1

IP NETWORKING OVERVIEW

IP NETWORKING 101

Each party engaged in a communication, whether two people speaking or two computers exchanging information, must comply with a set of conventions that govern the rules of such communication. Language and culture generally guide such conventions for human conversation. A *protocol* defines these conventions for computers. And it's usually easier to get computers to comply with these conventions than people. A protocol dictates the sequence and syntax of communications as well as recovery mechanisms required during error conditions. There are actually several protocols or protocol layers that are used during computer communications, each providing a specific set of functions to support a level of commonality for communicating over a variety of media. We'll delve deeper into this later in this chapter, but let's start out with a simple analogy to human communications to introduce the key aspects of Internet Protocol (IP) addressing and why address management is important.

When two people converse, one person may initiate the discussion in one of many ways: by physically approaching the other and speaking, calling him or her on the telephone, sending him or her an instant message, and so on. In each of these scenarios, the initiator of the conversation identifies and locates the

Introduction to IP Address Management, By Timothy Rooney

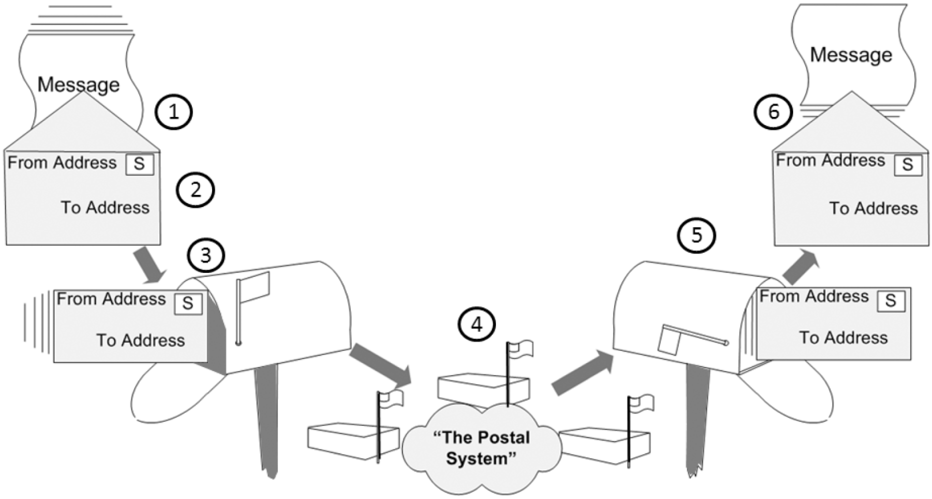


Figure 1-1: The postal delivery analogy

intended recipient, then attempts to begin a conversation using the chosen medium. When I want to talk to my friend Steve, for example, I can look up his number online or in a phone book, dial his number, and when he answers the phone, I can identify myself and begin the conversation. At a basic level, IP communications follows a similar process. When an IP device seeks to communicate with another, it must identify and locate the intended recipient, then initiate communications over a link, while also identifying itself to the recipient in the process.

Perhaps the best, though admittedly overused, analogy for IP communications is that of postal letter delivery. Nevertheless, let's consider this process of "sneaker mail," then relate it to IP communications. The basic postal delivery process is depicted in Figure 1-1, beginning with my writing a letter to Steve and communicating it via postal mail.

After writing my letter, I enclose it in an envelope. This is step 1. Next, I write my return (From) address and Steve's (To) address on the envelope, and stamp it to pay my postal service provider. At this point, I'm ready to mail it, so step 3 consists of depositing my letter in my outgoing mailbox. After my mailperson picks up my letter, the fourth step entails forwarding of the letter within the postal system to the local post office serving Steve's address. After the letter has been delivered to the post office or distribution center serving Steve's address, a local delivery mailperson drops the letter in Steve's mailbox. When Steve walks out to the mailbox, he can open the letter and read my letter. Message delivered!

So let's map this postal message flow to sending a message over an IP network, referring to Figure 1-2. In this case, we're communicating electronically over the Internet, though this analogy holds whether communication ensues over

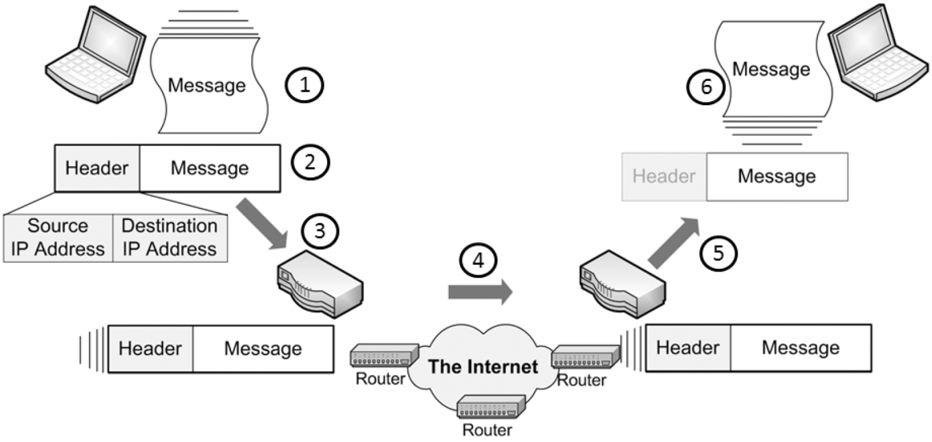


Figure 1-2: Internet Protocol communications

a private enterprise, service provider, home IP network, or a combination thereof. Just as Steve and I have postal mail addresses, we both need *IP addresses* to communicate with each other over the Internet. No one else in the world has the same mailing address as Steve; likewise, no one else in the world has the same IP address as Steve (technically this isn't necessarily true when IP addresses are translated between me and Steve, but let's go with it for now). Let's assume that each of our computers is configured with its respective IP address and that I know Steve's IP address.

Step 1 entails the creation or typing of my message to Steve. In step 2, my computer, knowing my IP address and Steve's, places my message within a data packet, or specifically an *IP packet*. An IP packet is simply the message to be communicated, prefixed with an IP header. The IP header, like our letter envelope, contains my (From) source IP address as well as Steve's (To) destination IP address, among other fields. Having formulated my IP packet, I'm now ready to send it. From my home network, I have a broadband router, to which my computer transmits my IP packet as step 3. This transmission may occur over a cable or a wireless connection between my computer and the router.

In step 4, my router forwards my IP packet to the Internet via my broadband service provider (no stamp required, they'll bill me later). Devices in the Internet called *routers* forward my IP packet ultimately to Steve's broadband service provider and the broadband router in his house. Routers examine each IP packet's header information to determine where to forward the packet to reach its destination IP address efficiently. Having been delivered to Steve's broadband router, step 5 consists of forwarding the packet to Steve's computer, whose IP address matches the IP packet's destination IP address field. In step 6, Steve's computer strips off the IP header to yield the message I had typed. Message delivered!

