4

Using the Internet Protocol in Local and Internet Communications

The protocols in the IEEE 802.3 Ethernet standard enable the computers in a local network to exchange messages with each other. In practice, most Ethernet networks also use Internet protocols such as TCP or UDP and IP. These provide defined and well-supported methods for accomplishing common tasks such as flow control and flexible addressing and routing of messages.

Messages that travel on the Internet must use IP. And because TCP and UDP are designed to work along with IP, local communications that use TCP or UDP also use IP, even if they wouldn't otherwise require it.

This chapter begins with a guide to connecting embedded systems to the Internet. Following this is an introduction to the Internet Protocol, including when and how embedded systems can use it in local and Internet communications.

Quick Start: Connecting to the Internet

To communicate over the Internet, a computer must have three things: an IP address that identifies the computer on the Internet, the ability to send and receive IP datagrams, and a connection to a router that can access the Internet.

An Internet Service Provider (ISP) can provide one or more IP addresses and a connection to a router that can communicate over the Internet. Customers use a variety of ways to connect to ISPs. A high-volume user, including the networks at some large businesses, government offices, and schools, may have a dedicated, high-speed connection to an ISP. If your network is located at a facility that has this type of access, your network administrator can tell you if your system can use the connection. Connections that support low to moderate traffic typically connect to the ISP via a modem or other device that interfaces to a phone line or a cable from a cable-TV provider.

Considerations in Obtaining Internet Service

The type of Internet connection to use depends in part on its intended use. A computer that hosts a Web page that other computers can request has different requirements than a computer used only to request Web pages but not serve them.

In many Internet communications, one computer functions as a client, and the other as a server. A client requests resources from a server. A resource may be a Web page, file, or other data. In response to a request, a server sends the client the requested resource or a response such as an error message.

Microsoft's Internet Explorer and other Web browsers are clients. The text that you type or copy into the browser's Address text box (such as `http://www.Lvr.com` or `http://192.168.111.1`) identifies the resource you're requesting and the server you're requesting it from. The computers that host the resources are functioning as servers, which detect, interpret, and respond to requests from computers on the Internet or in a local network.

Many servers are huge systems that store thousands of files, but a server can also be a small embedded system that serves a few basic Web pages or other information on request. As Chapter 3 showed, many Ethernet-capable modules for embedded systems include software that enables the modules to function as Web servers.

If you want users on the Internet to be able to request Web pages, download or upload information, or access other resources on your system, you'll need three things: a computer that functions as a server, an Internet account that permits hosting a server, and network-security settings that enable the server to receive and respond to requests from other computers in the network without putting other local resources at risk.

When selecting a method of connecting, you need to consider the speed in both the upstream (towards the Internet) and downstream (from the Internet) directions. For many inexpensive accounts, the upstream speed is slower than the downstream speed. This arrangement is generally fine for home users, who tend to use Internet connections for activities such as Web surfing, where most of the traffic is downloads. Typical uploading activities for home users, such as sending moderate amounts of e-mail, aren't time-critical, so a slower upload speed is fine.

In contrast, a server sends most of its data upstream. Still, an embedded system that serves very basic Web pages or transfers moderate amounts of data may function fine with a slower connection.

To host a server, it's likely that you'll need a business, or commercial, account with your ISP. In addition to limited speed for upstream communications, accounts offered to home users typically forbid hosting servers because a server is likely to draw more traffic than the ISP can support at

home-user prices. For home accounts, some ISPs block unsolicited requests to port 80, which is the default port where Web servers receive requests.

One option that uses a different approach is worth a mention for applications where an embedded system only needs to provide information periodically to a server on the Internet. Many ISPs and other companies offer Web hosting services that enable you to host Web pages on one of the company's servers. You upload the files, typically via FTP, to the server, and the server responds to requests to view the pages. For some applications, you can program a device to send files to the server as needed and let the server handle the work of serving requests on the Internet. With this arrangement, the device doesn't have to function as a server; it just needs to be able to transfer files as needed to a remote server.

Technologies for Connecting

There are several options for obtaining an Internet connection. A long-popular way for home users to connect to the Internet is via dial-up connections on phone lines. For higher speeds, alternatives are a Digital Subscriber Line (DSL), an Integrated Services Digital Network (ISDN) line, or a cable modem. Satellite connections are also possible. Table 4-1 compares the capabilities of the different methods. Not every connection type is available in all locations.

Depending on the type of access and the equipment that connects to the provider, Internet communications may use Ethernet, serial port (RS-232), or USB. Ethernet is fast and flexible, and an Ethernet network enables multiple computers to share a connection. Hardware support for RS-232 is very inexpensive. Most microcontrollers have an on-chip UART and require only a TTL-to-RS-232 converter. A computer that connects to the Internet via an RS-232 connection to a modem doesn't have to support Ethernet at all. Instead, the computer can use the Point-to-Point Protocol (PPP) to send and receive IP datagrams over the RS-232 connection.

Generally, a USB connection isn't practical for small embedded systems. USB modems must connect to a PC or other USB host, while most

Table 4-1: The speed of an Internet connection depends in part on the method of connecting. Downstream speeds are often faster than upstream.

Access Type	Downstream Speed (kb/s, typical maximum)	Upstream Speed (kb/s, typical maximum)	Transmission Medium
Dial up	56	56	phone line
ADSL	1500	384	phone line
SDSL	2000	2000	phone line
BRI ISDN	128	128	phone line
PRI ISDN	1500 (23 channels)	1500 (23 channels)	phone lines
Cable modem	1500, shared	384, shared	TV cable
Satellite	500	50	wireless

USB-capable embedded systems are USB devices. Also, USB modems typically come with driver software for Windows only.

Dial Up

A dial-up connection is available anywhere there is phone service. A modem provides an interface between a computer that wants to access the Internet and an ordinary phone line (Figure 4-1). To make a connection, the computer instructs the modem to dial a number that connects to a modem at the ISP. The ISP's modem in turn connects to a router with an Internet connection. A PC's modem may be on the motherboard or an expansion card, or the modem may connect to the PC via an RS-232 or USB port. An

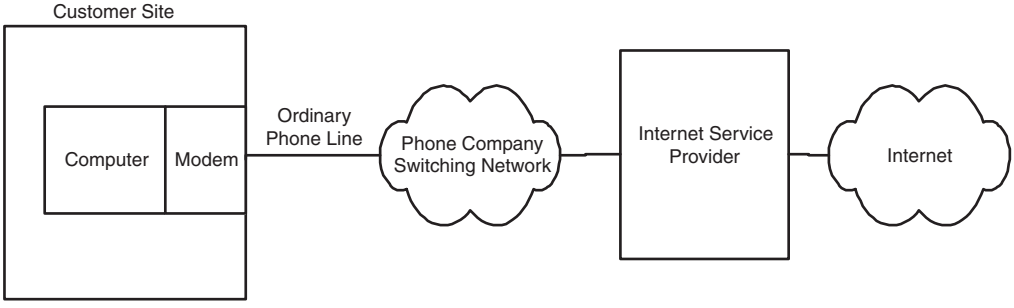

Figure 4-1: In a dial-up connection, the computer uses a modem to connect to an ISP over an ordinary phone line.

embedded system may also contain a modem or connect to an external modem, usually via RS-232.

The computer uses the Point-to-Point Protocol (PPP) to manage the modem connection and to send and receive IP datagrams over the serial link. Rabbit Semiconductor's Dynamic C has an optional module with libraries and example code for PPP communications. For TINI users, the `com.dalsemi.tininet.ppp` package supports PPP. *RFC 1661: The Point-to-Point Protocol (PPP)* defines the protocol.

Limitations of dial-up connections are a maximum speed of 56 kilobits per second and the need to provide a phone line for the connection. Advantages are low cost and availability anywhere there is phone service.

In general, a dial-up connection isn't the best option for a server because of limited speed. But dial up can be useful for some computers that occasionally communicate on the Internet. For example, a series of data loggers might periodically dial in to send readings to a central computer that is on the Internet and programmed to accept the communications from the data loggers. A system with a dial-up connection may also communicate by sending and receiving e-mail. Multiple systems can share a dial-up account if each calls in turn.

A computer that connects to an ISP via dial-up may also use Ethernet to connect to a local network.

DSL

DSL uses a conventional phone line with equipment at each end to enable the line to carry voice and Internet communications at the same time. Although the exact setup can vary with the provider, Figure 4-2 shows a typical configuration, where the customer's site has a DSL modem and a splitter. In the upstream direction, a splitter combines phone and Internet traffic on a single pair of wires. In the downstream direction, the splitter routes the phone and Internet traffic onto the appropriate wires inside the customer's premises. Another name for the splitter is network interface device (NID).

The line carrying Internet traffic in the customer's premises connects to a DSL modem, which has a USB or Ethernet connection to the customer's

computer. At the phone company's central office, phone traffic is routed to and from the company's switching equipment, and Internet traffic is routed to and from a DSL Access Module (DSLAM). The DSLAM interfaces to the company's DSL equipment, which connects to the Internet.

DSL connections often use Point-to-Point Protocol over Ethernet (PPPoE). PPPoE requires logging on with a user name and password but doesn't require dialing a phone number to connect to the ISP. Dynamic C's PPP module supports PPPoE and includes an example application.

DSL has several variants with differing speed and distance limits. Not all providers offer all variants. Two popular options are asymmetric DSL (ADSL) and single-line, or symmetric DSL (SDSL). With ADSL, traffic in each direction has a different speed, with the downstream speed typically much faster than the upstream speed. Embedded systems that host busy Web or FTP servers will probably find SDSL, with equal speeds in both directions, more suitable.

