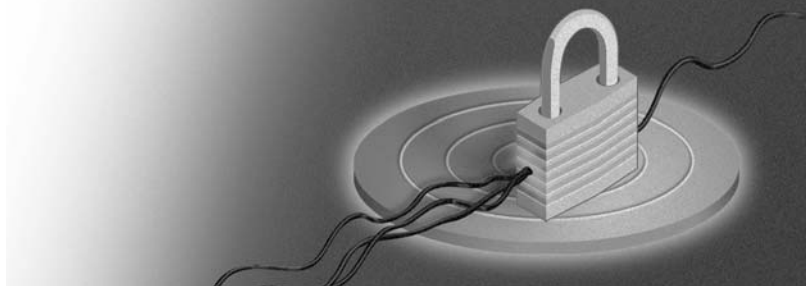


3



Using Vulnerability Scanning to Assess Network Security

In an ideal world, you could scan for vulnerabilities instantly by pulling up the complete configuration of all computers and network devices as well as the applications that run on them. Unfortunately, no current technology does this. Furthermore, because most networks are highly heterogeneous, have clients separated by high-latency links, and have administrative zones, remote clients, and unmanaged clients, this technology is not likely to appear anytime soon. Many network administrators engage in vulnerability scanning by running a tool they downloaded from the Internet, but because they don't have any real goals in mind or an understanding of what the tool is actually doing, the result of the scan is at best temporarily interesting and not entirely useful. To ensure that vulnerability scanning actually works to improve security on your network, you need to create a plan, select a technology or tool, and then execute the plan. This chapter discusses these three aspects of vulnerability scanning: planning, technology, and process.

Planning a vulnerability scanning project, though not necessarily complex, is critical. Like most IT projects, vulnerability scanning projects are most likely to fail because of ineffective planning, which inevitably increases the chance of ineffective execution. Remember, the goal of any vulnerability scanning project is to improve the security of your network—not just to find vulnerabilities. By completing the following steps, you can create a solid plan for improving your network security through vulnerability scanning:

1. Set a scope.
2. Determine goals.
3. Choose a technology.
4. Create a process for scanning for vulnerabilities.
5. Create a process for analyzing the results.

Setting a Scope for the Project

Attempting to tackle a vulnerability scanning project without a well-defined scope not only makes the project a lot harder to do but is also a certain path to failure. There are several key elements to setting the scope of a vulnerability scanning project:

- Define the target.
- Define the target scope.
- Define the types of vulnerabilities you will scan for.

At the end of this planning phase, you have a statement or series of statements that discretely define the boundaries of the project, provide the basis for project goals, and guide your choice of technology.

Defining the Target

Before you begin defining the target of your vulnerability scanning, you have to know what you have on your network. This inventory will determine the type of vulnerabilities that you look for and consequently the types of vulnerability scanning software you will use and the skills that the project will require. Always consider whether the target of your vulnerability scanning project will be a managed or an unmanaged target. This will have a big impact on the type of tool you will need to use. The more specific you can be about the target, the more detailed your scan will be. One way to think about your network is to break it down into components:

- **Network segments** Network segments, which include wired and wireless networks, internal networks, extranets, and publicly available networks, are very flexible. Administrators can easily create or remove segments or change the configuration of existing segments. Without centralized control or good record keeping, existing network diagrams can quickly fall out of date.

- **Devices** Common devices that exist on most networks include routers, firewalls, wireless access points, and other types of network appliances such as storage devices. For each device, information such as the version, configuration, and patch status is part of the baseline knowledge you should have about your organization's network.
- **Operating systems** Like a network device, an operating system used on your organization's network presents a unique set of vulnerabilities. As with devices, knowing the version, configuration, and patch status of each server and workstation on your network is almost as important as knowing of the existence of the computer.
- **Mobile devices** Once generally considered geek toys, mobile devices, such as personal digital assistants (PDAs) and other small form factor computers, are now part of the mainstream IT infrastructure. Because of the ascent of these devices from the realm of technocrats to the world of corporate executives, few are network-manageable; however, they still might be gateways to the network.
- **Applications** Applications also are part of the network ecosystem and thus present their own vulnerabilities. Applications are particularly difficult to account for because users can change their configurations, or they can be impossible to query.

