Chapter 15
Security Assessment and Testing

THE CISSP TOPICS COVERED IN THIS CHAPTER INCLUDE:

- ✓ **Domain 6.0: Security Assessment and Testing**
  - 6.1 Design and validate assessment, test, and audit strategies
    - 6.1.1 Internal (e.g., within organization control)
    - 6.1.2 External (e.g., outside organization control)
    - 6.1.3 Third-party (e.g., outside of enterprise control)
    - 6.1.4 Location (e.g., on-premise, cloud, hybrid)
  - 6.2 Conduct security controls testing
    - 6.2.1 Vulnerability assessment
    - 6.2.2 Penetration testing (e.g., red, blue, and/or purple team exercises)
    - 6.2.3 Log reviews
    - 6.2.4 Synthetic transactions/benchmarks
    - 6.2.5 Code review and testing
    - 6.2.6 Misuse case testing
    - 6.2.7 Coverage analysis
    - 6.2.8 Interface testing (e.g., user interface, network interface, application programming interface (API))
    - 6.2.9 Breach attack simulations
    - 6.2.10 Compliance checks
  - 6.3 Collect security process data (e.g., technical and administrative)
    - 6.3.1 Account management
    - 6.3.2 Management review and approval

- 6.3.3 Key performance and risk indicators
- 6.3.4 Backup verification data
- 6.3.5 Training and awareness
- 6.3.6 Disaster recovery (DR) and Business Continuity (BC)

- 6.4 Analyze test output and generate report
  - 6.4.1 Remediation
  - 6.4.2 Exception handling
  - 6.4.3 Ethical disclosure
- 6.5 Conduct or facilitate security audits
  - 6.5.1 Internal (e.g., within organization control)
  - 6.5.2 External (e.g., outside organization control)
  - 6.5.3 Third-party (e.g., outside enterprise control)
  - 6.5.4 Location (e.g., on-premise, cloud, hybrid)

✓ **Domain 8.0: Software Development Security**

- 8.2 Identify and apply security controls in software development ecosystems
  - 8.2.9 Application security testing (e.g., static application security testing (SAST), dynamic application security testing (DAST), software composition analysis, Interactive Application Security Test (IAST))

Throughout this book, you've learned about many of the different controls that information security professionals implement to safeguard the confidentiality, integrity, and availability of data. Among these, technical controls play an important role in protecting servers, networks, and other information processing resources. Once security professionals build and configure these controls, they must regularly test them to ensure that they continue to properly safeguard information.

Security assessment and testing programs perform regular checks to ensure that adequate security controls are in place and that they effectively perform their assigned functions. In this chapter, you'll learn about many of the assessment and testing controls used by security professionals around the world.

Building a Security Assessment and Testing Program

The cornerstone maintenance activity for an information security team is their security assessment and testing program. This program includes tests, assessments, and audits that regularly verify that an organization has adequate security controls and that those security controls are functioning properly and effectively safeguarding information assets.

In this section, you will learn about the three major components of a security assessment program:

- Security tests
- Security assessments
- Security audits

Whenever you design a security testing, assessment, and audit program, you should ensure that it will be effective in all locations where you operate. This includes on-premise, cloud, and hybrid locations.

Security Testing

Security tests verify that a control is functioning properly. These tests include automated scans, tool-assisted penetration tests, and manual attempts to undermine security. Security testing should take place on a regular schedule, with attention paid to each of the key security controls protecting an organization. When scheduling security controls for review, information security managers should consider the following factors:

- Availability of security testing resources

- Criticality of the systems and applications protected by the tested controls
- Sensitivity of information contained on tested systems and applications
- Likelihood of a technical failure of the mechanism implementing the control
- Likelihood of a misconfiguration of the control that would jeopardize security
- Risk that the system will come under attack
- Rate of change of the control configuration
- Other changes in the technical environment that may affect the control performance
- Difficulty and time required to perform a control test
- Impact of the test on normal business operations

After assessing each of these factors, security teams design and validate a comprehensive assessment and testing strategy. This strategy may include frequent automated tests supplemented by infrequent manual tests. For example, a credit card processing system may undergo automated vulnerability scanning on a nightly basis with immediate alerts to administrators when the scan detects a new vulnerability. The automated scan requires no work from administrators once it is configured, so it is easy to run quite frequently. The security team may wish to complement those automated scans with a manual penetration test performed by an external consultant for a significant fee. Those tests may occur on an annual basis to minimize costs and disruption to the business.

Many security testing programs begin on a haphazard basis, with security professionals simply pointing their fancy new tools at whatever systems they come across first. Experimentation with new tools is fine, but security testing programs should be carefully designed and include rigorous, routine testing of systems using a risk-prioritized approach.

Of course, it's not sufficient to simply perform security tests. Security professionals must also carefully review the results of those tests to ensure that each test was successful. In some cases, these reviews consist of manually reading the test output and verifying that the test completed successfully. Some tests require human interpretation and must be performed by trained analysts.

Other reviews may be automated, performed by security testing tools that verify the successful completion of a test, log the results, and remain silent unless there is a significant finding. When the system detects an issue requiring administrator attention, it may trigger an alert, send an email or text message, or automatically open a trouble ticket, depending on the severity of the alert and the administrator's preference.

Security Assessments

Security assessments are comprehensive reviews of the security of a system, application, or other tested environment. During a security assessment, a trained information security professional performs a risk assessment that identifies vulnerabilities in the tested environment that may allow a compromise and makes recommendations for remediation, as needed.

Security assessments normally include the use of security testing tools but go beyond automated scanning and manual penetration tests. They also include a thoughtful review of the threat environment, current and future risks, and the value of the targeted environment.

The main work product of a security assessment is normally an assessment report addressed to management that contains the results of the assessment in nontechnical language and concludes with specific recommendations for improving the security of the tested environment.

Assessments may be conducted by an internal team, or they may be outsourced to a third-party assessment team with specific expertise in the areas being assessed.

NIST SP 800-53A

The National Institute for Standards and Technology (NIST) offers a special publication that describes best practices in conducting security and privacy assessments. *NIST Special Publication 800-53A—Assessing Security and Privacy Controls in Information Systems and Organizations* is available for download:

http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53Ar5.pdf

Under NIST 800-53A, assessment objects include four components:

- Specifications are the documents associated with the system being audited. Specifications generally include policies, procedures, requirements, specifications, and designs.
- Mechanisms are the controls used within an information system to meet the specifications. Mechanisms may be based in hardware, software, or firmware.
- Activities are the actions carried out by people within an information system. These may include performing backups, exporting log files, or reviewing account histories.
- Individuals are the people who implement specifications, mechanisms, and activities.

When conducting an assessment, assessors may examine any of the four components listed here. They may also interview individuals and perform direct tests to determine the effectiveness of controls.

Security Audits

Security audits use many of the same techniques followed during security assessments but must be performed by independent auditors. An organization's security staff may routinely perform security tests and assessments, but this is not the case for audits. Assessment and testing results are meant for internal use only and are designed to evaluate controls with an eye toward finding potential improvements. Audits, on the other hand, are evaluations performed with the purpose of demonstrating the effectiveness of controls to a third party. The staff who design, implement, and monitor controls for an organization have an inherent conflict of interest when evaluating the effectiveness of those controls.

Auditors provide an impartial, unbiased view of the state of security controls. They write reports that are quite similar to security assessment reports, but those reports are intended for different audiences that may include an organization's board of directors, government regulators, and other third parties. There are three main types of audits: internal audits, external audits, and third-party audits.

Real World Scenario

Government Auditors Discover Air Traffic Control Security Vulnerabilities

Federal, state, and local governments also use internal and external auditors to perform security assessments. The U.S. Government Accountability Office (GAO) performs federal government audits at the request of Congress, and these GAO audits often focus on information security risks. In 2015, the GAO released an audit report titled “Information Security: FAA Needs to Address Weaknesses in Air Traffic Control Systems.”

The conclusion of this report was damning: “While the Federal Aviation Administration (FAA) has taken steps to protect its air traffic control systems from cyber-based and other threats, significant security control weaknesses remain, threatening the agency's ability to ensure the safe and uninterrupted operation of the national airspace system (NAS). These include weaknesses in controls intended to prevent, limit and detect unauthorized access to computer resources, such as controls for protecting system boundaries, identifying and authenticating users, authorizing users to access systems, encrypting sensitive data, and auditing and monitoring activity on FAA's systems.”

The report went on to make 17 recommendations on how the FAA might improve its information security controls to better protect the integrity and availability of the nation's air traffic control system. The full GAO report may be found at www.gao.gov/assets/gao-15-221.pdf.

Internal Audits

Internal audits are performed by an organization's internal audit staff and are typically intended for internal audiences. The

internal audit staff performing these audits normally have a reporting line that is completely independent of the functions they evaluate. In many organizations, the chief audit executive reports directly to the president, chief executive officer (CEO), or similar role. The chief audit executive (CAE) may also have reporting responsibility directly to the organization's governing board.

External Audits

External audits are performed by an outside auditing firm. These audits have a high degree of external validity because the auditors performing the assessment theoretically have no conflict of interest with the organization itself. There are thousands of firms that perform external audits, but most large organizations use the so-called Big Four audit firms:

- Ernst & Young
- Deloitte
- PricewaterhouseCoopers
- KPMG

Audits performed by these firms are generally considered acceptable by most investors and governing body members.

Third-Party Audits

Third-party audits are conducted by, or on behalf of, another organization. For example, a regulatory body might have the authority to initiate an audit of a regulated firm under contract or law. In the case of a third-party audit, the organization initiating the audit generally selects the auditors and designs the scope of the audit.

Exam Tip

As you prepare for the exam, it may be helpful to remember that these three types of audits differ in who controls them. Internal audits are under the organization's control. External audits are performed by an independent firm that is outside the organization's control but the auditor is still hired by the organization. Third-party audits are performed by a firm that is hired by another organization, so they are outside the enterprise's control.

Organizations that provide services to other organizations are frequently asked to participate in third-party audits. This can be quite a burden on the audited organization if they have a large number of clients. The American Institute of Certified Public Accountants (AICPA) released a standard designed to alleviate this burden. The Statement on Standards for Attestation Engagements document 18 (*SSAE 18*), titled *Attestation Standards: Clarification and Recodification*, provides a common standard to be used by auditors performing assessments of service organizations with the intent of allowing the organization to conduct an external assessment instead of multiple third-party assessments and then sharing the resulting report with customers and potential customers. Outside of the United States, similar engagements are conducted under the International Standard on Assurance Engagements (ISAE) 3402, *Assurance Reports on Controls at a Service Organization*.

SSAE 18 and ISAE 3402 engagements are commonly referred to as *system and organization controls (SOC)* audits, and they come in three forms:

SOC 1 Engagements Assess the organization's controls that might impact the accuracy of financial reporting.

