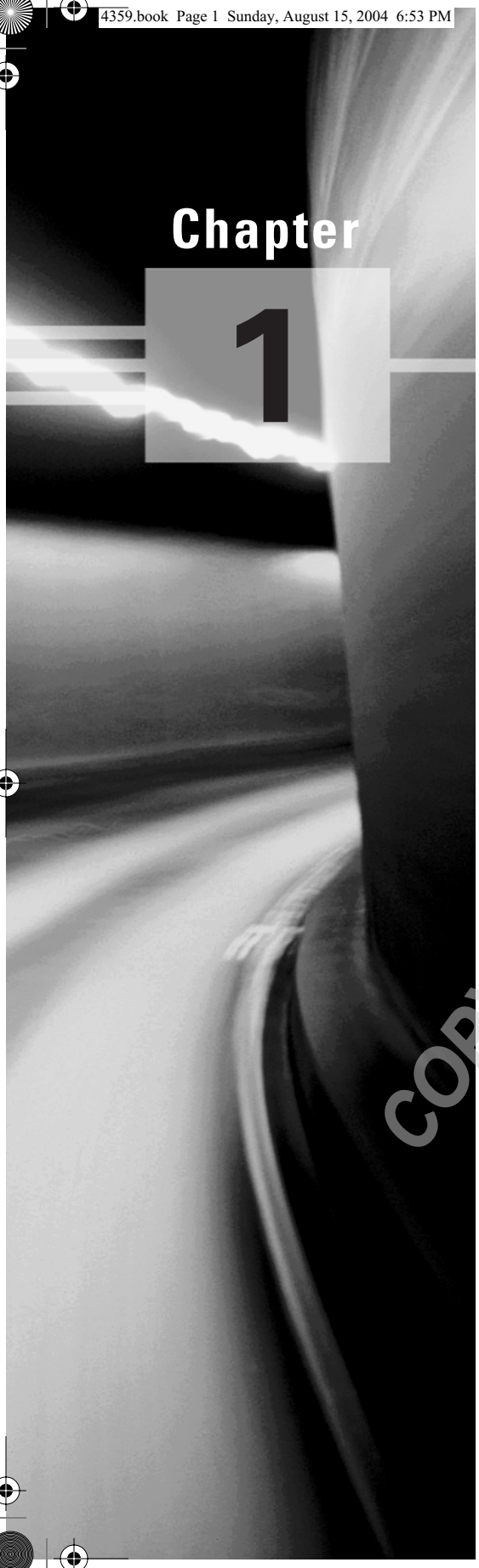
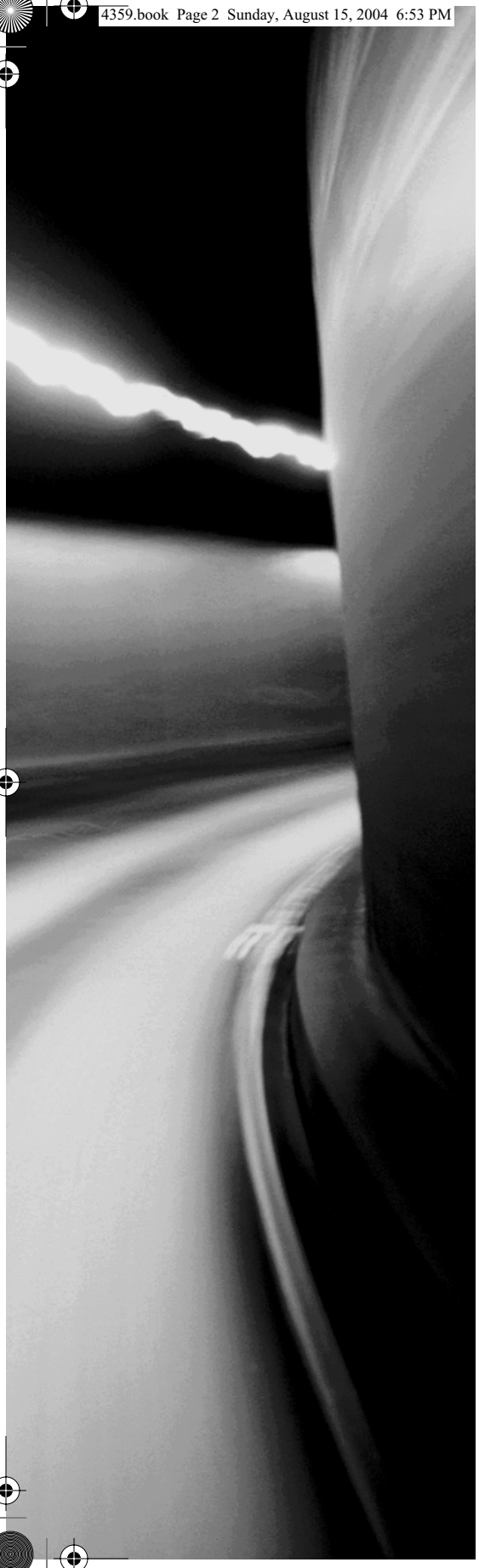


Chapter 1

General Security Concepts

COMPTIA SECURITY+ EXAM OBJECTIVES COVERED IN THIS CHAPTER:

- ✓ **1.1 Recognize and be able to differentiate and explain the following access control models**
  - MAC (Mandatory Access Control)
  - DAC (Discretionary Access Control)
  - RBAC (Role Based Access Control)
- ✓ **1.2 Recognize and be able to differentiate and explain the following methods of authentication**
  - Kerberos
  - CHAP (Challenge Handshake Authentication Protocol)
  - Certificates
  - Username/Password
  - Tokens
  - Multi-factor
  - Mutual
  - Biometrics
- ✓ **1.3 Identify non-essential services and protocols and know what actions to take to reduce the risks of those services and protocols**
- ✓ **1.4 Recognize the following attacks and specify the appropriate actions to take to mitigate vulnerability and risk**
  - DOS/DDOS (Denial of Service/Distributed Denial of Service)
  - Back Door
  - Spoofing



- Man in the Middle
- Replay
- TCP/IP Hijacking
- Weak Keys
- Mathematical
- Social Engineering
- Birthday
- Password Guessing
  - Brute Force
  - Dictionary
- Software Exploitation

✓ **1.5 Recognize the following types of malicious code and specify the appropriate actions to take to mitigate vulnerability and risk**

- Viruses
- Trojan Horses
- Logic Bombs
- Worms

✓ **1.6 Understand the concept of and know how reduce the risks of social engineering**

✓ **1.7 Understand the concept and significance of auditing, logging and system scanning**

The Security+ exam will test your basic IT security skills—those skills needed to effectively secure stand-alone and networked systems in a corporate environment. To pass the test and be effective in implementing security, you need to understand the basic concepts and terminology in this chapter. This chapter covers all of the seven major sections of the first domain of the Security+ objectives. These sections focus on the following topics: access control models, authentication methods, non-essential services, common attacks, malicious code, social engineering, and system auditing.

1.1 Identifying Access Control Models

The full name of this section from the Security+ objectives list is "Recognize and be able to differentiate and explain the following access control models." The mechanism by which users are granted or denied the ability to interact with and use resources is known as *access control*. Access control is often referred to using the term *authorization*. Authorization defines the type of access to resources users are granted—in other words, what users are authorized to do. Authorization is often considered the next logical step immediately after authentication. *Authentication* is the proving of your identity to a system or the act of logging on. With proper authorization or access control, a system will properly control access to resources in order to prevent unauthorized access.

For more information on this topic, refer to Chapter 1 of the *Security+ Study Guide, Second Edition*, by Sybex.

Critical Information

There are three common access control methods:

- *Mandatory Access Control* (*MAC*)
- *Discretionary Access Control* (*DAC*)
- *Role-Based Access Control* (*RBAC*)

These three models are widely used in today's IT environments. Familiarity with these three models is essential to the Security+ exam.

These three access control models aren't mutually exclusive. Two or more of them can be deployed simultaneously in a single environment.

Mandatory Access Control (MAC)

MAC is a form of access control commonly employed by government and military environments. MAC specifies that access is granted based on a set of rules rather than at the discretion of a user. The rules that govern MAC are hierarchical in nature and are often called *sensitivity labels, security domains, or classifications*. MAC environments define a few specific security domains or sensitivity levels and then use the associated labels from those domains to impose access control on objects and subjects.

A government or military implementation of MAC typically includes five levels:

- Unclassified
- Sensitive but unclassified
- Confidential
- Secret
- Top secret

This list is in order from least sensitive to most sensitive. Objects or resources are assigned sensitivity labels corresponding to one of these security domains. Each specific security domain or level defines the security mechanisms and restrictions that must be imposed in order to provide protection for objects within that domain.

MAC can also be deployed in private sector or corporate business environments. Such cases typically involve four levels of security domains:

- Public
- Sensitive
- Private
- Confidential

This list is also in order from least sensitive to most sensitive.

The primary purpose of a MAC environment is to prevent *disclosure*: the violation of the security principle of *confidentiality*. When an unauthorized user gains access to a secured resource, this is a security violation. Objects are assigned a specific sensitivity label based on the damage that would be caused if disclosure occurred. For example, if a top secret resource was disclosed, it could case grave damage to national security.