Note Knowing the composition of your organization's network is also essential to developing a successful patch management strategy. The information required for effective and cost-effective vulnerability scanning is strongly correlated to the information required for successful patch management, so gathering this information serves two important purposes. Your organization might already have this information as part of its patch management or license management processes.

In a nutshell, knowing the composition of even a small network can be very daunting, but it is particularly difficult to know about a network when an organization has undergone mergers or has grown by acquisition, or when the organization has, in effect, been managed by the end users. The three basic approaches to obtaining the composition of your organization's network are depicted in Figure 3-1. Not all organizations will use a strategy that falls into

only one of these categories, but the general approach that most organizations will take will align with one.

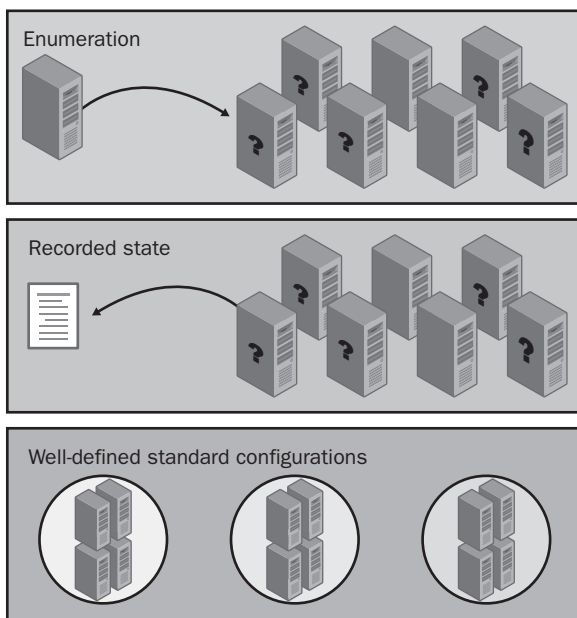


Figure 3-1 Approaches to obtaining composition of a network.

Enumeration

Unfortunately, most organizations don't know what devices they have or precisely how their network is configured at any given point in time. Thus, an organization's first attempt at vulnerability scanning ends up becoming an exercise in enumerating devices, operating systems, and applications that exist on the network. In fact, many vulnerability scanning software packages include software to remotely identify devices, operating systems, and applications, often through TCP/IP footprinting. For example, you can often determine the type of device or operating system by using the TCP options, unique ICMP responses, or other items not explicitly defined or listed as optional in the relevant RFCs. The all-purpose scanner Nmap has footprints for nearly 500 different IP stacks. Of course, there are less esoteric methods, too, for example, capturing banners (which is not always reliable, as you'll see later in this chapter); or port scanning for applications, such as SQLPing, which automates scanning networks for computers running Microsoft SQL Server.

Tip Many vulnerability scanning tools are also very useful for penetration testing, particularly those that do not require Administrative privileges on the target systems. Note that attackers also use tools, so getting to know what tools are out there and how they work will help you defend against them. This topic is discussed in more detail later in this chapter.

There are drawbacks to enumeration, namely that enumeration is a point-in-time event—if computers are turned off or are not attached to the network when the enumeration takes place, they will be missed. Similarly, if distributed firewalls are used, enumeration might fail.

Recorded State

An improvement over enumeration is to have a record of the state of the network and computers, devices, and applications. Having a record allows you to enumerate only portions of the network so that you can find the delta over time, an approach that is much more likely to identify transient components of the network, such as laptops.