The speed of a connection varies with the DSL variant, the distance from the phone company's central office, and the quality of the phone line. Theoretically, ADSL can support speeds as high as 6.1 Mb/s downstream and 1.5 Mb/s upstream. In practice, speeds are likely to be equal to or less than 1.5 Mb/s downstream and 384 kb/s upstream. The theoretical maximum for SDSL is 2 Mb/s in each direction. The maximum distance between the cus-

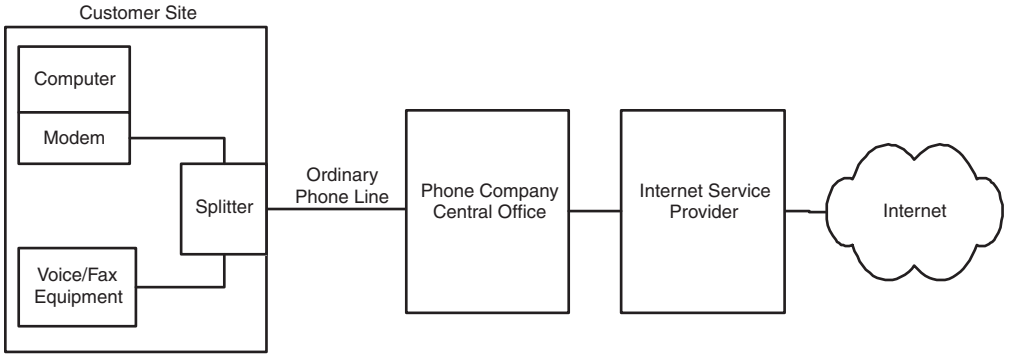

Figure 4-2: In a DSL connection, voice and fax lines can share the same phone line as data.

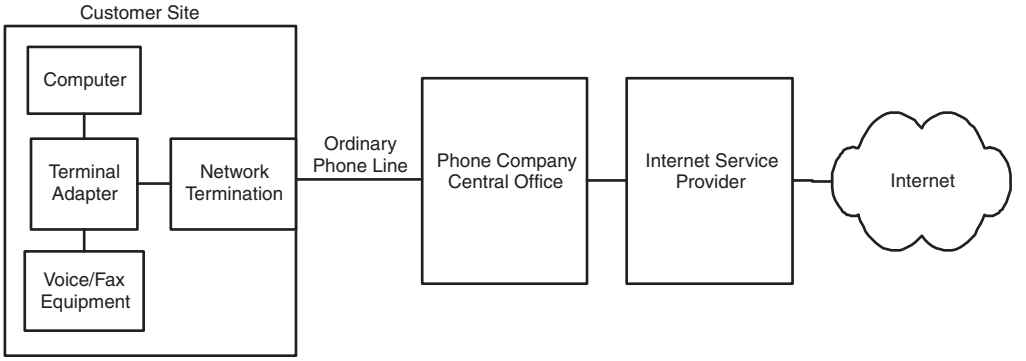

Figure 4-3: With BRI ISDN, one channel can carry voice or fax signals while the other carries data, or for a higher-speed connection, both channels can carry data.

tomer and the central office is around 18,000 feet for ADSL and 18,000 to 22,000 feet for SDSL.

ISDN

Like DSL, ISDN connections can use conventional phone lines. ISDN has two main variants. With Basic Rate Interface (BRI) ISDN, the phone line carries two 64-kb/s "B" channels that can be combined for a single 128-kb/s connection. A separate lower-speed "D" channel carries signaling information. As Figure 4-3 shows, the computer that wants to communicate over the Internet connects via Ethernet, RS-232, or USB to an ISDN terminal adapter, which in turn connects to a network termination. The customer's phone line connects the network termination to a switch at the phone company's central office, which routes the traffic to and from the ISP. It's also possible to use one ISDN channel for voice traffic and the other for a 64-kilobit Internet connection.

If BRI ISDN isn't enough, Primary Rate Interface (PRI) ISDN has 23 channels and speeds of up to 1.544 Mb/s. A BRI connection requires a T1 line, which is a special 4-wire phone line that carries digital data from the central office to the customer.

Cable Modem

A cable modem doesn't use phone lines, but instead uses a connection to a cable-TV provider that offers Internet access. The same cable can carry TV broadcasts and Internet traffic. As Figure 4-4 shows, the computer that wants to communicate over the Internet connects via Ethernet or USB to a cable modem. The cable modem in turn connects to a filter and splitter, then connects via coaxial cable to a neighborhood concentrator, which has a high-speed connection to the cable company's facility.

The cable's bandwidth is divided into channels. Each TV channel uses a 6-Mhz portion of the bandwidth. Internet traffic typically uses bandwidth above the TV channels for downstream traffic and bandwidth below the TV channels for upstream traffic.

With a cable modem, you share bandwidth with other customers in the neighborhood. So the performance of a cable-modem connection depends in part on the network speed provided by the account and in part on how much other traffic there is at the same time. Most cable-modem connections are asymmetrical, with higher downstream speeds. Typical network speeds for cable modems are from 256 kb/s to 1.5 Mb/s downstream and up to 384 kb/s upstream. Most providers encrypt the Internet traffic so customers who share a connection can't view each others' data.

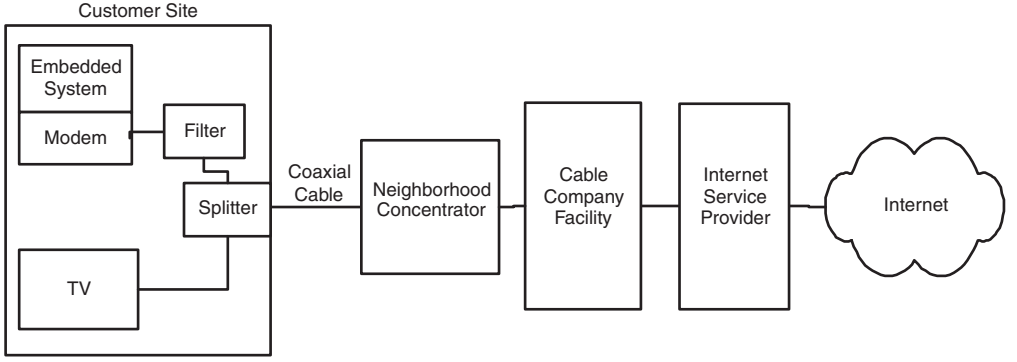

Figure 4-4: A cable-modem connection uses the same cable that carries TV programming.

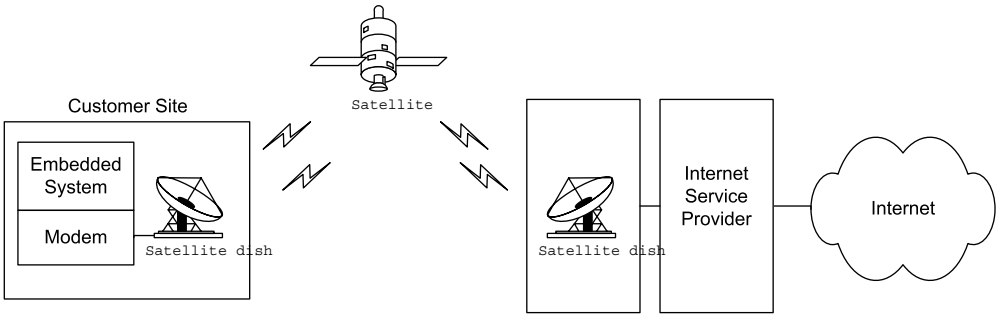

Figure 4-5: A satellite connection makes it possible to communicate from remote locations.

Because cable-TV providers market to residential customers, cable Internet may be unavailable at a business location. Because of the expense of running cable, cable Internet may be unavailable in remote locations.

Satellite

Another option for obtaining Internet access, especially for remote areas, is a satellite link (Figure 4-5). Early offerings of Internet access via satellite were downstream only, requiring a phone-line connection for upstream data. Newer systems offer 2-way communications via satellite. Download speeds range between 150 to 500 kb/s, with upstream speeds of around 50 kb/s. The low-speed upstream communications make satellite links less than ideal for hosting a server. The satellite dish requires a view of the southern sky. The satellite modem may connect via Ethernet or USB to a customer's computer.

Static and Dynamic IP Addresses

Every computer that communicates over the Internet must have an IP address, which the computer typically receives from its ISP. The IP address may be static or dynamic. A static IP address stays the same until someone explicitly changes it, while a dynamic IP address can change on every boot up or network connect (though the address typically changes only occasionally).

An embedded system may store a static IP address in non-volatile memory, either within an application or in memory where program code can retrieve the address when needed. Or the system may receive a static or dynamic IP address from a DHCP server on boot-up or network connect.

For hosting a domain, a static IP address is preferable because the name servers don't have to be updated unless the domain changes ISPs. If the computer hosting the domain has a dynamic IP address, the local name servers must be updated when the address changes, as described later in this chapter.

Connecting Multiple Computers to the Internet

A computer that connects to the Internet must have an IP address that is different from the addresses of all of the other computers on the Internet. When you contract with an ISP, you obtain the right for your computer to use one or more of the ISP's assigned IP addresses.

If you have a local network with multiple computers that need Internet access, it's often easier, more secure, and less expensive to have all of the computers share a single public IP address for Internet communications. Some ISPs charge for each connected computer whether or not they share an IP address, however.

Two ways to enable multiple computers to share a public IP address are with a router that supports the Network Address Translation (NAT) protocol and with a Windows PC configured as an Internet Connection Sharing host.

A router that supports the NAT protocol enables multiple computers to share a public IP address. The router connects to the ISP and to the computers in the local network. The router has two IP addresses: a public address for Internet communications and a local address for communicating with the local network. The router uses the NAT protocol to translate between the public and local addresses as needed.

To send a message on the Internet using a router with NAT support, a computer in the local network sends the message to the router's local address. The router creates a new IP datagram, placing the message in the datagram's

data area and the router's public IP address in the datagram's Source Address field. The router than forwards the datagram to a router at the ISP, which sends the datagram onto the Internet. On receiving a datagram from the ISP's router, the local router uses information in the IP header to determine where to forward the message. The router then creates a new datagram with the appropriate local IP address in the datagram's Destination Address field and forwards the datagram to its destination.