A MAC environment works by assigning subjects a *clearance level* and objects a *sensitivity label*; in other words, everything is assigned a classification marker. *Subjects* or users are assigned clearance levels. The name of the clearance level is the same as the name of the sensitivity label assigned to *objects* or resources. A person (or other subject, such as a program or a computer system) must have the same or greater assigned clearance level as the resources they wish to

access. In this manner, access is granted or restricted based on the rules of classification (that is, sensitivity labels and clearance levels).

MAC is named as it is because the access control it imposes on an environment is mandatory. Its assigned classifications and the resulting granting and restriction of access can't be altered by users. Instead, the rules that define the environment and judge the assignment of sensitivity labels and clearance levels control authorization.

MAC isn't a very granularly controlled security environment. An improvement to MAC includes the use of *need to know*: a security restriction where some objects (resources or data) are restricted unless the subject has a need to know them. The objects that require specific need to know are assigned a sensitivity label, but they're compartmentalized from the rest of the objects with the same sensitivity label (within the same security domain). The need to know is a rule in and of itself, which states that access is only granted to subjects who have been assigned work tasks that require access to the cordoned-off object. Even if a subject has the proper level of clearance, without need to know, they are denied access. Need to know is the MAC equivalent of the principle of least privilege from DAC (see the following section).

The Trusted Computer System Evaluation Criteria (TCSEC), or Orange Book specifications, which defined government computer security classifications before December 2000 (when they were replaced by the Common Criteria), reference only MAC and DAC methods of access control.

Discretionary Access Control (DAC)

DAC is the form of access control or authorization that is used in most commercial and home environments. DAC is user directed or, more specifically, controlled by the owner and creators of the objects (resources) in the environment. DAC is identity based: Access is granted or restricted by an object's owner based on user identity and on the discretion of the object owner. Thus, the owner or creator of an object can decide whom to grant and deny access to their object.

DAC uses access control lists (ACLs). An *ACL* is a security logical device attached to every object and resource in the environment; it defines which users are granted or denied the various types of access available based on the object type. Individual user accounts or user groups can be added to an object's ACL and granted or denied access.

If your user account isn't granted access through an object's ACL, then you're denied access by default. If your user account is specifically granted access through an object's ACL, then you're granted the specific level or type of access defined. If your user account is specifically denied access through an object's ACL, then you're denied the specific level or type of access defined. In most cases, a Denied setting in an ACL overrides all other settings. Table 1.1 shows an access matrix for a user who is a member of three groups, and the resulting access to a folder on a network server. As you can see, the presence of the Denied setting overrides any other access granted from another group. Thus, if your membership in one user group grants you write access over an object, but another group specifically denies you write access to the same object, then you're denied write access to the object.

TABLE 1.1 Cumulative Access Based on Group Memberships

Sales Group	User Group	Research Group	Resulting Access
Change	Read	None specified	Change
Read	Read	Change	Change
None specified	Read	Denied	Denied
Full Control	Denied	None specified	Denied

The security concept of the *principle of least privilege* is connected to DAC. The principle of least privilege states that users should be granted only the minimum level of access that is necessary for them to perform their work tasks, and no more.

Role-Based Access Control (RBAC)

RBAC is another form of rules-based access control. It may be grouped with the nondiscretionary access control methods along with MAC. The rules used for RBAC are basically job descriptions: Users are assigned a specific role in an environment, and access to objects is granted based on the necessary work tasks of that role. For example, the role of backup operator may be granted the ability to back up every file on a system to a tape drive. The user given the backup operator role would then be able to perform that function.

RBAC is most suitable for environments with a high rate of employee turnover. It allows the job descriptions or roles to remain static even when the user performing that role changes often.

The acronym RBAC has a second definition. Rule-Based Access Control (RBAC) is typically used in relation to network devices that filter traffic based on filtering rules. Be sure you understand the context of the Security+ exam question before assuming *Role* or *Rule* when you see RBAC.

Exam Essentials

Mandatory Access Control (MAC) MAC is based on classification rules. Objects are assigned sensitivity labels. Subjects are assigned clearance labels. Users obtain access by having the proper clearance for the specific resource. Classifications are hierarchical.

Common MAC hierarchies Government or military MAC uses the following levels: unclassified, sensitive but unclassified, confidential, secret, and top secret. Private sector or corporate business environment MAC uses these: public, sensitive, private, and confidential.

Discretionary Access Control (DAC) DAC is based on user identity. Users are granted access through ACLs on objects, based on the discretion of the object's owner or creator.

Role-Based Access Control (RBAC) RBAC is based on job description. Users are granted access based on their assigned work tasks. RBAC is most suitable for environments with a high rate of employee turnover.

1.2 Identifying Authentication Methods

The full name of this section from the Security+ objectives list is "Recognize and be able to differentiate and explain the following methods of authentication." *Authentication* is the mechanism by which a person proves their identity to a system. Often the authentication process involves a simple username and password. But other more complex authentication factors or credential-protection mechanisms are involved in order to provide strong protection for the logon and account-verification process.

For more information on this topic, refer to Chapter 1 of the *Security+ Study Guide, Second Edition*, by Sybex.

Critical Information

Authentication is little more than the process of proving that a subject is the valid user of an account. The authentication process requires that the subject provide an identity and then proof of that identity. Identity proofing typically takes the form of one or more of the following three authentication factors:

- *Something you know* (such as a password)
- *Something you have* (such as a smartcard)
- *Something you are* (such as a fingerprint)

This section discusses the specific authentication factors of passwords, tokens, and biometrics. In addition, it addresses the concepts of multi-factor and mutual authentication.

The topic of authentication also includes the protection mechanisms used to secure the authentication credentials (identity claim and identity proofs) while they're in transit from the client to the authentication server. Three such mechanisms are addressed here: Kerberos, CHAP, and certificates.

Kerberos

Early authentication transmission mechanisms sent logon credentials from the client to the authentication server in clear text. Unfortunately, this solution is vulnerable to eavesdropping and interception, thus making the security of the system suspect (if not decimating it). What was needed was a solution that didn't transmit the logon credentials in a form that could be easily captured, extracted, and reused.

One such method for providing protection for logon credentials is *Kerberos*: a trusted third-party authentication protocol that was originally developed at MIT under Project Athena. The current version of Kerberos in widespread use is version 5. Kerberos is used to authenticate network principles (subjects) to other entities on the network (objects, resources, and servers). Kerberos is platform independent; however, some operating systems require special configuration adjustments to support true interoperability (for example, Windows Server 2003 or Windows 2000 Server with Unix).

Kerberos is a centralized authentication solution. The core element of a Kerberos solution is the Key Distribution Center (KDC), which is responsible for verifying the identity of principles and granting and controlling access within a network environment through the use of secure cryptographic keys and tickets.

Kerberos is a trusted third-party authentication solution because the KDC acts as a third party in the communications between a client and a server. Thus, if the client trusts the KDC and the server trusts the KDC, then the client and server can trust each other.

Kerberos is also a single sign-on solution. *Single sign-on* means that once a user (or other subject) is authenticated into the realm, they need not reauthenticate to access resources on any realm entity. (A *realm* is the network protected under a single Kerberos implementation.)

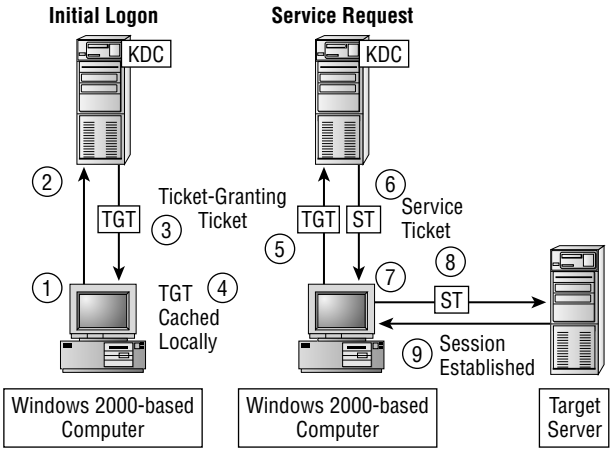
The basic process of Kerberos authentication is as follows:

1. The subject provides logon credentials.
2. The Kerberos client system encrypts the password with Data Encryption Standard (DES) and transmits the protected credentials to the KDC.
3. The KDC verifies the credentials and then creates a Ticket Granting Ticket (TGT—a hashed form of the subject's password with the addition of a timestamp that indicates a valid lifetime). The TGT is encrypted and sent to the client.
4. The client receives the TGT. At this point, the subject is an authenticated principle in the Kerberos realm.
5. The subject requests access to resources on a network server. This causes the client to request a Service Ticket (ST) from the KDC.
6. The KDC verifies that the client has a valid TGT and then issues an ST to the client. The ST includes a timestamp that indicates its valid lifetime.
7. The client receives the ST.
8. The client sends the ST to the network server that hosts the desired resource.
9. The network server verifies the ST. If it's verified, it initiates a communication session with the client. From this point forward, Kerberos is no longer involved.

Figure 1.1 shows the Kerberos authentication process.

The Kerberos authentication method helps to ensure that logon credentials aren't compromised while in transit from the client to the server. The inclusion of a timestamp in the tickets ensures that expired tickets can't be reused. This prevents replay and spoofing attacks against Kerberos.