In both postal and IP communications, the source and destination addresses are specified and are unique, an infrastructure of people and/or machines

TABLE 1-1: Key Similarities Between Postal and IP Communications

	Postal Communications	IP Communications
Message contents	Letter, package, or parcel	Application data such as an instant message
Message container	An envelope or box with To and From street addresses	An IP packet including an IP header with source and destination IP addresses
Sending of the message	Performed by depositing the letter in outgoing mailbox	Performed by transmitting the IP packet from the computer to the local router
Message routing	The letter is physically transported by air, sea, and/or ground via one or more postal offices or distribution centers, ultimately reaching the postal delivery center serving the To address specified	Routers forward the IP packet over a variety of media (e.g., fiber, copper, wireless) to other routers ultimately reaching the router serving the destination IP address
Message delivery	Postal personnel deliver the letter to the street address specified on the envelope	The local router delivers the IP packet to the computer configured with the destination IP address
Message receipt	The envelope is opened and discarded, and the letter is read	The IP header is stripped from the IP packet and the message or packet payload is delivered to the application (instant message window in this case)

successively forwards the message toward its addressed destination, and it is ultimately delivered to the recipient. Table 1-1 summarizes the key similarities among the postal and IP communications examples.

The two core concepts common to these communications analogies are routing and addressing. As we've implied so far, routing is dependent on proper addressing! Let's examine this relationship in more detail.

IP Routing

The postal system operates by "routing" letters and packages as efficiently as possible to regional distribution centers, local centers, and finally to the curb. Scanning and tracking systems along the way direct parcels closer to their ultimate destination via various means of transportation through one or more distribution centers along the way. Typically this routing is performed by first examining the "To" (destination) addressed country, postal code, city and state or province, and finally the street address. The encoding of the general (country, postal code) and the specific (street address) in the "To" address enables different entities in the postal system to use different portions of the address to route efficiently. Distribution centers can forward packages based only on country and postal code information; once the parcel arrives at a local center serving the destination postal code, the local center then needs to examine the street address for final delivery.

If Steve lives down the street, my letter will simply traverse my local post office, perhaps a distribution center, and back to Steve's local post office for delivery. If Steve lives across the country, my letter will likely route from my local post office through one or more regional centers, then to Steve's local distribution center for delivery. If Steve lives in a different country, my letter will likely be required to enter the country through a customs agent. The customs agent may analyze the letter and either allow its further delivery within the country or deny it by returning it to the sender or confiscating or disposing of it.

Routers perform analogous functions in routing IP packets. Routers mimic the scanning systems of the postal system by examining the network portion of the destination IP address within each IP packet and forwarding it on, getting closer to the destination with each hop. Upon reaching the local serving router, this router examines the full IP address in order to deliver it to the intended recipient. Hence, IP addresses are comprised of a network portion and a host* portion, concatenated together as we'll discuss in the next chapter.

If the packet is destined for a network operated privately, for example, by a corporation or enterprise, the packet will likely meet with examination analogous to the customs agent. And like the customs agent, this enterprise gateway or firewall can allow or deny further transmission of the packet to its destination. By the way, just as storms or other events can cause flight delays or mail reroutes in the postal system, routers can detect analogous outage or congestion events to reroute IP packets as needed. Yes neither rain, nor snow, nor dark of night will stop postal mail or IP packet delivery!

IP Addresses

As we've seen, each device on an IP network must be uniquely identifiable, by means of an IP address. Hence, each device desiring to communicate on an IP network requires an IP address. So your computer at home, your voice-over-IP phone at work, even your cell phone likely has an IP address, at least at the time it's powered up and ready to communicate. In our example above, we assumed that the IP address of each computer was already programmed in, but how does this IP address get in there? The IP address for each device can be assigned and configured in each device either manually or automatically.

The manual address assignment approach using a fixed IP address works well for fixed infrastructure IP devices like routers and servers. But for the vast majority of IP addressable devices such as laptops and cell phones, which are highly mobile, the fixed address assignment approach does not work well. This is because

*The term *host* refers to a device on an IP network. We will use the terms *device* and *host* interchangeably.

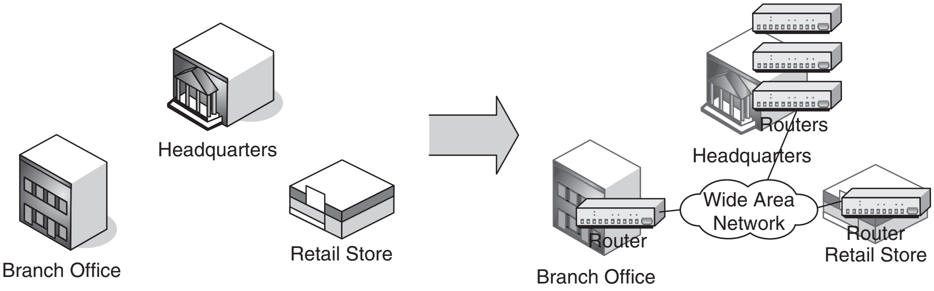


Figure 1-3 (a): Organization's locations (b): Overlay of IP network foundation

the assigned IP address must be relevant to the current network or subnet to which the IP device is connected. If these IP devices move about, they need to be IP addressable within the context of their current location on the IP network, rendering the manual method very cumbersome. Even the postal service doesn't offer a "find me" service to deliver my mail to me wherever I happen to be!

To illustrate this location-sensitivity requirement, consider a small organization with three offices as illustrated in Figure 1-3(a). To enable network communications among these sites, we interconnect them over a wide area network, which may be the Internet or, in this case, a private network from a service provider. To enable communications and routing, we've installed at least one router in each location as illustrated in Figure 1-3(b). This figure shows an overlay of a simple IP network among these locations.

To enable routing among these locations, we need to assign each location a unique set of IP addresses. In this way, the Branch Office will be home for one set of IP addresses (or one IP network), the Retail Store a different set, and Headquarters, yet another unique set of IP addresses. Let's use the set of IP addresses shown for each router as in Figure 1-3(c). Each router will support a set of IP addresses, from which individual IP addresses would be assigned to printers, laptops, voice over IP phones, and other IP devices in that location.

We'll describe the structure and format of IP addresses in more detail in Chapter 2, but an IP address is composed of four numbers, separated by decimal points or dots. Each of the four numbers can range in value from 0 to 255. In our example, IP address sets 10.0.1.0-255, 10.0.2.0-255, and 10.0.3.0-255 have been allocated to headquarters, 10.1.1.0-255 to the branch office, and 10.2.1.0-255 to the retail store. Each of these IP address sets is called a subnetwork or *subnet*, as each represents a portion or subset of the overall enterprise's set of IP addresses.

Note that each set of addresses falls within a contiguous IP address range that corresponds to the network portion of the IP addresses within each set. Recall from our postal analogy that the network portion of the address is used for efficient routing until the IP packet is delivered to the local serving router.