A router with NAT support also helps to keep a local network secure, as described in Chapter 10.

If your local network includes a PC running Windows XP, there is another option. You can enable multiple computers to share a public IP address by configuring the PC as an Internet Connection Sharing host. The PC requires two network interfaces, one to the local network and one to the modem or other connection to the ISP. In Windows XP's Network Setup Wizard, select *This computer connects directly to the Internet. The other computers on my network connect to the Internet through this computer.* All Internet communications for the local network then go through the interfaces on this computer. Windows Help has more information on using Internet Connection Sharing.

Communicating through a Firewall

Any PC or other large computer with Internet access should have a firewall. All communications from outside the local network should pass through the firewall to reach a computer in the local network. The firewall protects the local network by controlling what local resources external computers can access. A firewall may be software only or a combination of hardware and software.

Without a firewall, a computer from outside the network might be able to retrieve private files, install a program that deletes files, or use another computer to launch attacks on other computers. A firewall can also defend against denial-of-service attacks, where a computer attempts to overwhelm a server by bombarding it with requests using forged, invalid source addresses.

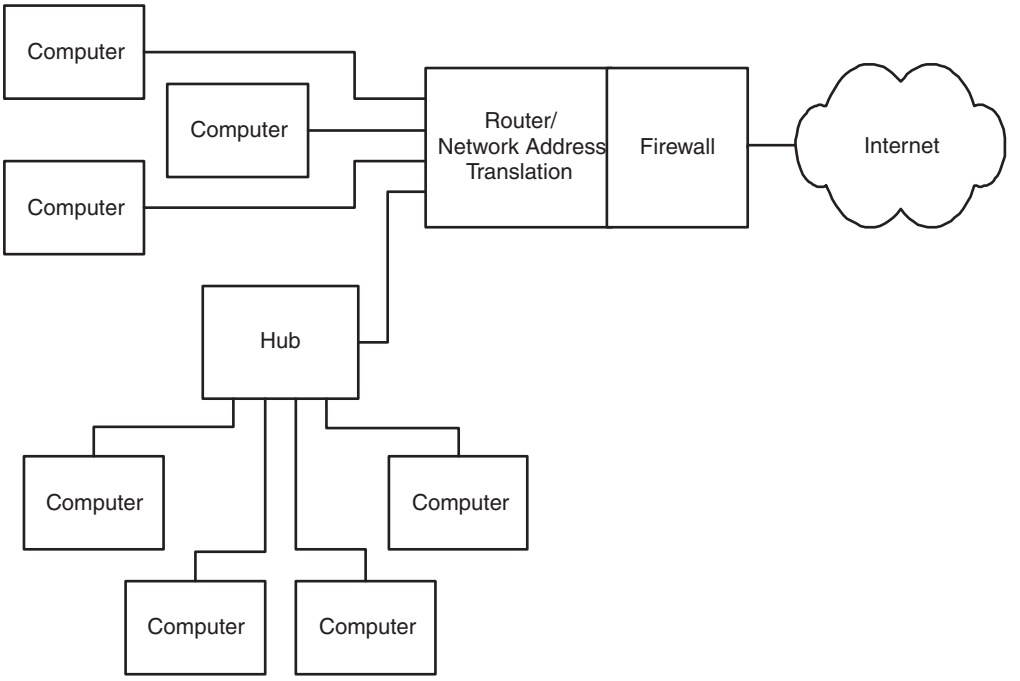

Figure 4-6: A firewall that supports the network address translation (NAT) protocol enables all of the computers in a local network to share a single public IP address.

In a local network, each computer may have its own firewall, or a single firewall may protect all of the computers in the network. The firewall may be software running on a PC or another general-purpose computer, or it may be a device designed specifically to function as a firewall. For networks that use a single firewall, the firewall is the only computer in the local network with a direct Internet connection. As Figure 4-6 shows, all of the other computers send and receive Internet communications by communicating with the computer that contains the firewall.

Some operating systems have firewall software built in. For example, Windows XP has an Internet Connection Firewall that you can configure for specific needs.

A hardware firewall for a small local network may provide additional capabilities, including functioning as a router with address translation and functioning as a DHCP server.

Even when an embedded system doesn't need a firewall to protect itself, many embedded systems are behind a firewall because they're in local networks that have firewall protection.

If your embedded system is behind a firewall, you may need to configure the firewall to enable your system to communicate. In a common setup, a firewall allows the local computers to request resources from computers on the Internet, but blocks all unsolicited incoming requests from the Internet. For example, the firewall typically enables local computers to request Web pages from computers on the Internet. The firewall stores information about each request, and when the computer returns an IP datagram containing the requested page, the firewall examines the header, determines that the datagram is in response to a previous request, and passes the datagram to the requesting computer. If the firewall doesn't recognize a datagram as a response to a previous request, the datagram doesn't pass through the firewall.

A computer that functions as a server available to all computers on the Internet must be able to receive unsolicited requests because the computer has no way of knowing where requests will come from. So you'll need to configure the firewall to allow the server to receive unsolicited communications on at least one port.

The details of how to configure a firewall vary with the product. Many stand-alone firewalls have a password-protected Web interface. Figure 4-7 shows an example configuration setup. Typically, to enable a specific computer to serve Web pages, you can configure the firewall to forward all open, or unsolicited, communications for port 80, which is the port used for HTTP requests, to the computer that serves the pages.

Chapter 10 has more about firewalls and security for networked embedded systems.

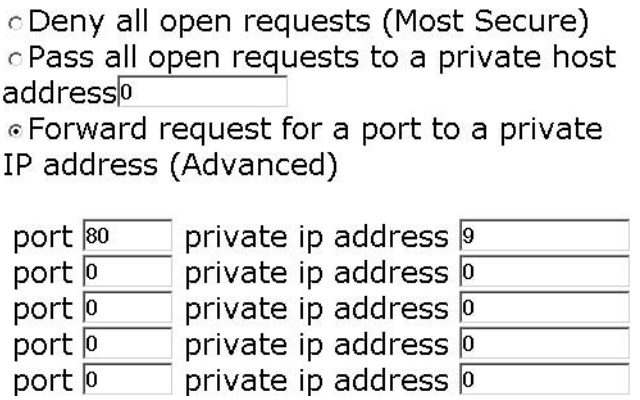

Figure 4-7: A firewall typically provides the option to forward unsolicited requests to a specific host or port on the host. In this example, all requests to port 80 are forwarded to host 9 in the subnet.

Obtaining and Using a Domain Name

After you obtain Internet access, connect your embedded system to the Internet, and configure your firewall to enable the embedded system to communicate, the system is ready to send and receive messages on the Internet.

Applications running on other computers on the Internet can access the embedded system by specifying its public IP address. For example, to view a server's home page, in the Address text box of a Web browser, you enter *http://* followed by the server's IP address.

Each IP address is 32 bits, typically expressed as four bytes in a format known as dotted quad, or dotted decimal, consisting of four decimal numbers separated by periods, or dots, as in 216.92.61.61.

An alternate, more human-friendly way to identify a computer on the Internet is with a domain name. Instead of remembering four numbers, users can provide a name such as *rabbitsemiconductor.com* or *dalsemi.com*. Another advantage of a domain name is that it can remain constant. The IP address of a particular Web page or other resource may change, either because the

owner of the domain has changed ISPs or because the ISP uses dynamic IP addresses that change from time to time.

Just about every major Web site available to the general public on the Internet has a domain name. The tiniest embedded system can also have a domain name, though not every system needs one. A system that functions as a client has no need for an easily remembered name because the client initiates all communications, and each request received from a client includes the IP address to respond to. A computer that only responds to communications from selected computers that know the computer's IP address doesn't need a domain name either. But a domain name can be useful and convenient for an embedded system that functions as a server that's available to any computer on the Internet.

To obtain the right to use a domain name, you need to register the name and provide two name servers that will respond to requests for the domain's IP address, as described later in this chapter.

Understanding Domain Names

A domain name consists of a name that is unique within its root domain, followed by a dot and the name of the root domain. Some examples are:

rabbitsemiconductor.com
dalsemi.com
rfc-editor.org

The original defined root domains were *.com, .edu, .gov, .mil, .net,* and *.org*. In recent years, more have been added.

A domain name may also contain a country-code top-level domain after the root:

number-10.gov.uk

And one or more names to the left of the main domain may identify subset(s) of a domain:

minordivision.majordivision.example.com

The order of the names of the subsets indicates their hierarchy. In the example above, *majordivision* is a subset of *example.com*, and *minordivision* is a subset of *majordivision*.

The letters *www* preceding a domain name specify that the request should be routed to the domain's Web server:

www.Lvr.com

Many domains are configured so that including *www* is optional. On receiving an HTTP request that doesn't include the *www*, the domain's software passes the request to the Web server by default.

The major documents describing the Internet's Domain Name System (DNS) are *RFC1034: Domain names - concepts and facilities* and *RFC1035: Domain names - implementation and specification*. Both are incorporated in standard document *STD0013*. All are available from *www.rfc-editor.org*.

How a URL Specifies a Resource

When requesting a file or other resource from a computer on the Internet, a computer provides a uniform resource locator (URL) that helps in identifying the location of the resource and tells the server how to respond to the request. A URL specifies the protocol to use in reading the request, the name or IP address of the server that hosts the requested resource, the path to the file on the server, and the name of the requested resource (or no name to request a default file).

The document that defines URLs is *RFC 1738: Uniform Resource Locators (URL)*. At minimum, a URL specifies a *scheme* that identifies a protocol such as HTTP, followed by scheme-specific information such as a host name that identifies the location of a requested file. A host name is either an IP address in dotted-quad format or a domain name. Here is an example of a URL that requests a page from a Web server:

http://www.example.com:80/data/testdata.htm

http:// contains the scheme that tells the server to use the hypertext transfer protocol (HTTP) in responding to the request. Other schemes include *ftp* for FTP transfers and *mailto* for links to e-mail messages. Many browsers

add *http://* if you omit the scheme when specifying a URL in the browser's Address text box.

example.com specifies the domain, and *www* specifies the Web server at the domain.