FIGURE 1.1 The Kerberos authentication process



Kerberos supports mutual authentication (client and server identities are proven to each other). It's scalable and thus able to manage authentication for large networks. Being centralized, Kerberos helps reduce the overall time involved in accessing resources within a network.

Kerberos is used to provide security and protection for authentication credentials alone. It isn't used in any way to provide encryption or security for other types of data transfer.

Challenge Handshake Authentication Protocol (CHAP)

CHAP is an authentication protocol used primarily over dial-up connections (usually Point-to-Point Protocol [PPP]) as a means to provide a secure transport mechanism for logon credentials. It was developed as a secure alternative and replacement to Password Authentication Protocol (PAP), which transmitted authentication credentials in clear text.

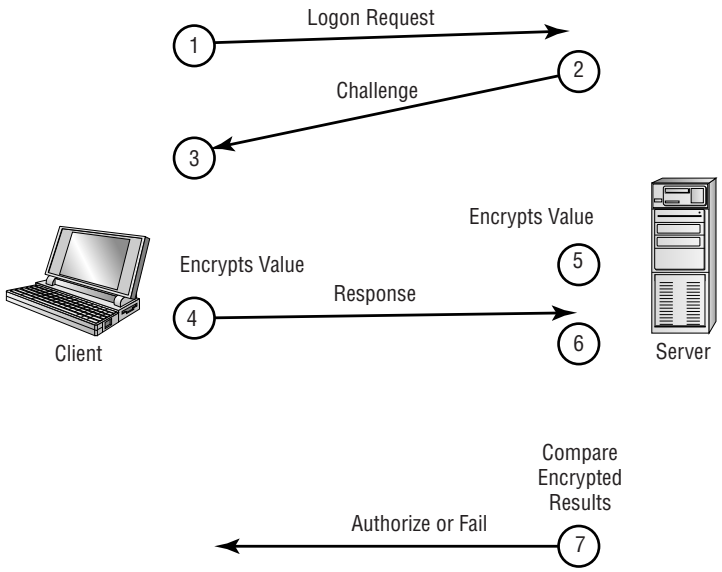
CHAP uses an initial authentication protection process to support logon and an ongoing verification process to ensure the subject/client is still who they claim to be. The process is as follows.

1. The initial authentication process performs a one-way hash function (specifically, MD5) on the subject's password and then passes the username and hash value to the authentication server.
2. The authentication server compares the username to its accounts database and the hash value to that stored for the identified user in its database.
3. If there is a match, the server transmits a challenge to the client.

4. The client produces the correct response and transmits it back to the server.
5. The server computes the response.
6. The server compares the response to that received by the client.
7. If everything matches, the subject is authenticated and allowed to communicate over the link.

Figure 1.2 shows the CHAP authentication process.

FIGURE 1.2 CHAP authentication



Once the client is authenticated, CHAP periodically sends a challenge to the client at random intervals. The client must compute the correct response to the issued challenge; otherwise the connection is automatically severed. This post-authentication verification process ensures that the authenticated session hasn't been hijacked.

Certificates

Certificates are another means of authentication. They typically involve a trusted third-party *certificate authority* (*CA*): a private or public entity that issues certificates to entities serving as either clients (subjects) or servers (objects). It's the responsibility of the CA to verify the identity of each entity before issuing a certificate. Once a certificate is issued, the entity can use the certificate as proof of their identity.

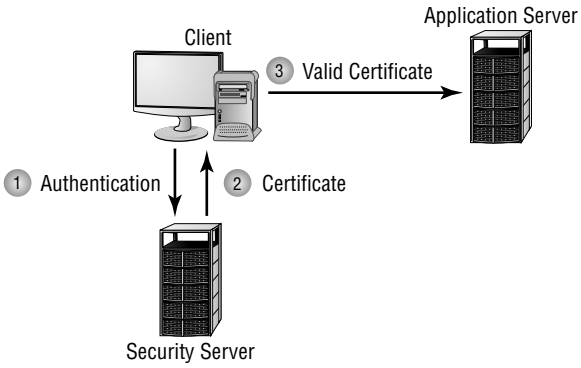
When you're using certificates in a private network, you are your own trusted third party (that is, you are your own CA). Such a private certificate solution is known as a Public Key

Infrastructure (PKI). A PKI is the definition (like a blueprint or schematic) of the mechanisms involved in implementing certificates. For the most part, you must deploy one or more servers with Certificate Services in order to create your own hierarchy of CAs.

Certificates can be used logically or electronically only (such as by the operating system or web browser directly) or via physical access control devices (such as a smartcard). In a logical deployment, certificates are installed into a client operating system or a specific application. Whenever identity proofing is required, the operating system transmits its certificate to the requesting party. In a physical deployment, the certificate is stored on a smartcard or some form of removable media. When the system needs user authentication, it requests the physical access control device.

When a user requests a certificate, they must usually provide proof of their identity along with their public key. This means they must use the same PKI solution as the CA issuing the certificate. The CA then uses the public key from the subject as the basis to generate the certificate returned to the user (see Figure 1.3). In this fashion, the certificate is tied to the subject's public key–private key pair and provides a mechanism for identity proofing.

FIGURE 1.3 A certificate being issued once identification has been verified



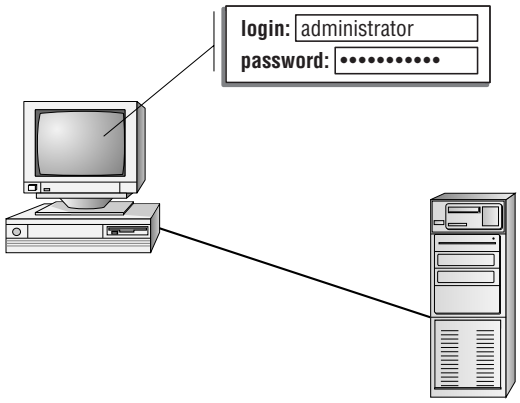
Certificates are commonly used over the Internet as a means of logical or electronic identity proofing. As long as both parties in a communication or transaction trust a specific third-party CA, such as VeriSign, then the two entities can trust that each is who they claim to be. It's very important to understand that a certificate only provides proof of identity, and that proof is based on trusting a third party (the CA that performed testing on the subject's identity). It in no way ensures that the subject or object so identified has benevolent motives or will function, perform, or operate in any specific manner.

Certificates are discussed in more depth in Chapter 4, "Basics of Cryptography."

Username/Password

The combination of a username and a *password* is the most common form of authentication (see Figure 1.4). If the provided password matches the password stored in a system's accounts database for the specified user, then that user is authenticated to the system. However, just because using a username and password is the most common form of authentication, that doesn't mean it's the most secure. On the contrary, it's generally considered to be the least secure form of authentication.

FIGURE 1.4 A basic logon process employing a username and password



Numerous means to improve the basic username/password combination have been developed. First is the storage of passwords in an accounts database in an encrypted form. Typically that form is the hash value from a one-way hash function (see Chapter 4). Second is the use of an authentication protocol (or mechanism) that prevents the transmission of passwords in an easily readable form over a network or especially the Internet. Third, strong (complex) passwords are often enforced at a programmatic level. This is used to ensure that only passwords that are difficult for a password-cracking tool to discover are allowed by the system.

The strength of a password is generally measured in the amount of time and effort—that is, work—involved in breaking the password through various forms of cryptographic attacks. These attacks are collectively known as *password cracking* or *password guessing* (discussed in more depth in Chapter 4). A weak password invariably uses only alphanumeric characters; often employs dictionary or other common words; and may include user profile–related information such as birthdates, social security numbers, and pet names.

Strong passwords consist of numerous characters (8 or more); include at least three types of characters (uppercase, lowercase, numerals, and keyboard symbols); are changed on a regular basis (every 30 days); don't include any dictionary or common words or acronyms; and don't include any part of the subject's real name, user name, or e-mail address.

Passwords should be strong enough to resist discovery through attack but easy enough for the person to remember. This can sometimes be a difficult line to walk.

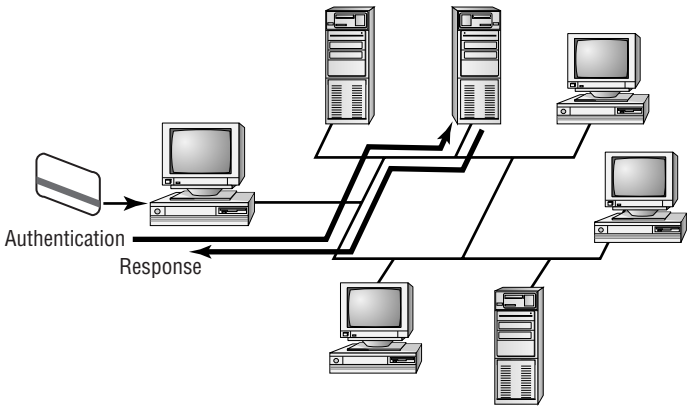
Tokens

A *token* is a form of authentication factor that is something you have. It's usually a hardware device, but it can be implemented in software. A token is used to generate temporary single-use passwords for the purpose of creating stronger authentication. In this way, a user account isn't tied to a single static password. Instead, the user must be in physical possession of the password-generating device. The user enters the currently valid password from the token as their password during the logon process.