SOC 2 Engagements Assess the organization's controls that affect the security (confidentiality, integrity, and availability) and privacy of information stored in a system. SOC 2 audit results are

confidential and are normally only shared outside the organization under an NDA.

SOC 3 Engagements Assess the organization's controls that affect the security (confidentiality, integrity, and availability) and privacy of information stored in a system. However, SOC 3 audit results are intended for public disclosure.

In addition to the three categories of SOC assessment, there are two different types of SOC report. Both reports begin with providing a description by management of the controls put in place. They differ in the scope of the opinion provided by the auditor:

Type I Reports These reports provide the auditor's opinion on the description provided by management and the suitability of the design of the controls. Type I reports also cover only a specific point in time, rather than an extended period. You can think of the Type I report as more of a documentation review where the auditor is checking things out on paper and making sure that the controls described by management are reasonable and appropriate.

Type II Reports These reports go further and also provide the auditor's opinion on the operating effectiveness of the controls. That is, the auditor actually confirms that the controls are functioning properly. The Type II report also covers an extended period of time: at least six months of operation. You can think of the Type II report as more like a traditional audit. The auditors are not just checking the paperwork; they are also going in and verifying that the controls function properly.

Type II reports are considered much more reliable than Type I reports because they include independent testing of controls. Type I reports simply take the service organization at their word that the controls are implemented as described.

Information security professionals are often asked to participate in internal, external, and third-party audits. They commonly must provide information about security controls to auditors through interviews and written documentation. Auditors may also request the participation of security staff members in the execution of

control evaluations. Auditors generally have carte blanche access to all information within an organization, and security staff should comply with those requests, consulting with management as needed.

Real World Scenario

When Audits Go Wrong

The Big Four didn't come into being until 2002. Up until that point, the Big Five also included the highly respected firm Arthur Andersen. Andersen, however, collapsed suddenly after they were implicated in the downfall of Enron Corporation. Enron, an energy company, suddenly filed for bankruptcy in 2001 after allegations of systemic accounting fraud came to the attention of regulators and the media.

Arthur Andersen, then one of the world's largest auditing firms, had performed Enron's financial audits, effectively signing off on their fraudulent practices as legitimate. The firm was later convicted of obstruction of justice and, although the conviction was later overturned by the Supreme Court, quickly collapsed due to the loss of credibility they suffered in the wake of the Enron scandal and other allegations of fraudulent behavior.

Auditing Standards

When conducting an audit or assessment, the team performing the review should be clear about the standard that they are using to assess the organization. The standard provides the description of control objectives that should be met, and then the audit or assessment is designed to ensure that the organization properly implemented controls to meet those objectives.

One common framework for conducting audits and assessments is the *Control Objectives for Information and Related Technologies (COBIT)*. COBIT describes the common

requirements that organizations should have in place surrounding their information systems. The COBIT framework is maintained by ISACA and is available at www.isaca.org/resources/cobit.

The International Organization for Standardization (ISO) also publishes a set of standards related to information security. ISO 27001 describes a standard approach for setting up an information security management system, and ISO 27002 goes into more detail on the specifics of information security controls. These internationally recognized standards are widely used within the security field, and organizations may choose to become officially certified as compliant with ISO 27001.

Performing Vulnerability Assessments

Vulnerability assessments are some of the most important testing tools in the information security professional's toolkit. Vulnerability scans and penetration tests provide security professionals with a perspective on the weaknesses in a system or application's technical controls by identifying technical vulnerabilities that they contain. Vulnerabilities are weaknesses in systems and security controls that might be exploited by a threat. Vulnerability assessments examine systems for these weaknesses, commonly using automated means, and help security professionals develop a roadmap for remediating those that pose an unacceptable risk to the business.

Describing Vulnerabilities

The security community depends on a common set of standards to provide a common language for describing and evaluating vulnerabilities. NIST provides the community with the *Security Content Automation Protocol (SCAP)* to meet this need. SCAP provides this common framework for discussion and also facilitates the automation of interactions between different security systems. The components of SCAP most directly related to vulnerability assessment include these:

- *Common Vulnerabilities and Exposures (CVE)* provides a naming system for describing security vulnerabilities.
- *Common Vulnerability Scoring System (CVSS)* provides a standardized scoring system for describing the severity of security vulnerabilities.
- *Common Configuration Enumeration (CCE)* provides a naming system for system configuration issues.
- *Common Platform Enumeration (CPE)* provides a naming system for operating systems, applications, and devices.
- *Extensible Configuration Checklist Description Format (XCCDF)* provides a language for specifying security checklists.
- *Open Vulnerability and Assessment Language (OVAL)* provides a language for describing security testing procedures.

For more information on SCAP, see the NIST website at http://csrc.nist.gov/Projects/Security-Content-Automation-Protocol.

Vulnerability Scans

Vulnerability scans automatically probe systems, applications, and networks, looking for weaknesses that may be exploited by an attacker. The scanning tools used in these tests provide quick, point-and-click tests that perform otherwise tedious tasks without requiring manual intervention. Most tools allow scheduled scanning on a recurring basis and provide reports that show differences between scans performed on different days, offering administrators a view into changes in their security risk environment.

There are four main categories of vulnerability scans: network discovery scans, network vulnerability scans, web application

vulnerability scans, and database vulnerability scans. A wide variety of tools perform each of these types of scans.

Remember that information security professionals aren't the only ones with access to vulnerability testing tools. Attackers have access to the same tools used by the "good guys" and often run vulnerability tests against systems, applications, and networks prior to an intrusion attempt. These scans help malicious actors zero in on vulnerable systems and focus their attacks on systems where they will have the greatest likelihood of success.

Network Discovery Scanning

Network discovery scanning uses a variety of techniques to scan a range of IP addresses, searching for systems with open network ports. Network discovery scanners do not actually probe systems for vulnerabilities but provide a report showing the systems detected on a network and the list of ports that are exposed through the network and server firewalls that lie on the network path between the scanner and the scanned system.

Network discovery scanners use many different techniques to identify open ports on remote systems. Some of the more common techniques are as follows:

TCP SYN Scanning Sends a single packet to each scanned port with the SYN flag set. This indicates a request to open a new connection. If the scanner receives a response that has the SYN and ACK flags set, this indicates that the system is moving to the second phase in the three-way TCP handshake and that the port is open. TCP SYN scanning is also known as "half-open" scanning.

TCP Connect Scanning Opens a full connection to the remote system on the specified port. This scan type is used when the user running the scan does not have the necessary permissions to run a half-open scan. Most other scan types require the ability to send raw packets, and a user may be

restricted by the operating system from sending handcrafted packets.

TCP ACK Scanning Sends a packet with the ACK flag set, indicating that it is part of an open connection. This type of scan may be done in an attempt to determine the rules enforced by a firewall and the firewall methodology.

UDP Scanning Performs a scan of the remote system using the UDP protocol, checking for active UDP services. This scan type does not use the three-way handshake, because UDP is a connectionless protocol.

Xmas Scanning Sends a packet with the FIN, PSH, and URG flags set. A packet with so many flags set is said to be "lit up like a Christmas tree," leading to the scan's name.

If you've forgotten how the three-way TCP handshake functions, you'll find complete coverage of it in Chapter 11, "Secure Network Architecture and Components."

The most common tool used for network discovery scanning is an open source tool called Nmap. Originally released in 1997, Nmap is, remarkably, still maintained and in general use today. It remains one of the most popular network security tools, and almost every security professional either uses Nmap regularly or has used it at some point in their career. You can download a free copy of Nmap or learn more about the tool at http://nmap.org.

When Nmap scans a system, it identifies the current state of each network port on the system. For ports where Nmap detects a result, it provides the current status of that port:

Open The port is open on the remote system and there is an application that is actively accepting connections on that port.

Closed The port is accessible on the remote system, meaning that the firewall is allowing access, but there is no application accepting connections on that port.

Filtered Nmap is unable to determine whether a port is open or closed because a firewall is interfering with the connection attempt.

Unfiltered The port is accessible, but Nmap cannot determine whether it is open or closed. It is unfiltered because the port is exposed to the packet probes sent by Nmap, but no conclusive evidence can determine the port's status.

Open | Filtered Nmap cannot establish whether the port is open or filtered. This state occurs when a port does not respond to Nmap's probes, which could be due to packet filtering preventing Nmap's requests from reaching the port, or the port is open but designed not to respond to the probes used by Nmap.

Figure 15.1 shows an example of Nmap at work. The user entered the following command at a Linux prompt:

```
nmap -vv 52.4.85.159
```

To interpret these results, you must know the use of common network ports as discussed in Chapter 12, "Secure Communications and Network Attacks." (You'll also find a reference listing of common ports later in this chapter.) Let's walk through the results of this Nmap scan:

- The first line of the port listing, `22/tcp open ssh`, indicates that the system accepts connections on TCP port 22. The Secure Shell (SSH) service uses this port to allow administrative connections to servers.
- The second line of the port listing, `80/tcp closed http`, indicates that a firewall rule exists to allow access to port 80 but no service is listening on that port. Port 80 is used by HTTP to accept unencrypted web server connections.
- The final line of the port listing, `443/tcp open https`, indicates that the system is accepting connection requests on port 443, which is used by HTTPS to deliver web pages over encrypted connections, a secure alternative to the use of unencrypted connections over port 80.

```
scanner $ nmap -vv 52.4.85.159

Starting Nmap 6.40 ( http://nmap.org ) at 2021-01-19 11:20 EDT
Initiating Ping Scan at 11:20
Scanning 52.4.85.159 [2 ports]
Completed Ping Scan at 11:20, 0.00s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 11:20
Completed Parallel DNS resolution of 1 host. at 11:20, 0.00s e
Initiating Connect Scan at 11:20
Scanning 52.4.85.159 [1000 ports]
Discovered open port 443/tcp on 18.213.119.84
Discovered open port 80/tcp on 18.213.119.84
Discovered open port 22/tcp on 18.213.119.84
Completed Connect Scan at 11:20, 4.71s elapsed (1000 total por
Nmap scan report for 52.4.85.159
Host is up (0.00060s latency).
Scanned at 2020-10-19 11:20:24 EDT for 4s
Not shown: 997 filtered ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  closed  http
443/tcp open  https
```

FIGURE 15.1 Nmap scan of a web server run from a Linux system

What can we learn from these results? The system being scanned is probably a web server that is openly accepting connection requests from the scanned system. The firewalls between the scanner and this system are configured to allow both secure (port 443) and insecure (port 80) connections, but the server is not set up to actually allow unencrypted transactions. The server also has an administrative port open that may allow command-line connections.

Port scanners, network vulnerability scanners, and web application vulnerability scanners use a technique called *banner grabbing* to identify the variant and version of a service running on a system. This technique opens a connection to the service and reads the details provided on the welcome screen, or banner, to assist with version fingerprinting.