Note Not all servers face the same threats. For example, a vulnerability for an Internet-facing Web server is not necessarily a vulnerability for an intranet server, or it can be a different level of vulnerability, depending on the server.

You can use a network diagram and configuration records to define the target of your vulnerability scanning. You can also use configuration records to better prioritize your target systems and to help you quickly identify whether you need to deploy a security patch. For example, in February 2002, CERT announced a critical vulnerability in SNMP that affected many products from many vendors (<http://www.cert.org/advisories/CA-2002-03.html>). Organizations that had a record of devices as well as operating systems and their model numbers could contact each vendor for the security patch or implement vendor-specific workarounds. Organizations that did not have a record faced a much longer period of exposure and very possibly increased the costs of patching this vulnerability.

Configuration records and network diagrams have one serious enemy: change. Configuration records and network diagrams fall out of date quickly, which can cause confusion or be downright misleading. More than one IT staff member has lost track of a WAN link this way!

Well-Defined Configurations

Ideally, you can deploy devices, computers, and applications in well-defined, standard configurations. For example, your organization might have two user operating system configurations for each business unit: one for desktops and one for laptops based on the Microsoft Windows XP security templates and a standard Microsoft Office configuration using the Microsoft Office Resource Kit Tools. By knowing exactly how each computer is configured, you might actually be able to find and mitigate vulnerabilities without using vulnerability scanning software and instead use vulnerability scanning to validate your search results.

More Info You can download the security templates for Windows XP from the Microsoft website at <http://www.microsoft.com/technet/security/prodtech/winclnt/secwinxp/default.asp> and the Microsoft Office Resource Kit Tools from <http://www.microsoft.com/office/downloads>.

Although implementing well-defined configurations is obviously *much* more easily said than done, by achieving this state of network management, you realize major benefits for your security administration that are not limited to vulnerability scanning. Having well-defined, standard configurations enables you to perform vulnerability scanning in one instance of the system and then implement the changes across all systems. This strategy greatly expedites the process of improving network security through vulnerability scanning by reducing the time it takes to scan the network and implement changes.

The completeness of your scan depends on the degree to which you know what devices, operating systems, and applications are running on your network. If you will be scanning client operating systems or applications, keep in mind that unlike servers and network devices, these network components might not be attached to the network or powered on at the time of the check; consequently, you should consider running multiple or regular passes of the vulnerability scanning software. This is discussed in more detail later in the chapter.

Defining the Target Scope

When you define the target scope of your vulnerability scanning project, you must determine the shape and size of your project. The shape of a project can be to scan vertical or horizontal, or to scan both. The shape of the scan helps determine what tools your project requires.

- **Vertical scan** This type of scan searches single host or single-type hosts for multiple vulnerabilities. For example, you might scan all computers running Windows XP for security patch level, common Windows vulnerabilities, and weak passwords. Vertical scans are useful when you are responsible for the security of only a certain type of network component or are looking for a particular vulnerability on a particular platform.
- **Horizontal scan** This type of scan searches different types of hosts or applications for the same vulnerability. For example, you might scan all network devices and computers on your network for vulnerability to land denial of service (DoS) attacks. Horizontal scans are useful for scanning for vulnerabilities that span platforms.
- **Vertical and horizontal scan** This type of scan combines the benefits of both vertical and horizontal scanning: searching for multiple vulnerabilities across multiple platforms. Although this tool performs both scanning functions and thus might seem appealing, like many all-in-one tools, it might not perform either function very well.

Tip Be careful about allowing a vulnerability scanning tool determine the scope of your project.

Determining the size of the scan is essential for ensuring that you deliver your project on time and on budget. For example, if you start your project without clearly defining the breadth of the scanning, you might end up scanning and remediating many more systems than you initially planned. This planning element is particularly important when your vulnerability scanning tool is licensed on a per-host basis or your project requires contract personnel.

