11 UNIX in Internet and Computer Networking

In previous chapters, the local applications and services of UNIX in individual computers have been introduced. This chapter will present the remote and network functions and services of UNIX in servers and workstations. After explanation of general concepts about the Internet and computer networking, Transmission Control Protocol/Internet Protocol (TCP/IP) suit will be presented, and contents associated with layers of TCP/IP model from down to up will be discussed. Finally, we will explain how to use some commands in the application layer of the TCP/IP model, such as telnet, ping, ftp, etc.

11.1 UNIX's Contribution to Development of Computer Networking

In the reality, the UNIX operating system is widely applied in servers and workstations, and have many original contributions to the development history of the computer networking and Internet. With the background information of computer networking, readers should have known that at the very beginning, the development of computer networking was funded by Advanced Research Projects Agency (ARPA, also named Defense Advanced Research Projects Agency, DARPA) in 1960s. This project resulted in the ARPANet, in which the UNIX time-sharing system allegedly performed well as an ARPA network mini-host according to RFC 681 (Request for Comments, which will be introduced in the following section) (Holmgren 1975). Within the document of RFC 681, it is detailed explained why UNIX was chosen as the operating system of an ARPANet mini-host. One reason is that UNIX has many beneficial features, among which are the network control program (NCP) integrated in the UNIX kernel and network connections accessible through UNIX I/O standard system calls. In the ARPANet, the first connected nodes included UCLA, Stanford University, University of California at Santa Barbara, and the University of Utah. In 1980s, BSD added the TCP/IP network function to the UNIX kernel, which made the Internet infrastructure expand

Y. Liu et al., *UNIX Operating System*
© Higher Education Press, Beijing and Springer-Verlag Berlin Heidelberg 2011

TCP/IP networks into not only the U.S. military and academic institutions but also commercial services.

Since most of the networking protocols were initially implemented on UNIX and most of the Internet services are provided by server processes running on the UNIX operating system, UNIX has a fundamental and profound influence on computer networking.

11.2 General Concepts of Computer Networks and Internet

As Computer Networking is also one of the most important disciplines in computer science and technology and there are many published books specialized to the Computer Networking topics, this chapter will present some general concepts of the computer networks and Internet, which are close to or involved with operating systems, and the detailed information about Computer Networking should be referred to the special books listed at the end of this chapter (Comer 1998; Comer et al 1998; Stevens 2002; Wright et al 2002). And having the knowledge of Computer Networking will be helpful for readers to understand this chapter.

11.2.1 Request for Comments

In the late 1960s, the first Request for Comments (RFC) was used by authors in the ARPA-funded projects to take notes of their research findings and circulated their notes with other ARPA researchers. In modern computer network engineering, the RFCs are memos published by the Internet Engineering Task Force (IETF) and the Internet Architecture Board (IAB) in order to describe research findings and achievements that can be applied to the computer network systems and Internet. Today, RFCs are the official publications that can help exchange information among the global computer network researchers.

As their main goals were requests for comments, the early RFCs were different from the modern RFCs, written not necessarily in a formal way and often left questions open for other researchers' extension. Nowadays, this type of writing style can be used for Internet Draft documents before being approved and revised as an RFC. The RFCs that are adopted by the IETF from proposals can be published as Internet standards while the RFCs can also be used for computer scientists and engineers in the Internet Society to publish their latest research achievements. Readers can refer to the RFCs through the Internet. Some of them are listed in the references of this chapter.

Another benefit of RFCs is that they are standards directly from the experience and practices of computer scientists and engineers who have been doing researches on computer networking by themselves. For this reason, these standards are operational and pragmatic. It is also the reason why compared to the ISO model, the TCP/IP model is viable and popular in the computer world.

11.2.2 Computer Networks and Internet

Before the advent of computer networks, communication between computers and even between the computing system and I/O systems was relied on human beings moving storage media (usually tapes) between them. Today, computer networks are the core of modern communication. The following advantages that computer networks have brought to the modern world may be so familiar for us to take them for granted.

- To share computer resources among different users. Users in a computer network can share printers, disks, files, and even CPUs.
- To make cooperation of researchers in different locations to develop one big project. Researchers with different talents in different cities can work together for one project.
- To enhance the reliability of a whole computer system. If one computer in a network is broken, other computers in the same network can undertake its work.

