her favorite Internet radio station. An application running on her computer (say, a web browser) has to tell her local router she wants to get frames with this particular multicast address passed her way. The local router must tell the router upstream, and this process continues so each router between the source and destination knows where to pass this multicast data. This ensures that the user can get her rock music without other networks being bothered with this extra data.

IPv4 multicast protocols use a Class D address (224.0.0.0 to 239.255.255.255), which is a special address space reserved for multicasting. IPv6 multicast addresses start with eight 1's (that is, 1111 1111). Multicasting can be used to send out information; multimedia data; and even real-time video, music, and voice clips.

Internet Group Management Protocol (IGMP) is used to report multicast group memberships to routers. When a user chooses to accept multicast traffic, she becomes a member of a particular multicast group. IGMP is the mechanism that allows her computer to inform the local routers that she is part of this group and to send traffic with a specific multicast address to her system. IGMP can be used for online streaming video and gaming activities. The protocol allows for efficient use of the necessary resources when supporting these types of applications.

Like most protocols, IGMP has gone through a few different versions, each improving upon the earlier one. In version 1, multicast agents periodically send queries to systems on the network they are responsible for and update their databases, indicating which system belongs to which group membership. Version 2 provides more granular query types and allows a system to signal to the agent when it wants to leave a group. Version 3 allows the systems to specify the sources it wants to receive multicast traffic from. Each version is backward-compatible because versions 1 and 2 are still in use in legacy equipment.

NOTE The previous statements are true pertaining to IPv4. IPv6 is more than just an upgrade to the original IP protocol; it functions differently in many respects, including how it handles multicasting, which has caused many interoperability issues and delay in its full deployment.

Layer 2 Security Standards

As frames pass from one network device to another device, attackers could sniff the data; modify the headers; redirect the traffic; spoof traffic; carry out man-in-the-middle attacks, DoS attacks, and replay attacks; and indulge in other malicious activities. It has become necessary to secure network traffic at the frame level, which is layer 2 of the OSI model.

802.1AE is the IEEE MAC Security (MACSec) standard, which defines a security infrastructure to provide data confidentiality, data integrity, and data origin authentication.

Where a VPN connection provides protection at the higher networking layers, MACSec provides hop-by-hop protection at layer 2, as shown in Figure 11-16.

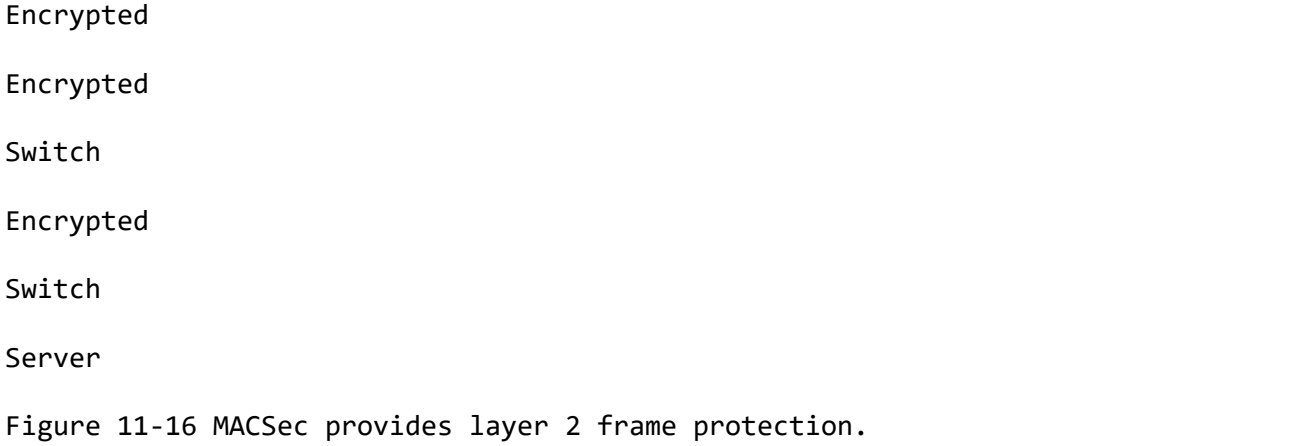

Figure 11-16 MACSec provides layer 2 frame protection.

MACSec integrates security protection into wired Ethernet networks to secure LANbased traffic. Only authenticated and trusted devices on the network can communicate with each other. Unauthorized devices are prevented from communicating via the network, which helps prevent attackers from installing rogue devices and redirecting traffic between nodes in an unauthorized manner. When a frame arrives at a device that is configured with MACSec, the MACSec Security Entity (SecY) decrypts the frame if necessary and computes an integrity check value (ICV) on the frame and compares it with the ICV that was sent with the frame. If the ICVs match, the device

processes the
frame. If they do not match, the device handles the frame according to a preconfigured
policy, such as discarding it.
The IEEE 802.1AR standard specifies unique per-device identifiers (DevID) and the
management and cryptographic binding of a device (router, switch, access point) to its
identifiers. A verifiable unique device identity allows establishment of the trustworthiness
of devices, and thus facilitates secure device provisioning.
As a security administrator you really only want devices that are allowed on your
network to be plugged into your network. But how do you properly and uniquely
identify devices? The manufacturer's serial number is not available for a protocol to
review. MAC addresses, hostnames, and IP addresses are easily spoofed. 802.1AR defines
a globally unique per-device secure identifier cryptographically bound to the device
through the use of public cryptography and digital certificates. These unique
hardwarebased credentials can be used with the Extensible Authentication Protocol-Transport
Layer Security (EAP-TLS) authentication framework. Each device that is compliant with
IEEE 802.1AR comes with a single built-in initial secure device identity (iDevID). The
iDevID is an instance of the general concept of a DevID, which is intended to be used
with authentication protocols such as EAP, which is supported by IEEE 802.1X.
So 802.1AR provides a unique ID for a device. 802.1AE provides data encryption,
integrity, and origin authentication functionality. 802.1AF carries out key agreement
functions for the session keys used for data encryption. Each of these standards provides
specific parameters to work within an 802.1X EAP-TLS framework, as shown in Figure 11-17.
As Figure 11-17 shows, when a new device is installed on the network, it cannot just
start communicating with other devices, receive an IP address from a Dynamic Host
Configuration Protocol (DHCP) server, resolve names with the Domain Name System
(DNS) server, and so on. The device cannot carry out any network activity until it is
authorized to do so. So 802.1X port authentication kicks in, which means that only
authentication data is allowed to travel from the new device to the authenticating server.
The authentication data is the digital certificate and hardware identity associated with
that device (802.1AR), which is processed by EAP-TLS. Once the device is authenticated,
usually by a Remote Authentication Dial-In User Server (RADIUS) server, encryption

keying material is negotiated and agreed upon between surrounding network devices.
Once the keying material is installed, then data encryption and frame integrity checking
can take place (802.1AE) as traffic goes from one network device to the next.
These IEEE standards are new and evolving and at different levels of implementation
by various vendors. One way the unique hardware identity and cryptographic material

IEEE
802.1AF
IEEE
802.1AR

Internal
network
Authentication
server
Certificate
authority

Upstream
device

New
infrastructure
device

IEEE
802.1AE

IETF
Key mgt
framework

0. New device is physically installed.
1. An 802.1X conversation starts.
2. EAP-TLS messages are forwarded.
3. Key material is returned and stored.
4. Session keys are generated.
5. MACSec encryption is enabled.

Figure 11-17 Layer 2 security protocols

are embedded in new network devices is through the use of a Trusted Platform Module
(TPM; described in Chapter 9).

Unless your network consists of only a few devices, isolated from the Internet (and what good is that?), you will need to move from layer 2 into layer 3 and above to do anything meaningful. Recall that the data link layer is concerned with exchanging data between devices that are directly connected to each other (in other words, in the same collision domain). Beyond that, we need layer 3 (network) and 4 (transport) protocols, such as TCP/IP.

The Transmission Control Protocol/Internet Protocol (TCP/IP) is a suite of protocols that governs the way data travels from one device to another. IP is a network layer protocol and provides datagram routing services. IP's main task is to support internetwork addressing and packet routing. It is a connectionless protocol that envelops data passed to it from the transport layer. The IP protocol addresses the datagram with the source and destination IP addresses. The protocols within the TCP/IP suite work together to break down the data passed from the application layer into pieces that can be moved along a network. They work with other protocols to transmit the data to the destination

IP

IP is a connectionless protocol that provides the addressing and routing capabilities for each package of data. The data, IP, and network relationship can be compared to the relationship between a letter and the postal system:

- Data = Letter
- IP = Addressed envelope
- Network = Postal system

The message is the letter, which is enveloped and addressed by IP, and the network and its services enable the message to be sent from its origin to its destination, like the postal system.