The best way to determine the size of the project is to use existing logical segments, such as subnets, or physical segments, such as buildings or floors within buildings. Using this approach not only sets the scope for the project but

also allows you to better track progress and check for completeness. For example, suppose your scanning project encompasses all computers in three offices, and you know that each office has 110 computers in it. If your scan returns results for only 309 computers, you know that something is preventing the software from locating or scanning all the computers.

If the target of your vulnerability scanning project includes the analysis of application source code, be sure to define which source code files are in scope and what programming languages are used in those source code files. Analyzing source code for a console application written in C requires significantly different tools and staffing than for an ASP.NET Web application written in C#.

Defining Types of Vulnerabilities

After you determine the target of your project, including which devices, operating systems, and applications will be scanned and the size and shape of the scanning, you should define the type of vulnerabilities that you will scan for. For example, if you decide to scan all servers for Windows 2000 and Windows Server 2003 on three specific subnets, you might decide to search only for susceptibility to exploits of known product vulnerabilities. This is the final step in setting the scope of your project and also the one that is most frequently ignored. Other types of potential vulnerabilities that are commonly scanned for include:

- Password vulnerabilities
- Weak operating system and application default settings
- Common configuration and coding mistakes
- Protocol vulnerabilities (such as the TCP/IP stack vulnerabilities)
- Administration vulnerabilities (such as having 73 administrators on a router)

In an ideal world, at this point, you would be able to discretely list each vulnerability that you would scan for; however, in reality, unless the project is very small, listing each vulnerability requires a lot of expertise and experience. For most administrators, the first few vulnerability scanning projects are learning experiences as much as they are occasions for yielding solid results. For example, you might find that even after setting scope and goals, when choosing a toolset for your scanning project, you identify additional items you would like to scan for. Because of this, for your first few projects, you will have to balance the impact of scope creep with meeting the ultimate goal of improving the security of your network.

At the end of this planning phase, you can construct scope statements that are essential for setting the project's goals. Table 3-1 illustrates how you can build these statements. You can use this table as a template for your own projects. If you have many hosts or are scanning for multiple vulnerabilities, you might want to use a spreadsheet or database to track this information. If you are planning to outsource vulnerability scanning to a contract firm, building scope documents will be essential for tracking the progress of the project.

Table 3-1 Example Vulnerability Scanning Scope

| Statement components | Example |
|-----------------------------|--|
| Target | Windows 2000 Server and Windows Server 2003 |
| Target area | All servers on the subnets: 192.168.0.0/24 192.168.1.0/24 |
| Vulnerabilities to scan for | RPC over DCOM vulnerability (MS 03-026) Anonymous SAM enumeration Guest account enabled Greater than 10 accounts in the local Administrator group |

Determining Goals

The *overarching* goal of any vulnerability scanning project is to locate weakness in hosts or the network and remediate them. After you define your scope, you should then determine *discrete* goals for the project. The overall success of the project can be determined by how well it achieves its goals. As an administrator conducting the project, you must set clear and attainable goals in the planning phase to help ensure that all members of the project team understand the project—what it aims to do, the time frame for the project, and so on—so that the project stays on task.

Tip On any project, when there is disagreement about what to do or how to proceed, it is helpful to ask yourself or the project team, “How will this help us accomplish our goals?” If your project has discrete goals, this question will generally settle the dispute.

The good news is that by establishing solid scope statements, the goals have practically written themselves! The scope statement provides the first part of the goal and the remediation covers the second. You can transform the scope template in Table 3-1 into discrete project goals.

The goal presented in Table 3-2 is likely to be much simpler than your project's goal, but it does provide a nice framework for documenting the real goal: increasing the security of the network. Chapter 6, "Reporting Your Findings," discusses how you can present the results of your findings to management to achieve the best results.