An attacker reading these results would probably make a few observations about the system that would lead to some further probing:

- Pointing a web browser at this server would likely give a good idea of what the server does and who operates it. Simply typing the IP address of the system in the address bar of the browser may reveal useful information. Figure 15.2 shows the result of performing this; the site is running a default installation of the Apache web server.

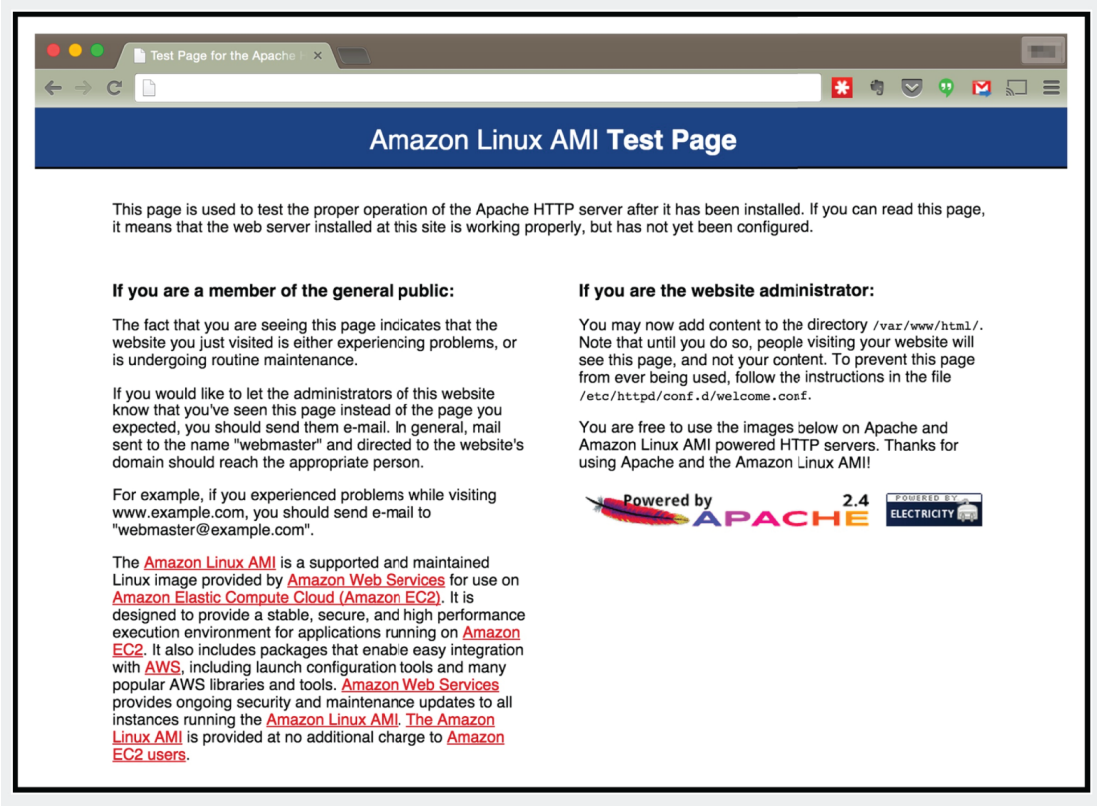

FIGURE 15.2 Default Apache server page running on the server scanned in Figure 15.1

HTTP connections to this server are encrypted. Eavesdropping on those connections is likely not possible.

- The open SSH port is an interesting finding. An attacker may try to conduct a brute-force password attack against administrative accounts on that port to gain access to the system.

In this example, we used Nmap to scan a single system, but the tool also allows scanning entire networks for systems with open ports. The scan shown in Figure 15.3 scans across the 192.168.1.0/24 network, including all addresses in the range 192.168.1.1–192.168.1.254.

The fact that you *can* run a network discovery scan doesn't mean that you *may* or *should* run that scan. You should only scan networks where you have explicit, and hopefully *written*, permission from the network owner to perform security scanning. Some jurisdictions consider unauthorized scanning a violation of computer abuse laws and may prosecute individuals for an act as simple as running Nmap on a coffee shop wireless network.

```
MacBook$ nmap 192.168.1.0/24

Starting Nmap 6.01 ( http://nmap.org )
Strange error from connect (65):No route to host
Nmap scan report for 192.168.1.65
Host is up (0.036s latency).
All 1000 scanned ports on 192.168.1.65 are closed

Nmap scan report for 192.168.1.69
Host is up (0.0017s latency).
All 1000 scanned ports on 192.168.1.69 are closed

Nmap scan report for 192.168.1.73
Host is up (0.021s latency).
Not shown: 994 closed ports
PORT     STATE SERVICE
80/tcp   open  http
515/tcp  open  printer
631/tcp  open  ipp
8080/tcp open  http-proxy
8290/tcp open  unknown
9100/tcp open  jetdirect

Nmap scan report for 192.168.1.94
Host is up (0.00089s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
5009/tcp  open  airport-admin
10000/tcp open  snet-sensor-mgmt

Nmap scan report for 192.168.1.114
Host is up (0.0015s latency).
Not shown: 962 closed ports, 37 filtered ports
PORT     STATE SERVICE
4242/tcp open  vrml-multi-use
```

FIGURE 15.3 Nmap scan of a large network run from a Mac system using the Terminal utility

The `netstat` command is a useful tool for examining the active ports on a system. This command lists all active network connections on a system as well as those ports that are open and awaiting new connections.

Network Vulnerability Scanning

Network vulnerability scans go deeper than discovery scans. They don't stop with detecting open ports but continue on to probe a targeted system or network for the presence of known vulnerabilities. These tools contain databases of thousands of known vulnerabilities, along with tests they can perform to identify whether a system is susceptible to each vulnerability in the system's database.

When the scanner tests a system for vulnerabilities, it uses the tests in its database to determine whether a system may contain the vulnerability. In some cases, the scanner may not have enough information to conclusively determine that a vulnerability exists and it reports a vulnerability when there really is no problem. This situation is known as a *false positive* report and is sometimes seen as a nuisance to system administrators. Far more dangerous is when the vulnerability scanner misses a vulnerability and fails to alert the administrator to the presence of a dangerous situation. This error is known as a *false negative* report.

Traditional vulnerability scans are unable to detect zero-day vulnerabilities that have not yet been identified by the scanner vendor. You'll learn more about zero-day vulnerabilities in Chapter 17, “Preventing and Responding to Incidents.”

By default, network vulnerability scanners run unauthenticated scans. They test the target systems without having passwords or other special information that would grant the scanner special privileges. This allows the scan to run from the perspective of an attacker but also limits the ability of the scanner to fully evaluate possible vulnerabilities. One way to improve the accuracy of the scanning and reduce false positive and false negative reports is to perform *authenticated scans* of systems. In this approach, the scanner has read-only access to the servers being scanned and can use this access to read configuration information from the target system and use that information when analyzing vulnerability testing results.

Figure 15.4 shows the results of a network vulnerability scan performed against the same system subjected to a network discovery scan earlier in this chapter.

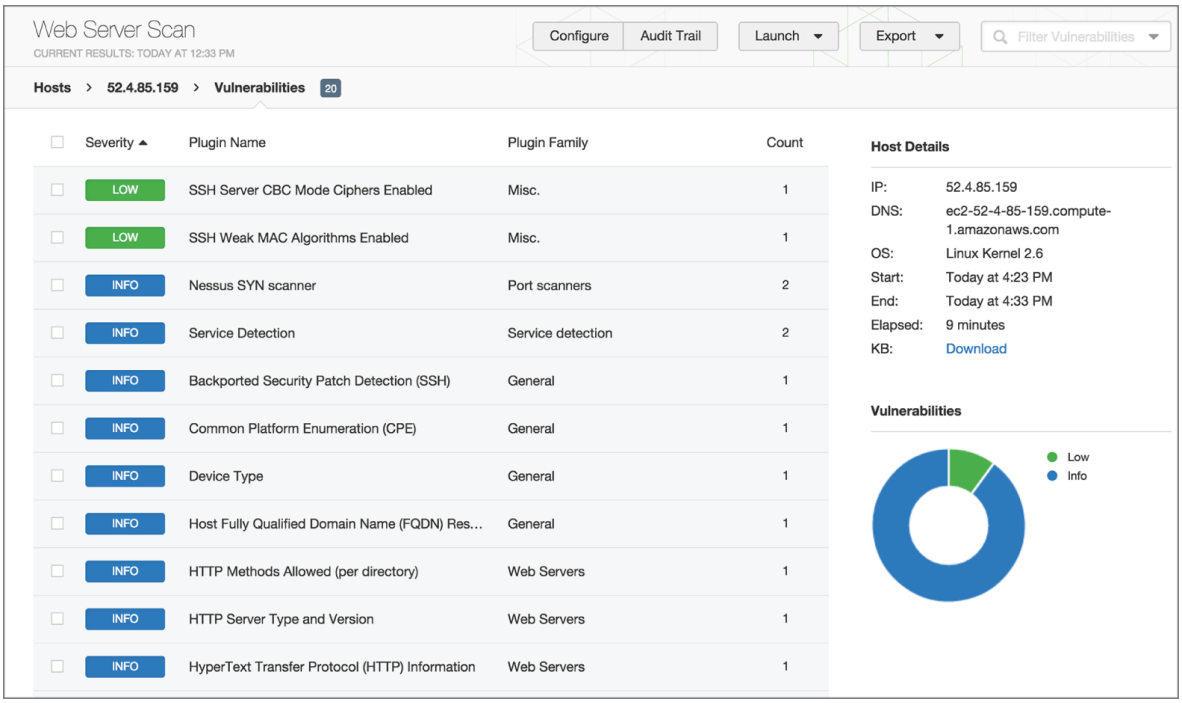

FIGURE 15.4 Network vulnerability scan of the same web server that was port scanned in Figure 15.1

The scan results shown in Figure 15.4 are very clean and represent a well-maintained system. There are no serious vulnerabilities and only two low-risk vulnerabilities related to the SSH service running on the scanned system. The system administrator may wish to tweak the SSH cryptography settings

to remove those low-risk vulnerabilities, but this is a very good report for the administrator and provides confidence that the system is well managed.

Learning TCP Ports

Interpreting port scan results requires knowledge of some common TCP ports. Here are a few that you should commit to memory when preparing for the CISSP exam:

- FTP: 20/21
- SSH: 22
- Telnet: 23
- SMTP: 25
- DNS: 53
- HTTP: 80
- POP3: 110
- NTP: 123
- Windows File Sharing: 135, 137–139, 445
- HTTPS: 443
- LPR/LPD: 515
- Microsoft SQL Server: 1433/1434
- Oracle: 1521
- H.323: 1720
- PPTP: 1723
- RDP: 3389
- HP JetDirect printing: 9100

There are many commercial vulnerability scanning tools available in today's marketplace. The Open Worldwide Application Security

Project (OWASP) maintains a comprehensive list at https://owasp.org/www-community/Vulnerability_Scanning_Tools. The open source OpenVAS scanner also has a growing community of users.