TCP

TCP is referred to as a connection-oriented protocol because before any user data is actually sent, handshaking takes place between the two systems that want to communicate. Once the handshaking completes successfully, a virtual connection is set up between the two systems. UDP is considered a connectionless protocol because it

does
not go through these steps. Instead, UDP sends out messages without first contacting
the destination computer and does not know if the packets were received properly or
dropped. Figure 11-18 shows the difference between a connection-oriented protocol
and a connectionless protocol.
UDP and TCP sit together on the transport layer, and developers can choose which to
use when developing applications. Many times, TCP is the transport protocol of choice
because it provides reliability and ensures the packets are delivered. TCP
provides a fullduplex, reliable communication mechanism, and if any packets are lost or damaged, they
are re-sent; however, TCP requires a lot of system overhead compared to UDP.

computer and then reassemble the data back into a form that the application layer can
understand and process.
Two main protocols work at the transport layer: TCP and UDP. TCP is a reliable and
connection-oriented protocol, which means it ensures packets are delivered to the destination
computer. If a packet is lost during transmission, TCP has the ability to identify this issue
and resend the lost or corrupted packet. TCP also supports packet sequencing (to ensure
each and every packet was received), flow and congestion control, and error detection
and correction. UDP, on the other hand, is a best-effort and connectionless protocol. It has
neither packet sequencing nor flow and congestion control, and the destination does not
acknowledge every packet it receives.

Connection-oriented communication
performs handshaking, sets up a virtual
circuit, and verifies that each packet
reaches its destination.

Host A

Host B

Connectionless communication
just puts the packets
on the wire.

Figure 11-18 Connection-oriented protocol vs. connectionless protocol functionality

If developers know that data being dropped during transmission is not detrimental to the application, they may choose to use UDP because it is faster and requires fewer resources. For example, UDP is a better choice than TCP when a server sends status information to all listening nodes on the network. A node will not be negatively affected if, by some chance, it did not receive this status information, because the information will be re-sent every 60 seconds.

UDP and TCP are transport protocols that applications use to get their data across a network. They both use ports to communicate with upper OSI layers and to keep track of various conversations that take place simultaneously. The ports are also the mechanism used to identify how other computers access services. When a TCP or UDP message is formed, source and destination ports are contained within the header information along with the source and destination IP addresses. The combination of protocol (TCP or UDP), port, and IP address makes up a socket, and is how packets know where to go (by the address) and how to communicate with the right service or protocol on the other computer (by the port number). The IP address acts as the doorway to a computer, and the port acts as the doorway to the actual protocol or service. To communicate properly, the packet needs to know these doors. Figure 11-19 shows how packets communicate with applications and services through ports.

Web browser

Telnet

SMTP

E-mail client

DNS

HTTP

Ping utility

POP

SSH client

ECHO

FTP

SSH

DNS

SNMP

7

21

22

53

161

Application

23

25

53

80

110

UDP

Transport

Network
Data link
Physical

Figure 11-19 The packet can communicate with upper-layer protocols and services through a port.

The difference between TCP and UDP can also be seen in the message formats. Because TCP offers more services than UDP, it must contain much more information within its packet header format, as shown in Figure 11-20. Table 11-4 lists the major
differences between TCP and UDP.

TCP

Source port

Destination port

Source port

Destination port

Length

Sequence number

Data

Acknowledgment number
Offset Reserved Flags

Window

Checksum

Urgent pointer

Options

Padding

Checksum

UDP format

Data

TCP format

Figure 11-20 TCP carries a lot more information within its segment because it offers more
services than UDP.

Property

TCP

UDP

Reliability

Ensures that packets reach their

	destinations, returns ACKs when packets are received, and is a reliable protocol.
	Does not return ACKs and does not guarantee that a packet will reach its destination. Is an unreliable protocol.
Connection	
	Connection-oriented. It performs handshaking and develops a virtual connection with the destination computer.
	Connectionless. It does no handshaking and does not set up a virtual connection.
Packet sequencing	
	Uses sequence numbers within headers to make sure each packet within a transmission is received.
	Does not use sequence numbers.
Congestion controls	
	The destination computer can tell the source if it is overwhelmed and thus slow the transmission rate.
	The destination computer does not communicate back to the source computer about flow control.
Usage	
	Used when reliable delivery is required. Intended for relatively small amounts of data transmission.
	Used when reliable delivery is not required and high volumes of data need to be transmitted, such as in streaming video and status broadcasts.
Speed and overhead	
	Uses a considerable amount of

resources and is slower than UDP.

Uses fewer resources and is faster
than TCP.

Table 11-4

Major Differences Between TCP and UDP

Port Types

Port numbers up to 1023 (0 to 1023) are called well-known ports, and almost every computer in the world has the exact same protocol mapped to the exact same port number. That is why they are called well known—everyone follows this same standardized approach. This means that on almost every computer, port 25 is mapped to SMTP, port 80 is mapped to HTTP, and so on. This mapping between lowernumbered ports and specific protocols is a de facto standard, which just means that we all do this and that we do not have a standards body dictating that it absolutely has to be done this way. The fact that almost everyone follows this approach translates to more interoperability among systems all over the world. Because this is a de facto standard and not a standard that absolutely must be followed, administrators can map different protocols to different port numbers if that fits their purpose. However, one thing to note is that ports 0 to 1023 can be used only by privileged system or root processes.

The following shows some of the most commonly used protocols and the ports to which they are usually mapped:

Registered ports are 1024 to 49151, which can be registered with the Internet Assigned Numbers Authority (IANA) for a particular use. Vendors register specific ports to map to their proprietary software. Dynamic ports (also known as ephemeral ports) are 49152 to 65535 and are available to be used by any application on an "as needed" basis. Typically, these ports are used on the client side of a connection.

For instance, if you look at a specific connection between your web browser and a website you visit, you may notice that the destination port is 80 and the source port (on your client) is 53042. In some cases, however, the server may be listening on a well-known port (e.g., 135 for RPC) and hand off the server port to an ephemeral one. This means that in some cases, you will see both the client and the server port for a connection in the range of 49152 to 65535.

- Secure Shell (SSH) port 22
- SMTP port 25
- DNS port 53
- HTTP port 80
- NTP port 123
- IMAP port 143
- HTTP Secure (HTTPS) port 443

TCP Handshake

TCP must set up a virtual connection between two hosts before any data is sent. This means the two hosts must agree on certain parameters, data flow, windowing, error detection, and options. These issues are negotiated during the handshaking phase, as shown in Figure 11-21.

The host that initiates communication sends a synchronization (SYN) packet to the receiver. The receiver acknowledges this request by sending a SYN/ACK packet. This packet translates into, “I have received your request and am ready to communicate with you.” The sending host acknowledges this with an acknowledgment (ACK) packet, which translates into, “I received your acknowledgment. Let’s start transmitting our data.” This completes the handshaking phase, after which a virtual connection is set up, and actual data can now be passed. The connection that has been set up at this point is considered full duplex, which means transmission in both directions is possible using the same transmission line.

If an attacker sends a target system SYN packets with a spoofed address, then the victim system replies to the spoofed address with SYN/ACK packets. Each time the victim system receives one of these SYN packets, it sets aside resources to manage the new connection.

If the attacker floods the victim system with SYN packets, eventually the victim system allocates all of its available TCP connection resources and can no longer process new requests. This is a type of DoS attack that is referred to as a SYN flood. To thwart this type of attack you can use a number of mitigations, the most common of which are described in the Internet Engineering Task Force’s (IETF) Request for Comments (RFC) 4987. One of the most effective techniques described in RFC 4987 is the use of SYN caches,

which
delays the allocation of a socket until the handshake is completed.
Another attack vector we need to understand is TCP sequence numbers. One of the values that is agreed upon during a TCP handshake between two systems is the sequence
numbers that will be inserted into the packet headers. Once the sequence number is
agreed upon, if a receiving system receives a packet from the sending system that does
not have this predetermined value, it disregards the packet. This means that an attacker
cannot just spoof the address of a sending system to fool a receiving system; the attacker
has to spoof the sender’s address and use the correct sequence number values. If an
attacker can correctly predict the TCP sequence numbers that two systems will use,
then she can create packets containing those numbers and fool the receiving system
into thinking that the packets are coming from the authorized sending system. She can
then take over the TCP connection between the two systems, which is referred to as
TCP session hijacking.

Figure 11-21
The TCP threeway handshake

Host A

1.

SYN

2.

SYN/ACK

3.

ACK

Host B

Data Structures
As stated earlier, the message is formed and passed to the application layer from a program and sent down through the protocol stack. Each protocol at each layer adds its
own information to the message to create a PDU and passes it down to the next layer.
This activity is referred to as encapsulation. As the message is passed down the

stack, it goes through a sort of evolution, and each stage has a specific name that indicates what is taking place. When an application formats data to be transmitted over the network, the PDU is called a message or data. The message is sent to the transport layer, where TCP does its magic on it. The PDU is now a segment. The segment is sent to the network layer. The network layer adds routing and addressing, and now the PDU is called a packet. The network layer passes off the packet to the data link layer, which frames the packet with a header and a trailer, and now it is called a frame. Figure 11-22 illustrates these stages.