As a reader, you can count more items of the advantages. And even more, just to image the situation if without the computer networks and Internet, you will realize how consequential their seeping effect on our real activities is.

As known, computer networking supports the communication between computer systems in or between computer networks. Computer networks are formed by connecting two or more computer hardware resources which can be computers, printers, scanners, screens, etc. If given an intuitive definition, a computer network is a group of computers or devices connected to each other in order to exchange data, and computer networking is to connect different computers with devices and media in order to make them communicate each other. In computer networking, a host can be a computer holding information resources as well as application software for providing network services; and routers or gateways are dedicated computers that are responsible to connect other computers or networks. According to their scales, there are three types of computer networks: LAN, MAN, and WAN:

- Local area network (LAN), a small network is located in a small geographic area, such as a building or a campus, and provides services to a small group of people who belong to one community. LANs can adopt a peer-to-peer or client-server networking model, which will be discussed in the following section.

- Metropolitan area network (MAN), a medium-sized network usually covers a city. When many LANs over a specific geographical area (for example a city) are integrated into one larger network, a MAN for that area is built up. Over a MAN, the flow volume of communications increases with the network growth.
- Wide area network (WAN), a large network often crosses different cities, which is a network composed of numerous smaller networks with a wide variety of resources. The Internet is the giant WAN. Through the Internet, a multi-national corporation can build up a WAN to interconnect their branch offices in different countries. Also, a wide variety of technologies involve in the communications in WANs, for example, Asynchronous Transfer Mode (ATM), Frame Relay, Point-to-Point Protocol (PPP), and Synchronous Optical Network (SONET). These technologies are distinguished by their switching capabilities and data transmission speeds.

As the technologies in telecommunications, computer science and computer engineering developing, the communication media in the computer networking are not only a variety of tangible wires (such as coaxial cable, twisted-pair copper wire cable, power lines, and optical fiber) but also wireless technologies. Therefore, the Wireless LANs and WANs are alternatives for LANs and WANs, which make computer networking mobile. A wireless LAN or WAN is almost the same as a LAN or WAN, except without wires between computers or devices. The data are transmitted via radio transceivers. Communications within large areas can be performed through communications satellites, cellular network, wireless local area network (WLAN, such as Wi-Fi) or Wireless local loop. According to the capabilities of antennas, Wireless LANs or WANs can cover areas ranged from hundreds of meters to a few kilometers. And Bluetooth makes the devices within a few meters communicate with each other wirelessly. Wireless technologies can save the cost and inconvenience of laying cables.

According to the geographic reference to a network, networks can also be divided into three types of networks: Intranet, Extranet, and Internet.

- Intranet, a network is usually for trust communications inside an organization, university, or enterprise. It is only accessible by authorized users, such as the faculty of the university or employees of the enterprise. For security, Intranets are generally limited to connect to the Internet.
- Extranet, an outside extension of an intranet allows secure communications to users outside an organization, university, or enterprise, such as the visitors of the university or the customers of the enterprise. With the Virtual Private Network (VPN) technology, Intranets and Extranets can be securely joined the Internet and escape the access of general Internet users.
- Internet, a generalized inter-network that is a ubiquitous network of networks. In other words, The Internet is a platform where a set of Internet Service providers and consumers are interconnected together and provide them the Internet access. The Internet usually includes end users, enter-

prises, businesses, and organizations interconnected by Internet Service Providers (ISP, also referred as Internet Access Providers, IAP), which are companies that offers its customers access to the Internet. The Internet Services include email accounts, remote data files storage and transmission, on-line chatting, commerce transactions, etc.

Along with electronic commerce development, commerce transactions can be done over the Internet via the communication security mechanism. The typical communication styles include business-to-business (B2B), business-to-consumer (B2C), and consumer-to-consumer (C2C). Along with the social needs' evolving, communication types extend to consumer-to-business (C2B), government-to-business (G2B), business-to-government (B2G), etc. Here give the descriptions of business-to-business, business-to-consumer and consumer-to-consumer.

- Business-to-business (B2B), the communication for electronic commerce transactions between businesses or enterprises. It can be between a manufacturer and a wholesaler, or between a wholesaler and a retailer.
- Business-to-consumer (B2C), the communication for the electronic commerce activities of businesses' serving end consumers with products or services. In a supply chain, there can be several B2B transactions (typically one from a manufacturer to a wholesaler, and one from a wholesaler to a retailer) but only one B2C transaction (one from a retailer to an end customer).
- Consumer-to-consumer (C2C), the communication for the electronic commerce transactions between consumers through the third party. The popular online auction is a C2C example. eBay is an example of the third party for C2C.