Organizations may also conduct specialized vulnerability assessments of wireless networks. Aircrack-ng is a tool commonly used to perform these assessments by testing the encryption and other security parameters of wireless networks. It can be used in conjunction with passive monitoring techniques that may identify rogue devices on the network.

Web Vulnerability Scanning

Web applications pose significant risk to enterprise security. By their nature, the servers running many web applications must expose services to Internet users. Firewalls and other security devices typically contain rules allowing web traffic to pass through to web servers unfettered. The applications running on web servers are complex and often have privileged access to underlying databases. Attackers often try to exploit these circumstances using SQL injection and other attacks that target flaws in the security design of web applications.

You'll find complete coverage of SQL injection attacks, cross-site scripting (XSS), cross-site request forgery (XSRF), and other web application vulnerabilities in Chapter 21, “Malicious Code and Application Attacks.”

Web vulnerability scanners are special-purpose tools that scour web applications for known vulnerabilities. They play an important role in any security testing program because they may discover flaws not visible to network vulnerability scanners. When an administrator runs a web application scan, the tool probes the web application using automated techniques that manipulate inputs and other parameters to identify web vulnerabilities. The tool then provides a report of its findings,

often including suggested vulnerability remediation techniques. Figure 15.5 shows an example of a web vulnerability scan. This scan ran against the web application running on the same server as the network discovery scan in Figure 15.1 and the network vulnerability scan in Figure 15.4. As you read through the scan report in Figure 15.5, notice that it detected vulnerabilities that did not show up in the network vulnerability scan.

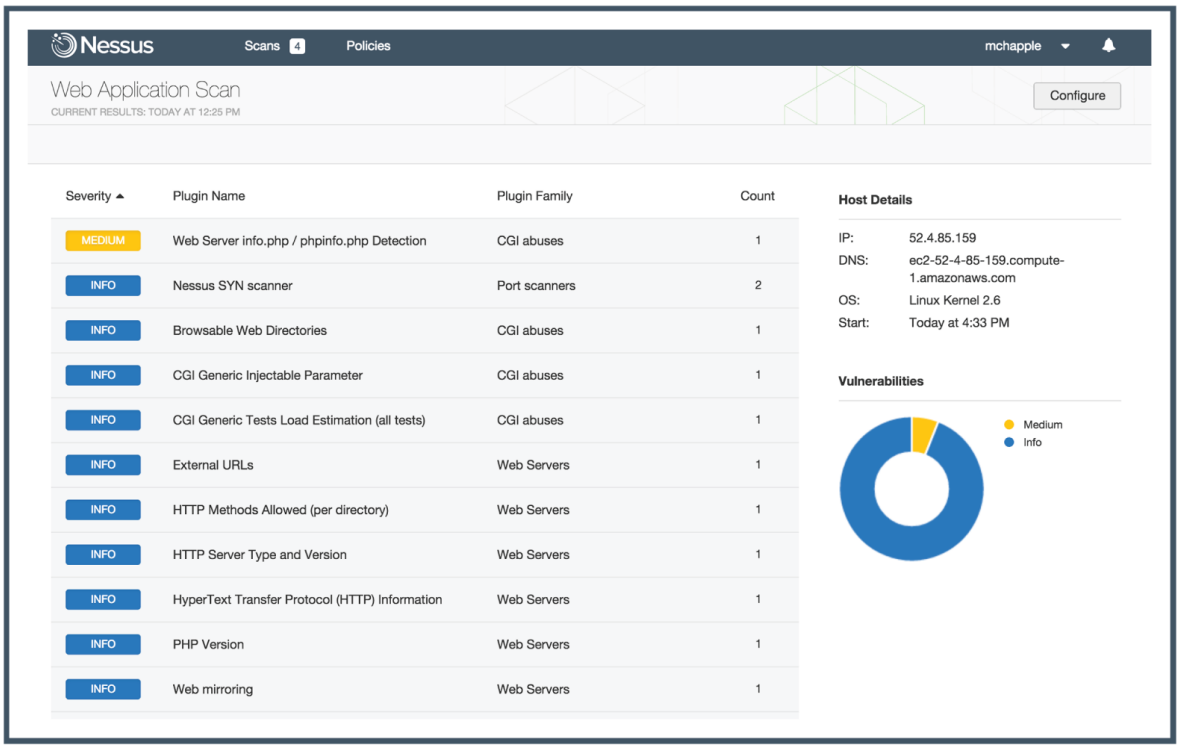

FIGURE 15.5 Web application vulnerability scan of the same web server that was port scanned in Figure 15.1 and network vulnerability scanned in Figure 15.2

Do network vulnerability scans and web vulnerability scans sound similar? That's because they are! Both probe services running on a server for known vulnerabilities. The difference is that network vulnerability scans generally don't dive deep into the structure of web applications, whereas web application scans don't look at services other than those supporting web services. Many network vulnerability scanners do perform basic web vulnerability scanning tasks, but deep-dive web vulnerability scans require specialized, dedicated web vulnerability scanning tools.

You may have noticed that the same vulnerability scanner performed both the network vulnerability scan shown in Figure 15.4 and the web vulnerability scan shown in Figure 15.5. This is an example of a hybrid tool that can perform both types of scan.

As with most tools, the capabilities for various vulnerability scanners vary quite a bit. Before using a scanner, you should research it to make sure it meets your security control objectives.

Web vulnerability scans are an important component of an organization's security assessment and testing program. It's a good practice to run scans in the following circumstances:

- Scan all applications when you begin performing web vulnerability scanning for the first time. This will detect issues with legacy applications.
- Scan any new application before moving it into a production environment for the first time.
- Scan any modified application before the code changes move into production.
- Scan all applications on a recurring basis. Limited resources may require scheduling these scans based on the priority of the application. For example, you may wish to scan web applications that interact with sensitive information more often than those that do not.

In some cases, web application scanning may be required to meet compliance requirements. For example, the Payment Card Industry Data Security Standard (PCI DSS), discussed in Chapter 4, "Laws, Regulations, and Compliance," requires that organizations either perform web application vulnerability scans at least annually or install dedicated web application firewalls to add additional layers of protection against web vulnerabilities.

OWASP provides a list of open source and commercial tools commonly used for web application vulnerability scanning at https://owasp.org/www-community/Vulnerability_Scanning_Tools.

Database Vulnerability Scanning

Databases contain some of an organization's most sensitive information and are lucrative targets for attackers. Although most databases are protected from direct external access by firewalls, web applications offer a portal into those databases, and malicious actors may leverage database-backed web applications to direct attacks against databases, including SQL injection attacks.

SQL injection attacks and other web application vulnerabilities are discussed in more detail in Chapter 21. Database security issues are covered in Chapter 9, "Security Vulnerabilities, Threats, and Countermeasures."

Database vulnerability scanners are tools that allow security professionals to scan both databases and web applications for vulnerabilities that may affect database security. Sqlmap is a commonly used open source database vulnerability scanner and penetration testing tool that allows security administrators to probe web applications for database vulnerabilities and exploit vulnerabilities that do exist. Figure 15.6 shows an example of Sqlmap scanning a web application.

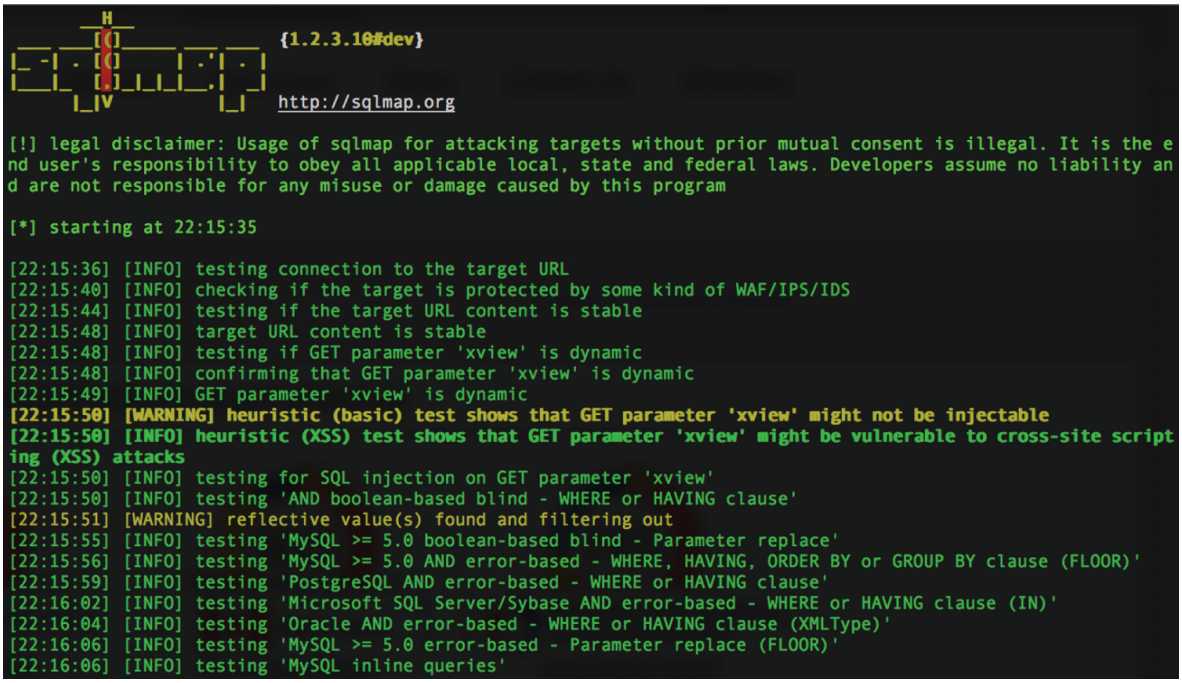

FIGURE 15.6 Scanning a database-backed application with Sqlmap

Vulnerability Management Workflow

Organizations that adopt a vulnerability management system should also develop a workflow approach to managing vulnerabilities. The basic steps in this workflow should include the following:

1. *Detection:* The initial identification of a vulnerability normally takes place as the result of a vulnerability scan.
2. *Validation:* Once a scanner detects a vulnerability, administrators should confirm the vulnerability to determine that it is not a false positive report.
3. *Remediation:* Validated vulnerabilities should then be remediated. This may include applying a vendor-supplied security patch, modifying a device configuration, implementing a workaround to avoid the vulnerability, or installing a web application firewall or other control that prevents the exploitation of the vulnerability.

