1

Introduction

There is no Internet without the Internet Protocol (IP). IP sets the rules for communicating among Internet connected devices and serves as the foundation for every Internet app. IP is not only the protocol for the Internet but is the de facto enterprise network protocol as well. This chapter introduces the basics of IP networking and motivations for managing IP addresses within your own network.

IP Networking Overview

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 in response to error conditions. There are actually several protocols or protocol layers that facilitate 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 more into this later in this chapter, but let's start out with a simple analogy to human communications to introduce the key aspects of IP addressing and why address management is important for those who manage IP networks.

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 their mobile, sending him or her an instant message, and so on. In each of these scenarios, the initiator of the conversation identifies and locates the intended

IP Address Management, Second Edition. Michael Dooley and Timothy Rooney.

Published 2021 by John Wiley & Sons, Inc.

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 on-line 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 trite, 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 me 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!

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 a

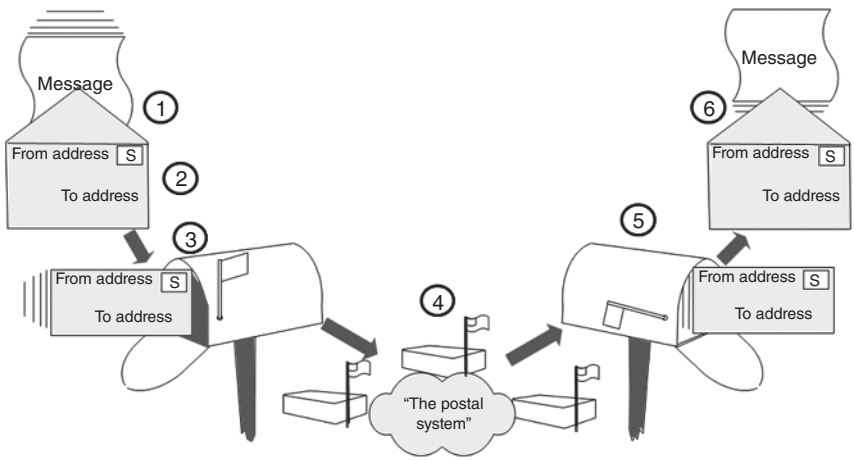

Figure 1.1 The postal delivery analogy.

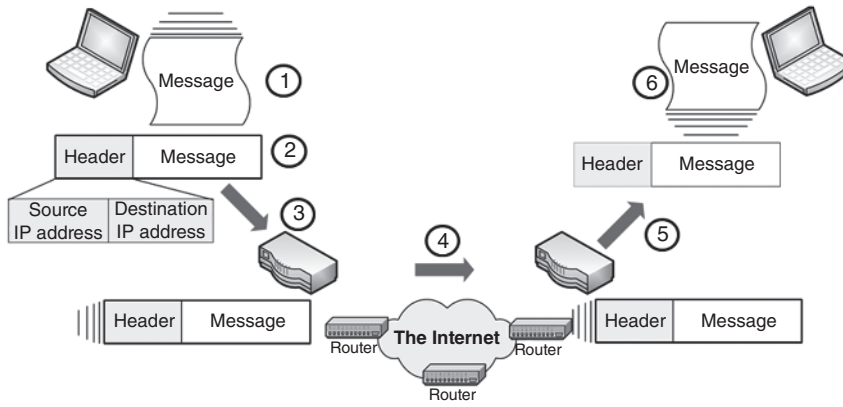

Figure 1.2 Internet protocol communications.

private enterprise, broadband, wireless, or 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 (including upper layer headers), 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 successively forward the message toward its addressed destination, and it is ultimately delivered to the recipient. The table below summarizes the key similarities among the postal and IP communications examples.