The interconnecting wide area network (WAN) also has a network address (10.254.1.0-255). The routers in each location must be configured with this network information in order to properly route IP communications traffic. In this

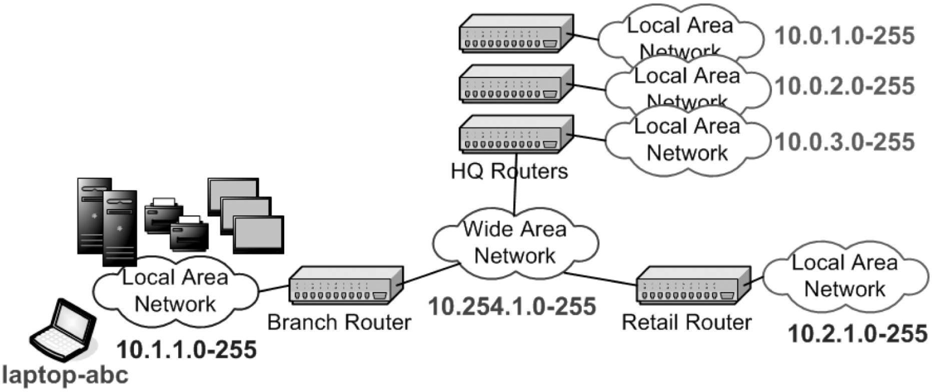


Figure 1-3 (c): More detailed IP network breakdown

way, our Branch Office router is responsible for IP addresses 10.1.1.0-255, and so all IP packets with a destination IP address falling in this range will be forwarded to the Branch Office router. This partitioning of IP addresses to particular sites or routers is analogous to the splitting of geographic locations by zip code and corresponding postal distribution centers.

Now that we've partitioned our IP addresses and configured our routers in accordance with our addressing plan, let's look at address assignment for a given device, say my laptop. Let's say I'm in the branch office on Monday as signified by *laptop-abc* in Figure 1-3(c). I'll need an IP address from the 10.1.1.0-255 subnet, let's say 10.1.1.52. This is because the Branch router "owns" the 10.1.1.0-255 subnet and serves as my "local post office" for delivery of IP packets to devices in the branch office. When I send an instant message to a colleague at Headquarters, my messages are routed to a Headquarters router and responses are routed to me via the Branch router.

Now let's assume I'm called to a meeting at the retail store on Tuesday. When I arrive at the retail store and connect to the store's network with my manually configured 10.1.1.52 IP address, I will quickly realize that I cannot communicate on the network. This is because my IP address, part of the 10.1.1.0-255 network, is served by the Branch router. Thus, when I begin sending an IP communication,* say by opening a web page, entering a www address, my web browser sends an IP packet to the destination web site IP address, using my laptop's IP address, 10.1.1.52, as the source IP address. The web server acknowledges the communication and responds with the requested web page, addressing it back to IP address 10.1.1.52. The routers all "know" that the Branch router delivers IP packets to IP addresses on the 10.1.1.0-255 subnet, so they route the response IP packet intended for my web browser back to the branch office!

*Technically my IP communication will not even get past the Retail router which would drop my packets because my source IP address is not valid for the retail subnet. But this simplified example illustrates the fallacy of this communications attempt even if it would be technically feasible.

From my perspective, I'm not getting a response from the web server. Is the network down or is the web server down? As I call the help desk complaining about the network outage, the network team eventually discovers that my IP address is not appropriate for the subnet to which I am connected. They walk me through the cumbersome process of manually changing my laptop's IP address to 10.2.1.187 (only to walk me back through the reverse process when I return to the branch office on Wednesday). Once my address is relevant to my location, I'm able to communicate bi-directionally with the web server and other IP application servers such as email servers because my address falls within the set assigned to the local serving router.

This simple example illustrates the importance of not only having an IP address for IP communications, but one that's appropriate to the subnet to which you're connected. Thus, with even minimal mobility of associates going to meetings, visiting customers, or generally traveling about, the clumsy manual help desk process outlined above is impractical. To require people to call the help desk when they need a new IP address (not only when the network really is down) reduces end user productivity (and patience!), as well as increases the costs of help desk operations and network and server technical support. Plus it's extremely difficult to walk a "technically challenged" person through the process of manually entering an IP address on a device! And as the variety of devices that people use to connect increases, the variety of IP address entry methods likely increases as well, adding to the support team burden. Clearly, an automated mechanism for assigning a unique IP address relevant to the subnet of connection is crucial to reducing costs while maintaining overall end user and support staff satisfaction and productivity. The Dynamic Host Configuration Protocol (DHCP) is one such mechanism that enables an IP device connecting to a network to automatically obtain a unique and location-relevant IP address.

After the address assignment problem is solved, our next objective is to eliminate error-prone human entry of IP addresses. Earlier, we glossed over the fact that I already knew Steve's IP address or that the www address I typed was magically translated into an IP address to enable creation of an IP packet. So if communication on an IP network requires IP addresses, how do we get away with entering text names to send emails or connect to websites? The solution is the Domain Name System (DNS), which enables users to enter names for services, web sites, or email boxes, obviating the need to enter IP addresses.

We generally take this for granted, which is a good thing! Imagine carting around the equivalent to an Internet phone book with web sites and associated IP addresses. DNS works "behind the scenes" to provide a name-to-address look-up mechanism to bridge this gap between human consumable names to network consumable IP addresses. Unless you're a numbers wizard, entering http://www.ipamworldwide.com in your web browser is vastly more simple than entering (and remembering) http://192.0.2.201. Fortunately, this usability problem was recognized early in the development of the Internet and DNS was devised to automate this directory lookup function.

Once you type in a www address, your computer looks up the www address in DNS and obtains the corresponding IP address, which it then uses as the destination IP address in the IP header. Other forms of name-to-address translation have been developed over the years, including hosts files, Yellow Pages (YP), and Windows Internet Naming Service (WINS), but DNS has sustained its dominant status as the de facto, production network-proven name-to-address translation service for IP communications today. Before further exploring the core elements of IP address management, namely IP address allocation, DHCP, and DNS, let's take a more detailed look at the basics of IP networking by exploring the inner workings of how routers deliver IP packets to their respective destinations.

IP NETWORKING 102

After reviewing our simple IP addressing example above, you may be wondering, why don't we just eliminate the routers and use one massive network that everyone can share instead of employing this subnetting process that leads to readdressing of my laptop when I move from location to location? After all, I could just use one IP address anywhere in the network. While this approach is theoretically possible utilizing a bridged network across all of these sites, this does not scale well because performance of communications will suffer with growing intersite distance and number of devices on the network. This is due to that fact that in a shared or bridged network, every device's messages are sent to every other device on the network. And because the network is a shared medium, collisions may result when two or more devices attempt to send messages at the same time.

Collisions in the networking world have the same effect as those in interpersonal communications. As more members join the "conversation," collisions arise more frequently. As collisions occur parties to the conversation (at least the polite ones) back off momentarily before reattempting to initiate communications. The more parties involved in the conversation, the more frequently collisions occur, and the larger the backlog of messages awaiting communications. This attempt/back-off/reattempt process escalates quickly until ultimately no party attempting to communicate gets a message to a recipient. As the backlog of reattempts to communicate builds, frustration escalates and gridlock results. The same effect occurs when too many devices attempt to communicate on a large, monolithic, shared medium. And the same general solution, which for human conversation consists of groups naturally branching off from the main conversation into subgroups of smaller sets of people, can be applied to IP networks.