The goal of a workflow approach is to ensure that vulnerabilities are detected and resolved in an orderly fashion. The workflow

should also include steps that prioritize vulnerability remediation based on the severity of the vulnerability, the likelihood of exploitation, and the difficulty of remediation.

You'll find more discussion of the vulnerability management process in Chapter 16, “Managing Security Operations.”

Penetration Testing

The *penetration test* goes beyond vulnerability testing techniques because it actually attempts to exploit systems. Vulnerability scans merely probe for the presence of a vulnerability and do not normally take offensive action against the targeted system. (That said, some vulnerability scanning techniques may disrupt a system, although these options are usually disabled by default.) Security professionals performing penetration tests, on the other hand, try to defeat security controls and break into a targeted system or application to demonstrate the flaw.

Penetration tests require focused attention from trained security professionals, to a much greater extent than vulnerability scans. When performing a penetration test, the security professional typically targets a single system or set of systems and uses many different techniques to gain access. *NIST SP 800-115—Technical Guide to Information Security Testing and Assessment* defines the penetration testing process as consisting of the four phases illustrated in Figure 15.7:

- *Planning* includes agreement on the scope of the test and the rules of engagement. This is an extremely important phase because it ensures that both the testing team and management are in agreement about the nature of the test and that the test is explicitly authorized.
- *Discovery* includes (1) information gathering and scanning and (2) vulnerability analysis. It uses manual and automated tools to collect information about the target environment. This includes performing basic reconnaissance to determine system function (such as visiting websites hosted on the system) and conducting network discovery scans to identify

open ports. Testers also use automated tools during this phase to probe for system weaknesses using network vulnerability scans, web vulnerability scans, and database vulnerability scans.

- *Attack* seeks to use manual and automated exploit tools to attempt to defeat system security. This step is where penetration testing goes beyond vulnerability scanning, as vulnerability scans do not attempt to actually exploit detected vulnerabilities.
- *Reporting* summarizes the results of the penetration testing and makes recommendations for improvements to system security.

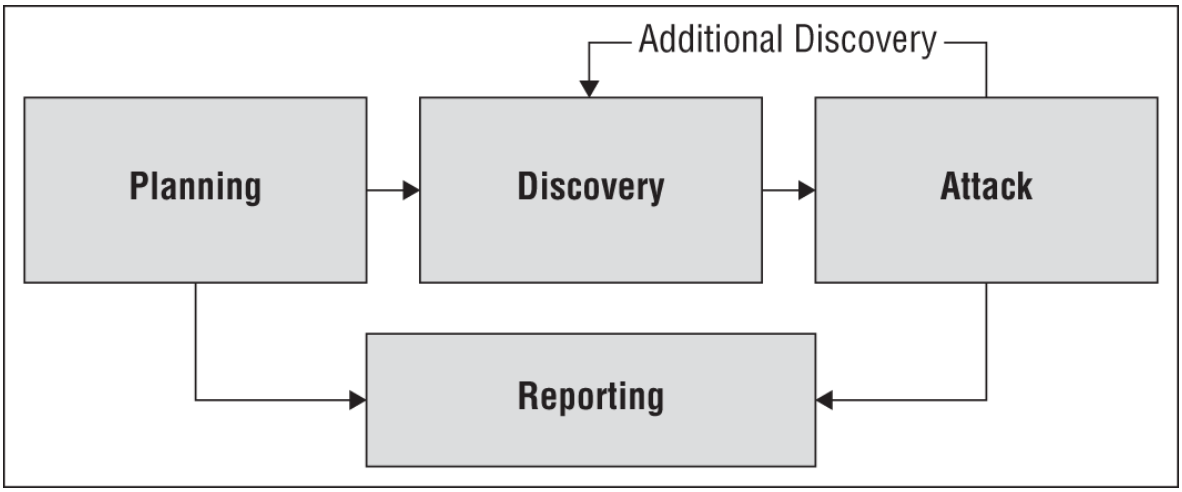

FIGURE 15.7 Penetration testing process

Penetration testers commonly use a tool called *Metasploit Framework* to automatically execute exploits against targeted systems. Metasploit Framework, shown in Figure 15.8, uses a scripting language to allow the automatic execution of common attacks, saving testers (and malicious actors) quite a bit of time by eliminating many of the tedious, routine steps involved in executing an attack.

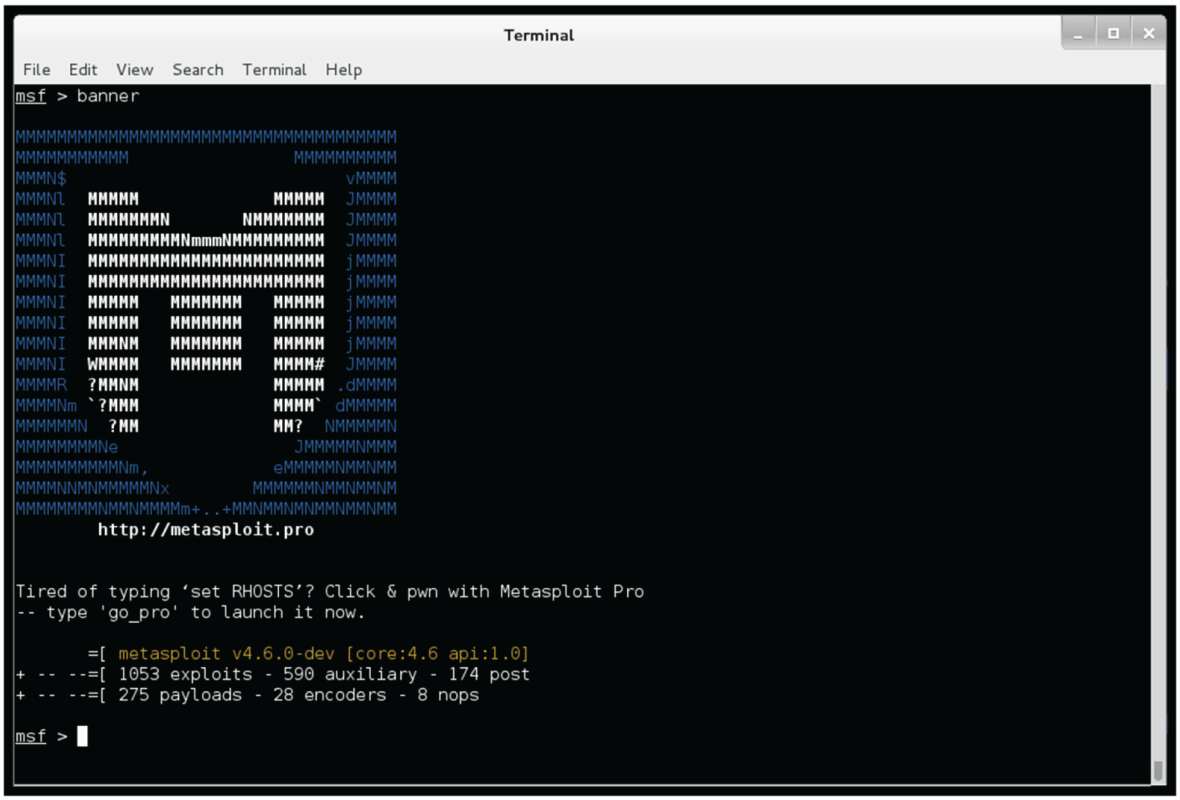

FIGURE 15.8 The Metasploit Framework automated system exploitation tool allows attackers to quickly execute common attacks against target systems.

Penetration testers may be company employees who perform these tests as part of their duties or external consultants hired to perform penetration tests. The tests are normally categorized into three groups:

White-Box Penetration Test Provides the attackers with detailed information about the systems they target. This bypasses many of the reconnaissance steps that normally precede attacks, shortening the time of the attack and increasing the likelihood that it will find security flaws. These tests are sometimes called "known environment" tests.

Gray-Box Penetration Test Also known as partial knowledge tests, these are sometimes chosen to balance the advantages and disadvantages of white- and black-box penetration tests. This is particularly common when black-box results are desired but costs or time constraints mean that some knowledge is needed to complete the testing. These tests are sometimes called "partially known environment" tests.

Black-Box Penetration Test Does not provide attackers with any information prior to the attack. This simulates an external attacker trying to gain access to information about the business and technical environment before engaging in an attack. These tests are sometimes called "unknown environment" tests.

Organizations performing penetration testing should be careful to ensure that they understand the hazards of the testing itself. Penetration tests seek to exploit vulnerabilities and consequently may disrupt system access or corrupt data stored in systems. This is one of the major reasons that it is important to clearly outline the rules of engagement during the planning phase of the test as well as have complete authorization from a senior management level prior to starting any penetration testing.

Breach and Attack Simulations

Breach and attack simulation (BAS) platforms seek to automate some aspects of penetration testing. These systems are designed to inject threat indicators onto systems and networks in an effort to trigger other security controls. For example, a BAS platform might place a suspicious file on a server, send beaconing packets over a network, or probe systems for known vulnerabilities.

In a well-functioning security program, detection and prevention controls would immediately detect and/or block this traffic as potentially malicious. The BAS platform is not actually waging attacks, but it is conducting automated testing of those security controls to identify deficiencies that may indicate the need for control updates or enhancements.

Penetration tests are time-consuming and require specialized resources, but they play an important role in the ongoing operation of a sound information security testing program.

There are many industry-standard penetration testing methodologies that make a good starting point when designing your own program. Consider using the OWASP Web Security Testing Guide, *Open Source Security Testing Methodology Manual* (OSSTMM), NIST 800-115— FedRAMP Penetration Test Guidance, or PCI DSS Information Supplement on Penetration Test Guidance v3 as references.

Compliance Checks

Organizations find themselves subject to a wide variety of compliance requirements. You learned about many of these laws and regulations in Chapter 4.

Savvy organizations create and maintain compliance plans documenting each of their regulatory obligations and map those to the specific security controls designed to satisfy each objective.

Compliance checks are an important part of security testing and assessment programs for regulated firms. These checks verify that all of the controls listed in a compliance plan are functioning properly and are effectively meeting regulatory requirements. Performing these checks on a periodic basis maintains the health of the organization's compliance program and avoids unforeseen regulatory issues.

Testing Your Software

Software is a critical component in system security. Think about the following characteristics common to many applications in use throughout the modern enterprise:

- Software applications often have privileged access to the operating system, hardware, and other resources.
- Software applications routinely handle sensitive information, including credit card numbers, Social Security numbers, and

proprietary business information.

- Many software applications rely on databases that also contain sensitive information.
- Software is the heart of the modern enterprise and performs business-critical functions. Software failures can disrupt businesses with very serious consequences.

Those are just a few of the many reasons that careful testing of software is essential to the confidentiality, integrity, and availability requirements of every modern organization.