:80 specifies the port the client sends the request to. If the URL doesn't include a port number, the client uses the protocol's default port. RFC 1738 specifies default port numbers for standard protocols. The default for HTTP is port 80.

/data/ names a folder within the server's root folder. A small embedded system may store all of its files in the server's root folder. Forward slashes separate folder and file names even if the server's file system uses different separators.

The name of the requested file is *testdata.htm*. When a URL doesn't specify a filename, most Web servers are configured to serve a default home page, often titled *index.html*.

In many cases, you don't need to type the full URL in the browser's window. If you leave off the *http://*, most browsers insert it for you. Every domain should have a default page to serve if no page is specified. And many servers are configured to serve a Web page even if the URL doesn't contain *www.* So typing just the domain name, such as *example.com*, often causes the Web server at the specified domain to return the same default home page that would be returned by requesting *http://www.example.com:80/index.html*.

Registering a Domain Name

If you want to be able to access your embedded system by specifying a domain name, you must register the name with an appropriate authority. Registering in turn requires providing two name servers that respond to requests for the domain's IP address.

For all domains except those with country-code top-level domains, you can register the name with any of a number of domain name registrars accredited by the Internet Corporation for Assigned Names and Numbers (ICANN) at *www.icann.org*. The registrar pays a yearly fee to ICANN for

each registered domain. The registrar in turn typically charges a yearly fee to the person or entity registering the domain. The domains managed by ICANN are available to registrants in any country.

In addition, each country-code top-level domain has a sponsoring organization for registering domains. The Internet Assigned Numbers Authority (IANA) at *ww.iana.org* has information about registering these domains.

Matching a Domain Name to Its IP Address

Name servers enable computers to match a domain name with the IP address required to access the domain's resources. Domain names are convenient for humans who are requesting resources, but each request ultimately must translate into one or more IP datagrams that contain the IP address of the datagram's destination. So a computer requesting a resource by domain name needs a way to learn the IP address that corresponds to the domain.

A system that communicates only with a defined set of hosts could store a lookup table that matches each host name with its IP address. If a host changes its IP address, the lookup table will need updating, however.

More commonly, matching a domain name to its IP address involves communications between one or more domain-name servers and a resolver. A domain-name server is a computer that stores records that match domain names with their IP addresses. The resolver is a program or process that uses the domain-name-system (DNS) protocol to communicate with name servers to find a match between a domain name and its IP address.

Each registered domain name must have two name servers that respond to queries for the domain's IP address. The ISP that provides the domain's IP address typically provides the name servers. Some registrars will provide name servers if you aren't ready to host the domain right away.

Once the name servers are set up and operating, the computers on the Internet need to learn about their existence. The Internet has a series of root name servers that store root zone files containing the IP addresses of the name servers for all registered domains. Each server stores records for one of the root domains such as *.com, .edu*, or *.mil*. To ensure that the information is always available even if a server fails, each root domain has multiple serv-

ers. The root name servers operate under the direction of IANA and are updated regularly. The servers are in varied locations and are owned by different entities.

To learn a domain's IP address, a computer uses the DNS protocol to send a query to a resolver, which may reside in the same computer that originated the query or elsewhere. The resolver first searches its own cache and returns the answer if found. If not, the resolver attempts to find the answer by querying a name server.

A local network may have an assigned local name server that functions as the resolver for queries from the local network. The local name server knows the addresses of the root name servers and maintains a database of information obtained from previous queries. If the local name server doesn't have the answer in its database, it queries a root name server or another server that it thinks may have the information.

On receiving a query, a name server may return the requested IP address or the IP address of another server that is likely to have the information. For example, to learn the IP address for *www.example.com*, a resolver may send a query to a *.com* root domain server that returns the address of the name server for *example.com*. The resolver can then query this name server for the address of *www.example.com*.

To learn the IP addresses of a local network's name servers, in Windows XP, click **Start** > **Run**, type **cmd**, and click **OK**. In the window that appears, type **ipconfig /all**. In the information displayed are the IP addresses of two DNS servers.

Although an embedded system with a domain name must have name servers that other computers can access to learn the domain's IP address, many embedded systems don't need to communicate with name servers themselves. An embedded system functioning as a server just needs to respond to requests that contain a source IP address to respond to. Other systems may communicate only with computers with known IP addresses. Systems that communicate only in a local network don't need to support domain names at all, though local computers may have locally assigned host names that correspond to local IP addresses.

In Depth: Inside the Internet Protocol

The Internet Protocol (IP) helps data find its way to its destination even if the data must travel through other networks, including the many and varied networks that make up the Internet. Although it's called the Internet Protocol, local networks can use IP as well. Many communications in local networks use IP because they use its companion protocols, TCP and UDP.

This section introduces IP, including how computers obtain IP addresses, the format of IP datagrams, how IP and the domain name system help in getting messages to their destinations, and how embedded systems can use IP in communicating in local networks and on the Internet.

What IP Does

Figure 4-8 shows the place of the IP layer in network communications in the networking stack introduced in Chapter 1. In transmitting, the IP layer receives a message to send from a higher-level protocol layer such as TCP or UDP. The IP layer places the message in an IP datagram that consists of an IP header, followed by the message to send. The IP layer then passes the datagram to a lower layer such as an Ethernet driver, which sends the datagram on the network.

On the way to its destination, a datagram may pass through one or more routers. The router examines the destination's IP address and uses the address in deciding where to forward the datagram.

At the destination computer, the Ethernet layer or another network interface passes the IP datagram to the IP layer, which removes the IP header. Information in the header tells the computer what protocol layer, such as TCP or UDP, should receive the datagram's message.

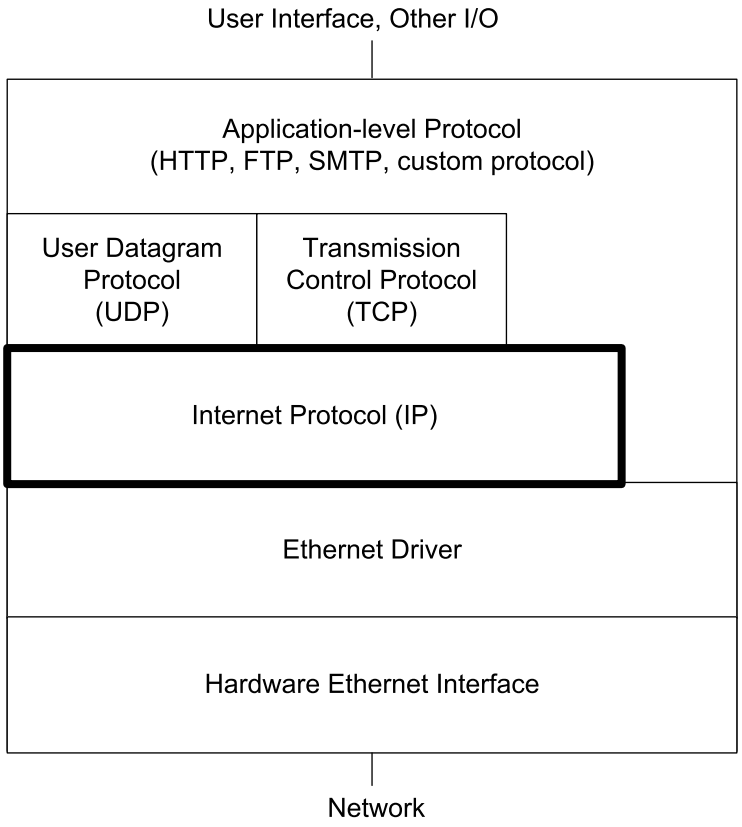

Figure 4-8: In an Ethernet network, the Internet Protocol layer communicates with the Ethernet driver and either a UDP or TCP layer or the application layer.

The Internet Protocol performs two major functions.

- It defines a way to specify source and destination addresses for use with any network interface and across networks that use different interfaces.
- It enables a datagram to pass through networks of varying capabilities by defining a protocol that allows a router to fragment, or divide, a datagram into multiple, smaller datagrams and enables the destination to reassemble the original message from the fragments.

Two things IP doesn't provide are flow control and error checking of the data payload. When needed, a higher-level protocol such as TCP can pro-

vide these. For local communications, Ethernet frames also provide error checking.

Two protocols can help in matching an IP address to a computer, or to be more precise, to a network interface (because a single computer can have multiple network interfaces). The Domain Name System (DNS) protocol described earlier in this chapter enables a computer to learn the IP address that corresponds to a domain name. And in Ethernet networks, the Address Resolution Protocol (ARP) described later in this chapter enables the sender of an IP datagram to match an Ethernet hardware address with an IP address in the local network.

The examples in this book use version 4 of IP (IPv4), which most networks are using at this writing. The expected replacement for IPv4 is IP version 6 (IPv6), which greatly increases the number of available IP addresses and adds other improvements for more efficient and secure transfers. It's likely that IPv6 routers will continue to support IPv4 for some time, so computers that support only IPv4 should have no trouble communicating with any destination.

The standards for IP and related protocols are the responsibility of the Internet Engineering Task Force (IETF) and its working groups (*www.ietf.org*). The IETF is open to anyone who has the necessary skills and abilities and wants to contribute.

The documents that define IP and many other networking protocols are available from the Request for Comments (RFC) Web site (*www.rfc-editor.org*). This book contains a number of references to RFC documents, so perhaps it's appropriate to say a few words about the documents and where they come from. The RFC Editor is a group funded by the Internet Society (ISOC). ISOC in turn is an organizational home for groups who are responsible for various standards relating the Internet's infrastructure.

The RFC Web site is a repository for RFC documents, which include standards-track documents as well as technical and organizational notes relating to networking and the Internet. The standards-track documents contain specifications that have undergone a review process to become approved standards.