Table 3-2 Vulnerability Scanning Project Goal Example

| Project goal | |
|--|--|
| In the vulnerability scanning project, all computers running Windows 2000 Server and Windows Server 2003 on the subnets 192.168.0.0/24 and 192.168.1.0/24 will be scanned for the following vulnerabilities and be remediated as stated. | |
| Vulnerability | Remediation |
| RPC over DCOM vulnerability (MS 03-026) | Install Microsoft security patches 03-026 and 03-39. |
| Anonymous SAM enumeration | Configure <i>RestrictAnonymous</i> to 2 on Windows 2000 Server and <i>RestrictAnonymous-Sam</i> to 1 on Windows Server 2003. |
| Guest account enabled | Disable Guest account. |
| Greater than 10 accounts in the local Administrator group | Minimize the number of accounts on the Administrator group. |

Choosing a Technology

Once your project scope and goals are in place, you should have a pretty good idea of what the vulnerability scanning tool requirements will be. The larger your scope, the more likely it is you will need to use multiple tools. Often, using multiple tools can yield fewer false-positive results because it provides cross-verification.

Before running any vulnerability scanning tool, know what the tool is scanning for and how it is conducting the scan. Occasionally, vulnerability scanning tools detect the presence of a vulnerability by directly testing for the weakness through an actual exploit. For example, you might have a packaged vulnerability scanning application that checks for a system's susceptibility to weak passwords by checking enumerated accounts for common passwords. If

your network has an account lockout policy configured, the vulnerability software could easily lock out all accounts on the network.

Tools and Managed vs. Unmanaged Targets

As discussed earlier, one major consideration when choosing a vulnerability scanning technology is whether the tool explicitly requires administrator access to the target system. This requirement largely determines how the tool scans for the weakness (and where the tool could produce misleading or incorrect information). A good example of this is scanning for computers that are vulnerable to the DCOM over RPC vulnerabilities described in Microsoft Security Bulletin 03-026 and 03-039. (See http://www.microsoft.com/security/security_bulletins/ms03-039.asp for more information about these vulnerabilities.) Here you can compare two tools provided by Microsoft that scan for this product vulnerability: Microsoft Baseline Security Analyzer (MBSA) Version 1.2 and a command-line scanner named KB824146Scan.exe.

More Info You can download both of these tools from the Microsoft website. For MBSA, see <http://www.microsoft.com/technet/security/tools/mbsahome.msp>. For KB824146Scan.exe, see <http://support.microsoft.com/?kbid=827363>. (Incidentally, KB824146Scan was written by this book's technical reviewer, Ramsey Dow.)

MBSA checks for the vulnerability by using the HFNetChk engine (developed by Shavlik Technologies, LLC) to determine whether the two patches that fix the vulnerability are installed. It does this by either checking the file version number on the files installed in the patch or looking at the registry to see whether the patch self-reported its installation. In effect, MBSA is not detecting the presence of the vulnerability itself but rather an artifact of the patch to the problem. Thus, the results might be misleading; for example, the patch could be flawed or the host needs to be restarted after the patch is installed before it takes effect. Although MBSA has some weaknesses, these do not equate to MBSA being a poorly designed product. Rather, this example highlights why knowing how the vulnerability scanning tool determines the presence of vulnerabilities is important. Figure 3-2 shows the output of MBSA scanning for this vulnerability when run under the security context of an administrator on a computer running Windows XP.

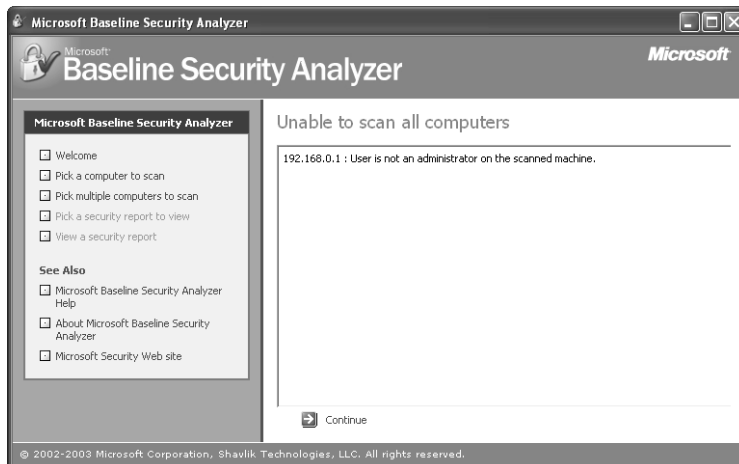


Figure 3-2 Running MBSA under a non-administrator security context.