Software should be designed in a manner that considers the possible threats to these objectives and responds appropriately. One of the core design principles supporting this goal is that software should never depend on users behaving properly. Instead, software should expect the unexpected and gracefully handle invalid input, improperly sequenced activity, and other unanticipated situations. This process of handling unexpected activity is known as *exception handling*.

In this section, you'll learn about the many types of software testing that you can integrate into your organization's software development life cycle.

This chapter provides coverage of software testing topics. You'll find deeper coverage of the software development life cycle (SDLC) and software security issues in Chapter 20, “Software Development Security.”

Code Review and Testing

One of the most critical components of a software testing program is conducting code review and testing. These procedures provide third-party reviews of the work performed by developers before moving code into a production environment. Code reviews and tests may discover security, performance, or reliability flaws

in applications before they go live and negatively impact business operations.

You will learn more about how code review and testing fits into the software development life cycle in Chapter 20.

Code Review

Code review is the foundation of software assessment programs. During a code review, also known as a *peer review*, developers other than the one who wrote the code review it for defects. Code reviews may result in approval of an application's move into a production environment, or they may send the code back to the original developer with recommendations for rework of issues detected during the review.

Code review takes many different forms and varies in formality from organization to organization. The most formal code review processes, known as Fagan inspections, follow a rigorous review and testing process with six steps:

1. Planning
2. Overview
3. Preparation
4. Inspection
5. Rework
6. Follow-up

An overview of the Fagan inspection appears in Figure 15.9. Each of these steps has well-defined entry and exit criteria that must be met before the process can formally transition from one stage to the next.

The Fagan inspection level of formality is normally found only in highly restrictive environments where code flaws may have catastrophic impact. Most organizations use less rigorous processes, using code peer review measures that include the following:

- Developers walking through their code in a meeting with one or more other team members
- A senior developer performing manual code review and signing off on all code before moving the code to production
- Use of automated review tools to detect common application flaws before moving the code to production

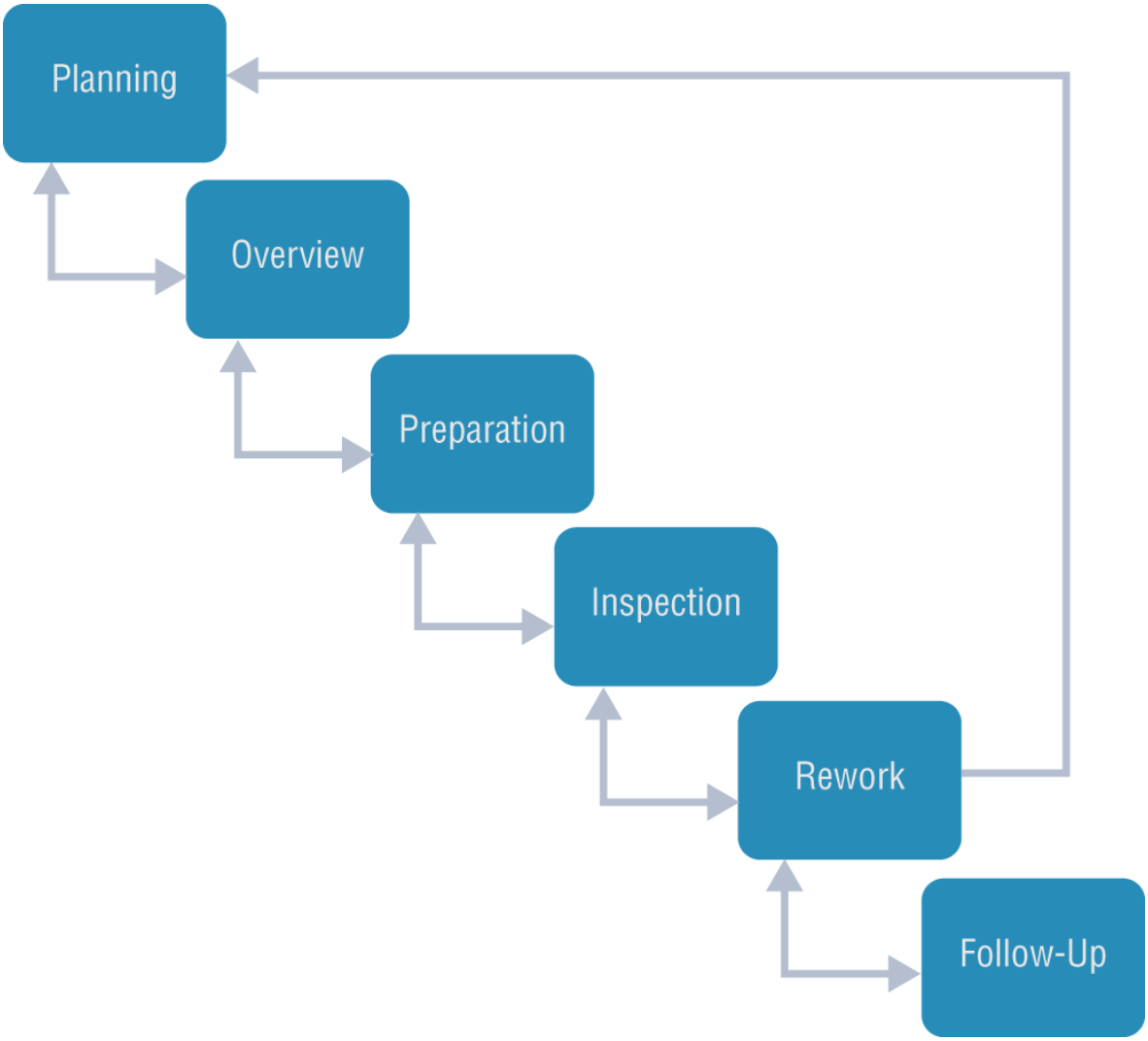

FIGURE 15.9 Fagan inspections follow a rigid formal process, with defined entry and exit criteria that must be met before transitioning between stages.

Each organization should adopt a code review process that suits its business requirements and software development culture.

Static Testing

Static application security testing (SAST) evaluates the security of software without running it by analyzing either the source code or the compiled application. Static analysis usually involves the use of automated tools designed to detect common software flaws, such as buffer overflows. In mature development environments, application developers are given access to static analysis tools and use them throughout the design, build, and test process.

Dynamic Testing

Dynamic application security testing (DAST) evaluates the security of software in a runtime environment and is often the only option for organizations deploying applications written by someone else. In those cases, testers often do not have access to the underlying source code. One common example of dynamic software testing is the use of web application scanning tools to detect the presence of cross-site scripting, SQL injection, or other flaws in web applications. Dynamic tests on a production environment should always be carefully coordinated to avoid an unintended interruption of service.

Dynamic testing may include the use of *synthetic transactions* and *benchmarks* to verify system performance. Synthetic transactions are scripted transactions with known expected results. The testers run the synthetic transactions against the tested code and then compare the output of the transactions to the expected state. Any deviations between the actual and expected results represent possible flaws in the code and must be further investigated.

Benchmarks are predefined standards or baseline values against which the performance, efficiency, or effectiveness of a system is measured. They provide a set of criteria that a system should meet or exceed to be considered adequately performing. Benchmarks often involve specific performance metrics such as response time, throughput, error rates, and resource utilization, among others. These metrics are used to evaluate the system's behavior under normal conditions or stress, such as high traffic

or data volume. By comparing the system's performance with these benchmarks, testers can determine whether the system is performing within acceptable parameters or if there are performance issues that need to be addressed.

Two terms you might encounter when dealing with code review and testing are IAST and RASP. Interactive application security testing (IAST) performs real-time analysis of runtime behavior, application performance, HTTP/HTTPS traffic, frameworks, components, and backend connections. Runtime Application Self-Protection (RASP) is a tool that runs on a server and intercepts calls to and from an application and validates data requests.

Ethical Disclosure

While conducting security testing, cybersecurity professionals may discover previously unknown vulnerabilities in products or systems operated by other vendors. They may implement compensating controls to correct these situations but find themselves unable to correct the underlying issue because it resides in code outside of their control.

The security community embraces the concept of *ethical disclosure*. This principle says that security professionals who detect a vulnerability have a responsibility to report that vulnerability to the vendor, providing them with an opportunity to develop a patch or other remediation to protect their customers.

This disclosure should first be made privately to the vendor, allowing them to correct the problem before it becomes public knowledge. However, the ethical disclosure principle also suggests that those reporting a vulnerability should provide the vendor with a reasonable amount of time to correct the vulnerability and, if it is not corrected, then publicly disclose the vulnerability so that other security professionals may make informed decisions about their future use of the product.

Fuzz Testing

Fuzz testing is a specialized dynamic testing technique that provides many different types of input to software to stress its limits and find previously undetected flaws. Fuzz testing software supplies invalid input to the software, either randomly generated or specially crafted to trigger known software vulnerabilities. The fuzz tester then monitors the performance of the application, watching for software crashes, buffer overflows, or other undesirable and/or unpredictable outcomes.

There are two main categories of fuzz testing:

Mutation (Dumb) Fuzzing Takes previous input values from actual operation of the software and manipulates (or mutates) it to create fuzzed input. It might alter the characters of the content, append strings to the end of the content, or perform other data manipulation techniques.

Generational (Intelligent) Fuzzing Develops data models and creates new fuzzed input based on an understanding of the types of data used by the program.

The zzuf tool automates the process of mutation fuzzing by manipulating input according to user specifications. For example, Figure 15.10 shows a file containing a series of 1s.