Request for Comments may sound like an odd designation for an approved standard, and in fact, approved standards have alternate designations that use the STD prefix. For example, the document that defines IP is *RFC0791: Internet Protocol*. The standards-track document that includes RFC0791 and related documents is STD0005. The IETF's Internet Engineering Steering Group (IESG) is responsible for approving specifications as standards. A protocol doesn't have to be an approved standard before becoming widely implemented, however.

IP Addresses

A computer that uses the Internet Protocol must have an IP address. A network administrator may manually assign an IP address to each computer or the network may have a way of assigning addresses automatically to computers that connect to the network.

An IPv4 address is 32 bits. As explained earlier in this chapter, the conventional way to express an IP address is in dotted-quad format, such as 192.168.111.1.

Assigning Addresses

Each IP datagram includes the IP addresses of the datagram's source and destination. A computer's IP address must be unique within the network or networks that the computer can communicate with. In a local network with no direct connection to other networks, the address only needs to be different from the other addresses in the local network. In theory an isolated local network could use any IP addresses, but the IP standard reserves three blocks of addresses for local use.

For communicating over the Internet, the address must be different from the address of every other computer on the Internet. As described earlier in this chapter, the network administrator typically obtains the right to use one or more IP addresses from the ISP that supplies the network's Internet connection.

An ISP in turn obtains the right to use addresses via a system that involves a variety of organizations that manage the allocating and assigning of

addresses. At the top is the Internet Corporation for Assigned Names and Numbers (ICANN), at *www.icann.org*. ICANN is a non-profit corporation that manages the top-level assigning and allocating of IP addresses. ICANN also manages the Internet's domain name system, the root server system that supports the domain name system, and the assigning of numbers to Internet protocols.

Under ICANN are several regional registries that manage the assigning and allocating of IP addresses in specific geographic areas. For example, the American Registry for Internet Numbers (ARIN) at *www.arin.net* allocates and assigns Internet addresses in North and South America and a few other areas. The regional registries assign and allocate addresses to some large end users and Internet Service Providers (ISPs). The ISPs may in turn assign some of their allocated addresses to end users and may allocate blocks of addresses to other ISPs, who may assign and allocate their addresses, and so on down the line.

The Network Address and Host Address

Each IP address has two parts: a network address, which is the same for all of the interfaces in the network, and a host address, which is unique to the interface within the network. The leftmost bits of the IP address are the network address and the rightmost bits are the host address.

Routers use network addresses to help in determining where to forward received datagrams. The hosts in a local network are generally located near each other physically. So a router can have a table entry that tells the router to forward all datagrams directed to a specific network address to a router that is physically closer to the network. Without network addresses, routers would have to have a separate entry for each IP address, which would quickly become unmanageable.

The number of bits allocated to the network address and host address depends on the network's size. A network with a 24-bit network address and 8-bit host addresses can have up to 254 hosts. (Host and network addresses of all zeros or all 1s have special meanings and can't be assigned to individual

hosts or networks.) A network with an 8-bit network address and 24-bits host addresses can have over 2 million hosts.

To keep from running out of available IP addresses, network addresses should be as long as possible while still enabling every host on the Internet to have a unique host address. If every network had an 8-bit network address, there could be no more than 254 networks on the Internet. But if every network had a 24-bit network address, each network could have no more than 254 hosts.

There are two protocols for assigning network addresses on the Internet. The original protocol, called classful addressing, defines three network classes with network addresses of 8, 16, and 24 bits. By examining the first three bits of the IP address, a router can determine what class of network the host belongs to, and thus how many bits make up the network address.

Many networks with classful addressing are also divided into subnetworks, or subnets. For each subnet, the routers in the local network store an additional 32-bit value called the subnet mask, which enables routers to determine which subnet a datagram is directed to.

A newer, more flexible and efficient alternative to classful addressing is classless addressing, where a network address can be any number of bits. A value called the IP prefix, or network prefix, specifies the number of bits in the network address. Routers that support classless addressing use the IP prefixes in determining where to forward datagrams.

Classful Addressing

Table 4-2 shows the five network classes defined by RFC0791 for classful addressing. The most significant bits of an IP address indicates the class of the network the host belongs to and how many bytes make up the network address. You can identify the class from the decimal value of the first byte or from the binary value of the few most significant bits.

In a Class A network, the first byte is between 1 and 126, and the most significant bit is 0. The network address is 1 byte, leaving three bytes for the host address. There can be up to 126 Class A networks.

Table 4-2: A network's class determines how many hosts the network can contain.

Network Class	Most Significant Bit(s) in Network Address	Range of Most Significant Byte in Network Address	Number of Bytes in Network Address	Maximum Number of Networks	Number of Bytes in Host Address	Maximum Number of Hosts
A	0	1-126	1	126	3	16 million+
B	10	128-191	2	16,384	2	65,534
C	110	192-223	3	2 million+	1	254
D	1110	224-239	reserved for multicasting			
E	1111	240-255	reserved for future use			

In a Class B network, the first byte is between 128 and 191, and the two most significant bits are 10. The network address is 2 bytes, leaving two bytes for the host address. There can be up to 65,534 Class B networks.

In a Class C network, the first byte is between 192 and 223, and the three most significant bits are 110. The network address is 3 bytes, leaving 1 byte for the host address. There can be up to 16,777,214 Class C networks.

In a Class D network, the first byte is between 224 and 239, and the four most significant bits are 1110. Class D networks are reserved for multicasting, described later in this chapter.

In a Class E network, the first byte is between 240 and 255, and the four most significant bits are 1111. Class E is reserved for future use.

Using Subnets

Subnetting is the process of dividing a network into groups called subnetworks, or subnets. The hosts within a subnet are typically physically near each other and may belong to the same department or facility within an organization.

In the same way that routers use network addresses to decide where to route traffic on the Internet, routers can use subnet IDs to decide where to route traffic within a network.

A small, isolated local network doesn't have to concern itself with subnets. A large local network might use subnets for easier routing of messages. A public IP address obtained from an ISP is likely to be in a subnet, so even if your embedded system is in a small network, if the system connects to the Internet, the public IP address is likely to be in a subnet.

Besides helping in routing, subnetting helps to solve the shortage of available network addresses. With only three general-purpose network classes, many organizations requesting network addresses would have to request much larger blocks of addresses than needed. For example, a network of 300 hosts is too large for Class C, but with a Class B address, tens of thousands of addresses would be unused. With subnets, a 300-host network can reserve a portion of a Class B network, leaving the remaining addresses for other subnets.

In a subnet, the host-address portion of an IP address has two parts: a subnet ID and a host ID. The subnet ID is the same for all hosts in the subnet, while each host ID is unique in the subnet. The network-address portion of the IP address is the same for all of the hosts in all of the subnets in the network. A subnet ID can be any combination of bits in the host-address portion of the IP address, but in practice it's almost always the most significant bits.

Figure 4-9 shows an example. Three Ethernet networks are subnets in a Class B network. A hub in each subnet connects to a router that enables the computers in the subnets to communicate with computers in other subnets and on the Internet. In each of the IP addresses, the first two bytes (172.16) are the network address, the third byte is the subnet ID (1, 2, or 3), and the fourth byte is the host ID.

A subnet ID may use any number of the bits in the host address. For example, a Class C network that uses four bits for the subnet mask can have up to 14 subnets (2^4-2) and each subnet can have up to 14 hosts.

As explained earlier, you can determine how many bits of an IP address are the network address by examining the most significant bits in the address.

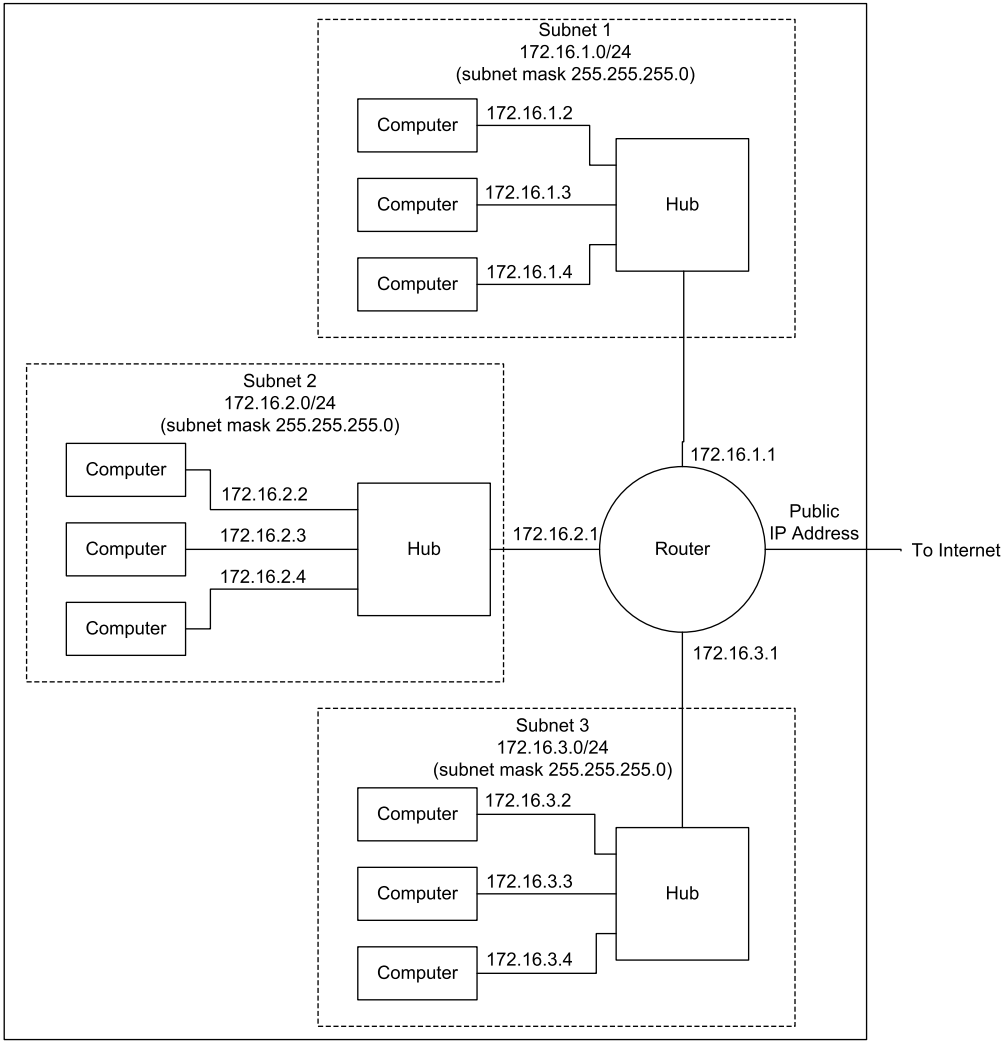

Figure 4-9: A router can enable multiple networks or subnets to communicate with each other and the Internet.