	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 an outgoing mailbox	Performed by transmitting the IP packet from the device 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)

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 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, e.g. 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. Your computer at home, your voice-over-IP phone at work, and your mobile all have IP addresses, at least at the time they're 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 mobile phones, which are highly mobile, the fixed address assignment approach does not work well. This is because 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 the left of Figure 1.3. To enable network communications among these sites, we interconnect them over a wide area network (WAN), 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 the right of Figure 1.3. 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 1.4. 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 both versions (IPv4 and internet protocol version 6 [IPv6]) of IP addresses in more detail in Chapter 2, but an IPv4 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 set 10.1.1.0–255 has been allocated to headquarters, 10.2.1.0–255 to the branch office, and 10.3.1.0–255 to the retail store. IPv6 addresses, also assigned to each location to support a dual-stack structure, are comprised of up to thirty-two hex

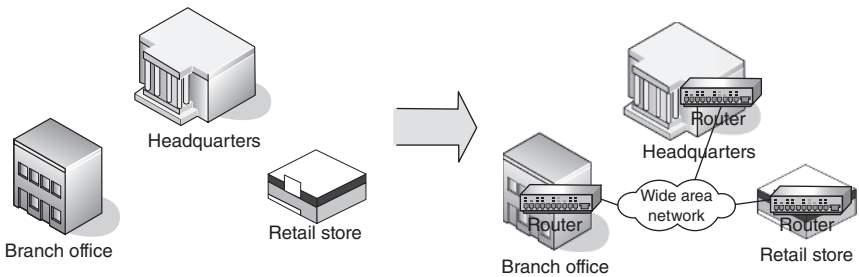

Figure 1.3 Organization's locations with IP network foundation.

digits, grouped into sets of four, and separated by colons, with strings of zeroes indicated by double colons. 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 IP addresses falls within a contiguous IP address range which 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 analogous to the postal distribution center which is used for efficient routing until the post or IP packet in this case is delivered to the local serving router.

The interconnecting WAN also has a network address to enable inter-site communications. The routers in each location must be configured with this network information in order to properly route IP communications traffic. In this way, our Branch Office router is responsible for IP addresses 10.2.1.0–10.2.1.255 and 2001:db8:0:2::-2001:db8:0:2:ffff:ffff:ffff:ffff, so all IP packets with a destination IP address falling in these ranges 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 1.4, I'll need an IP address from the branch office subnet, let's say 10.2.1.52 and 2001:db8:0:2::80e1:4d. This is because the Branch router "owns" the 10.2.1.0–255 and 2001:db8:0:2:: subnets 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 10.2.1.52 and 2001:db8:0:2::80e1:4d IP addresses, I will quickly realize

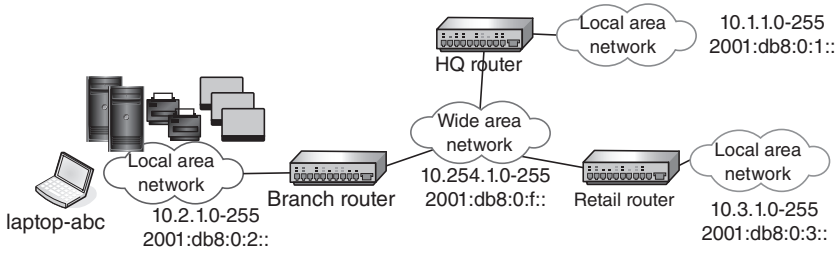

Figure 1.4 A more detailed IP network breakdown.

that I cannot communicate on the network. This is because my configured IP addresses are part of the branch office network, not the retail store networks or 10.3.1.0 and 2001:db8:0:3::. 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 website IP address, using my laptop's IP address, 10.2.1.52, as the source IP address. The web server acknowledges the communication and responds with the requested web page, addressing it back to IP address 10.2.1.52. The routers all "know" that the Branch Router serves IP addresses on the 10.2.1.0-255 subnet, so they route the response IP packet intended for my web browser back to the branch office!

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 quickly 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 10.3.1.187 or would you believe 2001:db8:0:3:7fe:d912:af99:3476, 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 bidirectionally 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 IPv4 and/or IPv6 address. Stateless address autoconfiguration (SLAAC) for IPv6 is another.