EXAM TIP If the message is being transmitted over TCP, it is referred to as a "segment." If it is being transmitted over UDP, it is referred to as a "datagram."

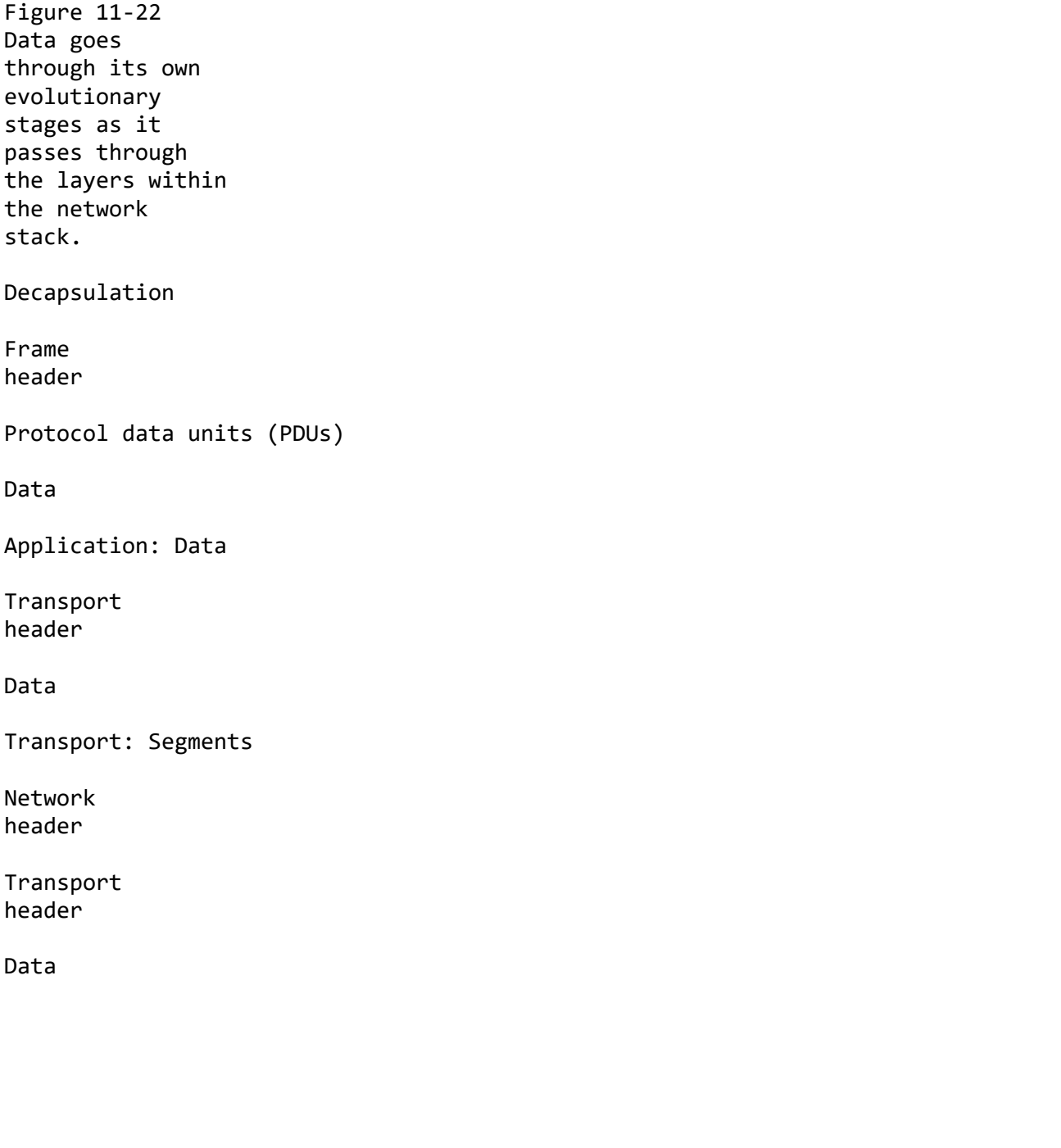

Figure 11-22 Data goes through its own evolutionary stages as it passes through the layers within the network stack.

Network: Packets

Network
header

Transport
header

Data

Frame
trailer

Data link:
Frames

1 0 1 0 0 1 0 0 0 1 0 1 0 1 0 0 0 1 1 1 0 1 0 1 0 1 0 0 0 0 1 0 0 Bits

PART IV

Sometimes when an author refers to a segment, she is specifying the stage in which
the data is located within the protocol stack. If the literature is describing routers, which
work at the network layer, the author might use the word “packet” because the data at
this layer has routing and addressing information attached. If an author is describing
network traffic and flow control, she might use the word “frame” because all data actually
ends up in the frame format before it is put on the network wire. The important thing
here is that you understand the various steps a data package goes through when it moves
up and down the protocol stack.

IP Addressing
Each node on a network must have a unique IP address. Today, the most commonly used
version of IP is IP version 4 (IPv4), which is used by roughly 70 percent of Internet hosts
as we write these words. IP version 6 (IPv6), which was created in part to address the
shortage of IPv4 addresses (IPv6 also has many security features built into it that are not
part of IPv4), is steadily gaining ground, however. IPv6 is covered later in this chapter.
IPv4 uses 32 bits for its addresses, whereas IPv6 uses 128 bits; thus, IPv6 provides

more possible addresses with which to work. Each address has a host portion and a network portion, and the addresses are grouped into classes and then into subnets. The subnet mask of the address differentiates the groups of addresses that define the subnets of a network. IPv4 address classes are listed in Table 11-5.

For any given IP network within an organization, all nodes connected to the network can have different host addresses but a common network address. The host address identifies every individual node, whereas the network address is the identity of the network all the nodes are connected to; therefore, it is the same for each one of them. Any traffic meant for nodes on this network will be sent to the prescribed network address.

A subnet is created from the host portion of an IP address to designate a "sub" network. This allows us to further break the host portion of the address into two or more logical groupings, as shown in Figure 11-23. A network can be logically partitioned to reduce administration headaches, increase traffic performance, and potentially strengthen security. As an analogy, let's say you work at Toddlers R Us and you are responsible for babysitting 100 toddlers. If you kept all 100 toddlers in one room, you would probably end up crazy. To better manage these kids, you could break them up into groups. The three-year-olds go in the yellow room, the four-year-olds go in the green room, and the five-year-olds go in the blue room. This is what a network administrator would do—break up and separate computer nodes to be able to better control them. Instead of putting them into physical rooms, the administrator puts them into logical rooms (subnets).

To continue with our analogy, when you put your toddlers in different rooms, you would have physical barriers that separate them—walls. Network subnetting is not physical; it is logical. This means you would not have physical walls separating your individual subnets, so how do you keep them separate? This is where subnet masks

Class

Address Range

Description

A

0.0.0.0 to 127.255.255.255

The first byte is the network portion, and the remaining 3 bytes are the host portion.

B

128.0.0.0 to 191.255.255.255

The first 2 bytes are the network portion, and the
remaining 2 bytes are the host portion.

C

192.0.0.0 to 223.255.255.255

The first 3 bytes are the network portion, and the
remaining 1 byte is the host portion.

D

224.0.0.0 to 239.255.255.255

Used for multicast addresses.

E

240.0.0.0 to 255.255.255.255

Reserved for research.

Table 11-5

IPv4 Addressing

Router

Figure 11-23
Subnets create
logical partitions.

Subnet 1
132.201.1.x

Subnet 2
132.201.2.x

Subnet 3
132.201.3.x

come into play. A subnet mask defines smaller networks inside a larger network, just like
individual rooms are defined within a building.
Subnetting allows larger IP address ranges to be divided into smaller, logical,

and more
tangible network segments. Consider an organization with several divisions, such as IT,
Accounting, HR, and so on. Creating subnets for each division breaks the networks
into logical partitions that route traffic directly to recipients without dispersing data all
over the network. This drastically reduces the traffic load across the network, reducing
the possibility of network congestion and excessive broadcast packets in the network.
Implementing network security policies is also much more effective across logically
categorized subnets with a demarcated perimeter, as compared to a large, cluttered, and
complex network.
Subnetting is particularly beneficial in keeping down routing table sizes because
external routers can directly send data to the actual network segment without having to
worry about the internal architecture of that network and getting the data to individual
hosts. This job can be handled by the internal routers, which can determine the individual
hosts in a subnetted environment and save the external routers the hassle of analyzing all
32 bits of an IP address and just look at the “masked” bits.
TIP You should not have to calculate any subnets for the CISSP exam, but
for a better understanding of how this stuff works under the hood, check out
the article “IP Tutorial: Subnet Mask and Subnetting” at https://www.lifewire
.com/internet-protocol-tutorial-subnets-818378 (keep in mind that URLs are
subject to change from time to time).