The Subnet Mask

Determining which bits in the host address are the subnet ID requires using a 32-bit value called the *subnet mask*.

In the subnet mask, the bits that correspond to the bits in the network address and the subnet ID are ones, and the bits that correspond to the bits in the host ID are zeros.

For example, in a Class B network, two bytes are the network address and two bytes are the host address. The subnet mask for a Class B network with eight bits of subnet ID is:

255.255.255.0

With eight bits of subnet ID, the network can have up to 254 subnets, and each subnet can have up to 254 hosts.

In a similar way, the subnet mask for a Class C network with four bits of subnet ID is:

255.255.255.240

(Decimal 240 equals binary 11110000.)

Program code can use the subnet mask to determine if a destination address is in the same subnet. To do so, perform a logical AND of the destination address and the subnet mask and compare the result to a logical AND of the host address and the subnet mask. If the values match, the destination is in the same subnet. Figure 4-10 illustrates.

Classless Addressing

With classless addressing, the network address and IP prefix are often expressed in the form:

```
xxx.xxx.xxx.xxx/n
```

where `xxx.xxx.xxx.xxx` is the lowest IP address in the network and `n` is the number of bits in the network-address portion of the IP address. For example, with a network address and IP prefix of 192.0.2.0/24, the network address is 192.0.2 (three bytes, or 24 bits), and the final eight bits in the IP address are the host address.

In routing datagrams for addresses that use classless addressing, routers use Classless Inter-domain Routing (CIDR) protocols defined in RFC 1519.

Example 1

```
Source address =                          192.168.0.229
Source subnet mask =                      255.255.255.224
Destination address =                     192.168.0.253
Subnet mask AND Destination address =     192.168.0.224
Subnet mask AND Source address =          192.168.0.224
192.68.0.224 XOR 192.68.0.224 =           0.0.0.0
```

The values match, so the destination address and the host are in the same subnet.

Example 2

```
Source address =                          10.2.1.3
Source subnet mask =                      255.255.0.0
Destination address =                     10.1.2.1
Subnet mask AND Destination address =     10.1.0.0
Subnet mask AND Source address =          10.2.0.0
10.1.0.0 XOR 10.2.0.0 =                   0.3.0.0
```

The values don't match, so the source and destination are not in the same subnet.

Figure 4-10: To determine whether a destination IP address is in the same subnet as the source IP address, perform a logical AND of each IP address with the source's subnet mask. If the two values are the same, the destination is in the same subnet and the source can use direct routing.

IP Addresses Reserved for Special Uses

Some IP addresses are reserved for special uses. A network address or host address can never be all zeros or all ones. So, for example, in a network with an IP address and IP prefix of 192.0.2.0/24, the hosts can have a host address of any value from 1 to 254, but not 0 or 255. There is no network at 255.255.255 or 0.0.0.0

The Local Host

The address 0.0.0.0 refers to the local host or network, also called "this" host or network. In a network with a DHCP server, a host sends a datagram with a source address of 0.0.0.0 to request the server to assign an IP address.

Broadcast Addresses

A destination address of all ones is a broadcast to all hosts in a network or subnet. A destination of 255.255.255.255 would appear to be a broadcast to the entire Internet, but in fact, Internet routers and most other routers ignore broadcasts, so the datagram only goes to the hosts in the local network or subnet. Individual hosts may also be configured to accept or ignore broadcasts.

A broadcast can also specify a network or subnet, with the host address and subnet ID, if any, set to all ones. For example, a network with this network address and IP prefix:

192.168.100.0/28

can have up to 14 hosts (192.168.100.241 through 192.168.100.254)

And a broadcast to:

192.168.100.255

is directed to all hosts in the network.

As Chapter 1 explained, an Ethernet frame with a destination address of all ones is another way to do a broadcast.

Loopback Addresses

Addresses with the most significant byte equal to 127 are loopback addresses reserved for loopback tests. On receiving data to transmit to a loopback address, the IP layer passes the data back up to the source instead of passing the datagram down for transmitting on the network. Transmitting to the loopback address can be a useful test of the local networking software.

Multicasting

Another option for sending datagrams to multiple hosts is multicasting, where a source addresses a datagram to a specific group of hosts that may reside in different networks and subnets. Uses for multicasting include sending audio and video to subscribers.

Classful addressing reserves the Class D addresses for multicasting. In practice, multicasting on the Internet has been uncommon because all routers between the source and destination must support multicasting, and many routers don't. Multicasting is feasible within local networks, however.

As explained in Chapter 1, in Ethernet networks, destination addresses can also identify multicast groups.

Local Addresses

In a local network that doesn't connect to the Internet, the IP addresses only have to be unique within the local network. An address range in each class is reserved for local networks that don't communicate with outside networks:

Class A: 10.0.0.0 to 10.255.255.255
Class B: 172.16.0.0 to 172.31.255.255
Class C: 192.168.0.0 to 192.168.255.255

These ranges are preserved with classless addressing as well.

The addresses are for use within networks where the network administrator can ensure that no two hosts have the same address. A network that uses addresses in these ranges should not connect directly to the Internet or to another local network that might use the same addresses. However, as explained earlier in this chapter, it's possible to connect computers with local addresses to the Internet by using a router that performs Network Address Translation (NAT).

Other Reserved Addresses

RFC 3330: Special-Use IPv4 Addresses lists other reserved ranges of IP addresses.

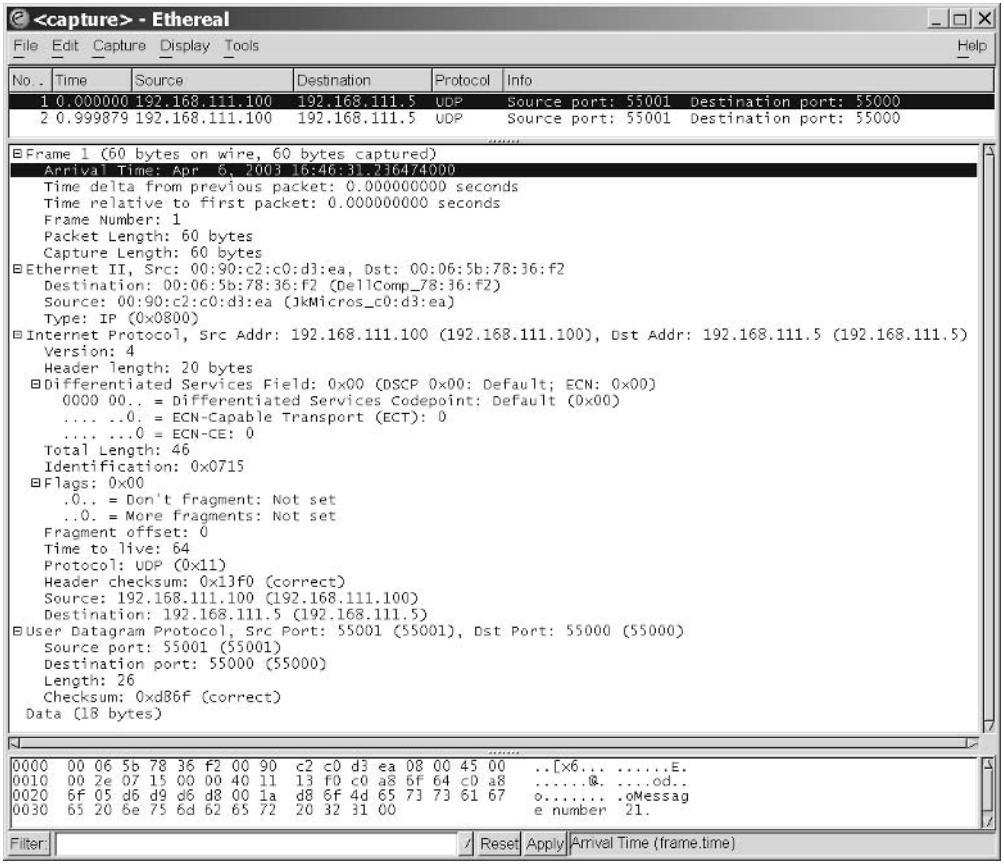

Figure 4-11: This capture from Ethereal shows an Ethernet frame whose data field contains an IP datagram. The data area of the IP datagram contains a UDP datagram.

The IP Header

An IPv4 header has twelve required fields and optional IP Options fields that precede the data, or message, being sent. Table 4-3 shows the fields in an IP header, and Figure 4-11 shows the contents of an example IP datagram. If you're using a provided library or other component for the IP layer, you normally won't have to concern yourself with the contents of most of the fields in the header, though the program code will need to provide

Table 4-3: Preceding the data portion of an IP datagram is a header with 12 or 13 fields.