Once the address assignment process is automated, we've eliminated the error-prone human entry of device IP addresses. But what about the destination IP

address? 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 communications 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, websites, 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 websites and associated IP addresses. DNS works "behind the scenes" to provide a name-to-address lookup 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 simpler than entering (and remembering) https://192.0.2.201, let alone https://2001:db8:7e9:31a:ce00::90aa. 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 (IPAM), 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.

At this point, 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 inter-site 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, Wi-Fi, 5G, and so on. Figure 1.5 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. For example, a web browser, file transfer program, or email client are examples of applications.

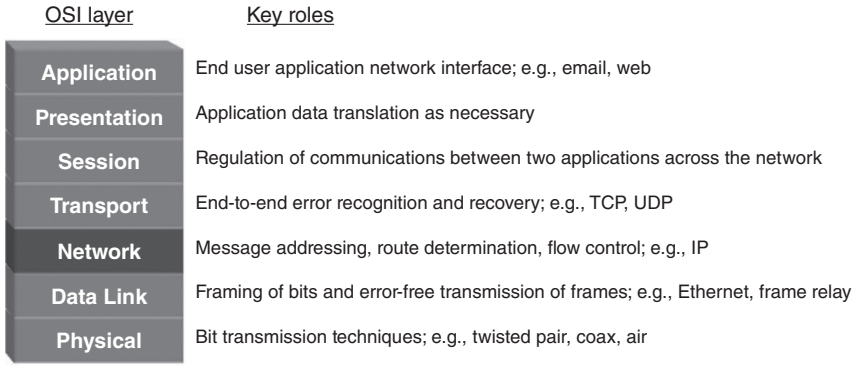

Figure 1.5 OSI protocol stack summary. Source: ISO/IEC [1].

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.

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, and Wi-Fi 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 but 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 1.5 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 1.6. Protocol layering enables not only the transmission of IP packets over a variety of media but 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 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, Wi-Fi, or other layer 2 protocol, and ultimately a particular physical layer. Another example illustrated in Figure 1.6 features hypertext transfer protocol (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

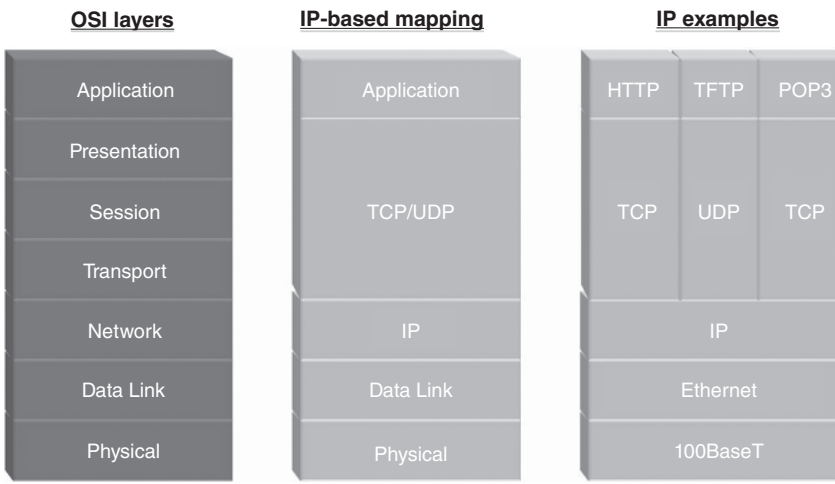

Figure 1.6 The OSI and TCP/IP protocol stacks.

common applications whether communicating over an Ethernet 1000BaseT network, an 802.11 wireless network, or an asynchronous transfer mode (ATM)-based 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.

TCP/UDP Ports

The "IP Examples" stack on the right of Figure 1.6 illustrates that a device may be running multiple applications, all reachable by its IP address. Once an IP packet has been successfully routed to the destination denoted by its IP address, the stack must deliver the packet to the correct application. Port or socket numbers for the source and destination are specified in the transport layer (TCP or UDP) header, which enable specification of a desired application. A given TCP/IP connection between two endpoints is defined by this four-tuple: {Source IP address, Source port, Destination IP address, and Destination port}. Sometimes the protocol, TCP or UDP is considered a fifth member of a five-tuple.