These communication models need the support of the communication security mechanism over the Internet and the services of networking operating systems. As UNIX usually works in servers, these issues have been considered in its development. And in the following sections, the services will be discussed gradually.

11.2.3 Client-server vs Peer-to-peer Models

In the computer networking, there are two types of networking models, the client-server and peer-to-peer models, both of which are based-on distributed computing.

11.2.3.1 Client-server Model

In a network with the client-server model, every client (usually a computer) is connected to the server (usually a high-performance host) and also each other. The resources in a server can be shared with clients, but a client does not share any of its resources with servers and other clients. A server computer executes one or more server programs waiting for clients' requests. A client initiates communication with a server for some service request. For one instance of a client program, a client can send a data request to one or more servers connected to it. In turn, the corresponding servers can receive the request, process it, and return the requested data to the client. A web browser is a typical client program through which a user can access a web server on the Internet. For instance, if a student wants to access remotely the library database server in a university to enquire some author's articles, he may first access the web server of the university library through a web browser from his computer and send a request to the web server. The server program may send the request to its own database client program that in turn forwards a request to a database server in the campus to retrieve the information of the articles. The information is then sent back to the database client, which in turn returned it to the web browser client displaying the enquired results on the screen of the student.

In addition to the web browser, the client-server model has a wide variety of applications. The main application protocols of the Internet, such as Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), Telnet (Telnet remote Protocol), and DNS (Domain Name System) (protocols will be discussed in the later sections), adopt this model. With these protocols, functions such as web access, email transfer, and database access, can be built. The corresponding servers include web, mail, ftp, name, file, database, application, print, and terminal servers; and the clients include web browsers, email clients, chat clients, and so on. Some servers can undertake just one task like a web server and play the role of single-service servers while other servers may perform the integration of more than one task, such as the combination of ftp server and file server, or the composition of the computing function and database service, dependent on the capacity of the hardware and the correlation between the tasks. However, as a user of a computer network or the Internet, a connected computer can be installed with different client programs and initiate each of them when needed. Although the client-server model can be used in a variety of applications, the architecture is basically the same.

The client-server model has its advantages and disadvantages. The advantages are:

- Its architecture is maintainable when the volume of servers and clients change. As the roles of clients and servers are concise, their responsibilities are also clear. Therefore, it is easy to replace, repair, and add a new service to it without a high impact on the main part of the network.
- The security cost is low. As all data storage is concentrated in the servers, the security can be controlled more easily than the data distributed in different computers.
- The system update is convenient. Also as all data storage is centralized in the servers and the roles in the model are definite, the data update can be maintained just in the servers and the function update can be done in

the clients or servers relied on the division of responsibilities.

The disadvantages of the client-server model are:

- The bottleneck for traffic on the client-server model can be caused at the server when the volume of client requests increases rapidly. If a large number of client requests are initiated at the same time, the server can be so overloaded that the response from the server may be sluggish and even fail.
- The reliability of the client-server model is lower than the peer-to-peer model. If the server without the backup system is disrupted, the service can be broken off and even all the data can be lost.
- The scalability of the client-server model is less than the peer-to-peer model. As the capability of a central server is limited, the extension of the whole network can be restrained.

11.2.3.2 Peer-to-peer Model

In a network with the peer-to-peer (abbreviated to P2P) model, the resources of each computer peer can be shared with other computer peers. The shared resources can be files, disk storage, network bandwidth, and CPUs. In the peer-to-peer architecture, there is not a host that works as the central coordination server. On the contrary to the client-server model where servers provide resources or services of which clients make use, the status of each computer peer in the peer-to-peer model is equal to others' and can not only provide its resources to others but also make use of the resources of others.

File sharing has made the peer-to-peer model popular and is also criticized drastically for the copyright protection. In file sharing systems (such as Napster, Freenet, and Gnutella), users can connect their computers to a peer-to-peer network with some software and search for shared files on other computer peers that are in the network as well. Files of interest can then be transferred directly from other computer peers. To speed up the transference, large files can be decomposed into smaller parts, obtained from different peers simultaneously, and then reassembled after transference. At the same time, one peer can also upload the parts it has received to other peers.