Field	Number of Bits	Description
Version	4	IP version being used
Internet Header Length	4	Total length of the header in 32-bit words
Type of Service	8	Suggestions as to the importance of minimizing delay, maximizing throughput, and maximizing reliability in routing
Total Length	16	Total length of the datagram in bytes
Identification	16	Identifier for use in reassembling fragments
Flags	3	Information used in fragmenting
Fragment Offset	13	Position of a fragment in units of 64 bits
Time to Live	8	Maximum time or number of router hops a datagram may live
Protocol	8	Protocol identifier for the data portion of the datagram
Header Checksum	16	Error-checking value for the header
Source Address	32	IP address of source
Destination Address	32	IP address of destination
Options (optional)	varies	Additional information for security, routing, identification, and/or time stamping

source and destination IP addresses. Understanding the IP header can help in troubleshooting and in understanding IP's capabilities and limits, however. These are the functions of the fields in an IP header:

Version

The Internet Protocol has been through various revisions over the years. RFC0791, dated 1981, describes IP version 4, which is the version in popular use at this writing. Replacing IPv4 is IPv6, described in RFC2460. The field is 4 bits.

Header Length

The Header Length is the length of the datagram's header in 4-byte words. The length of the header can vary because of the optional IP Options field. The required fields use 20 bytes (for a Header Length of 5), and IP Options

can use up to 40 additional bytes (for a Header Length of 15). The field is 4 bits.

Type of Service

The Type of Service bits offer a way for the sending process to advise routers how to handle the segment. The options are to maximize reliability, minimize delay, maximize throughput, or minimize cost. Routers may ignore these bits. The field is 8 bits.

Total Length of Datagram

The Total Length of Datagram field is the length of the header plus the data payload in bytes. The maximum is 65,535 bytes. The field is 16 bits.

Datagram Identification

The host that originates the datagram assigns a unique Datagram Identification value to the datagram. If a router fragments the datagram as it travels to its destination, each fragment will have the same Datagram Identification value. This field is 16 bits.

Flags

Two bits in the Flags field relate to fragmenting.

Bit 0 is unused.

Bit 1: Don't Fragment. If this bit is 1, routers should not fragment the datagram. If possible, a router should route the datagram to a network that can accept the datagram in one piece. Otherwise, the router discards the datagram and may return an error message indicating that the destination is unreachable. The IP standard requires hosts to accept datagrams of up to 576 bytes, so if the datagram may pass through unknown hosts and you want to be sure it won't be discarded due to size, use datagrams of 576 bytes or less.

Bit 2: More Fragments. When this bit is 1, the datagram is a fragment, but not the last fragment of the fragmented datagram. When the bit is 0, the datagram isn't fragmented or it's the final fragment.

The field is 3 bits.

Fragment Offset

The Fragment Offset field identifies the location of a fragment in a fragmented datagram. The value is in units of eight bytes, with a maximum of 8191, which corresponds to a 65,528-byte offset.

For example, to send 1024 bytes in two fragments of 576 and 424 bytes, the first fragment has a Fragment Offset of 0 and the second fragment has a Fragment Offset of 72 (because 72*8 = 576). The field is 13 bits.

Time to Live

If a datagram doesn't reach its destination in a reasonable time, the network discards it. The Time to Live field determines when it's time to discard a datagram.

Time to Live expresses the time remaining for the datagram, with each router decrementing the value by 1 or the number of seconds needed to process and forward the datagram, whichever is greater. In practice, routers typically take less than one second to process and forward a datagram, so instead of measuring time, the value measures the number of hops, or network segments between routers. The computer sending the datagram sets the initial value. The field is 8 bits.

Protocol

The Protocol field specifies the protocol used by the datagram's data payload so the IP layer will know where to pass received data. The document *RFC0790: Assigned Numbers* specifies the values for different protocols. TCP is 6. UDP is decimal 17. The field is 8 bits.

Header Checksum

The Header Checksum enables the receiver of a datagram to check for errors in the IP header only, not including the contents of the data area, or message. The checksum is calculated on the values in the header, with the Header Checksum bits assumed to be zero. Error checking of the message is

required in Ethernet frames and TCP segments and optional in UDP datagrams. Figure 4-12 illustrates a checksum calculation.

To calculate a checksum on an IP header, do the following

1. Divide the header into a series of 16-bit words.

2. Add the first two words. If the result has a carry bit (if the result is greater than FFFFh), drop the carry bit and add 1 to the sum.

3. Add the next 16-bit word to the sum. Again, if the value has a carry bit, drop the carry bit and add 1 to the sum.

4. Repeat step 3 until all of the 16-bit words have been added in.

5. Find the one's complement of the result. To obtain the one's complement, in the binary value, change each 0 to 1 and change each 1 to 0. The result is the checksum.

RFC 791 says that the checksum appears to provide adequate protection, but may be replaced by a CRC calculation.

If you use software with built-in support for IP, you don't have to worry about providing code to calculate the checksum.

The field is 16 bits.

Source IP Address

The Source IP Address identifies the sender of the datagram. The receiver of a datagram can use this field to find out where to send a reply. The field is 32 bits.

Destination IP Address

The Destination IP Address identifies the destination of the datagram. The field is 32 bits.

Assigning an IP Address to a Host

A network may use any of a variety of ways of assigning IP addresses to its hosts. One approach is to have a network administrator configure the address at each host. This can work fine for small networks, especially if the

Contents of the IP header in Figure 4-11 expressed as 16-bit hexadecimal words:

```
4500
002E
0715
0000
4011
13F0 (checksum)
C0A8
6F64
C0A8
6F0F
```

Calculations to obtain the checksum:

```
4500 + 002E = 452E
452E + 0715 = 4C43
4C43 + 0000 = 4C43
4C43 + 4011 = 8C54
(Skip the checksum value.)
8C54 + C0A8 = 14CFC
14CFC - 10000 + 1 = 4CFD (Drop the carry bit and add 1.)
4CFD + 6F64 = 6C61
6C61 + C0A8 = 17D09
17D09 -10000 + 1 = 7D0A (Drop the carry bit and add 1.)
7D0A + 6F0F = EC0F
One's complement of EC0F = 13F0
The checksum is 13F0.
```

Figure 4-12: Calculating the checksum for this IP header verifies that the value is 13F0h.

hosts seldom change. But often, it makes more sense to have a single location in charge of assigning IP addresses. The Dynamic Host Configuration Protocol (DHCP) defines three ways of doing this.

DCHP: Three Options

The alternatives described in *RFC2131: Dynamic Host Configuration Protocol* are manual, automatic, and dynamic allocation. Table 4-4 compares the

Table 4-4: A network may use any of a number of methods to assign IP addresses to its computers.

Method of Assigning IP Addresses	Stores Addresses in a Single Server?	Method of Adding a Host	Method of Removing a Host	Requires the Host to Renew Its Lease Periodically?
Per Host Manual	no	manual	manual	no
DHCP Manual	yes	manual	manual	no
DHCP Automatic	yes	automatic	manual	no
DHCP Dynamic	yes	automatic	automatic	yes

capabilities of the Dynamic Host Configuration Protocol (DHCP)'s methods and manual assignment at the individual hosts.

All three DHCP methods require a computer that functions as a DHCP server. The other computers in the network are DHCP clients, which request IP addresses from the server. The server uses one of the three methods in responding to the requests.

On connecting to the network, a DHCP client uses UDP to broadcast a DHCPDISCOVER message to request an assigned IP address. Because the host doesn't have an IP address yet, it uses a source IP address of 0.0.0.0 in the request. The server must have another way of identifying the sender of the message. In an Ethernet network, the server can use the hardware address in the Source Address field of the Ethernet frame. The DHCP server responds to the DHCPDISCOVER message by returning an IP address to the requesting host, which uses the new address in future communications. RFC2131 and *RFC1533: DHCP Options and BOOTP Vendor Extensions* specify the format of DHCP requests and replies.

Manual Allocation

In manual allocation, the network administrator specifies an address for each host, but instead of configuring the addresses at each host, the administrator configures all of the addresses at the DHCP server. On receiving a DHCPDISCOVER message, the DHCP server returns the address assigned to the requesting host. For example, in an Ethernet network, the network

administrator can provide the server with a table that matches an IP address to the Ethernet hardware address of each Ethernet controller in the network. The DHCP server reads the source's Ethernet address from the Ethernet frame, finds the corresponding IP address in the table, and returns the address to the requesting host's Ethernet address.

Manual allocation is more convenient than configuring an address at each host, but the allocation still requires the administrator to know each host's hardware address and to assign an address every time the network gains a new host.

Automatic Allocation

In automatic allocation, instead of maintaining a table of values matched to hardware addresses, the DHCP server begins with a list of available IP addresses. On receiving a DHCPDISCOVER message, the server selects any unassigned address to return to the requesting host and marks the address in the table as assigned to that host.

Dynamic Allocation

One thing that automatic allocation doesn't define is a way to reclaim addresses that are no longer in use. Reclaiming addresses is essential in networks that have more potential hosts than available IP addresses. For example, the hosts connected to an ISP at any one time will vary as different customers go on and off line. If the ISP assigns a permanent, or static, address to every computer that connects, it will eventually run out of addresses, even if only a few customers connect at once. A solution is to use dynamic allocation, which reclaims IP addresses that are no longer in use.

As with automatic allocation, in dynamic allocation, the DHCP server begins with a list of available IP addresses and returns addresses in response to DHCPDISCOVER messages. But instead of assigning a permanent address, the server leases the address to the client for a specified time. To keep an address, the client must periodically send a request to renew the lease. If the client disconnects from the network or for any other reason fails to renew its lease, the server is free to assign the address to another computer.

A client may request an infinite lease or suggest a lease time, but servers aren't required to comply with these requests. The lease time is a 32-bit value in seconds, with FFFFFFFFh indicating an infinite lease.

On receiving a request for an IP address, a DHCP server uses the previously assigned address for that host if available. A computer can also request a specific IP address. But with dynamic allocation, there is no guarantee that a request for an IP address will return a specific value. For some small embedded systems, it may be easier to store a static IP address in firmware. The DHCP server must then be configured to reserve this address.