The source port is typically selected randomly by the sending device, while the destination port is defined by the corresponding application to which a connection is desired. Example "well-known" TCP/UDP ports include 53 for DNS, 67 for DHCP, 547 for DHCPv6, 80 for HTTP, and 443 for HTTPS. A device desiring to issue a DNS query would populate its source IP address, a random source UDP port, the DNS server's IP address within the Destination IP address field and port 53 as the destination port.

Intra-Link Communications

To get our arms around how data flows through these layers within protocol stacks, consider Figure 1.7. Beginning with the application data in the upper left of the figure, notice the addition of headers as the application data payload traverses down the stack. The data link frame that is transmitted on the local network

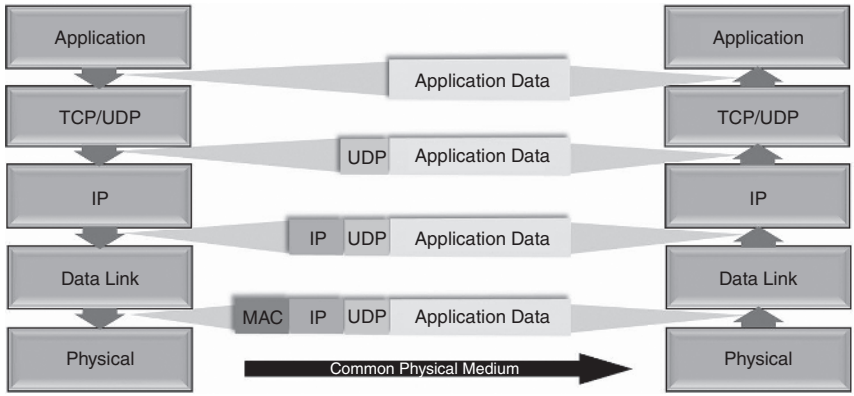

Figure 1.7 Protocol layering.

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 specifies the UDP or TCP header parameters, which consist of the application-specific port number as well as checksum, flow control, and other data. 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 or SLAAC), 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 an 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. 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 which 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 is typically the IP address of a router serving the subnet to which all outbound packets destined beyond the subnet are to be sent. This router is also referred to as the *default gateway*.

For example, as per Figure 1.4, if I use my laptop to transmit data to another device within the branch office, i.e. with destination IP address in the range of 10.1.1.0–10.1.1.255 or 2001:db8:0:2::, it can be sent directly (intra-link); if I transmit data to a device in HQ or the retail store, my laptop must send it to the branch router for routing to the destination via the WAN. 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 generally 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 its 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 the need for repeated ARP queries, e.g. for multi-frame or frequent 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 data link receive frames sent from every other connected device. 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 certainly offers 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 1.8 for an example of VLAN segmentation. The figure shows the common implementation of associating different VLANs with different subnet addresses. This is contrasted against the single broadcast domain per hub shown on the left of the figure.

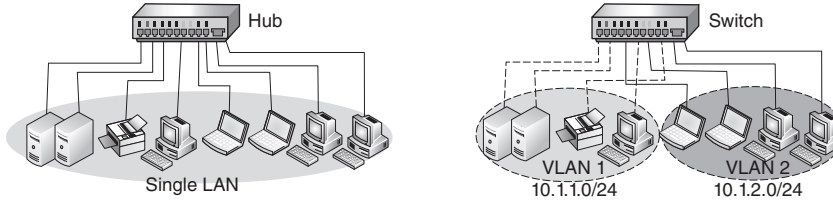

Figure 1.8 Hub vs. switch/VLAN architecture.