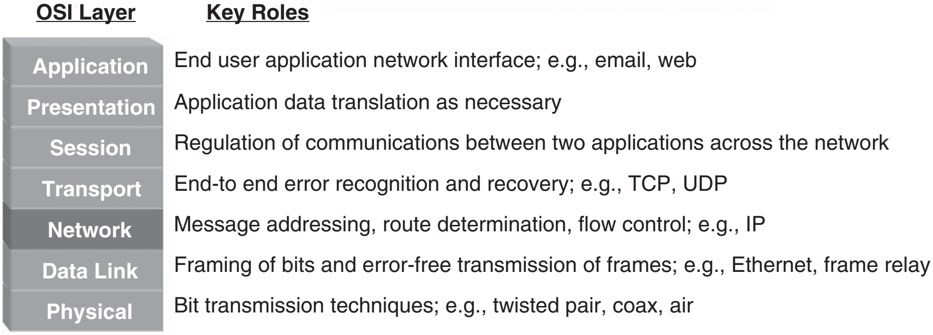
By reducing and limiting the number of parties communicating on the same medium, we can localize the *collision domain* and reduce the number of collisions and backlog on the network as a whole. The deployment of switches and routers supports this partitioning of the network into separate collision domains. Switches enable partitioning of the collision domain itself by virtue of interconnecting pair-wise switch ports, each of which is physically connected to a host. Routers provide collision domain boundary points by terminating yet interconnecting collision domains. Routers in particular leverage the concept of protocol layering to separate collision domains. Let's review the concept of protocol layering, which will lead us back to a more detailed discussion of the roles of switches and routers.

Protocol Layering

The International Standards Organization (ISO) has defined a layered protocol model, separating responsibilities for different aspects of controlling communications (1). The layered model consists of seven layers and is denoted the Open Systems Interconnect (OSI) model. The term *protocol stack* refers to the fact that several protocol layers are "stacked" one upon another to usher data and commands from my web browser onto the wire or over the air and through the network to the destination. In fact, the Internet Protocol's ability to run over various media such as these and others is a powerful consequence of the use of a layered protocol stack. The OSI model enables a common implementation of the Internet Protocol across a variety of lower layer data link and physical layers, including cable, Digital Subscriber Line (DSL), fiber, Ethernet, wireless and so on. Figure 1-4 illustrates the OSI model with a brief summary of key functions of each layer.

Application Layer (Layer 7). The application layer provides the primary end user exposure and functionality. A web browser, file transfer program, or email client are examples of applications.

Presentation Layer (Layer 6). The presentation layer is responsible for defining the data format and syntax between application endpoints in the communication. For example, this layer specifies standard graphics formatting and translation for communications across the network.



OSI Layer	Key Roles
Application	End user application network interface; e.g., email, web
Presentation	Application data translation as necessary
Session	Regulation of communications between two applications across the network
Transport	End-to end error recognition and recovery; e.g., TCP, UDP
Network	Message addressing, route determination, flow control; e.g., IP
Data Link	Framing of bits and error-free transmission of frames; e.g., Ethernet, frame relay
Physical	Bit transmission techniques; e.g., twisted pair, coax, air

Figure 1-4: OSI Protocol stack summary (1)

Session Layer (Layer 5). The session layer is responsible for regulating the end-to-end communications of applications across the network, providing such services as security and authentication. NetBIOS is an example of a session layer protocol.

Transport Layer (Layer 4). The transport layer is responsible for end-to-end communications integrity, assuring flow control between the two endpoints, as well as data checking, requesting retransmissions, and proper ordering of information. Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are transport layer protocols within the TCP/IP stack.

Network Layer (Layer 3). The network layer is responsible for the formatting of information into packets and/or packet fragments for communications and for routing over one or more networks. IP is a network layer protocol.

Data Link Layer (Layer 2). The data link layer is responsible for formatting of information into frames for communications over the physical network, including error checking for data integrity. Ethernet, Token Ring, ATM, and frame relay are examples of data link layer protocols. The data link layer is commonly split into Logical Link Control (LLC) and Media Access Control (MAC) sublayers. The familiar term "MAC address" refers to a device's layer 2 or media access control address.

Physical Layer (Layer 1). The physical layer defines the electronic interfaces and characteristics including voltage and current specifications for transmission of data and control (e.g., preamble) bits. EIA-232 (RS-232) provides an example of a physical layer specification.

OSI and TCP/IP Layers

These protocol layers not only permit interoperability for multiple applications and underlying physical networks, they also segment the responsibility required for successfully communicating over a data network. For example, some layers such as the data link and network layers, provide error checking and correction to facilitate accurate communications and reduce retransmission requirements. Others, such as the transport layer, are responsible for end-to-end communications integrity and proper ordering of information. Overall, the standardization of protocol layer definitions enables successful end-to-end communications while facilitating interoperability.

This seven-layer stack shown in Figure 1-4 is sometimes portrayed as a five-layer stack in the Internet context with the Application layer sitting above TCP, IP, data link, and physical layers, respectively, as shown in Figure 1-5. Protocol layering enables not only the transmission of IP packets over a variety of media, it also permits a variety of end user applications to communicate over IP, which in turn run over various lower layer protocols. For example, an email client

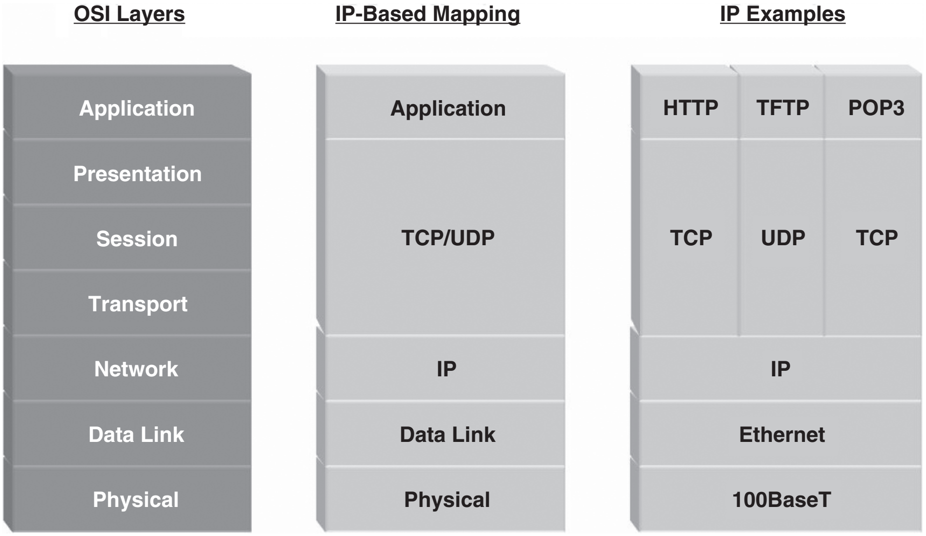


Figure 1-5: The OSI and TCP/IP Protocol stacks

application can communicate to an email server using a Post Office Protocol version 3 (POP3) application, which is layered on TCP, and in turn IP, which can then be layered on an Ethernet, Token Ring, frame relay, or other layer 2 protocol, and ultimately a particular physical layer. Another example illustrated in Figure 1-5 features HTTP (web browsing), TFTP (trivial file transfer protocol), and POP3 running over respective transport protocols and IP, Ethernet, and 100BaseT. This provides a seamless end user experience in using common "desktop applications" whether communicating over an Ethernet 100BaseT network, an 802.11 wireless network, or an ATM-based DSL network.

