable in truly securing an application or Web site. Perform a Web search for the server's name occasionally to see if any attackers or online tools are targeting the system or application.

CHAPTER SUMMARY

The chapter has covered in detail how important it is to test Web applications. Testing Web applications involves not just the underlying database, but also the Web server front-end software and Web server operating system, because all are interconnected. Testing may require the use of testing tools.

Many of the tools available are so feature-rich that using them without a plan can be time-consuming and possibly even dangerous to the system. Spend the extra time needed to run non-invasive discovery and enumeration tools. This will help determine which tools, or options within tools, are worth running and could save scan time and data. It will also help identify the technologies involved and the techniques, both automated and manual, that would be most effective in performing deeper penetration tests on the system or application.

Finally, a security assessment cannot be considered a success unless the resulting data, or findings, are clearly documented and communicated. Resolving security issues likely will involve various parties, and the data from the security assessment often must be fashioned so that it's relevant to each audience. A good executive summary is important for getting traction with higher-level management and can also be a valuable means of satisfying external parties who need to see evidence of security testing. A clear set of recommendations is also important for remediation.

In summary, an effective Web site vulnerability and security assessment relies on a combination of fundamental technical know-how, thoughtful planning, a variety of tools, and effective communication.

KEY CONCEPTS AND TERMS

Enumeration
Fingerprinting
Penetration testing
Ping sweep

CHAPTER 11 ASSESSMENT

1. The "percentage of vulnerabilities not found" metric is a useful way of reporting assessment data.

   A. True
   B. False

2. How many tiers are commonly used for Web sites?

   A. 2
   B. 1
   C. 3
   D. 4

3. The act of fixing vulnerabilities or findings resulting from an assessment is known as _____.

4. Which of the following activities are considered parts of a Web server OS assessment? (Select two.)

   A. Identifying the source code author

B. Identifying the patches and updates that have been installed
C. Identifying the services and ports that are active
D. Identifying the databases that are running

5. Ping sweeps are a part of what process?

A. Code review
B. Discovery
C. Attack vectors
D. Remediation

6. Web site forms and user input fields are often attacked using cross-site scripting.

A. True
B. False

7. Which section of the assessment report is intended to be a high-level briefing of the findings?

A. Summary of findings
B. Vulnerability findings
C. Recommendations
D. Executive summary

8. An in-depth security assessment of a Web server application includes performing which of the following?

A. Error-based code compiling
B. OS patching
C. A source code review
D. TCP/IP routing

9. SQL _____ is an attempt to manipulate a database by inserting commands into a field or URL.

10. Nmap's primary features include which of the following? (Select three.)

A. Password cracking
B. OS fingerprinting
C. Port scanning
D. Code analysis
E. Ping sweeps

11. What is the purpose of exploiting a vulnerability or flaw in a system to gain access to resources not otherwise available to the attacker or tester?

A. Acceleration
B. Enumeration
C. Privilege escalation
D. Field injection

12. OWASP is the organization known for developing secure application development standards and practices.

A. True
B. False

13. Nessus uses thousands of _____ to identify vulnerabilities associated with services, applications, and operating systems.

14. Which attack involves exploring the files and folders of a Web server by manipulating URLs?

A. Man-in-the-middle
B. Buffer underflow
C. Brute force password attacks
D. Directory traversal attacks

15. Unauthenticated scanning requires the scanner logging onto the systems being assessed.

A. True
B. False