Interlink 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 1.9, which is a recasting of Figure 1.7 with an intervening router. The left side of the figure is identical to that in Figure 1.7. 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 the purposes of this discussion, we'll consider their routing functions only). As the router's data link layer checks the frame integrity, it strips off the layer 2 frame header (and footer if used) 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

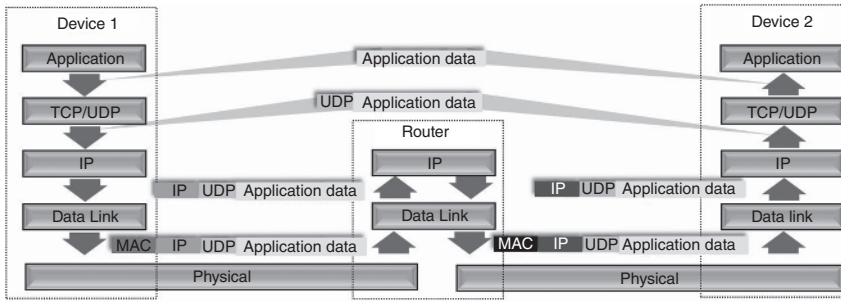

Figure 1.9 Protocol stacks with an intervening router.

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 transmitted. 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 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 1.9 from two devices interconnected via a single gateway to a scenario where I'm using my computer at home over my broadband connection to access a web server halfway around the world, as shown in Figure 1.10. When I browse the website hosted on the server, the site address I type or click is captured and formulated into IP packets (thanks in part to DNS) 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 1.9. 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 the next hop. Notice that there are multiple paths from my PC to the web server in Figure 1.10.

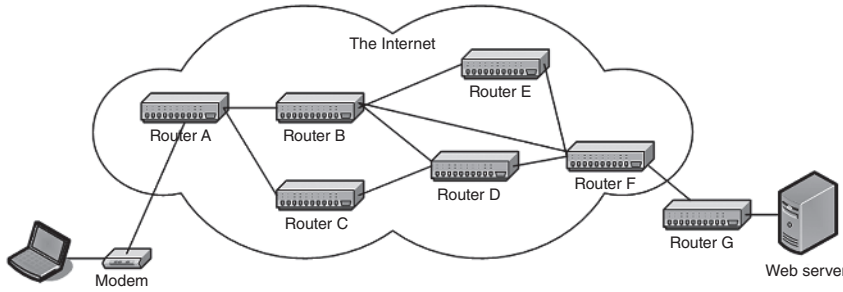

Figure 1.10 Simple view of an IP network such as the Internet.

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 setup of a connection which 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 acknowledgement. While IP itself is considered an unreliable datagram service, the transport layer above IP, namely the 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 which provides unreliable data delivery. Conversely, a pseudo-layer 2 technology such as multi-protocol label switching, MPLS, could 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 dynamic routing advantage.

Dynamic Routing

The connectionless scheme for IP enables a router to detect a break in the communications path and to re-route packets automatically through an alternate route, 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 provide 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 1.10, Router B will receive advertisements of reachability to the network on which the web server resides from Routers A, 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 and Router A a hop count of 4. Presumably, the chosen next hop will be closer to the intended destination, i.e. 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, e.g. 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. 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), which is far and away the most popular. Exterior routing protocols, such as intermediate system to intermediate system (IS-IS) protocol 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 [ISP]) using an exterior routing protocol. BGP is the de facto Internet standard exterior routing protocol.

Organizations typically run an interior routing protocol like OSPF on their internal routing network and BGP on their externally facing router interfaces, e.g. those connected to their *ISPs*. However, BGP is not necessary for organizations with a single ISP connection not providing downstream routing, e.g. 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 (RIRs)* to uniquely identify an organization or more accurately, a routing domain. We'll delve into the role of RIRs in Chapter 3.

What is meant by the statement that "BGP summarizes reachability information"? When communicating subnet reachability (known as *advertising*) 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 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 RIR 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 IPAM 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 packets based on layer 3 (IP) information, each link needs to be assigned a set of IP addresses, i.e., 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 generally 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 IPAM, 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 intraorganization 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 1.4, the Branch office router advertises direct reachability to the 10.2.1.0 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. Likewise, the advertisement of the 2001:db8:0:2:: network affords one route instead of 1.8×10^{19} individual IP addresses. And this is why we don't allow the Retail Store router to contend with the Branch office router for serving the IP address 10.2.1.52 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 three networks were served by a single router, the router could communicate its proximity to these networks in one statement instead of $3 \times 256 = 768$ for each IP address, or at worst 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. And 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.