There are several forms of tokens. Some generate passwords based on time, whereas others generate passwords based on challenges from the authentication server. In either case, the user can use (or attempt to use) the generated password just once before they must either wait for the next time window or request another challenge. Passwords that can only be used once are known as *one-time passwords*. This is the most secure form of password, since regardless of whether its use results in a successful logon, that one-use password is never valid again for reuse. One-time passwords can only be employed when a token device is used, due to the complexity and ever-changing nature of the passwords.

A token may be a device like a small calculator with or without a keypad. It may also be a high-end smartcard (see Figure 1.5). When properly deployed, a token-based authentication system is more secure than a password-only system.

FIGURE 1.5 The smartcard authentication process



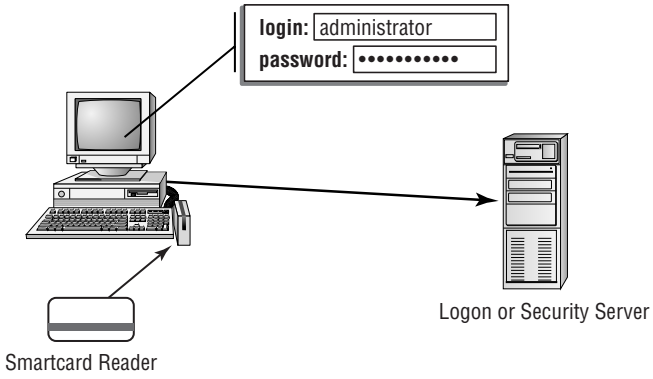
Multi-Factor Authentication

Multi-factor authentication is the requirement that a user must provide two or more authentication factors in order to prove their identity. There are three generally recognized categories of authentication factors:

- Something you know (such as a password)
- Something you have (such as a token)
- Something you are (such as a biometric)

When two different authentication factors are used, this is known as *two-factor authentication* (see Figure 1.6). If two of the same authentication factors are used, this is known as *strong authentication*. Whenever different factors are used (whether two or three), this is always a more secure solution than any number of the same authentication factors. This is due to the fact that with two or more different factors, two or more different types of attacks must take place in order to steal or capture the authentication factor itself. With strong authentication, even if 10 passwords are required, only a single type of password-stealing attack needs to be waged to break through the authentication security.

FIGURE 1.6 Two-factor authentication



Mutual Authentication

Mutual authentication is two-way authentication. The subject (user) authenticates to the object (server), and the object (server) authenticates back to the subject (user). In this way, both ends of the communication have proof of the identity of the other partner.

Biometrics

Biometrics is the term used to describe the collection of physical attributes of the human body that can be used as authentication factors. Biometrics fall into the authentication factor category of "something you are": You, as a human, have the element of identification as part of your physical body. Biometrics include fingerprints, palm scans (use of the entire palm as if it were a fingerprint), hand geometry (geometric dimensions of the silhouette of a hand), retinal scans (pattern of blood vessels at the back of the eye), iris scans (colored area of the eye around the pupil), facial recognition, voice recognition, signature dynamics, and keyboard dynamics.

Although biometrics are a stronger form of authentication than passwords alone, biometrics in and of themselves aren't the best solution. Even with biometrics, implementing multi-factor authentication is the most secure solution.

Exam Essentials

Kerberos Kerberos is a trusted third-party authentication protocol. It uses encryption keys as tickets with timestamps to prove identity and grant access to resources. Kerberos is a single sign-on solution employing a Key Distribution Center (KDC) to manage its centralized authentication mechanism.

Challenge Handshake Authentication Protocol (CHAP) CHAP is an authentication protocol used primarily over dial-up connections (usually PPP) as a means to provide a secure transport mechanism for logon credentials. CHAP uses a one-way hash to protect passwords and periodically reauthenticates clients.

Certificates Certificates are an electronic means of proving subject and object identity. They are issued by certificate authorities (CAs).

Passwords Passwords are the most common form of authentication, but they're also the weakest. Passwords can be improved through the use of strong password policies enforced by software.

Tokens Tokens are a "something you have" type of authentication factor. They support one-time passwords, which are the most secure form of passwords.

Multi-factor authentication Multi-factor authentication is the requirement that a user must provide two or more authentication factors in order to prove their identity.

Mutual authentication Mutual authentication is two-way authentication. The subject (user) authenticates to the object (server), and the object (server) authenticates back to the subject (user).

Biometrics Biometrics is the collection of physical attributes of the human body that can be used as authentication factors ("something you are"). Biometrics include fingerprints, palm scans (use of the entire palm as if it were a fingerprint), hand geometry (geometric dimensions of the silhouette of a hand), retinal scans (pattern of blood vessels at the back of the eye), iris scans (colored area of the eye around the pupil), facial recognition, voice recognition, signature dynamics, and keyboard dynamics.

1.3 Identifying Non-Essential Services

The full name of this section from the Security+ objectives list is "Identify non-essential services and protocols and know what actions to take to reduce the risks of those services and protocols." It's important to realize that a key element in securing a system is to reduce its attack surface. The *attack surface* is the area that is exposed to untrusted networks or entities and that is vulnerable to attack. If a system is hosting numerous services and protocols, then its attack surface is larger than that of a system running only essential services and protocols.

For more information on this topic, refer to Chapter 1 of the *Security+ Study Guide, Second Edition*, by Sybex.

Critical Information

It's tempting to install every service, component, application, and protocol available to you on every computer system you deploy. However, this temptation is in direct violation of a security best practice stating that you should have each system host only those services and protocols that are absolutely essential to its mission-critical operations.

The real issue is that software isn't trusted. Software is what services, applications, components, and protocols are; unfortunately, software is written by people and therefore in all likelihood isn't perfect. But even if software was without bugs, errors, oversights, mistakes, and so on, it would still represent a security risk. Software that is working as expected can often still be exploited by a malicious entity. Therefore, every instance of software deployed onto a computer system represents a collection of additional vulnerability points that may be exposed to external, untrusted, and possibly malicious entities.

From this perspective, you should understand that all non-essential software elements should be removed from a system before it's deployed on a network, especially if that network has Internet connectivity. But how do you know what is essential and what isn't? Here is a basic methodology:

1. Plan the purpose of the system.
2. Identify the services, applications, and protocols needed to support that purpose. Make sure these are installed on the system.
3. Identify the services, applications, and protocols that are already present on the system. Remove all that aren't needed.

Often, you won't know if a specific service that appears on a system by default is needed. Thus, a trial-and-error test is required. If software elements aren't clearly essential, disable them one by one and test the capabilities of the system. If the system performs as you expect, the software probably isn't needed. If the system doesn't perform as expected, then the software will need to be re-enabled. This process is known as application and system hardening.

Application and system hardening is discussed in greater detail in Chapter 3, "Infrastructure Security."

You may discover that some services and protocols offer features and capabilities that aren't necessary to the essential functions of your system. If so, find means to disable or restrict those characteristics. This may include restricting ports or reconfiguring services through a management console.

The essential services on a system are usually easy to identify; they generally have recognizable names that correspond to the function of the server. However, you must determine which services are essential on your specific system. Services that are essential on a web server may not be essential on a file server or an e-mail server. Some examples of possible essential services include the following:

- File sharing
- E-mail
- Web
- File Transfer Protocol (FTP)
- Telnet
- Remote access
- Network news (Network News Transfer Protocol [NNTP])
- Domain Name Service (DNS)
- Dynamic Host Configuration Protocol (DHCP)

Non-essential services are more difficult to identify. Just because a service doesn't have the same name as an essential function of your server doesn't mean that it isn't used by the underlying operating system or as a support service. It's extremely important to test and verify whether any service is being depended on by an essential service. However, several services are common candidates for non-essential services that you may want to locate and disable first (assuming you follow the testing method described earlier). These may include:

- NetBIOS
- Unix RPC
- Network File System (NFS)
- X services
- R services
- Telnet
- FTP
- Trivial File Transfer Protocol (TFTP)
- NetMeeting
- Instant messaging
- Remote control software
- Simple Network Management Protocol (SNMP)

Exam Essentials

Security risks of non-essential software Non-essential software increases the attack surface of your systems. Removing every element of software that isn't required will improve the security of a system.

1.4 Identifying Attack Methods

The full name of this section from the Security+ objectives list is "Recognize the following attacks and specify the appropriate actions to take to mitigate vulnerability and risk." The rate at which systems connected to networks are attacked is increasing at an alarming rate. Systems that aren't even connected to the Internet, but just to a private network, may come under attack. There are myriad ways to attack a computer system. Your familiarity with a small collection of these attacks and how to respond to them is an essential skill for the Security+ exam.

For more information on this topic, refer to Chapters 2 and 7 of the *Security+ Study Guide, Second Edition*, by Sybex.

Critical Information

Any computer system connected to any type of network is subject to various methods of attack. The following section discusses common attack methods.

Denial of Service/Distributed Denial of Service (DOS/DDOS) Attacks

Denial of Service (*DoS*) is a form of attack that has the primary goal of preventing the victimized system from performing legitimate activity or responding to legitimate traffic. There are two basic types of DoS attack. The first form exploits a weakness, error, or standard feature of software to cause a system to hang, freeze, consume all system resources, and so on. The end result is that the victimized computer is unable to process any legitimate tasks. The second form floods the victim's communication pipeline with garbage network traffic. The end result is that the victimized computer is unable to send or receive legitimate network communications. In either case, the victim has been denied the ability to perform normal operations (services).