Layering also enables components of the stack to be developed and offered by different organizations. In practice, protocol layering works effectively at the application to TCP/IP boundary and the TCP/IP to data link boundary, though TCP and UDP generally operate exclusively with IP at the transport-network layer boundary. An application programming interface (API) enables a variety of applications from different vendors to utilize common function calls into the TCP/IP stack, which is commonly included with the operating system. The de facto API for TCP/IP applications is the *sockets* interface originally implemented on BSD UNIX (on which DNS BIND was also originally implemented). The sockets interface defines program calls to enable applications to interface with TCP/IP layers to communicate over IP networks. Microsoft's Winsock API is also based on the sockets interface.

Intra-Link Communications. To get our arms around how data flows through these layers within protocol stacks, consider Figure 1-6. Beginning with the application data in the upper left of the figure, notice the addition of headers

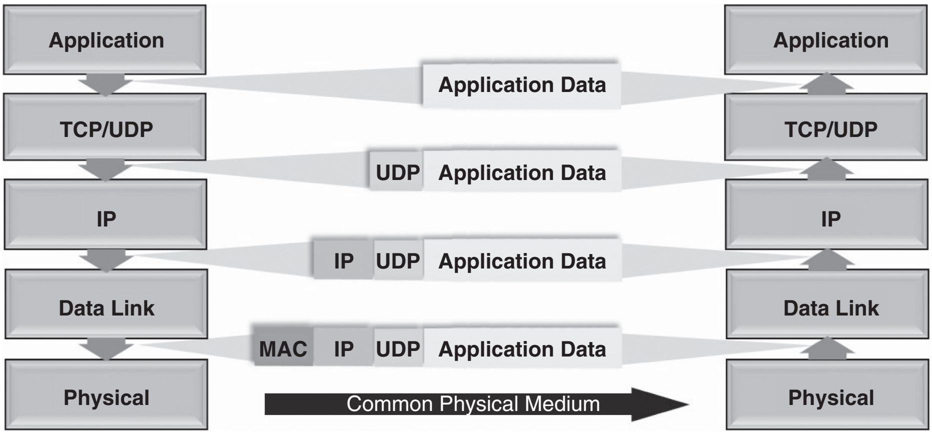


Figure 1-6: Protocol layering

as the application data payload traverses down the stack. The data link frame that is transmitted on the local network encloses the application data and upper layer headers. The frame recipient on the right side of the figure then processes and strips off each header at corresponding layers as the data is passed up the stack to the destination application. The headers and payload shown at each layer boundary are exactly the same on both ends of this intra-link communication.

As application data is prepared for transmission on an IP network using a TCP/IP stack, the application calls the sockets API to communicate the data down the stack. A TCP or UDP header is added based on application selection, and then an IP header is appended. Recalling our earlier postal analogy, we can now see that the IP header was just one such header added to a network message. The "message" to which the IP header was appended included the application data (e.g., the instant message text) plus the transport layer (TCP or UDP) header. In this multi-layered stack model, each layer adds a header to enable it to perform its respective function.

The application generally dictates the UDP or TCP header parameters, which consists of application-specific port numbers as well as checksum, flow control, and other data. As we've discussed, the IP header contains the sender's and the receiver's IP addresses. The sender's IP address is assigned manually or automatically (e.g., via DHCP) and the recipient's IP address is entered either via the application user interface or is fetched from a DNS server.

Below the IP layer, the IP packet, which itself comprises and IP header, a TCP or UDP header, and application data, is enclosed within a data link layer frame for transmission over the physical network. Communication on a given data link requires encapsulation of the IP packet within a data link frame, which itself requires source and destination data link (MAC) addresses. To transmit the IP packet within a data link frame, the transmitting host must determine the recipient's data link (MAC) address. So the device must map the destination IP address to a destination MAC address for transmission within a layer 2 frame on the physical network. But another wrinkle arises depending on whether the destination IP address is on the same data link as the sender.

Are We on the Same Link? Generally each device can determine if the intended recipient resides on the same link by virtue of its configured IP address and subnet address range. So if the destination IP address falls within the same subnet range as the sender's IP address, the device is considered on the same link. Otherwise, if the intended recipient does not reside on the same subnet, the source device must identify a router on the link that can forward or route the data to the intended destination. Using either a routing table, or for most non-router devices, a configured *default route*, the device can determine the next hop to which to send the data. A default route is the next hop destination to which all packets are sent in the absence of a known next hop toward the intended destination address. This is kind of like my outgoing mailbox in the postal analogy, which is where I place all of my outgoing communiqués. This default route, commonly referred to as a default gateway, is typically the IP address of a router serving the subnet to which all nonlink local packets are to be sent.

For example, per Figure 1-3(c), if *laptop-abc* transmits data to another device within the branch office, that is, with destination IP address in the range of 10.1.1.0–10.1.1.255, it can be sent directly (intra-link); if it transmits data to a device in HQ or the retail store, it must send it to the branch router for routing. Whether sending the data to an intra-link destination or a router, the sending device must determine the recipient's or router's data link (MAC) address in order to formulate and transmit the IP packet within a data link layer frame. This MAC address is the data link layer address for the intended device. But we determined the destination IP address from DNS, not the MAC address; so how is this MAC address determined?

The *Address Resolution Protocol (ARP)* enables a device to determine the MAC address of a device on the same link corresponding to an IP address. The device transmitting the message knowing the intended next hop IP address (recipient on-link or router) formulates an ARP data link broadcast frame requesting MAC resolution of the IP address. A *broadcast frame* is a data link frame addressed to all devices on the link. For Ethernet data links, for example, a broadcast frame uses the destination Ethernet address of all 1's, meaning the broadcast address for all devices connected to this Ethernet link. The device that is configured with the sought IP address responds with an ARP response frame indicating its corresponding layer 2 address. This enables the source device to formulate the Ethernet frame for transmission of the IP packet. Most devices cache this information in an ARP cache, providing a temporary storage of this IP-to-MAC address correlation to reduce repeated ARP queries, for example, for multi-frame communications.

Limiting Broadcast Domains. Data links like Ethernet comprise the collision domains (also referred to as broadcast domains) we referred to earlier. The data link layer is chiefly concerned with accessing the network for transmission, detecting collisions, and performing error checking on frames. All devices connected to a common Ethernet network receive frames sent from every other connected device. Hence it's easy to see how our earlier analogy about backlogs and gridlock as the number of parties attempting to communicate holds true. As the number of devices within the collision domain increases, and/or the number of communications attempts per device increases and/or the volume of communications per device increases, the more likely data collisions will occur, degrading network performance. If we can confine the number of participants in each collision domain, we can improve overall performance and end user productivity and satisfaction.