Assigning IP addresses

The subnet and network plan serves as the foundation on which individual devices can obtain IP addresses(es) relevant to their respective points of connection. Tracking of address assignments is critical given that a duplicate IP address assignment would render communications impossible for both claimants to the duplicate IP address. While some devices like routers, switches, and servers may be manually configured with IP addresses, these addresses must be tracked to eliminate IP address duplications. For other devices such as end user devices, DHCP serves as the Internet standard protocol for server-based automated IP address assignment. DHCP is a client/server protocol which "leases" IP addresses to clients for a duration configurable by an administrator from a few seconds to forever.

SLAAC is an IPv6 feature which enables a device to determine the relevant subnet address based on information from the serving router to derive a full IPv6 address. Duplicate address detection (DAD) is a process whereby each IPv6 address, whether derived via SLAAC or leased from DHCP, is verified as unique on the subnet.

The Human Element

All of this discussion of IP addresses, subnets, routers, etc. is exciting for us and hopefully for you, dear reader, but what of your network users? They likely have little desire to even know about networking details underpinning their access to email and websites. Fortunately, thanks to the wonder of IP routing we've discussed in this chapter, IP packets can usually get from point A to point B on the Internet. And fortunately for end users, the DNS makes identifying point B as simple as typing a www address in a browser.

Often termed "the directory of the Internet" or similarly banal labels, DNS is essentially a distributed Internet database of information, commonly consisting of IP addresses associated with domain names. When users enter a web address into a browser, the browser initiates a DNS lookup for the entered web address in text form to retrieve its corresponding IP address, which the browser populates as the destination IP address within its IP packet to connect to the webserver. DNS certainly makes the Internet usable for humans, at least mere mortals not well versed in IPAM technologies!

Why Manage IP Space?

Given the necessity of users automatically obtaining relevant IP address assignments for their devices and easily navigating the Internet thanks to the DNS, network managers clearly should be monitoring, tracking, and configuring their DHCP and DNS services in conjunction with their overall IP address plans. The practice of IPAM entails the application of network management disciplines to IP address space and associated network services, namely DHCP and DNS. The linkages among an IP address plan and configurations of DHCP and DNS servers are inseparable. A change of an IP address will affect DNS information and perhaps DHCP as well. These services provide the foundation for today's converged services IP networks, which offer ad hoc anytime, anyplace communications.

If end user devices such as laptops or voice over IP (VoIP) phones cannot obtain an IP address via DHCP, they will be rendered unproductive and users will contact the help desk. Likewise, if DNS is improperly configured, application navigation by name, phone number, or web address will likewise impair productivity and induce help desk calls.

Effective IPAM practice is a key ingredient in an enterprise or service provider IP network management strategy. As such, IPAM addresses configuration, change control, auditing, reporting, monitoring, security, trouble resolution, and related functions as applied to the three foundational IPAM technologies:

1) *IP address subnetting and tracking* – maintenance of a cohesive IP address plan that promotes route summarization, maintains accurate IP address inventory, and provides an automated individual IP address assignment and tracking mechanism. This tracking of individual IP address assignments on each subnet includes those assigned by hard-coding, e.g. routers or servers, and others assigned dynamically, e.g. via DHCP or SLAAC.
2) *DHCP* – automated IP address and parameter assignment relevant to location and device type. This requires tracking address assignments configured on devices and setting aside dynamically allocated address pools. These address pools can be configured on DHCP servers in order to enable devices to request an IP address, and receive a location-relevant address in reply.
3) *DNS* – lookup or resolution of host names, e.g. www entries to IP addresses. This third key aspect of IPAM deals with simplifying IP communications for humans through the use of names, and the mapped IP addresses must be consistent with the IP address plan.