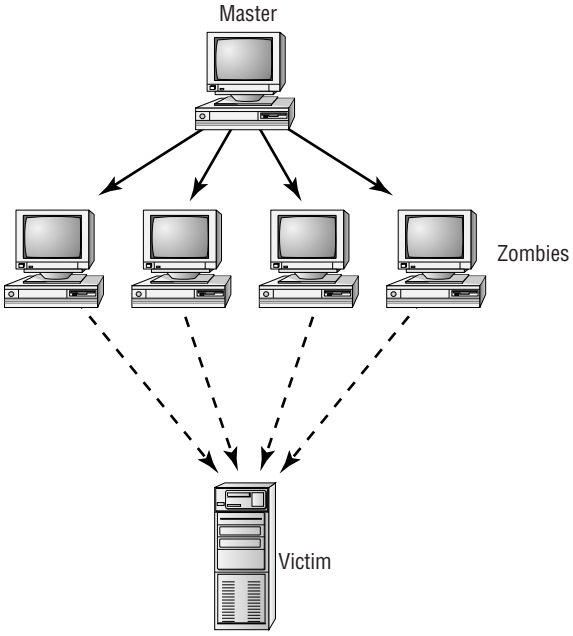
DoS isn't a single attack but rather an entire class of attacks. Some attacks exploit flaws in operating system software, whereas others focus on installed applications, services, or protocols. Some attacks exploit specific protocols, including Internet Protocol (IP), *Transmission Control Protocol (TCP)*, *Internet Control Message Protocol (ICMP)*, and *User Datagram Protocol (UDP)*.

DoS attacks typically occur between one attacker and one victim. However, they don't have to be waged in that simple a manner. Most DoS attacks employ some form of intermediary system (usually an unwilling and unknowing participant) in order to hide the attacker from the victim. For example, if an attacker sends attack packets directly to a victim, then it's possible for the victim to discover who the attacker is. This is made more difficult, although not impossible, through the use of spoofing (discussed later in this chapter).

Many DoS attack are waged by first compromising or infiltrating one or more intermediary systems that serve as launch points, or attack platforms. The attacker installs remote-control tools, often called *bots*, *zombies*, or *agents*, onto these systems. Then, at an appointed time or in response to a launch command from the attacker, the DoS attack is conducted against the victim. In this manner, the victim may be able to discover the one or many zombied systems that are

causing the DoS attack but probably won't be able to track down the actual attacker. This form of attack involving zombied systems is known as Distributed Denial of Service (DDoS), as shown in Figure 1.7.

FIGURE 1.7 Distributed Denial of Service attack



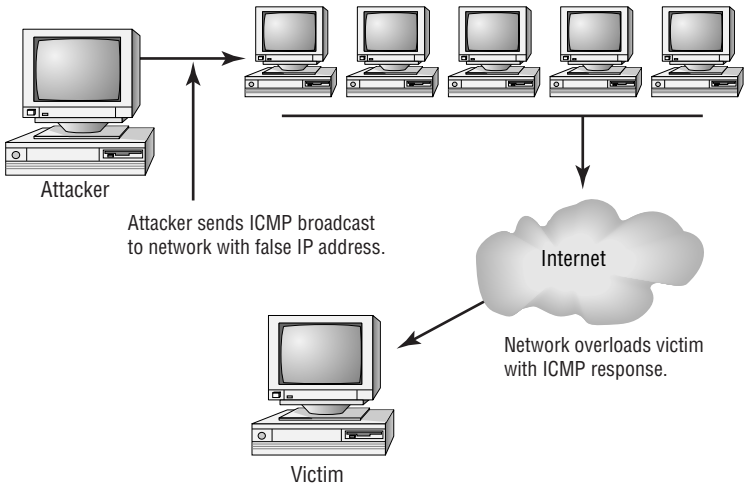
There is a third level of DoS known as Distributed Reflective Denial of Service (DRDoS). This form of attack employs an *amplification* or *bounce* network that is an unwilling and unknowing participant but that has unfortunately left on its ability to receive broadcast messages and create message responses, echoes, or bounces. In effect, the attacker sends spoofed message packets to the amplification network's broadcast address. This causes each single inbound received packet to be distributed to all the hosts inside that network (which could be in the 10,000 or 100,000 range). Each host then responds to each packet, but since the source of the original packet was falsified, the response goes to the victim instead of the true sender (the attacker). So, what originated from the attacker as a single packet is transformed into numerous packets exiting the amplification network and ultimately flooding the victim's communication link.

There are numerous specific DoS, DDoS, and DRDoS attack tools and methods. Here are a few that you should be able to recognize:

Smurf This form of DRDoS uses Internet Control Message Protocol (ICMP) echo reply packets (ping packets). See Figure 1.8 for an example.

Fraggle This form of DRDoS uses User Datagram Protocol (UDP) packets directed to port 7 (echo port) or 19 (chargen [character generator] port).

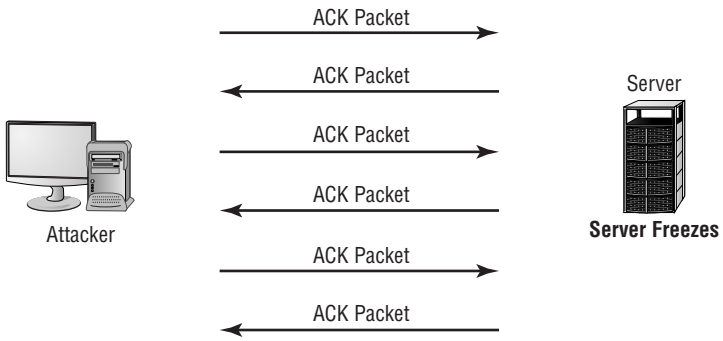
FIGURE 1.8 A smurf attack underway against a network



SYN flood This type of attack is an exploitation of a TCP three-way handshake. Every TCP session starts with the client sending a SYN (synchronize) packet to a server, the server responding with a SYN/ACK (synchronize/acknowledgment) packet, and the client sending a final ACK packet. The attack consists of the attacker serving as a client and sending numerous SYN packets but never any final ACK packets. This causes the server to consume all system resources opening incomplete communication sessions. Figure 1.9 shows an example of a TCP SYN flood attack.

Teardrop Numerous partial IP packets are sent to a victim with overlapping sequencing and offset values. The victim attempts to assemble complete IP packets from the received partials, but the fragments overwrite each other and may produce a packet of an invalid size. This causes the victim to freeze or crash.

FIGURE 1.9 TCP SYN flood attack



Land attack Numerous SYN packets are sent to the victim with source and destination addresses spoofed as the victim's address. The victim is confused because it's unable to respond to a packet it sent to itself that it has no record of sending. This often results in a freeze or crash.

Ping flood The attacker sends numerous ping echo requests to a victim. The victim responds with the echo. If enough inbound and outbound packets are transmitted, no legitimate traffic will be able to use the communication link.

Ping of death The attacker sends oversized ping packets to the victim; the victim doesn't know how to handle invalid packets, and it freezes or crashes.

Bonk The attacker sends a corrupt UDP packet to DNS port 53. This type of attack may cause Windows systems to crash.

Boink The same as Bonk, but the corrupt UDP packets are sent to numerous ports. The result may cause a Windows system to crash.

Fortunately, most of the basic DoS attacks that exploit error-handling procedures (such as ping of death, land attack, teardrop, bonk, boink, and so on) are now automatically handled by improved versions of the protocols installed in the operating system. However, most of the DDoS and DRDoS attacks aren't as easy to safeguard against. SYN flood, teardrop, land attack, ping flood, ping of death, bonk, and boink are typically labeled DoS attacks, but they can be waged as a DDoS if the attacker compromises several intermediary systems and uses those as launching points to attack the victim.

Some countermeasures and safeguards against these attacks include the following:

- Adding firewalls, routers, and intrusion detection systems (IDSs) that detect DoS traffic and automatically block the port or filter out packets based on the source or destination address
- Disabling echo replies on external systems
- Disabling broadcast features on border systems
- Blocking spoofed packets from entering or leaving your network
- Keeping all systems patched with the most current security updates from vendors

Back Door Attacks

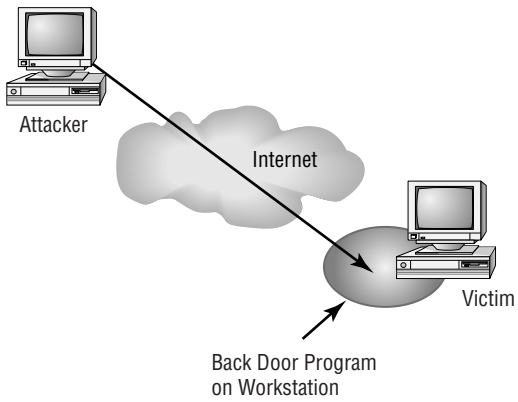
The term *back door* can refer to two types of problems or attacks on a system. The first and oldest meaning of back door was a developer-installed access method that bypassed all security restrictions. The back door was a special hard-coded user account, password, or command sequence that allowed anyone with knowledge of the access hook (sometimes called a *maintenance hook*) to enter the environment and make any changes. This sounds great from a developer's perspective, especially during the coding and debugging process. Unfortunately, such programming shortcuts are often overlooked when the product nears completion; thus, they end up in the final product. Fortunately, once a back door is discovered in a released product, the vendor usually releases a patch to remove the back door code from the installed product. The possible presence of back doors is another good reason to stay current with vendor-released updates and patches.