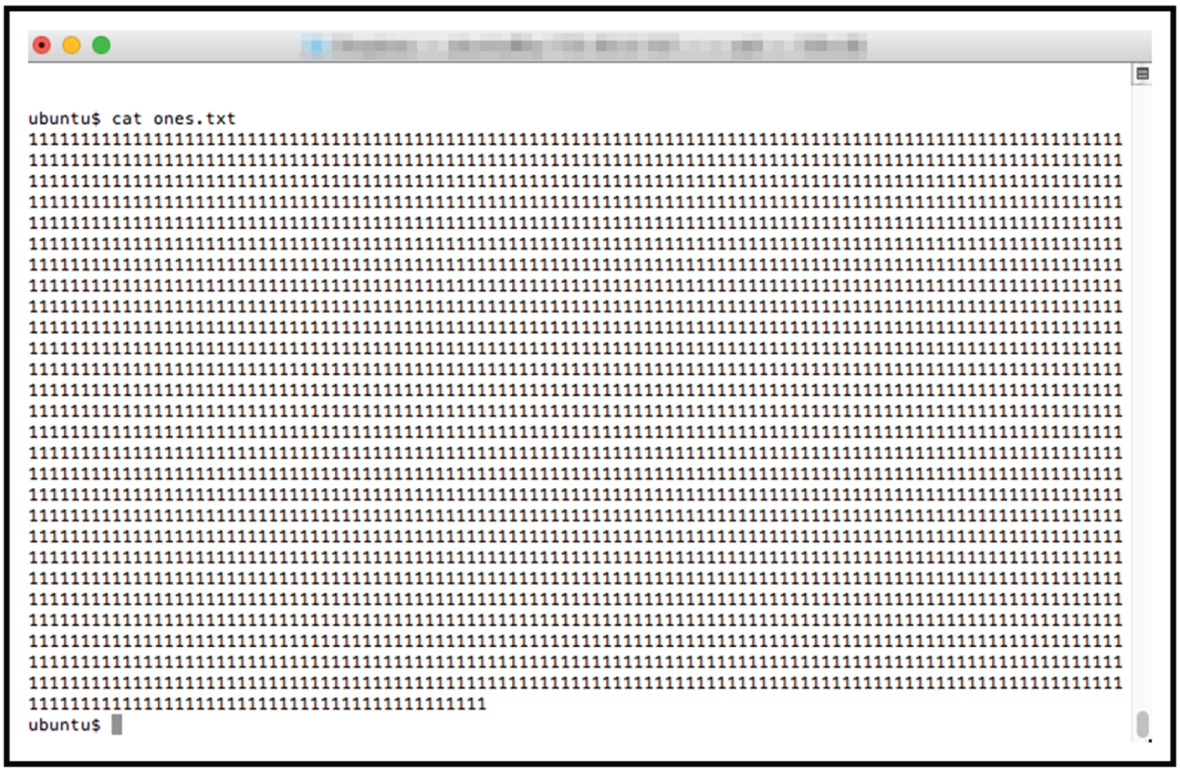

FIGURE 15.10 Prefuzzing input file containing a series of 1s

Figure 15.11 shows the zzuf tool applied to that input. The resulting fuzzed text is almost identical to the original text. It still contains mostly 1s, but it now has several changes made to the text that might confuse a program expecting the original input. This process of slightly manipulating the input is known as *bit flipping*.

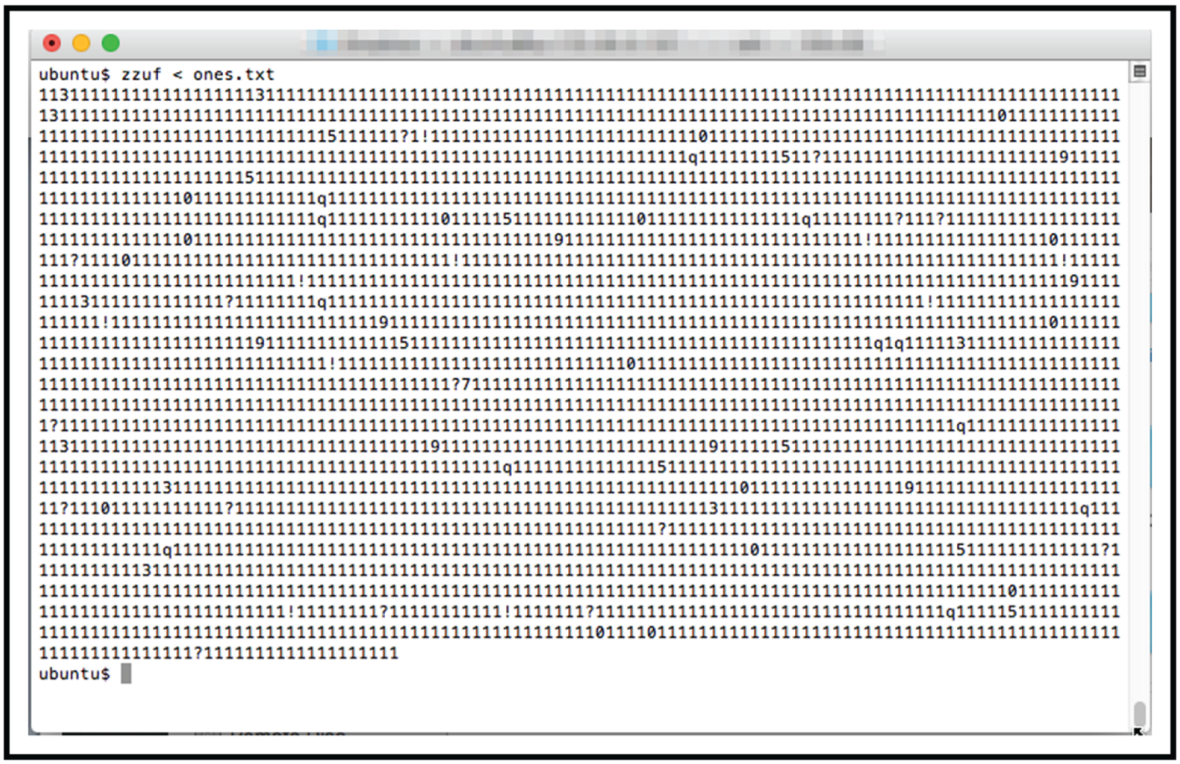

FIGURE 15.11 The input file from Figure 15.10 after being run through the zzuf mutation fuzzing tool

Fuzz testing is an important tool, but it does have limitations. Fuzz testing typically doesn't result in full coverage of the code and is commonly limited to detecting simple vulnerabilities that do not require complex manipulation of business logic. For this reason, fuzz testing should be considered only one tool in a suite of tests performed, and it is useful to conduct test coverage analysis (discussed later in this chapter) to determine the full scope of the test.

Interface Testing

Interface testing is an important part of the development of complex software systems. In many cases, multiple teams of developers work on different parts of a complex application that must function together to meet business objectives. The handoffs between these separately developed modules use well-defined interfaces so that the teams may work independently. Interface testing assesses the performance of modules against the interface specifications to ensure that they will work together properly when all the development efforts are complete.

Four types of interfaces should be tested during the software testing process:

Application Programming Interfaces (APIs) Offer a standardized way for code modules to interact and may be exposed to the outside world through web services. Developers must test APIs to ensure that they enforce all security requirements.

User Interfaces (UIs) Examples include graphical user interfaces (GUIs) and command-line interfaces. UIs provide end users with the ability to interact with the software. Interface tests should include reviews of all user interfaces to verify that they function properly.

Network Interfaces Serve as crucial communication gateways that allow different systems to connect and interact over various types of networks, such as local area networks (LANs), wide area networks (WANs), or the Internet. These interfaces handle the data exchange between systems, and their functionality is paramount in ensuring that data is transmitted accurately, securely, and efficiently. Testing of network interfaces should include validating the robustness of the connection, data transmission speed, security protocols used for data encryption, authentication processes, and error handling mechanisms. This testing is essential, especially in distributed applications, to ensure data integrity, confidentiality, and system interoperability across different network environments.

Physical Interfaces Exist in some applications that manipulate machinery, logic controllers, or other objects in the physical world. Software testers should pay careful attention to physical interfaces because of the potential consequences if they fail.

Interfaces provide important mechanisms for the planned or future interconnection of complex systems. The modern digital world depends on the availability of these interfaces to facilitate interactions between disparate software packages. However, developers must be careful that the flexibility provided by interfaces does not introduce additional security risk. Interface

testing provides an added degree of assurance that interfaces meet the organization's security requirements.

Misuse Case Testing

In some applications, there are clear examples of ways that software users might attempt to misuse the application. For example, users of banking software might try to manipulate input strings to gain access to another user's account. They might also try to withdraw funds from an account that is already overdrawn. Software testers use a process known as *misuse case testing* or *abuse case testing* to evaluate the vulnerability of their software to these known risks.

In misuse case testing, testers first enumerate the known misuse cases. They then attempt to exploit those misuse cases with manual and/or automated attack techniques.

Test Coverage Analysis

Testing is an important part of any software development process, but it is unfortunately impossible to completely test any piece of software. There are simply too many ways that software might malfunction or undergo attack. Software testing professionals often conduct a *test coverage analysis* to estimate the degree of testing conducted against the new software. The test coverage is computed using the following formula:

$$test\ coverage = \left(number\ of\ use\ cases\ tested\ /\ total\ number\ of\ use\ cases\right) * 100$$

Of course, this is a highly subjective calculation. Accurately computing test coverage requires enumerating the possible use cases, which is an exceptionally difficult task. Therefore, anyone using test coverage calculations should take care to understand the process used to develop the input values when interpreting the results.

The test coverage analysis formula may be adapted to use many different criteria. Here are five common criteria:

- *Branch coverage*: Has every `if` statement been executed under all `if` and `else` conditions?
- *Condition coverage*: Has every logical test in the code been executed under all sets of inputs?
- *Function coverage*: Has every function in the code been called and returned results?
- *Loop coverage*: Has every loop in the code been executed under conditions that cause code execution multiple times, only once, and not at all?
- *Statement coverage*: Has every line of code been executed during the test?

Website Monitoring

Security professionals also often become involved in the ongoing monitoring of websites for performance management, troubleshooting, and the identification of potential security issues. This type of monitoring comes in two different forms:

- *Passive monitoring* analyzes actual network traffic sent to a website by capturing it as it travels over the network or reaches the server. This provides real-world monitoring data that gives administrators insight into what is actually happening on a network. *Real user monitoring (RUM)* is a variant of passive monitoring where the monitoring tool reassembles the activity of individual users to track their interaction with a website.
- *Synthetic monitoring* (or *active monitoring*) performs artificial transactions against a website to assess performance. This may be as simple as requesting a page from the site to determine the response time, or it may execute a complex script designed to identify the results of a transaction.

These two techniques are often used in conjunction with each other because they achieve different results. Passive monitoring is only able to detect issues after they occur for a real user

because it is monitoring real user activity. Passive monitoring is particularly useful for troubleshooting issues identified by users because it allows the capture of traffic related to that issue. Synthetic monitoring may miss issues experienced by real users if they are not included in the testing scripts, but it is capable of detecting issues before they actually occur.

Training and Exercises

Organizations conduct a wide variety of training programs designed to help employees understand their cybersecurity role. Information security professionals often participate in training programs that are set up as exercises using a competition-style format, pitting a team of attackers against a team of defenders.

Running exercises helps to identify vulnerabilities in the organization's systems, networks, and applications, similar to the results achieved from penetration testing. Exercises also provide employees with hands-on experience both attacking and defending systems. This helps boost cybersecurity skills and awareness among the technical staff.

When conducting an exercise, participants are often divided into three teams:

- *Red team* members are the attackers who attempt to gain access to systems.
- *Blue team* members are the defenders who must secure systems and networks from attack. The blue team also monitors the environment during the exercise, conducting active defense techniques. The blue team commonly gets a head start with some time to secure systems before the attack phase of the exercise begins.
- *White team* members are the observers and judges. They serve as referees to settle disputes over the rules and watch the exercise to document lessons learned from the test. The white team is able to observe the activities of both the red and blue teams and is also responsible for ensuring that the exercise does not cause production issues.