Each network may have its own DHCP server, or multiple networks may use relay agents to share a DHCP server. A relay agent accepts DHCPDISCOVER messages from hosts in a network and sends the messages to a DHCP server. The server replies to the relay agent, which then sends the message to the host that requested it.

In a network that has a NAT router that connects to a cable modem or DSL modem, the router typically can function as a DHCP server for the local network and as a DHCP client for the public network. The server can assign addresses to the hosts in the local network. The client enables the router to request an IP address from a DHCP server at the ISP.

Windows XP can function as a DHCP client for a server at an ISP or other location. Using Internet Connection Sharing, Windows XP can function as a DHCP server that assigns IP addresses to computers in a local network.

Considerations when Using Dynamic IP Addresses

When a domain's IP address changes, the DHCP server or other entity that changed the address must send an updated resource record to the domain's name servers.

If your ISP uses DHCP in assigning your network's public IP address and your embedded system has a domain name, you'll need to update the domain's name servers when the system's IP address changes. A way to achieve this is by using a service that handles the updates automatically. One provider of this service is Tzolkin Corporation (*www.tzo.com*).

To use Tzolkin's service with an embedded system, you'll need a PC in the same local network as the embedded system, and both computers must share a public IP address (using a NAT router as described earlier).

At the registrar where you registered the domain, you must change the registration to indicate that Tzolkin's name servers are the name servers for your domain.

On a PC that shares the embedded system's public IP address, run Tzolkin's application, which monitors the PC's current public IP address. When the address changes, the application automatically informs the name servers of the change. For best results, the PC running the Tzolkin application should be on line all of the time.

Matching an IP Address to an Ethernet Interface

Every IP datagram must include the IP address of its destination. A host can use a variety of ways to learn the IP address of a destination the host wants to communicate with.

A network administrator can provide each host with the IP addresses the host will communicate with. The hosts will need a way to update their lists when a host is added, removed, or changes its address, but if changes are rare, the updates can be done manually.

Some computers only need to reply to received communications using the source address in received datagrams. For example, a host that functions as a Web server that sends Web pages on request only needs to respond to received requests that include the IP address to reply to.

A host that wants to request a Web page or other resource or send other communications over the Internet must know the destination's IP address. As described earlier in this chapter, if a host knows only the domain name, a name resolver can query name servers to learn the IP address that corresponds to the domain name.

Using ARP

In a local network, the Address Resolution Protocol (ARP) can match an IP address with the Ethernet hardware address of the computer with that IP address. The document that defines ARP is *RFC 0826: An Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware*, also available as standards-track document STD0037.

To learn the Ethernet hardware address that corresponds to an IP address, a host broadcasts an Ethernet frame containing an ARP packet. In the Ethernet header, the Type field contains 0806h, which indicates that the frame is carrying an ARP message. The destination address is all ones or a broadcast address for a specific network or subnet.

In a similar way, a computer can broadcast a RARP (reverse ARP) request to learn the IP address that corresponds to a hardware address, including the computer's own IP address. RARP is defined in *RFC0903: A Reverse Address Resolution Protocol*, also available as standards-track document STD0038.

In Chapter 1, Figure 1-5 showed an example ARP request captured with the Ethereal Ethernet analyzer. Figure 4-13 shows the request's reply.

ARP and RARP Format

ARP and RARP requests and replies transmit in the data fields of Ethernet frames. Each request or reply has nine fields. The purpose of each field is as follows:

Hardware address space. Indicates the hardware interface being matched to a protocol address. Ethernet=0001h. 2 bytes.

Protocol address space. Indicates the protocol being matched to a hardware address. IP=0800h (specified in RFC1010). 2 bytes.

Length in bytes of a hardware address. Ethernet hardware addresses are 6 bytes. 1 byte.

Length in bytes of a protocol address. IPv4 addresses are 4 bytes. 1 byte.

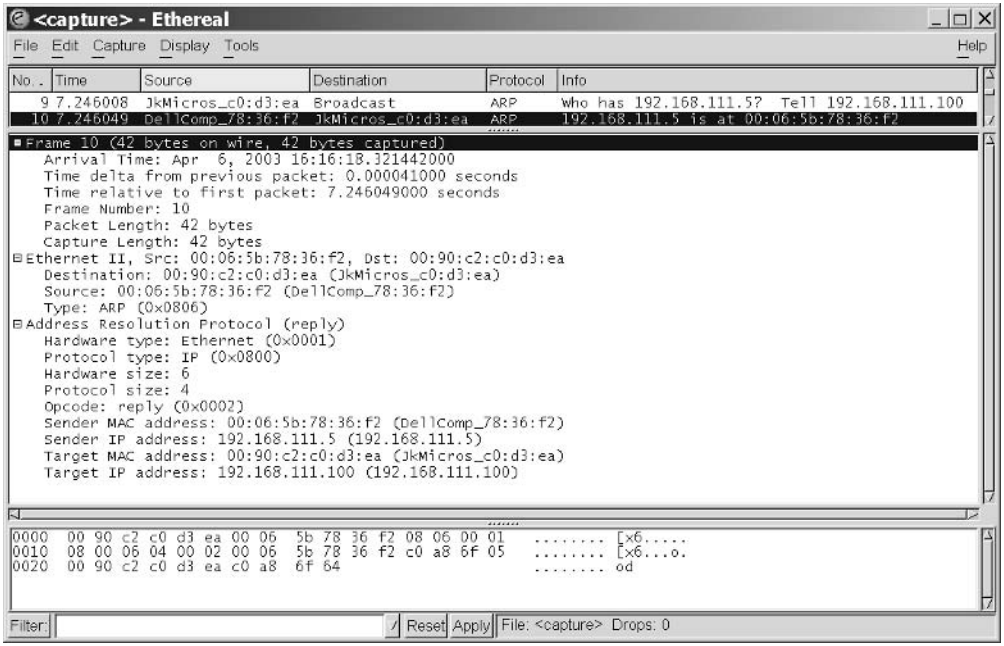

Figure 4-13: In this ARP response, the host with the specified target IP address responds with its Ethernet hardware address (00:06:5b:78:36:f2).

Opcode. Indicates the operation to perform:

1=ARP request
2=ARP reply
3=RARP request
4=RARP reply

2 bytes.

Source Ethernet hardware address. 6 bytes.

Source IP address. 4 bytes.

Destination Ethernet hardware address. For ARP requests, this value is undefined because it's the value being requested. For ARP replies, this value contains the hardware address for the request's IP address. For RARP requests and replies, this value is the hardware address whose IP address is being requested. 6 bytes.

Destination IP address. For RARP requests, this value is undefined because it's the value being requested. For RARP replies, this value contains the IP address that corresponds to the request's hardware address. For ARP requests and replies, this value is the IP address whose hardware address is being requested. 4 bytes.

To prevent having to send an ARP request before every communication to a host in the local network, a host can maintain a cache of ARP entries. To eliminate entries that are no longer valid, the cache must use timeouts or other methods.

How a Datagram Finds Its Way to Its Destination

When a host wants to send a message and knows the IP address of its destination, it's ready to send the IP datagram on the network. But how does the datagram find its way to its destination? The IP address contains no information about the physical location of the destination.

Direct Routing

Messages whose destination is within the local subnet, or within the local network when there is no subnet, use direct routing. In direct routing in an Ethernet network, the originating host sends an IP datagram in an Ethernet frame that contains the destination's Ethernet hardware address, as described in Chapter 1. The originating host uses ARP if needed to learn the destination's hardware address.

Within an Ethernet network, hosts connected by repeater hubs receive all valid frames sent by any of the hosts. An Ethernet switch forwards frames to a specific port if possible, and otherwise forwards the frame to all of the switch's ports.

Indirect Routing

Messages whose destination is outside the local subnet or network use indirect routing. With indirect routing, a designated default router accepts messages destined for outside the local subnet or network. An Ethernet network

that connects to the Internet should have a default router for messages whose destination is outside the local network.

For example, if a computer in an Ethernet network wants to send a message on the Internet, the computer places the message in an IP datagram in an Ethernet frame. The destination address in the Ethernet frame is the default router's hardware address. The default router uses the destination address in the IP datagram to decide where to forward the datagram.

To decide where to forward the datagram, the router first checks its internal forwarding table for a matching IP address. Each entry in the table has the IP address of the router that is the next hop, or the next router on the way, for datagrams going to a specific address, network, or subnet.

A router builds its forwarding table by saving entries containing the source address of received datagrams and the router port that the datagram arrived on. To ensure that there's room for new entries, each entry has a timeout and is removed on timing out.

Of course, no forwarding table will contain an entry that matches every received destination address, if only because a router may begin with no entries other than a default router. On receiving a datagram with a destination address that isn't in the forwarding table, the router sends the datagram to another router designated as the first router's default router. In a similar way, the router that receives the datagram looks for a match in its forwarding table and sends the segment on either to a destination found in its forwarding table or to another default router.

An IP datagram may travel through a number of routers on the way to its destination. The source may have no way of knowing the maximum size of datagrams the routers or the destination can accept. If a datagram is too large for its destination, a router may send the data payload in multiple, smaller datagrams, with a portion, or fragment, of the data in each. On receiving the datagrams, the destination uses information in the IP header to put the fragments back together.

The Internet Control Message Protocol (ICMP)

Hosts that support IP must also support the Internet Control Message Protocol (ICMP) defined by *RFC 792: Internet Control Message Protocol*. ICMP is a basic protocol for sending messages. Some common uses for ICMP are to send a PING message to learn if a host is available on the network and to obtain the IP addresses of local routers.

ICMP messages travel in IP datagrams. The Protocol field in the IP header is 1 to indicate ICMP. The first byte in the data portion of the datagram is an ICMP Type code that determines the format of the data that follows. RFC 792, RFC 950, and RFC 1256 define the type codes listed in Table 4-5 and have further details about the message formats.