As you can see in Figure 3-2, when you run MBSA in a non-administrator security context, the tool cannot determine whether the patch was installed.

In contrast, the tool KB824146Scan.exe actually checks whether the computer is vulnerable to the exploit of the DCOM over RPC vulnerabilities by attempting a non-obstructive exploit (non-obstructive in that it does not affect the stability or security of the target host). Figure 3-3 shows the output of KB824146Scan.exe when run under the security context of a non-administrator.

```

C:\WINDOWS\System32\cmd.exe
C:\>kb824146scan 192.168.0.1

Microsoft (R) KB824146 Scanner Version 1.00.0257 for 80x86
Copyright (c) Microsoft Corporation 2003. All rights reserved.

<=> Starting scan (timeout = 5000 ms)

Checking 192.168.0.1
192.168.0.1: patched with both KB824146 (MS03-039) and KB823980 (MS03-026)

<=> Scan completed

Statistics:

Patched with both KB824146 (MS03-039) and KB823980 (MS03-026) .... 1
Patched with only KB823980 (MS03-026) ..... 0
Unpatched ..... 0
TOTAL HOSTS SCANNED ..... 1

DCOM Disabled ..... 0
Needs Investigation ..... 0
Connection refused ..... 0
Host unreachable ..... 0
Other Errors ..... 0
TOTAL HOSTS SKIPPED ..... 0

TOTAL ADDRESSES SCANNED ..... 1

C:\>

```

Figure 3-3 Running KB824146Scan.exe under a non-administrator security context.

There are two important lessons to be learned here. First, unless you are certain that all the computers in your scope are managed (that is, you have administrator privileges on all computers), using a vulnerability scanning tool that requires administrator access to carry out its scan might not give you a complete report of the vulnerable hosts on the network. Second, you should scrutinize your vulnerability scanning toolkit for tools that detect artifacts of the presence of the vulnerability rather than the vulnerabilities themselves. The most common situation in which you need to do this is when using banner text, such as HTTP or Telnet welcome banners, to footprint a host. Because administrators can edit the banner text for popular services on nearly every platform, using banners to footprint a host is entirely ineffective.

Author's Note

Let's say that you run a vulnerability scan for hosts vulnerable to a Microsoft SQL vulnerability, and the search turns up no vulnerable hosts. Does that mean that your network is not vulnerable to the exploit? Maybe not. What if, for instance, the scanning software looked for instances of Microsoft SQL Server through port 1433, and your database administrators had changed the default port to something else? Many administrators misinterpret what being a "well-known" port (those less than 1024) means. It does not mean that the service must run on that port. It means only that it commonly does and that hosts can generally depend on finding the service on that port. How many times have you heard someone refer to the SQL Server port or the HTTP port? They don't exist. A good vulnerability scanning tool does not depend on a service running on a specific port, but rather footprints any service it finds running and determines what service is being provided.