The second meaning of *back door* is a hacker-installed remote access client. These small, maliciously purposed tools can be easily deposited on a computer through a Trojan horse, a

virus, or a website mobile code download, or even as part of an intrusion activity. Once active on a system, they open up access ports and wait for an inbound connection. Thus, a back door serves as an access portal for a hacker so they can bypass any security restrictions and gain (or regain) access to a system. Some common back door tools include Back Orifice, NetBus, and Sub7 (all of which function on Windows). These and other common back door tools are detected and removed by virus scanners and spy-ware scanning tools.

Figure 1.10 shows a back door attack in progress.

FIGURE 1.10 A back door attack in progress



Preemptive measures against back doors include the following:

- Restricting mobile code from being automatically downloaded to your systems
- Using software policies to prevent unauthorized software from being installed
- Requiring software and driver signing

Spoofing Attacks

Spoofing is the act of falsifying data. Usually the falsification involves changing the source address of network packets. As a result of the changed source address, victims are unable to locate the true attackers or initiators of a communication. Also, by spoofing the source address, the attacker redirects responses, replies, and echoes to packets to some other system (such as in the case of smurf, fraggle, and land DoS attacks.

Spoofing is also a common activity for unsolicited e-mail, commonly known as spam. Spoofed e-mail means that you're unable to reply to the e-mail or determine where it originally came from.

There are innumerable forms of spoofing attacks. Spoofing can be used to redirect packets, bypass traffic filters, steal data, perform social engineering attacks, and even falsify websites.

Countermeasures against spoofing attacks include the following:

- Using e-mail spam and spoofing filters

- Dropping all inbound packets received by border systems that have a source destination from inside your private network (this indicates spoofing)
- Dropping all outbound packets received by border systems that have a source destination from outside your private network (this also indicates spoofing)

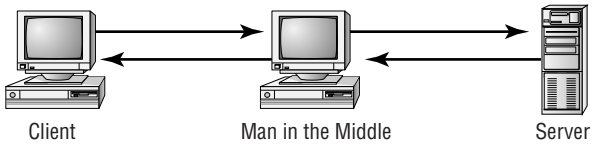
Man-in-the-Middle Attacks

A *man-in-the-middle attack* is a form of communications eavesdropping attack. Attackers position themselves in the communication stream between a client and server (or any two communicating entities). The client and server believe that they are communicating directly with each other; they may even have secured or encrypted communication links. However, the attacker can access and even modify the communications.

Waging a man-in-the-middle attack is complex. It involves altering network traffic and possibly poisoning name-resolution systems (such as DNS or WINS) in order to fool the client into perceiving the attacker as the server and to fool the server into perceiving the attacker as the client. Once that charade is successful, the client will submit their logon credentials to the fake server (the masked attacker), which in turn sends the credentials to the actual server while masquerading as the actual client. As a result, the client establishes a communication link (maybe even an encrypted link) with the attacker, and the attacker establishes a communication link with the server. As data is transmitted in either direction between the true client and server systems, the attacker can read and access all the data and can choose to modify the traffic to further the subterfuge.

Figure 1.11 shows a man-in-the-middle attack.

FIGURE 1.11 A man-in-the-middle attack occurring between a client and a web server



Such attacks are usually most successful when routing and name resolution systems are first compromised in order to position the attacker before the client-to-server communication is initiated. However, man-in-the-middle attacks can be conducted against existing client-server communication links (usually assuming they aren't encrypted). This second form of attack is much more difficult due to existing routing, name resolution, TCP sequencing, and the speed of the communication. However, several tools exist that perform the necessary operations against both sides of the communication connection in order to implement a man-in-the-middle injection. Some of these tools include Juggernaut, T-Sight, and Hunt.

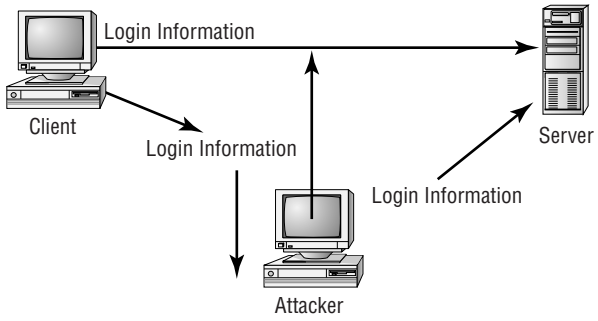
Countermeasures to man-in-the-middle attacks include strong encryption protocols, such as IPSec, and the use of strong authentication, such as Kerberos, certificates, multi-factor authentication, and mutual authentication.

Replay Attacks

A *replay attack* is just what it sounds like: An attacker captures network traffic and then replays the captured traffic in an attempt to gain unauthorized access to a system. Most commonly, the attacker focuses on network traffic that is the exchange between a client and server performing authentication. If an attacker can capture the authentication traffic—especially the packets containing the logon credentials, even if they are more than just username and password (such as certificates, token responses, or biometric values)—then a replay attack may grant the attacker the ability to log on to a system by retransmitting the captured packets.

Figure 1.12 shows a reply attack.

FIGURE 1.12 A replay attack occurring



If a replay attack succeeds, then the attacker gains the same level of access as the user that originally submitted the authentication information. Fortunately, most modern operating systems, networks, protocols, services, and applications use various replay protection mechanisms to directly prevent such attacks. Two of the most common countermeasures are packet sequencing and timestamps. Packet sequencing ensures that any packet received that isn't in the proper order (or within a reasonable margin) is dropped and ignored. Packet timestamps ensures that any packet received outside of a specific time window is dropped and ignored.

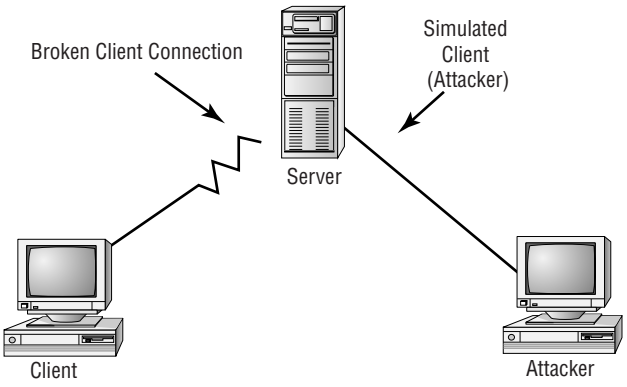
TCP/IP Hijacking Attacks

TCP/IP hijacking is a form of attack where the attacker takes over an existing communication session. The attacker can take over the role of the client or the server, depending on the purpose of the attack. Session hijacking is a simpler one-sided form of man-in-the-middle attack. Many of the same tools and techniques are used in both forms of attack. However, with a session hijacking, the other partner in the communication is disconnected—they're aware that they are no longer communicating and that their session was interrupted. However, they may not immediately realize that they were the collateral damage in a session hijacking attack.

Figure 1.13 shows a TCP/IP hijacking attack.

Countermeasures to TCP/IP hijacking attacks include using encrypted protocols and performing reauthentication during a session.

FIGURE 1.13 TCP/IP hijacking attack



Weak Keys

Weak keys isn't an attack but rather a vulnerability in how a cryptography system is used that makes various types of attacks possible. The term *weak keys* implies that the cryptographic key selected to encrypt a file or a communication session is either too short or too easily guessed. Weak keys are generally anything less than 64 bits in length. As a result of weak keys, attackers have an easier time cracking encrypted communications because they have to try fewer possible permutations during a full brute-force attack than they would for strong keys (those at least 128 bits long). In general, you should avoid using weak keys by only using cryptography solutions that use longer, stronger keys.

For more information on keys and cryptography, see Chapter 4.

Mathematical Attacks

Mathematical attacks are a specific type of attacks against cryptography. They're directed against the algorithm in an attempt to exploit the arithmetic it employs. The goal of a mathematical attack is to either decrypt an encrypted message or to discover the key used for encryption.

For more information on cryptography and related attacks, see Chapter 4.

Social Engineering

Social engineering is mentioned twice in the first domain of the Security+ exam objectives list. We discuss it later in this chapter, in the section "1.6 Understanding Social Engineering."

Birthday Attacks

A *birthday attack* is used against a specific form of cryptography called hashing. The birthday attack gets its name from a bar bet that exploits the mathematical probably of shared birthdays: It takes only 46 people in a room for there to be a greater than 50 percent probability that any two of those people share the same birthday. (The bar bet is that you'll drink for free if two people in the bar share a birthday; otherwise you'll buy the house a round of drinks.)

Hashing is a one-way function that creates a fixed-length output (known as the hash, hashing value, fingerprint, message digest, and so on) from an input of any length. A hash serves as an ID code to detect when the original data source has been altered, since no two data sources produce the same hash.