Purple Teaming

At the end of an exercise, it's common to bring the red and the blue teams together to share information about tactics and lessons learned. Each team walks the other through their role in the exercise, helping everyone learn from the process. This combination of knowledge from the red and blue teams is often referred to as *purple teaming*, because combining red and blue makes purple.

Capture the flag (CTF) exercises are a fun way to achieve training objectives. In a CTF exercise, the red team begins with set objectives, such as disrupting a website, stealing a file from a secured system, or causing other security failures. The exercise is scored based on how many objectives the red team was able to achieve compared to how many the blue team prevented them from executing.

Exercises don't need to take place using production systems. In many cases, an organization might set up a special environment solely for the purpose of the exercise. This provides a safe playground for the test and minimizes the probability that an attack will damage production systems. Other exercises may not even use real systems at all. *Tabletop exercises* simply gather participants in the same room to walk through their response to a fictitious exercise scenario.

Implementing Security Management Processes and Collecting Security Process Data

In addition to performing assessments and testing, sound information security programs also include a variety of management processes designed to oversee the effective operation of the information security program. These processes are a critical feedback loop in the security assessment process because they provide management oversight and have a deterrent

effect against the threat of insider attacks. They also provide an opportunity to collect security process data for use in other security tasks.

The security management reviews that fill this need include log reviews, account management, backup verification, and key performance and risk indicators. Each of these reviews should follow a standardized process that includes management approval at the completion of the review.

Log Reviews

In Chapter 16, you will learn the importance of storing log data and conducting both automated and manual log reviews. Security information and event management (SIEM) packages play an important role in these processes, automating much of the routine work of log review. These devices collect information using the syslog functionality present in many devices, operating systems, and applications. Some devices, including Windows systems, may require third-party clients to add syslog support. Administrators may choose to deploy logging policies through Windows Group Policy Objects (GPOs) and other mechanisms that can deploy and enforce standard policies throughout the organization.

Logging systems should also make use of the Network Time Protocol (NTP) to ensure that clocks are synchronized on systems sending log entries to the SIEM as well as the SIEM itself. This ensures that information from multiple sources has a consistent timeline.

Information security managers should also periodically conduct log reviews, particularly for sensitive functions, to ensure that privileged users are not abusing their privileges. For example, if an information security team has access to eDiscovery tools that allow searching through the contents of individual user files, security managers should routinely review the logs of actions taken by those administrative users to ensure that their file access relates to legitimate eDiscovery initiatives and does not violate user privacy.

Network flow (NetFlow) logs are particularly useful when investigating security incidents. These logs provide records of the connections between systems and the amount of data transferred.

Account Management

Account management reviews ensure that users only retain authorized permissions and that unauthorized modifications do not occur. Account management reviews may be a function of information security management personnel or internal auditors.

One way to perform account management is to conduct a full review of all accounts. This is typically done only for highly privileged accounts because of the amount of time consumed. The exact process may vary from organization to organization, but here's one example:

1. Managers ask system administrators to provide a list of users with privileged access and the privileged access rights. They may monitor the administrator as they retrieve this list to avoid tampering.
2. Managers ask the privilege approval authority to provide a list of authorized users and the privileges they should be assigned.
3. The managers then compare the two lists to ensure that only authorized users retain access to the system and that the access of each user does not exceed their authorization.

This process may include many other checks, such as verifying that terminated users do not retain access to the system, checking the paper trail for specific accounts, or other tasks.

Organizations that do not have time to conduct this thorough process may use sampling instead. In this approach, managers pull a random sample of accounts and perform a full verification

of the process used to grant permissions for those accounts. If no significant flaws are found in the sample, they make the assumption that this is representative of the entire population.

Sampling only works if it is random! Don't allow system administrators to generate the sample or use nonrandom criteria to select accounts for review, or you may miss entire categories of users where errors may exist.

Organizations may also automate portions of their account review process. Many identity and access management (IAM) vendors provide account review workflows that prompt administrators to conduct reviews, maintain documentation for user accounts, and provide an audit trail demonstrating the completion of reviews.

Disaster Recovery and Business Continuity

In Chapter 3, "Business Continuity Planning," you learned how organizations design continuity controls to maintain operations in the face of potential disruptions. In Chapter 18, "Disaster Recovery Planning," you will learn the importance of supplementing those continuity controls with disaster recovery programs that help organizations resume operations quickly after a disruption.

Consistent backup programs are an extremely important component of these efforts. Managers should periodically inspect the results of backups to verify that the process functions effectively and meets the organization's data protection needs. This may involve reviewing logs, inspecting hash values, or requesting an actual restore of a system or file.

Regular testing of disaster recovery and business continuity controls provides organizations with the assurance that they are effectively protected against disruptions to business operations.

Training and Awareness

Training and awareness programs play a crucial role in preparing an organization's workforce to support information security programs. These efforts educate employees about current threats and advise them on best practices for protecting information and systems under their care from attack.

These programs should begin with initial training designed to provide foundational knowledge to employees who are either joining the organization for the first time or moving into a new role with different security responsibilities. This initial training should be tailored to an individual's role, providing them with the specific, actionable information that they need to carry out their security responsibilities.

Recurring training and awareness efforts should take place throughout the year, reminding employees of their responsibilities and updating them on changes to the organization's operating environment and the threat landscape.

Many organizations use *phishing simulations* to evaluate the effectiveness of their security awareness programs. These simulations use fake phishing messages to determine whether users are susceptible to phishing attacks. Users who click the link or otherwise respond to the simulated attacks are redirected to training resources to help them better identify suspicious activity.

You'll find complete coverage of security training and awareness programs in Chapter 2, “Personnel Security and Risk Management Concepts.”

Key Performance and Risk Indicators

Security managers should also monitor key performance and risk indicators on an ongoing basis. The exact metrics they monitor will vary from organization to organization but may include the following:

- Number of open vulnerabilities
- Time to resolve vulnerabilities

- Vulnerability/defect recurrence
- Number of compromised accounts
- Number of software flaws detected in preproduction scanning
- Repeat audit findings
- User attempts to visit known malicious sites

Once an organization identifies the key security metrics it wishes to track, managers may want to develop a dashboard that clearly displays the values of these metrics over time and display it where both managers and the security team will regularly see it, such as on an intranet.

Summary

Security assessment and testing programs play a critical role in ensuring that an organization's security controls remain effective over time. Changes in business operations, the technical environment, security risks, and user behavior may alter the effectiveness of controls that protect the confidentiality, integrity, and availability of information assets. Assessment and testing programs monitor those controls and highlight changes requiring administrator intervention. Security professionals should carefully design their assessment and testing program and revise it as business needs change.

Security tests verify that a control is functioning properly. With vulnerability assessments, security professionals perform a variety of tests to identify misconfigurations and other security flaws in systems and applications. Network discovery tests identify systems on the network with open ports. Network vulnerability scans discover known security flaws on those systems. Web vulnerability scans probe the operation of web applications searching for known vulnerabilities.

Software plays a critical role in any security infrastructure because it handles sensitive information and interacts with critical resources. Organizations should use a code review process

to allow peer validation of code before moving it to production. Rigorous software testing programs also include the use of static testing, dynamic testing, fuzz testing, interface testing, and misuse case testing to robustly evaluate software.

Security management processes include log reviews, account management, disaster recovery and business continuity, backup verification, training and awareness, and tracking of key performance and risk indicators. These processes help security managers validate the ongoing effectiveness of the information security program. They are complemented by formal internal and external audits performed by third parties on a less frequent basis.

Exam Essentials

Understand the importance of security assessment and testing programs. Security assessment and testing programs provide an important mechanism for validating the ongoing effectiveness of security controls. They include a variety of tools, such as vulnerability assessments, penetration tests, software testing, audits, and security management tasks designed to validate controls. Every organization should have a security assessment and testing program defined and operational.

Conduct vulnerability assessments and penetration tests. Vulnerability assessments use automated tools to search for known vulnerabilities in systems, applications, and networks. These flaws may include missing patches, misconfigurations, or faulty code that exposes the organization to security risks. Penetration tests also use these same tools but supplement them with attack techniques where an assessor attempts to exploit vulnerabilities and gain access to the system. Vulnerability management programs take the results of these tests as inputs and then implement a risk management process for identified vulnerabilities.

Perform software testing to validate code moving into production. Software testing techniques verify that code functions as designed and does not contain security flaws. Code

review uses a peer review process to formally or informally validate code before deploying it in production. Interface testing assesses the interactions between components and users with API testing, user interface testing, and physical interface testing.

Understand the difference between static and dynamic software testing. Static software testing techniques, such as code reviews, evaluate the security of software without running it by analyzing either the source code or the compiled application. Dynamic testing evaluates the security of software in a runtime environment and is often the only option for organizations deploying applications written by someone else.

Explain the concept of fuzzing. Fuzzing uses modified inputs to test software performance under unexpected circumstances. Mutation fuzzing modifies known inputs to generate synthetic inputs that may trigger unexpected behavior. Generational fuzzing develops inputs based on models of expected inputs to perform the same task.

Perform security management tasks to provide oversight to the information security program. Security managers must perform a variety of activities to retain proper oversight of the information security program. Log reviews, particularly for administrator activities, ensure that systems are not misused. Account management reviews ensure that only authorized users retain access to information systems. Backup verification ensures that the organization's data protection process is functioning properly. Management should also monitor other security functions, such as disaster recovery, business continuity, and awareness/training programs. Key performance and risk indicators provide a high-level view of security program effectiveness.

Conduct or facilitate internal, external, and third-party audits. Security audits occur when a third party performs an assessment of the security controls protecting an organization's information assets. Internal audits are performed by an organization's internal staff and are intended for management use. External audits are performed by a third-party audit firm and are generally intended for the organization's governing body.

Collect logs and security process data. Many components of the information security program generate data that is crucial to security assessment processes. These components include the account management process, management review and approval, key performance and risk indicators, backup verification data, training and awareness metrics, and the data generated by disaster recovery and business continuity programs.

Know how to use cybersecurity exercises to ensure that teams are prepared for security incidents. Exercises are designed to test the skills of security professionals. Blue teams are responsible for managing the organization's defenses. Offensive hacking is used by red teams as they attempt to gain access to systems on the target network. White teams serve as the neutral moderators of the exercise. Purple teaming is conducted after an exercise to bring together the red and blue teams for knowledge sharing.

Written Lab

1. Describe the difference between TCP SYN scanning and TCP connect scanning.
2. What are the five port status values returned by the Nmap network discovery scanning tool?
3. What is the difference between static and dynamic code testing techniques?
4. What is the difference between mutation fuzzing and generational fuzzing?

Review Questions

1. Which one of the following tools is used primarily to perform network discovery scans?

   A. Nmap

   B. Nessus

   C. Metasploit