Checklist for Evaluating Tools

Common vulnerability scanning tools include all-purpose tools like Nessus that are available at no cost and commercial tools such as Internet Security Systems' Internet Scanner. There are many tools specific to applications or devices, such as NetStumbler or Kismet, or for locating open wireless networks. There are also tools available to scan code for common vulnerabilities, such as FXCop for

.NET Framework–based applications, which is available on the GotDotNet website at <http://www.gotdotnet.com/team/fxcop/>. When evaluating tools, use the following checklist of questions to ask:

- Does the tool require administrative privileges on the target, or will it work without any credentials?
- Does the tool determine susceptibility to exploit or determine the artifacts of the fix to the vulnerability?
- What platforms does the scanner run on and what platforms does the scanner run against?
- Does the scanner automatically detect the platform?
- Does the tool include risk assessments?
- How long does it take to scan a network? Can the scanning be distributed or scheduled easily?
- Will a DoS scan actually disrupt my network? (Most do.)
- How does the scanner handle hosts protected by firewalls?
- Is the tool extensible? Can you add your own tests?
- What are the licensing costs?
- How hard is the tool to use?
- Is the tool's vulnerability database updated regularly?
- Is the tool's reporting format easy to read or readily transformed for use in a database?

More Info To get a jump-start on learning about vulnerability assessment tools, see the research report written by Jeff Forristal and Greg Shipley for Network Computing. The report titled "Vulnerability Assessment Scanners" details the strengths and weaknesses of many popular tools and also gives you more insight into the type of questions you need to answer when selecting a tool. You can read the article on Network Computing's website at <http://www.nwc.com/1201/1201f1b1.html>.

Creating a Process for Scanning for Vulnerabilities

Running a vulnerability scanning tool once will certainly yield results that might help improve the security of your organization's network, but to make the tool truly effective, you need to develop a process for scanning for vulnerabilities that will do the following:

- Detect vulnerabilities
- Assign risk levels to vulnerabilities that are found
- Identify vulnerabilities that have not been remediated
- Determine improvement in network security over time

Detecting Vulnerabilities

Detecting vulnerabilities might sound like an obvious step, but it does not start and end with running an automated tool. Even the best tools do not present a comprehensive report, and the scanning tool itself could have flaws. After running a scan, you need to validate the results for completeness and accuracy. You also should compare the number of hosts scanned to the number of hosts in the scope and determine why discrepancies exist, if they do. For example, suppose you run the tool after business hours so that the scan has no impact on network bandwidth, not realizing that many employees power down their computers each night and that some hosts are running firewalls that block the scan. The scan might report that only 130 hosts are scanned on two /24 subnets.

Tip As a simple check to ensure that your scanning is working properly, in addition to lab testing, place a host with default security in the scope of systems that will be scanned. If you do this, though, be sure to remove it before a bad guy discovers it.

Ideally, you want the results of the vulnerability scan in a database where you can write complex queries to mine the data. Additionally, you can use this database to assign responsibility to researching and remediating the vulnerabilities.

Can I trust the output of my vulnerability scanning tool?

For the most part, yes, but you will need to be aware of possible false-positive and false-negative results. A *false positive* occurs when a scanning tool identifies a vulnerability that does not exist, whereas a *false negative* occurs when a scanning tool reports no vulnerabilities when they do exist. Obviously, false negatives, which can lead to a false sense of security or even perpetuate the existence of serious vulnerabilities on your organization's network, are much more dangerous than false positives, which will slow down your remediation effort or cause you to find a new scanning tool.

The DCOM scanning tool, KB824146Scan.exe, which was discussed earlier in the chapter, has a known false-positive condition (described in the Microsoft Knowledge Base article describing the tool) when scanning Windows 95 and Windows 98 computers. The tool always reports that these systems are vulnerable to the DCOM exploit because of the unsophisticated way these operating systems use RPC. Once you know about the false-positive report, you can analyze the result of your scan appropriately. However, you might not be able to do this with false negatives.

Many older scanning tools (and even some current ones) used the service banner presented by Web servers to determine whether they were vulnerable to known exploits. Unfortunately, as previously discussed, administrators can easily change the banner text. In an attempt to fool would-be attackers, a clever administrator at your company might modify the banner in Microsoft Internet Information Services (IIS) to appear as though it is running Apache. Tools that rely on the service banner for vulnerability scanning will scan this server for Apache vulnerabilities rather than IIS vulnerabilities. Because there are few common vulnerabilities, the scanning software might report that the host is not vulnerable. Clearly, this is a problem.