The birthday attack exploits a mathematical property that states that if the same mathematical function is performed on two values and the result for each is the same, then the original values are the same. This concept is often represented with the following syntax:

```
f(M)=f(M') therefore M=M'
```

Birthday attacks can be waged against any use of hashing. However, they're most commonly employed during password-guessing attacks (discussed in the following section). In a password-guessing attack, a program compares possible passwords with passwords stored in an accounts database. But passwords stored in an accounts database are secured because only their hash values are stored there. Thus, the password-cracking program first performs the same hashing function used by the secured system on each possible password before scanning the accounts database for a match. If a match is found, then the password-guessing tool has discovered a password based on the `f(M)=f(M')` property. This is more specifically known as *reverse hash matching*.

For more information on cryptography and related attacks, see Chapter 4.

Password-Guessing Attacks

Password guessing is an attack aimed at discovering the passwords employed by user accounts. Password guessing is often called password *cracking*. There are several forms of password-guessing attack tools: Some attempt to guess passwords by attacking a logon prompt, others try to extract passwords directly from an accounts database, and others attempt to capture authentication traffic and extract passwords out of the network packet. In most cases, the latter two options employ birthday attack (reverse hash matching) methods to discover the password used by a user account.

There are innumerable password-guessing tools on the Internet. However, they can be distilled into two primary categories based on the method used to select possible passwords for a direct logon prompt or birthday attack procedure: brute force and dictionary.

No matter what tool is used to discover passwords, the most important countermeasure against password crackers is to use complex passwords and change them on a regular basis. See the discussion in the section "Username/Password" for details on strong passwords.

Brute Force

A *brute-force* attack is designed to try every possible valid combination of characters to construct possible passwords in the attempt to discover the specific passwords used by user accounts. Brute-force attacks rarely begin at 0000001 or 000000a but rather take a statistical probability approach to the selection of possible passwords. Most noncomplex passwords under 8 characters long can be discovered in less than 30 minutes.

Longer and more complex passwords make brute-force attacks less successful. However, given enough time, a brute-force attack will always succeed. Thus, regularly changing a password will thwart the use of any discovered old passwords.

Dictionary

A *dictionary attack* performs password guessing by using a preexisting list of possible passwords. Password lists can include millions of possible passwords. Often password lists or dictionaries are constructed around topics. Thus, if an attacker knows basic information about you as a person, they can attempt to exploit human nature's propensity to select passwords using words common or familiar to you. For example, if an attacker knows that you work in the medical industry, you have cats, and you enjoy sailing, they can select password dictionaries that include words, acronyms, and phrases common to those issues.

Dictionary attacks are surprisingly effective against users who have not been trained in the methods and skills of creating complex passwords.

Software Exploitation Attacks

Software exploitation attacks are directed toward known flaws, bugs, errors, oversights, or normal functions of the operating system, protocols, services, or installed applications. One of the most common forms of software exploitation is a buffer overflow attack.

Buffer overflow attacks are discussed in detail in Chapter 2, "Communication Security."

A buffer overflow attack occurs when an attacker submits data to a process that is larger than the input variable is able to contain. Unless the program is properly coded to handle excess input, the extra data is dropped into the system's execution stack and may execute as a fully privileged operation. Buffer overflow attacks can result in system crashes, corrupted data, user privilege escalation, or just about anything a hacker can think of. The only countermeasures to buffer overflow attacks are to patch the software when issues are discovered and to properly code software to perform input validation checks before accepting input for processing.

Once a weakness is discovered in software, a hacker often writes an exploit or attack tool. These tools are easily accessible on the Internet. They then allow anyone to grab the tool and point it at a victim to perform the attack, even when the attacker has no knowledge of how to perform the attack.

Exam Essentials

Denial of Service attacks Denial of Service (DoS) is a form of attack that has the primary goal of preventing the victimized system from performing legitimate activity or responding to legitimate traffic. There are two basic types of DoS attack. The first form exploits a weakness, error, or standard feature of software on the victim system to cause the system to hang, freeze, consume all system resources, and so on. The second form floods the victim's communication pipeline with garbage network traffic.

Examples of DoS attacks Examples of DoS include smurf, fraggle, SYN flood, teardrop, land, ping flood, ping of death, bonk, and boink.

Back door attacks The term *back door* can refer to two types of problems or attacks on a system: a developer-installed access method that bypasses any and all security restrictions, or a hacker-installed remote access client.

Spoofing attacks Spoofing is the act of falsifying data. Usually the falsification involves changing the source address of network packets. Because the source address is changed, victims are unable to locate the true attackers or initiators of a communication. Also, by spoofing the source address, attackers redirect responses, replies, and echoes to packets to some other system.

Man-in-the-middle attacks A man-in-the-middle attack is a form of communications eavesdropping attack. Attackers position themselves in the communication stream between a client and server (or any two communicating entities). The client and server believe that they are communicating directly with each other.

Replay attacks In a replay attack, an attacker captures network traffic and then replays the captured traffic in an attempt to gain unauthorized access to a system.

TCP/IP hijacking attacks TCP/IP hijacking is a form of attack where the attacker takes over an existing communication session.

Weak keys Weak keys isn't an attack but rather a weakness in how a cryptography system is used that makes various types of attacks possible. Weak keys implies that the cryptographic key selected to encrypt a file or a communication session is either too short or too easily guessed. Weak keys are generally anything less than 64 bits in length.

Mathematical attacks Mathematical attacks are direct against the algorithm in an attempt to exploit the arithmetic it employs.

Birthday attacks The birthday attack exploits a mathematical property that states that if the same mathematical function is performed on two values and the result is the same, then the original values are the same. This concept is often represented with the syntax `f(M)=f(M') therefore M=M'`.

Password guessing Password guessing is an attack aimed at discovering the password employed by user accounts. It's often called password cracking. There are two primary categories of

password-guessing tools, based on the method used to select possible passwords for a direct logon prompt or birthday attack procedure: brute force and dictionary.

Software exploitation Software exploitation attacks are directed toward known flaws, bugs, errors, oversights, or normal functions of the operating system, protocols, services, or installed applications. One of the most common forms of software exploitation is a buffer overflow attack.

1.5 Identifying Malicious Code

The full name of this section from the Security+ objectives list is "Recognize the following types of malicious code and specify the appropriate actions to take to mitigate vulnerability and risk." Knowing how to recognize and respond to malicious code is an essential skill in today's networking environments.

For more information on this topic, refer to Chapter 2 of the *Security+ Study Guide, Second Edition*, by Sybex.

Critical Information

Malicious code is any element of software that performs an unwanted or undesired function from the perspective of the legitimate user or owner of a computer system. This section explores four specific types of malicious code.

Viruses

Viruses get their name from their biological counterparts. They are programs that are designed to spread from one system to another through self-replication and to perform any of a wide range of malicious activities. The malicious activities performed by viruses include data deletion, corruption, alteration, and theft. Some viruses replicate and spread so rapidly that they consume system and network resources, thus performing a type of DoS attack.

Most viruses need a host to latch onto. The host can be a file or a sector of a hard drive. Viruses that attach themselves to the boot sector of a hard drive and thus are loaded in memory when the drive is activated are known as *boot sector viruses*. *Polymorphic viruses* have the ability to alter their own code in order to avoid detection by antivirus scanners. *Macro viruses* live within documents or e-mails and exploit the scripting capabilities of productivity software.

The best countermeasure to viruses is an antivirus scanner, which is updated regularly and which monitors all local storage devices, memory, and communication pathways for viral activities. Other countermeasures include avoiding downloading software from the Internet, not opening e-mail attachments, and avoiding the use of removable media from other environments.

Trojan Horses

A *Trojan horse* is a form of malicious software that is disguised as something useful or legitimate. The most common forms of Trojan horses are games and screen savers, but any software can be made into a Trojan horse. The goal of a Trojan horse is to trick a user into installing it on their computer. This allows the malicious code portion of the Trojan horse to gain access to the otherwise secured environment. Some of the most common Trojan horses are tools that install DoS zombies onto systems.

Countermeasures for Trojan horses are the same as for viruses.

Logic Bombs

A *logic bomb* is a form of malicious code that remains dormant until a triggering event occurs. The triggering event can be a specific time and date, the launching of a specific program, or the accessing of a specific URL (such as your online banking logon page). Logic bombs can perform any malicious function the programmer wishes, from causing system crashes to deleting data to altering configurations to stealing authentication credentials.

Countermeasures for logic bombs are the same as for viruses.

Worms

Due to the expanding capabilities (although malicious) of viruses, *worms* are no longer an easily identifiable distinct category of malicious code. Worms are designed to exploit a single hole in a system (operating system, protocol, service, or application) and then use that flaw to spread themselves to other systems with the same flaw. Worms may be used to deposit viruses, Trojan horses, or logic bombs, or they may perform direct virus-like maelstrom activities on their own.

Countermeasures for worms are the same as for viruses with the addition of keeping systems patched.

Exam Essentials

Viruses Viruses are programs that are designed to spread from one system to another through self-replication and to perform any of a wide range of malicious activities.

Trojan horses A Trojan horse is a form of malicious software that is disguised as something useful or legitimate.

Logic bombs A logic bomb is a form of malicious code that remains dormant until a triggering event occurs. The triggering event can be a specific time and date, the launching of a specific program, or the accessing of a specific URL (such as your online banking logon page).