This brings us back to how switches and routers constrain broadcast domains. Historically, Ethernet local area networks (LANs) were deployed with wiring to each office, cubicle, or end station funneled back to one or more Ethernet hubs. Hubs literally broadcast frames received on any given port to every other port, thus comprising an indivisible collision domain. Switches were developed to directly interconnect (or switch) traffic between source and destination ports without blindly broadcasting all data to all ports. Switch ports effectively provide a direct point-to-point connection between the end device and its switch port. The switch detects the MAC address of each connected device then leverages this information to directly interconnect the port on which the frame is incoming to the appropriate destination port. This minimizes superfluous broadcast traffic to devices on all other ports. Of course layer 2 broadcast traffic is broadcast to all switch ports, but this is certainly an improvement!

Modern layer 2 switches have further evolved to enable definition of a subset of physical ports to a given broadcast domain. Thus, instead of hardwiring hub ports within a broadcast domain or LAN, one could partition which switch ports belong to independent broadcast domains. These independent broadcast domains are referred to as virtual LANs (VLANs), as the switch supports multiple logical LANs on one physical switch device. Broadcast traffic on ports within VLAN 1 will not be broadcast to switch ports associated with VLAN 2 for example. Consider Figure 1-7 for an example of VLAN segmentation. The figure shows the common implementation of associating different VLANs with different

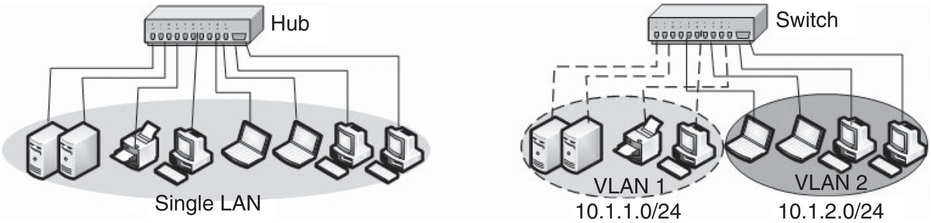


Figure 1-7: Hub vs. switch/VLAN architecture

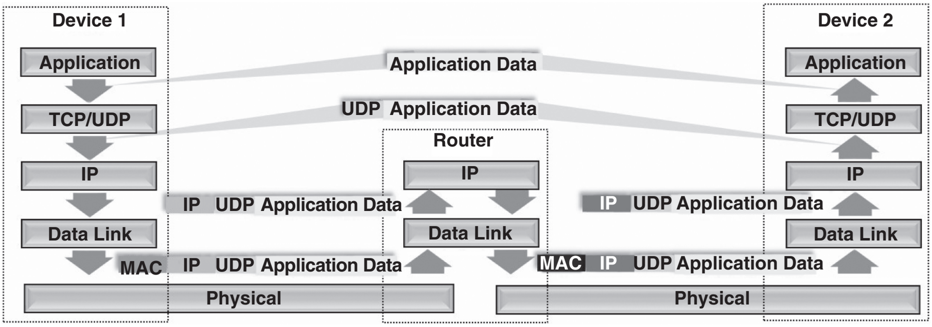


Figure 1-8: Protocol stacks with an intervening router

subnet addresses.* This is contrasted against the single broadcast domain per hub shown on the left of the figure.

Inter-Link Communications. Switches certainly help reduce the scope of collision domains, but there are only so many switch ports! The second method used to limit broadcast domains leverages protocol layering concepts and employs routers to separate layer 2 networks. Consider Figure 1-8, which is a recasting of Figure 1-6 with an intervening router. The left side of the figure is identical to that in Figure 1-6. Each protocol layer adds its respective header as the application data moves down the stack. Finally the Ethernet frame is sent over the physical [layer] network to a router. The router also has a protocol stack, but it consists only of layers 1–3 (routers generally do provide a configuration "application" layer but, for the purposes of this discussion, we'll consider their routing function only). As the router's data link layer checks the frame integrity, it strips off the layer 2 frame header (and footer) then passes the IP packet to the IP layer. The router analyzes the source and destination IP addresses (and potentially other IP header fields) to determine where next to route the packet. That is, the router is only an intermediate point on the way toward the ultimate destination, hence its protocol stack limitation to layers 1–3.

After the router determines where next to route the IP packet, it appends a modified IP header (note the differently shaded IP header in the figure—the source and destination IP addresses remain the same but at least one other field is changed, the Time to Live (TTL) field, which is decremented by each router along the path of the packet so that packets don't wander the Internet aimlessly forever). The router then passes the IP packet to the data link layer. The data link layer encapsulates the IP packet within a link layer frame appropriate to the corresponding link and physical layers over which the message will be routed. The router uses its source link layer (MAC) address based on its chosen outgoing interface and identifies the destination link layer address based on ARP or ARP

*Note that a single subnet address can be shared across multiple VLANs with the use of VLAN IDs in accordance with IEEE 802.1q.

cache mapping of the next hop IP address to its corresponding MAC address. Thus the router may receive the incoming packet on the left over a wireless link, then transmit it as modified appropriately over an Ethernet link.

Upon receiving the link layer frame, Device 2's protocol stack processes each successive layer, passing it up its stack to the intended application. Note that the data link and IP layer frames and packets respectively vary link-by-link. However, the transport layer message and application data itself are identical at both the sending and receiving ends of the connection. Of course, the goal is to transmit the application data intact, and the identical TCP/UDP layer message enables the intended end-to-end processing required of the transport layer.

We can now conclude that the router serves to terminate the data link layer or collision domain on the left side of the figure, only to modify the IP and MAC headers then forward the message to a second collision domain on the right side of the figure. For this reason, routers are also known as gateways, serving as gateways between layer 2 collision domains and IP networks. While switches utilize port VLAN configurations to separate collision domains at layer 2, routers use IP subnets to differentiate collision domains at layer 3.

Worldwide IP Communications. Let's extrapolate our view from Figure 1-8 from two devices interconnected via a single gateway to a scenario where I'm using my PC at home over my broadband connection to access a web server halfway around the world, as shown in Figure 1-9. When I browse the web site hosted on the server, the site address I type or click is captured and formulated into IP packets that are then sent via my broadband router to my service provider's router, Router A.