To better understand potential false-positive and false-negative reports in the tool that you are using, follow these guidelines:

- Read the scanning tool's documentation and check with the author for updates.
- Scan newsgroups or talk with other administrators who use the same tools.
- Run more than one scanning tool periodically.

Assigning Risk Levels to Vulnerabilities

Not all vulnerabilities are equal and not all hosts are equal. For example, an information disclosure vulnerability might not be as serious as a vulnerability that allows an attacker to remotely run code on the host. Similarly, an Internet-facing Web server faces more exposure to attackers than does a workstation computer. You need to assign risk values to each of the vulnerabilities that you discover. Ideally, you assign the vulnerabilities a risk level before the scan takes place—many scanning software packages do this for you. The risk level helps guide administrators and IT managers in determining what area of network security to address first or where to assign the most resources responsible for fixing.

More Info See Chapter 6, “Reporting Your Findings,” for more information about reporting your findings.

Identifying Vulnerabilities That Have not Been Remediated

You might not be able to remediate all vulnerabilities reported by the scanning software. For example, the remediation might break applications that run on the host or that communicate with it; there could be no effective remediation at the time of the scan; or the administrators might not make the necessary changes. By scanning the network periodically, you will know when previously identified vulnerabilities have not been addressed. Having this information will help you escalate security issues within your organization. If you are scanning across areas of security or network administration, you might also be able to use the delta reports to determine which IT administrators are more effective and to begin to understand why.

Determining Improvement in Network Security Over Time

Over time, you can measure improvement in network security by analyzing the results of the scans. This information will be important to your organization and might be the key to getting a bigger bonus or raise. It will also provide an objective dataset for assessing the cost-benefit of the vulnerability scanning project. Here is a list of items that you can measure over time:

- Number of new vulnerabilities discovered on hosts
- Number of hosts vulnerable to a new vulnerability

- Number of vulnerabilities unremediated and the number of hosts that are vulnerable
- Amount of time that a host remained vulnerable
- Overall vulnerabilities detected and remediated

You can use these values as absolute indicators or relative indicators. Odds are that your IT manager doesn't have any idea about security or insecurity but does very much understand the value that vulnerability scanning will have if you present the information properly.

Creating a Process for Analyzing the Results

The final step in vulnerability scanning involves developing a process to ensure the results are properly analyzed so that you can make the best use of your discoveries. Include your comparison of the current results to results from previous scans so that you can gauge the amount of change over time. This information will help you identify whether your organization's security posture is improving or degrading. And, finally, you will want to securely archive the results. Analyzing and reporting the results of vulnerability scans is an important part of performing a security assessment. You can find detailed information in Chapter 6.

Frequently Asked Questions

- Q.** Is vulnerability scanning really this complicated?
- A.** Scanning a network might be as easy as clicking a mouse button, but improving network security is more complicated. Don't be fooled—there is much more to vulnerability scanning than running MBSA. The better your preparation is, the better your results will be. It is cliché, but it's especially true with vulnerability scanning.
- Q.** What's in the fine print?
- A.** Great question. The biggest catch with vulnerability scanning is usually the limitations of the vulnerability scanning software, including your understanding of how the tool determines which hosts are vulnerable. If your scope includes many different platforms, you might not be able to familiarize yourself with all the vulnerabilities of each platform.

Q. Any more tips?

A. Yes—three, actually. First, separate scanning for vulnerabilities from remediating vulnerabilities. This will enable your organization to make good decisions about both vulnerability scanning and remediation. Second, be careful of crying wolf if the vulnerability scanning software lights up like a Christmas tree, because there might be rational reasons for this. Take your time and do your research before escalating your findings too far. Third, use more than one tool to, if nothing else, validate the results of your primary tool.