If the traditional subnet masks are used, they are referred to as classful or classical IP
addresses. If an organization needs to create subnets that do not follow these traditional
sizes, then it would use classless IP addresses. This just means a different subnet mask
would be used to define the network and host portions of the addresses. After it became
clear that available IP addresses were running out as more individuals and corporations
participated on the Internet, classless interdomain routing (CIDR) was created. A Class
B address range is usually too large for most organizations, and a Class C address range
is too small, so CIDR provides the flexibility to increase or decrease the class sizes as
necessary. CIDR is the method to specify more flexible IP address classes. CIDR is also

referred to as supernetting.
TIP To better understand CIDR, a good resource is “IP Classless Addressing: Classless Inter-Domain Routing (CIDR)/“Supernetting”: www.tcpipguide.com/free/t_IPClasslessAddressingClasslessInterDomainRoutingCI.htm.

Although each node has an IP address, people usually refer to their hostname rather than their IP address. Hostnames, such as www.mheducation.com, are easier for humans to remember than IP addresses, such as 198.105.254.228. However, the use of these two nomenclatures requires mapping between the hostnames and IP addresses because the computer understands only the numbering scheme. This process is addressed in the “Domain Name Service” section later in this chapter.
NOTE IP provides addressing, packet fragmentation, and packet timeouts. To ensure that packets do not continually traverse a network forever, IP provides a Time to Live (TTL) value that is decremented every time the packet passes through a router.

IPv6
IPv6, also called IP next generation (IPng), not only has a larger address space than IPv4 to support more IP addresses; it has some capabilities that IPv4 does not and it accomplishes some of the same tasks differently. All of the specifics of the new functions within IPv6 are beyond the scope of this book, but we will look at a few of them, because IPv6 is the way of the future. IPv6 allows for scoped addresses, which enables an administrator to restrict specific addresses for specific servers or file and print sharing, for example.
IPv6 has Internet Protocol Security (IPSec) integrated into the protocol stack, which provides end-to-end secure transmission and authentication. IPv6 has more flexibility and routing capabilities and allows for Quality of Service (QoS) priority values to be assigned to time-sensitive transmissions. The protocol offers autoconfiguration, which makes administration much easier, and it does not require network address translation (NAT) to extend its address space.
NAT was developed because IPv4 addresses were running out. Although the NAT technology is extremely useful, it has caused a lot of overhead and transmission problems because it breaks the client/server model that many applications use today. One reason the industry did not jump on the IPv6 bandwagon when it came out years ago is that NAT was developed, which reduced the speed at which IP addresses were being depleted.

Although the conversion rate from IPv4 to IPv6 is slow in some parts of the world
and the implementation process is quite complicated, the industry is making the shift
because of all the benefits that IPv6 brings to the table.
NOTE NAT is covered in the “Network Address Translation” section later in this chapter.

The IPv6 specification, as outlined in RFC 8200, lays out the differences and benefits
of IPv6 over IPv4. A few of the differences are as follows:

IPv4 header
0

4

8

Version

IHL

12

16

20

Type of service

28

31

Total length

Identification
Time to live

24

Flags

Protocol

Fragment offset
Header checksum

Source address
Destination address

IPv6 header
0

4

8

Version Traffic class

12

16

20

24

Flow label

28

32

36

44

Payload length
Source address

Destination address

Figure 11-24 IPv4 vs. IPv6 headers

40

48

52

56

Next header

60
Hop limit

63

PART IV

• IPv6 increases the IP address size from 32 bits to 128 bits to support more levels
of addressing hierarchy, a much greater number of addressable nodes, and simpler

autoconfiguration of addresses.
• The scalability of multicast routing is improved by adding a “scope” field to multicast addresses. Also, a new type of address called an anycast address is defined, which is used to send a packet to any one of a group of nodes.
• Some IPv4 header fields have been dropped or made optional to reduce the common-case processing cost of packet handling and to limit the bandwidth cost of the IPv6 header. This is illustrated in Figure 11-24.

• Changes in the way IP header options are encoded allow for more efficient forwarding, less stringent limits on the length of options, and greater flexibility
for introducing new options in the future.
• A new capability is added to enable the labeling of packets belonging to particular traffic “flows” for which the sender requests special handling, such as
nondefault QoS or “real-time” service.
• Extensions to support authentication, data integrity, and (optional) data confidentiality are also specified for IPv6.
IPv4 limits packets to 65,535 bytes of payload, and IPv6 extends this size to 4,294,967,295 bytes. These larger packets are referred to as jumbograms and improve
performance over high-MTU links. Currently most of the world still uses IPv4, but IPv6
is being deployed more rapidly. This means that there are “pockets” of networks using IPv4
and “pockets” of networks using IPv6 that still need to communicate. This communication
takes place through different tunneling techniques, which either encapsulate IPv6
packets within IPv4 packets or carry out automated address translations. Automatic
tunneling is a technique where the routing infrastructure automatically determines the
tunnel endpoints so that protocol tunneling can take place without preconfiguration.
In the 6to4 tunneling method, the tunnel endpoints are determined by using a wellknown IPv4 anycast address on the remote side and embedding IPv4 address data within
IPv6 addresses on the local side. Teredo is another automatic tunneling technique that
uses UDP encapsulation so that NAT address translations are not affected. Intra-Site
Automatic Tunnel Addressing Protocol (ISATAP) treats the IPv4 network as a virtual IPv6
local link, with mappings from each IPv4 address to a link-local IPv6 address.
The 6to4 and Teredo are intersite tunneling mechanisms, and ISATAP is an intrasite
mechanism. So the first two are used for connectivity between different networks,
and ISATAP is used for connectivity of systems within a specific network. Notice in

Figure 11-25 that 6to4 and Teredo are used on the Internet and ISATAP is used within an intranet.

While many of these automatic tunneling techniques reduce administration overhead, because network administrators do not have to configure each and every system and network device with two different IP addresses, there are security risks that need to be understood. Many times users and network administrators do not know that automatic tunneling capabilities are enabled, and thus they do not ensure that these different tunnels are secured and/or are being monitored. If you are an administrator of a network and have intrusion detection systems (IDSs), intrusion prevention systems (IPSs), and firewalls that are only configured to monitor and restrict IPv4 traffic, then all IPv6 traffic could be traversing your network insecurely. Attackers use these protocol tunnels and misconfigurations to get past these types of security devices so that malicious activities can take place unnoticed. If you are a user and have a host-based firewall that only understands IPv4 and your operating system has a dual IPv4/IPv6 networking stack, traffic could be bypassing your firewall without being monitored and logged. The use of Teredo can actually open ports in NAT devices that allow for unintended traffic in and out of a network.

Client connecting from globally routable IPv6 address

External CRL distribution

Domain controllers

NAP servers

Network location server

DNS servers

Certification

authority

IPv6
6to4
Client connecting from
public IPv4 address

Teredo
Client connecting from
private (NAT) IPv4 address

IPv4

ISATAP tun

neled IPV6

traffic

Internal CRL
distribution point

IPv6

IP-HTTPS
Client connecting from
behind a firewall, or unable
to connect via other
methods

Application servers
running IPv4

Internet

Application servers
running ISATAP

Application servers
running native IPv6

Intranet

It is critical that people who are responsible for configuring and maintaining systems
and networks understand the differences between IPv4 and IPv6 and how the various
tunneling mechanisms work so that all vulnerabilities are identified and properly
addressed. Products and software may need to be updated to address both traffic types,
proxies may need to be deployed to manage traffic communication securely, IPv6 should
be disabled if not needed, and security appliances need to be configured to

monitor all
traffic types.

Address Resolution Protocol
On a TCP/IP network, each computer and network device requires a unique IP address
and a unique physical hardware address. Each NIC has a unique 48-bit physical address
that is programmed by the manufacturer into the ROM chips on the card. The physical
address is also referred to as the Media Access Control (MAC) address. The network layer
works with and understands IP addresses, and the data link layer works with and
understands physical MAC addresses. So, how do these two types of addresses work together
while operating at different layers?
NOTE A MAC address is unique because the first 24 bits represent the manufacturer code and the last 24 bits represent the unique serial number assigned by the manufacturer.

PART IV

Figure 11-25 Various IPv4 to IPv6 tunneling techniques

When data comes from the application layer, it goes to the transport layer for sequence
numbers, session establishment, and streaming. The data is then passed to the network
layer, where routing information is added to each packet and the source and destination
IP addresses are attached to the data bundle. Then this goes to the data link layer, which
must find the MAC address and add it to the header portion of the frame. When a frame
hits the wire, it only knows what MAC address it is heading toward. At this lower layer of
the OSI model, the mechanisms do not even understand IP addresses. So if a computer
cannot resolve the IP address passed down from the network layer to the corresponding
MAC address, it cannot communicate with that destination computer.
NOTE A frame is data that is fully encapsulated, with all of the necessary headers and trailers.