This transmission follows the same process we reviewed as shown on the left half of Figure 1-8. Router A then routes my data packet via additional intervening routers through the Internet to the ultimate destination, served by Router G. On each link along the way, each router terminates the layer 2 frame and IP packet, determines where next to route it (next hop IP address from a particular outgoing interface), and formulates a corresponding frame for transmission to

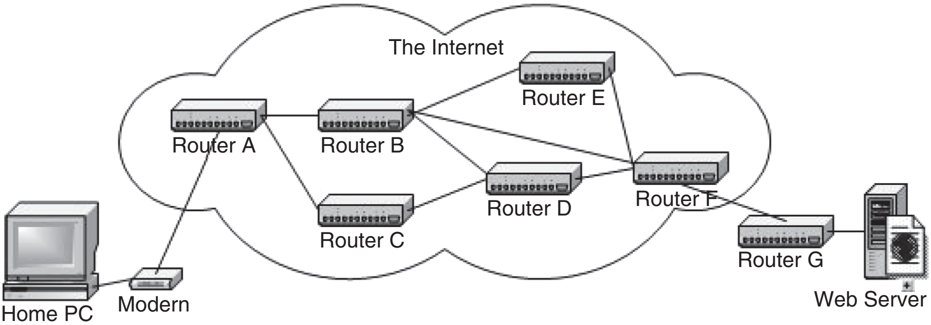


Figure 1-9: Simple view of an IP network such as the Internet

the next hop. Notice that there are multiple paths from my PC to the web server in Figure 1-9. One of my packets may travel from end to end over path A-B-E-F-G, while the next packet may take a different path, say A-C-D-F-G, and so on. Each IP packet is routed independently through the network.

Contrast this with an old-fashioned circuit-switched telephone call that temporarily "nails up" a dedicated connection through the traditional telephone network from my phone to the phone of the person I'm calling. Since our voice conversation requires set-up of a connection that comprises a physical path through the telephone network for the duration of our discussion, this type of connection is referred to as "connection-oriented." At the IP layer, IP does not establish a connection prior to communicating, and each IP packet is routed independently: all packets may happen to follow a common end-to-end path, they may all take different paths, or more likely, somewhere between these two extremes. IP is therefore considered "connectionless."

Connection-oriented communications generally provide a more reliable method of communications at the expense of tying up network resources and not dynamically rerouting around intermediate failure points during the connection session. The term *reliable* in this context means that there are means to detect and recover from dropped packets or packet fragments. "Pardon me?" usually works during a voice conversation and certain protocols such as TCP include an analogous construct with positive acknowledgment. While IP itself is considered an *unreliable datagram service*, the transport layer above IP, namely TCP, can be used to overlay connection-oriented controls to provide reliable communications between two devices or hosts. Without nailing up a physical connection, TCP provides mechanisms to properly order incoming IP packets and to request retransmission of IP packets should one or more get lost along the way. UDP is an alternative connectionless transport layer protocol within the TCP/IP protocol suite that provides unreliable data delivery. Conversely, a pseudo-layer 2 technology, multi-protocol label switching, MPLS, can be used to logically "nail up" a connection path over which IP packets can be transmitted between two endpoints. And finally, an IP static route could be configured along the path of intervening routers to deterministically route packets. Static routing emulates a connection-oriented session, at the expense of the connectionless rerouting advantage.

Dynamic Routing. The advantage of employing a connectionless scheme for IP is that a link outage between routers in the path of communications can be automatically detected and packets rerouted, keeping the lines of communication open. If you've ever been on a phone call and the call dropped, you've experienced this disadvantage of connection-oriented communications: if a link along the path fails, the entire session fails. Connectionless communications provides automated routing around outages, and this resilience was, in fact, one of the key design goals of the Internet Protocol.

Each router along the communications path of an IP packet examines the destination IP address in the IP header to identify whether it directly serves the network on which the destination IP address resides, or failing that, to which router to forward the packet that is "closer" to the ultimate destination. Each router consults an internal *routing table*, which stores information about where next to route packets destined for various IP networks. The routing table within each router governs routing decisions on each incoming IP packet and generally indicates one or more *next hops* in the path for a given destination network. A next hop is another router to which a given router can forward the packet directly. That is, the next hop is an adjacent or directly connected router, which itself may be directly connected or be multiple (hopefully fewer) hops away from the destination. In this way, each router need not be aware of every other router on the Internet; instead a given router must simply know where next among its directly connected peers to send a packet to get it closer to its ultimate destination.

A *routing protocol* is used by each router to periodically communicate with its neighboring routers to obtain their current routing and reachability information to keep routing tables up-to-date. Hence, dynamic routing makes use of recently updated routing information to make next-hop routing decisions. If a link or router fails, reachability changes will be detected and updated reachability metrics will ripple through the routing infrastructure via the chosen routing protocol. Routing protocols define the format and rules governing this "background" communication among routers, which enable each to maintain its routing table with the latest reachability information.

For example, considering Figure 1-9, Router B will receive advertisements of reachability to the network on which the web server resides from Routers D, E, and F. None of these routers directly serves this network but they offer an intermediate path. Using a simple hop count distance metric, Router F advertises a hop count of 2, while Routers D and E advertise a hop count of 3. Presumably the chosen next hop will be closer to the intended destination, that is, Router F, though other factors such as packet traffic congestion may come into play. More sophisticated metrics beyond hop count are now taken into account with modern routing protocols. Upon receiving the packet from Router B, the chosen next hop router then performs the same basic algorithm to determine whether it directly serves the IP network or if the packet must be sent on to another router, for example, Router G. Ultimately, the packet should reach a router serving the intended destination IP address for delivery.

Two types of routing protocols are generally used by an organization. *Interior* routing protocols enable routers within an organization to communicate subnet reachability. Popular interior routing protocols include Routing Information Protocol (RIP/RIP-2), Enhanced Interior Gateway Routing Protocol (EIGRP from Cisco Systems Inc.), and Open Shortest Path First (OSPF). *Exterior* routing protocols, such as Intermediate System to Intermediate System Protocol (IS-IS) or Border Gateway Protocol (BGP), enable updating of reachability and metric information across organizations or routing domains. Reachability to my network is communicated by my routers (or my Internet Service Provider) using an *exterior* routing protocol. BGP is the de facto Internet standard exterior routing protocol.

Organizations typically run an interior routing protocol on their internal routing network and run BGP on their externally facing router interfaces, for example, those connected to their Internet Service Providers (ISPs). However, BGP is not necessary for organizations with a single ISP connection not providing downstream routing, say, to other organizations. For example, BGP would not be required for a small office with a single-ISP connection. Such an end user is considered single-homed in contrast to a multi-homed organization with multiple Internet connections to one or multiple ISPs. BGP summarizes reachability information for the organization, which is identified via an Autonomous System (AS) number. AS numbers are simply organizational identifiers and are distributed and managed by Regional Internet Registries (see Chapter 2) to uniquely identify an organization or more accurately, a routing domain.

What is meant by the statement that "BGP summarizes reachability information"? When communicating (technically termed *advertising*) subnet reachability to the Internet using an exterior routing protocol, the routers do not list every subnet by hop number. This would create a massive amount of messaging overhead. A process called *summarization* or *aggregation* enables the communication of a single network address on the Internet for each such contiguous set of IP addresses. This is kind of like routing letters to a zip code distribution center. The center at the other end of the country needs only route to the destination postal code center and then allow that center to perform local delivery. Likewise, summarization enables routers to identify a contiguous set of IP addresses as a single network address along with its relative proximity or distance instead of communicating such reachability for every single IP address or subnet. Typically, an organization obtains a set of Internet addresses from an Internet Registry or ISP and simply advertises reachability to all such addresses by network address. It's then up to the organization to carve up this network allocation internally for Internet reachability as needed. This block allocation process is one of the key processes of IP address management as we'll discuss in detail in Chapter 6. A similar summarization process is utilized by interior routing protocols as well.