Worms Worms are designed to exploit a single flaw in a system (operating system, protocol, service or application) and then use that hole to replicate itself to other systems with the same flaw.

Malicious code countermeasures The best countermeasure to viruses and other malicious code is an antivirus scanner, which is updated regularly and which monitors all local storage devices, memory, and communication pathways for viral activities. Other countermeasures include avoiding downloading software from the Internet, not opening e-mail attachments, and avoiding the use of removable media from other environments.

1.6 Understanding Social Engineering

The full name of this section from the Security+ objectives list is "Understand the concept of and know how reduce the risks of social engineering." No matter how much technical protection has been deployed in an environment, there will always be a known weakness: people. *Social engineering* is a class of attacks directed toward people. Unless you can remove all personnel from your environment, you'll have to deal with social-engineering attacks.

For more information on this topic, refer to Chapter 2 of the *Security+ Study Guide, Second Edition*, by Sybex.

Critical Information

Social engineering is a unique type of attack that attempts to exploit human behavior. Social-engineering attacks can take many forms, from skillfully worded websites to clever e-mail messages to personnel impersonations over the phone to face-to-face acting. In just about every case, a social-engineering attack tries to convince the victim to perform some activity or reveal a piece of information that they shouldn't.

Any form of advertisement could be considered a form of social-engineering attack—ads appeal to you in an attempt to get you to purchase or use a product or service. Although an advertisement's motivation is profit, most social-engineering attacks' motives are more malevolent. Here are some example scenarios of common social-engineering attacks:

- You receive an e-mail warning you about a dangerous new virus spreading across the Internet. The message tells you to look for a specific file on your hard drive and delete it, since it indicates the presence of the virus. Often, however, the identified file is really an essential file needed by your system.
- A website claims to offer free temporary access to its products and services, but it requires that you alter the configuration of your Web browser and/or firewall in order to download the access software.
- A secretary receives a phone call from a person claiming to be a client who is running late to meet the CEO. She asks for the CEO's private cell phone numbers so she can call him.
- The Help Desk receives a call from an outside line. The caller claims to be a manager of a department who is currently involved in a sales meeting in another city. He claims he forgot his password and needs it to be reset so that he can log in remotely to download his essential presentation.
- Someone who looks like an A/C repairman enters the office and claims a service call was received for a malfunctioning unit in the building. He is sure the unit can be accessed from inside your office work area, and he asks to be given free roam to repair the A/C system.

These are just a few examples of possible social-engineering attacks. These may also be legitimate and benign occurrences, but you can see how they could mask the motives and purposes of an intentional attacker.

Methods to protect against social engineering include the following:

- Training personnel about social-engineering attacks and how to recognize common signs
- Requiring authentication when performing activities for personnel over the phone
- Defining restricted information that is never communicated over the phone
- Always verifying the credentials of a repair person and verifying that a real service call was placed by authorized personnel
- Never following the instructions of an e-mail without verifying the information with at least two independent and trusted sources
- Always erring on the side of caution when dealing with anyone you don't know or recognize, whether in person, over the phone, or over the Internet/network

Exam Essentials

Social-engineering attacks Social engineering is a unique type of attack that attempts to exploit human behavior. Such attacks can take many forms, from skillfully worded websites to clever e-mail messages to personnel impersonations over the phone to face-to-face acting. In just about every case, a social-engineering attack tries to convince the victim to perform some activity or reveal a piece of information that they shouldn't.

1.7 Understanding Auditing

The full name of this section from the Security+ objectives list is "Understand the concept and significance of auditing, logging and system scanning." The key aspect of this issue is understanding that auditing is the primary means by which security violations are discovered. Through the use of a system's auditing capabilities, audit trails or logs are created. Audit trails enable auditors or administers to detect when security violating activities have occurred and may assist in understanding how such a violation was allowed to occur.

For more information on this topic, refer to Chapter 2 of the *Security+ Study Guide, Second Edition,* by Sybex.

Critical Information

Auditing is the process of recording information about various online, electronic, digital events between subjects and objects. It's often synonymous with *logging* and *monitoring*. The goal of auditing is to check compliance with security policy and to discover security violations or system errors. Some common activities that are audited include logging on, logging off, reading files, deleting files, using remote access services, and using applications.

Effective auditing starts with detailed planning. You must know what you want to audit for in order to properly configure a system (if not an entire network) to record the desired information. Auditing everything will result in too much audit output, whereas auditing too little won't gather sufficient information for detection or event reconstruction.

The recorded results of auditing are called *audit trails* or *logs*. Logs are reviewed in order to discover abnormalities, discrepancies, or outright security violations. With sufficiently extensive logging, it's possible to reconstruct events leading up to a security violation. Auditing often enables system administrators to discover the perpetrator of the violation as well as pinpoint the systems and resources involved.

A closely related issue is *system scanning*. System scanning, penetration testing, and vulnerability scanning are all similar concepts. The point is to test the configuration of a system or network to determine if all known security weaknesses have been patched or addressed correctly. System scanning produces an audit report listing all of the weaknesses and vulnerabilities of the scanned system. Often the report includes suggestions or recommendations on how to resolve the discovered problems.

Exam Essentials

Auditing Auditing is the process of recording information about various online, electronic, digital events between subjects and objects. The goal of auditing is to check compliance with security policy and to discover security violations or system errors. Some common activities that are audited include logging on, logging off, reading files, deleting files, using remote access services, and using applications.

System scanning The point of system scanning is to test the configuration of a system or network to determine if all known security weaknesses have been patched or addressed correctly. System scanning produces an audit report listing all of the weaknesses and vulnerabilities of the scanned system. Often the report includes suggestions or recommendations on how to resolve the discovered problems.

Review Questions

1. What method of access control is best suited for environments with a high rate of employee turnover?
   - **A.** MAC
   - **B.** DAC
   - **C.** RBAC
   - **D.** ACL
2. Which of the following is the strongest form of authentication?
   - **A.** Biometric
   - **B.** Two-factor
   - **C.** Something you have
   - **D.** Username and password
3. Kerberos is used to perform what security service?
   - **A.** Authentication protection
   - **B.** File encryption
   - **C.** Secure communications
   - **D.** Protected data transfer
4. What form of authentication periodically reauthenticates the client during a logon session?
   - **A.** Kerberos
   - **B.** Certificates
   - **C.** Multi-factor
   - **D.** CHAP
5. Which is the strongest form of password?
   - **A.** More than eight characters
   - **B.** One-time use
   - **C.** Static
   - **D.** Uses different types of keyboard characters
6. Which of the following is commonly found to be a non-essential service on a web server?
   - **A.** Server service
   - **B.** DNS service
   - **C.** FTP service
   - **D.** Print spooler service

7. Which of the following is a Denial of Service attack that uses network packets that have been spoofed so that the source and destination address are that of the victim?
   - **A.** Land
   - **B.** Teardrop
   - **C.** Smurf
   - **D.** Fraggle
8. Which is the best countermeasure against malicious code?
   - **A.** Manage user behavior
   - **B.** Prevent reuse of external removable media
   - **C.** Use antivirus software
   - **D.** Disable mobile code on web browsers
9. Which is the best countermeasure to social-engineering attacks?
   - **A.** Preventing the download of mobile code from the Internet
   - **B.** Employee training
   - **C.** Strong password policies
   - **D.** Auditing user activities
10. What is the primary goal of auditing?
   - **A.** Detect virus infections.
   - **B.** Look for rogue services.
   - **C.** Scan for open ports.
   - **D.** Check compliance with security policy.

Answers to Review Questions

1. C. RBAC is best suited for environments with a high rate of employee turnover because access is defined against static job descriptions rather than transitive user accounts (DAC and ACL) or assigned clearances (MAC).

2. B. Two-factor is always more secure than any single factor of authentication.

3. A. Kerberos is a third-party authentication service; thus it provides authentication protection. Kerberos can't be used to encrypt files, secure nonauthentication communications, or protect data transfer.

4. D. CHAP periodically reauthenticates the client during a logon session. Kerberos, certificates, and multi-factor authentication mechanisms don't perform reauthentication.

5. B. A one-time password is always the strongest form of password. A static password is always the weakest form of password. A password with more than eight characters and using different types of keyboard characters is usually strong, but these factors alone are unable to indicate their strength.

6. D. A non-essential service is any element that isn't needed by the primary function of the server. In most cases, a web server doesn't use the print spooler service, but it often uses the server, DNS, and FTP services.

7. A. A land DoS attack uses network packets that have been spoofed so that the source and destination address are that of the victim. A teardrop attack uses fragmented IP packets. Smurf and fraggle attacks use spoofed ICMP and UDP packets, respectively, against an amplification network.

8. C. The most reliable countermeasure against malicious code is an antivirus scanner. User-behavior modification, managing media, and disabling mobile code are all countermeasures against malicious code, but they aren't as reliable and effective as antivirus scanners.

9. B. The only real countermeasure to social engineering is employee training. The other three options are good security measures; however, they aren't countermeasures to social engineering.

10. D. The primary goal of auditing is to check compliance with security policy. Virus infection detection is handed by a virus scanner. Rogue service detection is performed manually or with a sniffer. Port scanning is performed with a port-scanning tool.