MAC and IP addresses must be properly mapped so they can be correctly resolved. This happens through the Address Resolution Protocol (ARP). When the data link layer
receives a frame, the network layer has already attached the destination IP address to
it, but the data link layer cannot understand the IP address and thus invokes ARP for

help. ARP broadcasts a frame requesting the MAC address that corresponds with the destination IP address. Each computer on the broadcast domain receives this frame, and all but the computer that has the requested IP address ignore it. The computer that has the destination IP address responds with its MAC address. Now ARP knows what hardware address corresponds with that specific IP address. The data link layer takes the frame, adds the hardware address to it, and passes it on to the physical layer, which enables the frame to hit the wire and go to the destination computer. ARP maps the hardware address and associated IP address and stores this mapping in its table for a predefined amount of time. This caching is done so that when another frame destined for the same IP address needs to hit the wire, ARP does not need to broadcast its request again. It just looks in its table for this information.

Sometimes attackers alter a system's ARP table so it contains incorrect information. This is called ARP table cache poisoning. The attacker's goal is to receive packets intended for another computer. This is a type of masquerading attack. For example, let's say that Bob's computer has an IP address of 10.0.0.1 and a MAC address of bb:bb:bb:bb:bb:bb, Alice's computer has an IP address of 10.0.0.7 and a MAC address of aa:aa:aa:aa:aa:aa, and an attacker has an IP address of 10.0.0.3 and a MAC address of cc:cc:cc:cc:cc:cc, as shown in Figure 11-26. Suppose Bob wants to send a message to Alice. The message is encapsulated at the IP layer with information including Alice's IP address and then handed off to the data link layer. If this is the first message for Alice's computer, the data link process on Bob's computer has no way of knowing her MAC address, so it crafts an ARP query that (literally) says "Who has 10.0.0.7?" This ARP frame is broadcast to the network, where it is received by both Alice's computer and the attacker's computer. Both respond claiming to be the rightful owners of that IP address. What does Bob's computer do when faced with multiple different responses? The answer in most cases is that it uses the most recent response. If the attacker wants to ensure that Bob's ARP table remains poisoned, then he will have to keep pumping out bogus ARP replies.

IP: 10.0.0.7
MAC: aa:aa:aa:aa:aa:aa

Alice

Who has
10.0.0.7?
10.0.0.7 is at
aa: aa:aa:aa:aa:aa
Bob

IP: 10.0.0.1
MAC: bb:bb:bb:bb:bb:bb

10.0.0.7 is at
cc:cc:cc:cc:cc:cc:
10.0.0.7 is at
cc:cc:cc:cc:cc:cc:
10.0.0.7 is at
cc:cc:cc:cc:cc:cc:
10.0.0.7 is at
cc:cc:cc:cc:cc:cc:

Attacker

IP: 10.0.0.3
MAC: cc:cc:cc:cc:cc:cc

Figure 11-26 ARP poisoning attack

Dynamic Host Configuration Protocol
A computer can receive its IP addresses in a few different ways when it first boots up. If
it has a statically assigned address, nothing needs to happen. It already has the configuration settings it needs to communicate and work on the intended network. If a computer
depends upon a DHCP server to assign it the correct IP address, it boots up and makes
a request to the DHCP server. The DHCP server assigns the IP address, and everyone
is happy.
DHCP is a UDP-based protocol that allows servers to assign IP addresses to network
clients in real time. Unlike static IP addresses, where IP addresses are manually configured,
the DHCP server automatically checks for available IP addresses and correspondingly
assigns an IP address to the client. This eliminates the possibility of IP address conflicts
that occur if two systems are assigned identical IP addresses, which could cause loss of
service. On the whole, DHCP considerably reduces the effort involved in managing large-scale IP networks.

The DHCP server assigns IP addresses in real time from a specified range when a client connects to the network; this is different from static addresses, where each system

PART IV

So ARP is critical for a system to communicate, but it can be manipulated to allow
traffic to be sent to unintended systems. ARP is a rudimentary protocol and does not have any security measures built in to protect itself from these types of attacks.
Networks should have IDS sensors monitoring for this type of malicious activity so that
administrators can be alerted if it is underway. This is not difficult to detect, since, as
already noted, the attacker will have to constantly (or at least frequently) transmit bogus
ARP replies.

is individually assigned a specific IP address when coming online. In a standard DHCPbased network, the client computer broadcasts a DHCPDISCOVER message on the network in search of the DHCP server. Once the respective DHCP server receives the
DHCPDISCOVER request, the server responds with a DHCPOFFER packet, offering the client an IP address. The server assigns the IP address based on the subject of the
availability of that IP address and in compliance with its network administration policies.
The DHCPOFFER packet that the server responds with contains the assigned IP address
information and configuration settings for client-side services.
Once the client receives the settings sent by the server through the DHCPOFFER packet, it responds to the server with a DHCPREQUEST packet confirming its acceptance of the allotted settings. The server now acknowledges with a DHCPACK packet, which includes the validity period (lease) for the allocated parameters.
Client

DHCP server
1. DHCP discover

2. DHCP offer

3. DHCP request

4. DHCP ack

So as shown in Figure 11-27, the DHCP client yells out to the network, “Who can help me get an address?” The DHCP server responds with an offer: “Here is an address
and the parameters that go with it.” The client accepts this gracious offer with the

DHCPREQUEST message, and the server acknowledges this message. Now the client can start interacting with other devices on the network and the user can surf the Web
and check her e-mail.
Unfortunately, both the client and server segments of DHCP are vulnerable to falsified identity. On the client end, attackers can masquerade their systems to appear as
valid network clients. This enables rogue systems to become a part of an organization’s
network and potentially infiltrate other systems on the network. An attacker may create
an unauthorized DHCP server on the network and start responding to clients searching

519
Client

DHCP
server

• DHCPDISCOVER—Client searches for the presence of DHCP servers.

• DHCPOFFER—DHCP server offers the client an available IP address and service settings.

• DHCPREQUEST—Client confirms accepting the allocated settings.
Client

DHCP
server

• DHCPACK—DHCP server acknowledges that the particular IP address has now been allocated to the client system for a specific lease period.

Figure 11-27 The four stages of the Discover, Offer, Request, and Acknowledgment (D-O-R-A)
process

for a DHCP server. A DHCP server controlled by an attacker can compromise client system configurations, carry out man-in-the-middle attacks, route traffic to unauthorized
networks, and a lot more, with the end result of jeopardizing the entire network.
An effective method to shield networks from unauthenticated DHCP clients is through the use of DHCP snooping on network switches. DHCP snooping ensures that DHCP servers can assign IP addresses to only selected systems, identified by their MAC
addresses. Also, advanced network switches have the capability to direct clients toward
legitimate DHCP servers to get IP addresses and restrict rogue systems from

becoming
DHCP servers on the network.
Diskless workstations do not have a full operating system but have just enough code
to know how to boot up and broadcast for an IP address, and they may have a pointer to
the server that holds the operating system. The diskless workstation knows its hardware
address, so it broadcasts this information so that a listening server can assign it the
correct IP address. As with ARP, Reverse Address Resolution Protocol (RARP) frames go to
all systems on the subnet, but only the RARP server responds. Once the RARP server
receives this request, it looks in its table to see which IP address matches the broadcast
hardware address. The server then sends a message that contains its IP address back to
the requesting computer. The system now has an IP address and can function on the
network.
The Bootstrap Protocol (BOOTP) was created after RARP to enhance the functionality
that RARP provides for diskless workstations. The diskless workstation can receive its
IP address, the name server address for future name resolutions, and the default gateway
address from the BOOTP server. BOOTP usually provides more functionality to diskless
workstations than does RARP.

Internet Control Message Protocol
The Internet Control Message Protocol (ICMP) is basically IP’s “messenger boy.” ICMP
delivers status messages, reports errors, replies to certain requests, and reports routing
information and is used to test connectivity and troubleshoot problems on IP networks.
The most commonly understood use of ICMP is its use by the ping utility. When a
person wants to test connectivity to another system, he may ping it, which sends out
ICMP Echo Request frames. The replies on his screen that are returned to the ping utility
are called ICMP Echo Reply frames and are responding to the Echo Request frames. If a
reply is not returned within a predefined time period, the ping utility sends more Echo
Request frames. If there is still no reply, ping indicates the host is unreachable.
ICMP also indicates when problems occur with a specific route on the network and

tells
surrounding routers about better routes to take based on the health and congestion of the
various pathways. Routers use ICMP to send messages in response to packets that could
not be delivered. The router selects the proper ICMP response and sends it back to the
requesting host, indicating that problems were encountered with the transmission request.
ICMP is used by other connectionless protocols, not just IP, because connectionless
protocols do not have any way of detecting and reacting to transmission errors, as do
connection-oriented protocols. In these instances, the connectionless protocol may use
ICMP to send error messages back to the sending system to indicate networking problems.
As you can see in Table 11-6, ICMP is used for many different networking purposes.
This table lists the various messages that can be sent to systems and devices through ICMP.