Routers and Subnets. As routers serve as inter-link gateways and forward or route data based on layer 3 (IP) information, each link needs to be assigned a set of IP addresses, a subnet address. Each device on a given link will require an IP address from the corresponding subnet address associated with the link and will utilize ARP to identify a MAC address corresponding to the IP address to which to transmit IP packets over the link. The process of IP subnetting entails the partitioning of an IP network into contiguous sets or blocks of IP addresses which are then associated with each link or subnet. Based on the subnet plan, a key element of IP address management, routers can be provisioned accordingly.

Provisioning a subnet on a router is akin to the addition of a new housing development for postal delivery. Just as the postal system must be updated to reflect the newly available addresses, Internet routers must likewise be updated. Fortunately, in both cases, this is usually a simple process. In the postal case, most new neighborhoods fall within an existing postal code and as long as the rest of the "postal system" can continue to deliver letters to the postal distribution center serving this zip code, updating is limited to systems and personnel within the local center or zip code.

In the Internet world, each organization desiring to communicate on the Internet needs a set of Internet-unique IP addresses. An organization's set or block of IP addresses can be likened to a zip code. Any IP communications destined for devices within an organization's set of addresses are routed to the organization's routers, akin to zip code distribution centers. The organization's routers then handle "local delivery" within the organization. Hence, the addition of a subnet to a router's configuration affects only intra-organization routers, which need to be updated via the interior routing protocol to identify which router serves the new subnet.

Referring back to our example network in Figure 1-3(c), the Branch office router advertises direct reachability to the 10.1.1.0/24 network, not a listing of 0-255 IP addresses; this reduces the size of routing tables and update messages from 256 to 1 for each such subnet, thereby reducing overhead and improving router performance. And this is why we don't allow the Retail Store router to contend with the Branch office router for serving the IP address 10.1.1.52 when I'm at the Retail store for a day: the overhead in routing protocol messages would create needless traffic at the expense of end user productivity traffic. Overhead is also minimized by virtue of the fact that the router only analyzes up to the IP layer in the protocol stack. Without having to fully digest each frame, it is able to quickly discern where next to route the message.

In the case of our HQ networks, if these three networks were served by a single router, the router could communicate its proximity to these networks in one statement* instead of 3 × 256 = 768 for each IP address, or even just three, one for each network. Organizing your address space to promote this router aggregation is important to keeping routing protocol communications small to speed up communications of updates and routing outages while enabling scalability.

Within an organization, address space planning must consider address capacity needs for the IP device population in the context of the organization's routing topology. As we'll see in later chapters, alignment of address allocations with routing topology produces an efficient address plan that will minimize routing protocol update traffic and routing table sizes.

HIGHLIGHTS OF INTERNET PROTOCOL HISTORY

The Internet Protocol (IP) is the most widely deployed network layer protocol worldwide. Emerging from a U.S. government sponsored networking project for the U.S. Department of Defense begun in the 1960s, the TCP/IP suite was initially implemented in the early 1980s and has evolved and scaled to support

*An example of the sole summarized route in this case would be 10.0.0.0/22, though the mask size in this example network could vary. This will be discussed in detail in Chapter 2.

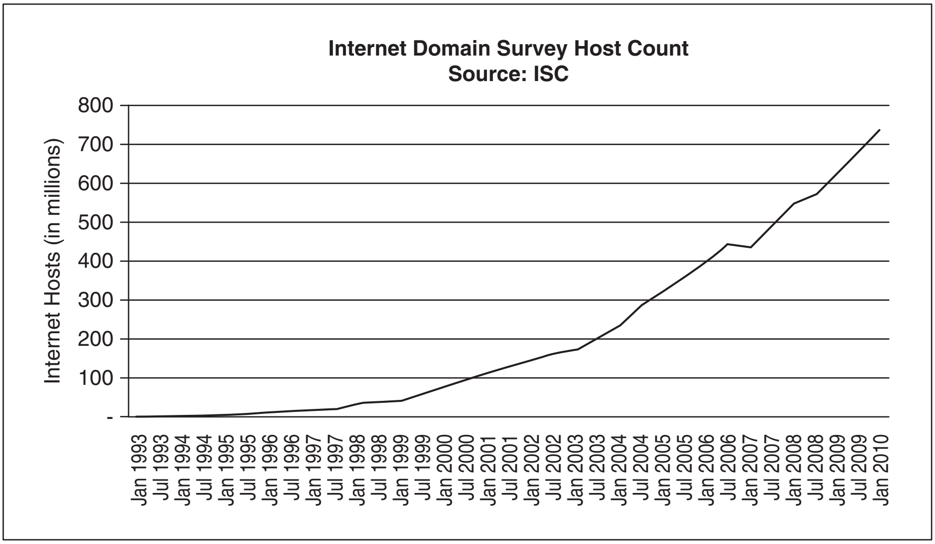


Figure 1-10: Growth of Internet hosts 1994–2009 (2)

networks from hundreds of computers to hundreds of millions today. In fact, according to Internet Systems Consortium (ISC) surveys (2), the number of devices or hosts on the Internet exceeded 730 million as of early 2010, with average annual additions of about 75 million hosts *per year* over each of the last five years (see Figure 1-10). The fact that the Internet has scaled rather seamlessly from a research project to a network of over 730 million computers is a testament to the vision of its developers and robustness of their underlying technology design.

The Internet Protocol was "initially" defined in 1980 in Request for Comments (RFC) 760 (3) and 791 (4), edited by the venerable Jon Postel. I've quoted "initially" because as Mr. Postel pointed out in his preface, RFC 791 is based on six earlier editions of the ARPA (Advanced Research Projects Agency, a U.S. Department of Defense agency) Internet Protocol, though it is referred to in the RFC as version 4 (IPv4). RFC 791 states that the Internet Protocol performs two basic functions: addressing and fragmentation. While this may appear to trivialize the many additional functions and features of the Internet Protocol implemented then and since, it actually highlights the importance of these two major topics for any protocol designer. Fragmentation deals with splitting a message into a number of IP packets so they can be transmitted over networks that have limited packet size constraints, and reassembly of packets at the destination in the proper order. Addressing is of course one of the key topics of this book, so assuring unique addressability of hosts requiring reachability is critical to basic protocol operation.

The Internet has become an indispensable tool for daily personal and business productivity with such applications as email, instant messaging, web browsing, and wireless, video, and voice communications. The Internet has indeed become a key element of modern society. The term "Internet," evolved from the lower case form of the term used by the early developers of Internet technology to refer to communications among interconnected networks or "internets."

Today, the capitalized "Internet" refers to the global Internet that we use on a daily basis, and it truly is a massive network of interconnected networks. Most enterprise networks also utilize the TCP/IP protocol stack, as do broadband, wireless, and landline service providers. Getting all of these networks and the hosts on them to cooperate and exchange user communications efficiently requires a robust set of rules for such communications. The Internet Protocol has proven remarkably robust, as well as extremely versatile and scalable.