Basic IPAM Approaches

Early History

With the growth in prominence of the use of TCP/IP within enterprise networks and for service provider Internet offerings starting in the mid-1990s, organizations initially managed the three cornerstones of IPAM independently. Smaller organizations maintained a paper log, spreadsheet, or in-house database for tracking IP address space and subnet assignments. DHCP and DNS configuration files for the few DHCP and DNS servers operating on the network were generally configured manually using text editors. Larger organizations built home-grown systems or procured commercial software solutions for all or portions of these three areas to provide some degree of automation and consistency among these key areas. The focus historically had been on simply providing a repository of information for IP address tracking and usually for some level of DHCP and DNS configuration and tracking.

Most currently available IPAM tools provide a repository and, in some cases, automated creation of DHCP and DNS configuration information. Indeed, there are many tools available in the market today, each providing varying levels of integration and functionality. Thus, some organizations continue to use spreadsheets or databases for IP inventory, while utilizing a software tool with a graphical user interface (GUI) for DHCP and/or DNS management. Even the use of spreadsheets or databases has historically proven adequate for "first generation" monolithic IP networks.

Today's IP Networks and IP Management Challenges

Today, many organizations have deployed wireless, VoIP, unified communications services, cloud-based infrastructure, Internet of Things (IoT) devices and more IP service offerings are continually emerging. These organizations commonly configure their routers to provide a differentiated "class of service" to certain devices like VoIP devices than to data devices such as laptops or PCs. This translates to IP packets for voice traffic garnering higher priority queuing than data traffic for example. The delay sensitivity requirements of voice communications often necessitate such a configuration. For example, if my email takes an additional 30 seconds to reach my email server, I would rarely even notice. But if portions of my voice conversation are delayed by even half a second, the communication will be rendered totally useless. But how can each router distinguish VoIP vs. data traffic? As we discussed earlier in this chapter, routers can use information in the IP header to differentiate these packets, including in some cases the source IP address field. Considering the use of the source IP address field, the routers must be configured with which source IP addresses represent VoIP phones and which represent data devices. Of course, this maps back directly to the IP address plan afforded by the IP inventory and assignment aspect of IPAM.

This address segmentation may require the organization to partition its address space essentially into parallel address spaces – parallel in that they each service the same physical networks, albeit for different applications. They could carve out a portion of the data network for allocation for VoIP devices. This slice of IP address space must then be reflected in the router configurations for packet processing and in each subnet providing VoIP and data services to end users. In addition, VoIP phones utilize DHCP to obtain an IP address and configuration, so configuration of the data and VoIP address pools within DHCP servers in accordance with the address segmentation plan is required.

Along with the pool configuration, the DHCP server must be configured to recognize a VoIP phone from a data device while deciding what address to assign. This is generally performed using a construct referred to as client classes. Finally, many organizations desire to have VoIP devices reside on a different DNS domain for administrative purposes. Whether on a separate domain or one common to other devices on the subnet, generally DNS requires updating the relevant name to address information and possibly even telephone numbers and other device or application specific information as well.

The upshot of this discussion is that the introduction of a new IP service, such as VoIP, introduces a substantial ripple effect on how IP address space is managed. It affects IP inventory, in tracking and allocating what amounts to a parallel address space across the organization's sites, as well as DHCP and DNS configurations. While the overall address space within the organization may not have

changed, the complexity and granularity of configuration required has essentially doubled. The introduction of additional IP services requiring "special treatment" in address assignment and configuration, such as video conferencing, would stimulate the requirement for a third parallel address space and the further increasing of management complexity.

Beyond the management complexity around the three basic IPAM cornerstones, an additional set of requirements has arisen. As more IP services are deployed, it's easy to see that the reliance on the IP network grows linearly if not exponentially. With more intraorganizational and even interorganizational communications relying on the integrity, performance, and availability of the IP network, as it extends into the cloud and perhaps into the field with IoT devices, the exigency to effectively manage it grows. With increased reliance comes increased risk to your organization should portions of your IPAM foundation degrade or fail. Securing your IPAM helps secure the foundation of your network. And you can employ several IPAM techniques to enhance to overall security of your network.

IPAM is a fundamental component of your overall IP network management strategy. The increasing dependence of your business on your IP network increases the importance of your IPAM processes, which must be reliable, highly available, accurate, secure, and ideally integrated into the broader IP network management processes and systems.