Attacks Using ICMP
ICMP was developed to send status messages, not to hold or transmit user data. But
someone figured out how to insert some data inside of an ICMP packet, which can be
used to communicate to an already compromised system. This technique is called ICMP
tunneling, and is an older, but still effective, client/server approach that can be used by
hackers to set up and maintain covert communication channels to compromised systems.
The attacker would target a computer and install the server portion of the tunneling
software. This server portion would “listen” on a port, which is the back door an attacker
can use to access the system. To gain access and open a remote shell to this computer,
an attacker would send commands inside of ICMP packets. This is usually successful
because many routers and firewalls are configured to allow ICMP traffic to come and go
out of the network, based on the assumption that this is safe because ICMP was developed to not hold any data or a payload.
Just as any tool that can be used for good can also be used for evil, attackers commonly
use ICMP to redirect traffic. The redirected traffic can go to the attacker’s dedicated
system, or it can go into a “black hole.” Routers use ICMP messages to update each other
on network link status. An attacker could send a bogus ICMP message with incorrect
information, which could cause the routers to divert network traffic to where

the attacker
indicates it should go.
ICMP is also used as the core protocol for a network tool called Traceroute. Traceroute
is used to diagnose network connections, but since it gathers a lot of important network
statistics, attackers use the tool to map out a victim’s network. This is similar to a burglar

521
Table 11-6
ICMP Message
Types

Name	
0	Echo Reply
1	Unassigned
2	Unassigned
3	Destination Unreachable
4	Source Quench
5	Redirect
6	Alternate Host Address
7	Unassigned
8	Echo Request

9

Router Advertisement

10

Router Solicitation

11

Time Exceeded

12

Parameter Problem

13

Timestamp

14

Timestamp Reply

15

Information Request

16

Information Reply

17

Address Mask Request

18

Address Mask Reply

19

Reserved (for Security)

20–29

Reserved (for Robustness Experiment)

30

Traceroute

31

Datagram Conversion Error

32

Mobile Host Redirect

33

IPv6 Where-Are-You

34

IPv6 I-Am-Here

35

Mobile Registration Request

36

Mobile Registration Reply

37

Domain Name Request

38

Domain Name Reply

39

SKIP

40

Photuris (Disambiguation)

41

ICMP messages utilized by experimental mobility protocols
such as Seamoby

PART IV

Type

“casing the joint,” meaning that the more the attacker learns about the environment,
the easier it can be for her to exploit some critical targets. So while the Traceroute tool

is a valid networking program, a security administrator might configure the IDS sensors to monitor for extensive use of this tool because it could indicate that an attacker is attempting to map out the network's architecture.

The countermeasures to these types of attacks are to use firewall rules that only allow the necessary ICMP packets into the network and the use of an IDS or IPS to watch for suspicious activities. Host-based protection (host firewalls and host IDS) can also be installed and configured to identify this type of suspicious behavior.

Simple Network Management Protocol

The Simple Network Management Protocol (SNMP) was released to the networking world in 1988 to help with the growing demand of managing network IP devices. Organizations use many types of products that use SNMP to view the status of their network, traffic flows, and the hosts within the network. Since these tasks are commonly carried out using graphical user interface (GUI)–based applications, many people do not have a full understanding of how the protocol actually works. The protocol is important to understand because it can provide a wealth of information to attackers, and you should understand the amount of information that is available to the ones who wish to do you harm, how they actually access this data, and what can be done with it.

The two main components within SNMP are managers and agents. The manager is the server portion, which polls different devices to check status information. The server component also receives trap messages from agents and provides a centralized place to hold all network-wide information. The agent is a piece of software that runs on a network device, which is commonly integrated into the operating system. The agent has a list of objects that it is to keep track of, which is held in a database-like structure called the Management Information Base (MIB). A MIB is a logical grouping of managed objects that contain data used for specific management tasks and status checks.

When the SNMP manager component polls the individual agent installed on a specific device, the agent pulls the data it has collected from the MIB and sends it to the manager. Figure 11-28 illustrates how data pulled from different devices is located in one centralized location (SNMP manager). This allows the network administrator to have a holistic view of the network and the devices that make up that network.

NOTE The trap operation allows the agent to inform the manager of an event, instead of having to wait to be polled. For example, if an interface on

a router goes down, an agent can send a trap message to the manager. This is the only way an agent can communicate with the manager without first being polled.

It might be necessary to restrict which managers can request information of an agent,
so communities were developed to establish a trust between specific agents and managers.
A community string is basically a password a manager uses to request data from the agent,
and there are two main community strings with different levels of access: read-only and

SNMP agent
MIB
SNMP agent
MIB

SNMP agent
MIB

SNMP agent
MIB

WAN

SNMP agent
MIB

SNMP agent
MIB
SNMP agent
MIB
Statistics

Alerts

Events

SNMP manager

read-write. As the names imply, the read-only community string allows a manager to read
data held within a device’s MIB, and the read-write string allows a manager to read the
data and modify it. If an attacker can uncover the read-write string, she could change
values held within the MIB, which could reconfigure the device.
Since the community string is a password, it should be hard to guess and be protected.
It should contain mixed-case alphanumeric strings that are not dictionary words.

This practice is not always the case in many networks. The usual default read-only community string is “public” and the read-write string is “private.” Many organizations do not change these, so anyone who can connect to port 161 can read the status information of a device and potentially reconfigure it. Different vendors may put in their own default community string values, but organizations may still not take the necessary steps to change them. Attackers usually have lists of default vendor community string values, so they can be easily discovered and used against networks.

To make matters worse, the community strings are sent in cleartext in SNMP v1 and v2, so even if a company does the right thing by changing the default values, the strings are still easily accessible to any attacker with a sniffer. If you absolutely have to use v1 or v2 (and you really shouldn’t because they are obsolete), make sure that different network segments use different community strings, so that if one string is compromised an attacker cannot gain access to all the devices in the network. The SNMP ports (161 and 162) should not be open to untrusted networks, like the Internet, and if needed they should be filtered to ensure only authorized individuals can connect to them. If these

Figure 11-28 Agents provide the manager with SNMP data.

ports need to be available to an untrusted network, configure the router or firewall to only allow UDP traffic to come and go from preapproved network-management stations.

While versions 1 and 2 of this protocol send the community string values in cleartext, version 3 has cryptographic functionality, which provides encryption, message integrity, and authentication security. So, SNMP v3 should be implemented for more granular protection.

If the proper countermeasures are not put into place, then an attacker can gain access to a wealth of device-oriented data that can be used in her follow-up attacks. The following are just some data sets held within MIB SNMP objects that attackers would be interested in:

.server.svSvcTable.svSvcEntry.svSvcName

Running services

.server.svShareTable.svShareEntry.svShareName

Share names

.server.sv.ShareTable.svShareEntry.svSharePath

Share paths

.server.sv.ShareTable.svShareEntry.svShareComment

Comments on shares

.server.svUserTable.svUserEntry.svUserName

Usernames

.domain.domPrimaryDomain8

Domain names

Gathering this type of data allows an attacker to map out the target network and enumerate the nodes that make up the network.
As with all tools, SNMP is used for good purposes (network management) and for bad
purposes (target mapping, device reconfiguration). We need to understand both sides of
all tools available to us.

Domain Name Service
Imagine how hard it would be to use the Internet if we had to remember actual specific
IP addresses to get to various websites. The Domain Name Service (DNS) is a method of
resolving hostnames to IP addresses so names can be used instead of IP addresses within
networked environments.
The first iteration of the Internet was made up of about 100 computers (versus over 22 billion now), and a list was kept that mapped every system's hostname to its
IP address. This list was kept on an FTP server so everyone could access it. It did not take
long for the task of maintaining this list to become overwhelming, and the computing
community looked to automate it.
When a user types a uniform resource locator (URL) into his web browser, the URL is made up of words or letters that are in a sequence that makes sense to that user, such
as www.google.com. However, these words are only for humans—computers work with IP addresses. So after the user enters this URL and presses □□□□□, behind the scenes
his computer is actually being directed to a DNS server that will resolve this

URL, or
hostname, into an IP address that the computer understands. Once the hostname has
been resolved into an IP address, the computer knows how to get to the web server
holding the requested web page.

Many organizations have their own DNS servers to resolve their internal hostnames.
These organizations usually also use the DNS servers at their Internet service providers
(ISPs) to resolve hostnames on the Internet. An internal DNS server can be used to
resolve hostnames on the entire LAN, but usually more than one DNS server is used so
the load can be split up and so redundancy and fault tolerance are in place.
Within DNS servers, DNS namespaces are split up administratively into zones.
One zone may contain all hostnames for the marketing and accounting departments,
and another zone may contain hostnames for the administration, research, and legal
departments. The DNS server that holds the files for one of these zones is said to be
the authoritative name server for that particular zone. A zone may contain one or more
domains, and the DNS server holding those host records is the authoritative name server
for those domains.
The DNS server contains records that map hostnames to IP addresses, which are
referred to as resource records. When a user’s computer needs to resolve a hostname to an
IP address, it looks to its networking settings to find its DNS server. The computer then
sends a request, containing the hostname, to the DNS server for resolution. The DNS
server looks at its resource records and finds the record with this particular hostname,
retrieves the address, and replies to the computer with the corresponding IP address.
It is recommended that a primary and a secondary DNS server cover each zone. The
primary DNS server contains the actual resource records for a zone, and the secondary
DNS server contains copies of those records. Users can use the secondary DNS server
to resolve names, which takes a load off of the primary server. If the primary server goes
down for any reason or is taken offline, users can still use the secondary server for name
resolution. Having both a primary DNS server and a secondary DNS server provides

fault
tolerance and redundancy to ensure users can continue to work if something happens to
one of these servers.
The primary and secondary DNS servers synchronize their information through a zone transfer. After changes take place to the primary DNS server, those changes must
be replicated to the secondary DNS server. It is important to configure the DNS server
to allow zone transfers to take place only between the specific servers. For years now,
attackers have been carrying out unauthorized zone transfers to gather very useful
network information from victims' DNS servers.
An unauthorized zone transfer provides the attacker with information on almost every
system within the network, including the hostname and IP address of each system, system
alias names, public key infrastructure (PKI) server, DHCP server, DNS servers, and so
on. This allows an attacker to carry out very targeted attacks on specific systems. If you
were the attacker and you had a new exploit for DHCP software, now you would know
the IP address of the company's DHCP server and could send your attack parameters
directly to that system. Also, since the zone transfer can provide data on all of the systems
in the network, the attacker can map out the network. He knows what subnets are being used, which systems are in each subnet, and where the critical network systems
reside. This is analogous to you allowing a burglar into your house with the freedom of
identifying where you keep your jewels, expensive stereo equipment, piggy bank, and
keys to your car, which will allow him to more easily steal these items when you are on

vacation. Unauthorized zone transfers can take place if the DNS servers are not properly
configured to restrict this type of activity.

Internet DNS and Domains

Networks on the Internet are connected in a hierarchical structure, as are the different
DNS servers, as shown in Figure 11-29. While performing routing tasks, if a router does
not know the necessary path to the requested destination, that router passes the packet
up to a router above it. The router above it knows about all the routers below it. This

router has a broader view of the routing that takes place on the Internet and has a better
chance of getting the packet to the correct destination. This holds true with DNS servers
also. If one DNS server does not know which DNS server holds the necessary resource
record to resolve a hostname, it can pass the request up to a DNS server above it.

Root-level
domain

Top-level domain

COM

EDU

ORG

Second-level domain

mheducation

usma

ieee

Figure 11-29 The DNS naming hierarchy is similar to the routing hierarchy on the Internet.

The naming scheme of the Internet resembles an inverted tree with the root servers at
the top. Lower branches of this tree are divided into top-level domains, with second-level
domains under each. The most common top-level domains are as follows:

- COM Commercial
- EDU Education
- MIL U.S. military organization
- INT International treaty organization
- GOV Government
- ORG Organizational
- NET Networks

DNS Resolution Components
Your computer has a DNS resolver, which is responsible for sending out requests to DNS servers for host IP address information. If your system did not have this resolver, when you type in www.google.com in your browser, you would not get to this website because your system does not actually know what www.google.com means. When you type in this URL, your system’s resolver has the IP address of

a DNS server it is supposed to send its hostname-to-IP address request to. Your resolver can send out a nonrecursive query or a recursive query to the DNS server.
A nonrecursive query means that the request just goes to that specified DNS server
and either the answer is returned to the resolver or an error is returned. A recursive
query means that the request can be passed on from one DNS server to another one until the DNS server with the correct information is identified.
(Continued)

PART IV

So how do all of these DNS servers play together in the Internet playground? When
a user types in a URL to access a website that sells computer books, for example, his
computer asks its local DNS server if it can resolve this hostname to an IP address. If
the primary DNS server cannot resolve the hostname, it must query a higher-level DNS
server, ultimately ending at an authoritative DNS server for the specified domain. Because
this website is most likely not on the corporate network, the local LAN DNS server will
not usually know the necessary IP address of that website. The DNS server does not
reject the user’s request, but rather passes it on to another DNS server on the Internet.
The request for this hostname resolution continues through different DNS servers until
it reaches one that knows the IP address. The requested host’s IP information is reported
back to the user’s computer. The user’s computer then attempts to access the website
using the IP address, and soon the user is buying computer books, happy as a clam.
DNS server and hostname resolution is extremely important in corporate networking
and Internet use. Without it, users would have to remember and type in the IP address
for each website and individual system instead of the name. That would be a mess.

In the following illustration, you can follow the succession of requests that commonly takes place. Your system’s resolver first checks to see if it already has the
necessary hostname-to-IP address mapping cached or if it is in a local HOSTS file.
If the necessary information is not found, the resolver sends the request to the local

DNS server. If the local DNS server does not have the information, it sends the request to a different DNS server.
DNS client (resolver)

Server-to-server
query (recursion)

Client-to-server query
Zones

Other DNS servers
Q3

Q1

DNS
resolver
cache

A1

A3

DNS
server
Q2

Q5
A5

A2
Q4

Web browser
A4

URL: www.google.com

HOSTS file

DNS server
cache

The HOSTS file resides on the local computer and can contain static hostnameto-IP address mapping information. If you do not want your system to query a
DNS server, you can add the necessary data in the HOSTS file, and your system will check its contents before reaching out to a DNS server. HOSTS files are like
two-edged swords: on the one hand they offer a degree of security by ensuring that certain hosts resolve to specific IP addresses, but on the other hand they are
attractive targets for attackers who want to redirect your traffic to specific hosts. The

key, as always, is to carefully analyze and mitigate the risks.

DNS Threats

As stated earlier, not every DNS server knows the IP address of every hostname it is asked to resolve. When a request for a hostname-to-IP address mapping arrives at a DNS server (server A), the server reviews its resource records to see if it has the necessary information to fulfill this request. If the server does not have a resource record for this hostname, it forwards the request to another DNS server (server B), which in turn reviews its resource records and, if it has the mapping information, sends the information back to server A.

Server A caches this hostname-to-IP address mapping in its memory (in case another client requests it) and sends the information on to the requesting client.

With the preceding information in mind, consider a sample scenario. Andy the attacker wants to make sure that any time one of his competitor's customers tries to visit the competitor's website, the customer is instead pointed to Andy's website. Therefore, Andy installs a tool that listens for requests that leave DNS server A asking other DNS servers if they know how to map the competitor's hostname to its IP address. Once Andy sees that server A sends out a request to server B to resolve the competitor's hostname, Andy quickly sends a message to server A indicating that the competitor's hostname resolves to Andy's website's IP address. Server A's software accepts the first response it gets, so server A caches this incorrect mapping information and sends it on to the requesting client. Now when the client tries to reach Andy's competitor's website, she is instead pointed to Andy's website. This will happen subsequently to any user who uses server A to resolve the competitor's hostname to an IP address because this information is cached on server A.

Previous vulnerabilities that have allowed this type of activity to take place have been addressed, but this type of attack is still taking place because when server A receives a response to its request, it does not authenticate the sender.

Mitigating DNS threats consists of numerous measures, the most important of

which
is the use of stronger authentication mechanisms such as the DNSSEC (DNS security,
which is part of many current implementations of DNS server software). DNSSEC implements PKI and digital signatures, which allows DNS servers to validate the origin
of a message to ensure that it is not spoofed and potentially malicious. If DNSSEC were
enabled on server A, then server A would, upon receiving a response, validate the digital
signature on the message before accepting the information to make sure that the response
is from an authorized DNS server. So even if an attacker sends a message to a DNS
server, the DNS server would discard it because the message would not contain a valid
digital signature. DNSSEC allows DNS servers to send and receive authorized messages
between themselves and thwarts the attacker's goal of poisoning a DNS cache table.
This sounds simple enough, but for DNSSEC to be rolled out properly, all of the DNS
servers on the Internet would have to participate in a PKI to be able to validate digital
signatures. The implementation of Internet-wide PKIs simultaneously and seamlessly has
proved to be difficult.
Despite the fact that DNSSEC requires more resources than the traditional DNS, more and more organizations globally are opting to use DNSSEC. As of this writing,
91 percent of the top-level domains implement DNSSEC. However, across the entire Internet, barely 3 percent of domains have implemented it. So we are getting there,
slowly but surely.

DNS Splitting
Organizations should implement split DNS, which means a DNS server in the DMZ handles external hostname-to-IP address resolution requests, while an internal DNS server handles only internal requests. This helps ensure that the internal
DNS server has layers of protection and is not exposed by being "Internet facing."
The internal DNS server should only contain resource records for the internal computer systems, and the external DNS server should only contain resource records
for the systems the organization wants the outside world to be able to connect to.
If the external DNS server is compromised and it has the resource records for all of
the internal systems, now the attacker has a lot of "inside knowledge" and can carry

out targeted attacks. External DNS servers should only contain information on the systems within the DMZ that the organization wants others on the Internet to be able to communicate with (web servers, external mail server, etc.).

Now let's discuss another (indirectly related) predicament in securing DNS traffic—manipulation of the HOSTS file, a technique frequently used by malware. The HOSTS file is used by the operating system to map hostnames to IP addresses as described before.

The HOSTS file is a plaintext file located in the %systemroot%\system32\drivers\etc folder in Windows, in /etc/hosts in Unix/Linux systems, and in /private/etc/hosts in macOS. The HOSTS file simply consists of a list of IP addresses with their corresponding hostnames.

Depending on its configuration, the computer refers to the HOSTS file before issuing a DNS request to a DNS server. Most operating systems give preference to details of IP addresses returned by the HOSTS file rather than querying the DNS server because the HOSTS file is generally under the direct control of the local system administrator.

As covered previously, in the early days of the Internet and prior to the adoption of DNS, HOSTS files were the primary source of determining a host's network addresses from its hostname. With the increase in the number of hosts connected to the Internet, maintaining HOSTS files became next to impossible and ultimately led to the creation of DNS.

Due to the important role of HOSTS files, they are frequently targeted by malware to propagate across systems connected on a local network. Once a malicious program takes over the HOSTS file, it can divert traffic from its intended destination to websites hosting malicious content, for example. A common example of HOSTS file manipulation carried out by malware involves blocking users from visiting antivirus update websites.

This is usually done by mapping target hostnames to the loopback interface IP address 127.0.0.1. The most effective technique for preventing HOSTS file intrusions is to set it as a read-only file and implement a host-based IDS that watches for critical file modification attempts.

Attackers don't always have to go through all this trouble to divert traffic to rogue destinations. They can also use some very simple techniques that are

surprisingly effective
in routing naive users to unintended destinations. The most common approach is known

as URL hiding. Hypertext Markup Language (HTML) documents and e-mail messages allow users to attach or embed hyperlinks in any given text, such as the “Click Here”
links you commonly see in e-mail messages or web pages. Attackers misuse hyperlinks to
deceive unsuspecting users into clicking rogue links.
Let’s say a malicious attacker creates an unsuspicious text, www.good.site, but embeds
the link to an abusive website, www.bad.site. People are likely to click the www.good
.site link without knowing that they are actually being taken to the bad site. In addition,
attackers also use character encoding to obscure web addresses that may arouse user
suspicion.

Network Address Translation

- 10.0.0.0–10.255.255.255 Class A networks
- 172.16.0.0–172.31.255.255 Class B networks
- 192.168.0.0–192.168.255.255 Class C networks

NAT is a gateway that lies between a network and the Internet (or another network)
that performs transparent routing and address translation. Because IP addresses were
depleting fast, IPv6 was developed in 1999, and was intended to be the long-term fix to
the address shortage problem. NAT was developed as the short-term fix to enable more
organizations to participate on the Internet. However, to date, IPv6 is slow in acceptance
and implementation, while NAT has caught on like wildfire. Many firewall vendors have
implemented NAT into their products, and it has been found that NAT actually provides
a great security benefit. When attackers want to hack a network, they first do what they
can to learn all about the network and its topology, services, and addresses. Attackers
cannot easily find out an organization’s address scheme and its topology when NAT is
in place, because NAT acts like a large nightclub bouncer by standing in front of the
network and hiding the true IP scheme.

When computers need to communicate with each other, they must use the same type of addressing scheme so everyone understands how to find and talk to one another. The Internet uses the IP address scheme as discussed earlier in the chapter, and any computer or network that wants to communicate with other users on the network must conform to this scheme; otherwise, that computer will sit in a virtual room with only itself to talk to. However, IP addresses have become scarce (until the full adoption of IPv6) and expensive. So some smart people came up with network address translation (NAT), which enables a network that does not follow the Internet's addressing scheme to communicate over the Internet.

Private IP addresses have been reserved for internal LAN address use, as outlined in RFC 1918. These addresses can be used within the boundaries of an organization, but they cannot be used on the Internet because they will not be properly routed. NAT enables an organization to use these private addresses and still be able to communicate transparently with computers on the Internet.

The following lists current private IP address ranges:

NAT hides internal addresses by centralizing them on one device, and any frames that leave that network have only the source address of that device, not of the actual internal computer that sends the message. So when a message comes from an internal computer with the address of 10.10.10.2, for example, the message is stopped at the device running NAT software, which happens to have the IP address of 1.2.3.4. NAT changes the header of the packet from the internal address, 10.10.10.2, to the IP address of the NAT device, 1.2.3.4. When a computer on the Internet replies to this message, it replies to the address 1.2.3.4. The NAT device changes the header on this reply message to 10.10.10.2 and puts it on the wire for the internal user to receive.

Three basic types of NAT implementations can be used:

• Static mapping The NAT software has a pool of public IP addresses configured. Each private address is statically mapped to a specific public address. So computer A always receives the public address x, computer B always receives the public address y, and so on. This is generally used for servers that need to

keep the same public address at all times.
• Dynamic mapping The NAT software has a pool of IP addresses, but instead of statically mapping a public address to a specific private address, it works on a first-come, first-served basis. So if Bob needs to communicate over the Internet, his system makes a request to the NAT server. The NAT server takes the first IP address on the list and maps it to Bob's private address. The balancing act is to estimate how many computers will most likely need to communicate outside the internal network at one time. This estimate is the number of public addresses the organization purchases, instead of purchasing one public address for each computer.
• Port address translation (PAT) The organization owns and uses only one public IP address for all systems that need to communicate outside the internal network. How in the world could all computers use the exact same IP address? Good question. Here's an example: The NAT device has an IP address of 127.50.41.3. When computer A needs to communicate with a system on the Internet, the NAT device documents this computer's private address and source port number (10.10.44.3; port 43,887). The NAT device changes the IP address in the computer's packet header to 127.50.41.3, with the source port 40,000. When computer B also needs to communicate with a system on the Internet, the NAT device documents the private address and source port number (10.10.44.15; port 23,398) and changes the header information to 127.50.41.3 with source port 40,001. So when a system responds to computer A, the packet first goes to the NAT device, which looks up the port number 40,000 and sees that it maps to computer A's real information. So the NAT device changes the header information to address 10.10.44.3 and port 43,887 and sends it to computer A for processing. An organization can save a lot more money by using PAT because it needs to buy only a few public IP addresses, which are used by all systems in the network.

Most NAT implementations are stateful, meaning they keep track of a communication between the internal host and an external host until that session is ended. The NAT device needs to remember the internal IP address and port to send the reply messages back. This stateful characteristic is similar to stateful-inspection firewalls, but NAT does not perform scans on the incoming packets to look for malicious characteristics. Instead, NAT is a service usually performed on routers or gateway devices within an organization's screened subnet.
Although NAT was developed to provide a quick fix for the depleting IP address problem, it has actually put the problem off for quite some time. The more organizations that implement private address schemes, the less likely IP addresses will become scarce.
This has been helpful to NAT and the vendors that implement this technology, but

it has
put the acceptance and implementation of IPv6 much farther down the road.

Routing Protocols

NOTE As an analogy, just as the world is made up of different countries, the Internet is made up of different ASs. Each AS has delineated boundaries just as countries do. Countries can have their own languages (e.g., Spanish, Arabic, Russian). Similarly, ASs have their own internal routing protocols. Countries that speak different languages need to have a way of communicating with each other, which could happen through interpreters. ASs need to have a standardized method of communicating and working together, which is where external routing protocols come into play.

The architecture of the Internet that supports these various ASs is created so that
no entity that needs to connect to a specific AS has to know or understand the interior
routing protocols that are being used. Instead, for ASs to communicate, they just have to
be using the same exterior routing protocols (see Figure 11-30). As an analogy, suppose
you want to deliver a package to a friend who lives in another state. You give the package
to your brother, who is going to take a train to the edge of the state and hand it to
the postal system at that junction. Thus, you know how your brother will arrive at the
edge of the state—by train. You do not know how the postal system will then deliver your
package to your friend’s house (truck, car, bus), but that is not your concern. It will get to
its destination without your participation. Similarly, when one network communicates
with another network, the first network puts the data packet (package) on an exterior
protocol (train), and when the data packet gets to the border router (edge of the state), the
data is transferred to whatever interior protocol is being used on the receiving network.

PART IV

Individual networks on the Internet are referred to as autonomous systems (ASs). These
ASs are independently controlled by different service providers and organizations. An
AS is made up of routers, which are administered by a single entity and use a common
Interior Gateway Protocol (IGP) within the boundaries of the AS. The boundaries of
these ASs are delineated by border routers. These routers connect to the border routers
of other ASs and run interior and exterior routing protocols. Internal routers