Beside the file sharing, there are many other applications of a peer-to-peer model. A peer-to-peer network can be used for indexing and resource discovery. For example, since resources in a network are manifold, a peer-to-peer model can be applied to fulfill the efficient resource discovery for grid computing systems, which usually undertake the responsibility to manage resources and schedule applications. Resource discovery involves searching for the suitable resource types to match the application demands of the user.

A network with the peer-to-peer model is usually implemented in the application layer of TCP/IP and runs over the underlying Internet Protocol (IP) network (the later sections will give detailed discussion on TCP/IP and IP).

The peer-to-peer model and client-server model can be merged into one network, which is usually called a hybrid peer-to-peer architecture. There are two types of hybrid peer-to-peer networks: one divides their clients into two groups – client nodes and overlay nodes – in which overlay nodes are used to coordinate the peer-to-peer structure; the other has central servers to index the peers and direct the traffic among the peers.

Compared to the client-server networks, peer-to-peer networks have their advantages and disadvantages, too. The advantages are:

- The reliability of the peer-to-peer model is higher than the client-server model. As peers in a peer-to-peer model have equivalent status and responsibility, if one peer shuts down, the rest of the peers are still capable to communicate each other.
- The requests on the peer-to-peer model are settled in balance. As it can distribute data among peers and there is not a central server in it, the bottleneck for traffic can be prevented in the peer-to-peer model.
- The resources of the peers in a peer-to-peer model can be utilized effectively to enhance the performance of the whole network. As the peers in a peer-to-peer model take both roles of providers and consumers of resources, each of them can contribute their parts to a cooperating execution.
- The peer-to-peer model can be scalable. As a peer-to-peer model is usually formed dynamically, the addition or removal of peers brings no significant influence to the whole network.

The disadvantages of the peer-to-peer networks are:

- The security of the peer-to-peer model is lower than a client-server model. As it is easy and convenient for a user to engage in the peers of a peer-to-peer model, the insecurity can also be brought easily into the network.
- The existence time of a peer-to-peer network is largely dependent on the peer loyalty and engagement. Since a user can join to a peer-to-peer network as their wish, peers may give up the network when it loses the attraction.
- The reliance on the common interest among the peers can restrain the applications of a peer-to-peer model. Because the effectiveness and efficiency of the peer-to-peer model come from the peers' cooperation, the search for some rare topic may fail in a peer-to-peer model.

Both client-server and peer-to-peer architectures are widely used today. In a workgroup on the same LAN, users can share printers and database servers via the client-server model. With common interests, users located arbitrarily can share a set of servers over the Internet through the client-server model and also share resources via the peer-to-peer software.

11.2.4 TCP/IP and ISO models

In computer networking, networks can be treated in a physical or logical perspective. In a physical perspective, just like the above discussion, it involves geographic locations, physical connection, and the network elements that are connected via varied media. As the technologies in the computer, telecommunication, and other relevant fields have developed so quickly in these decades, the technologies involved in the computer networking and Internet are diversified and complicated. However, no matter how complex and variant the technologies become, there are always some basic logical principles to support them. And logical networks in the TCP/IP and Open Systems Interconnection (OSI) models can map onto one or more physical media. In this section, it will be introduced how to view the computer networking in a logical way.

When considering the layout of networks, the below protocols (or rules) and facilities should be followed and established:

- The protocols are to implement the detail of software for the particular applications such as telnet, ftp, etc.
- The protocols are to fulfill the application data transportation between two processes over a network.
- The protocols are to carry out routing the application data between hosts over a network.
- The protocols are to execute access to the physical transmission medium for a host's data transmission on a network.
- The type of network topology, such as bus, ring, star, tree, or mesh, is adopted as the infrastructure layout of a network.
- The type of physical communication medium, such as coaxial cable, twisted-pair copper wire cable, power lines, optical fiber, or wireless technologies, is chosen as the connection between the resources over a network.

There are two popular models that are often referred in the computer networking; they are the Open Systems Interconnection (OSI) Reference Model and the TCP/IP Model (or Internet Protocol Model). The OSI model was recommended by the International Standards Organization in 1981 and the TCP/IP model was started by the Department of Defense Advanced Research Projects Agency (ARPA), which has been mentioned at the very beginning of this chapter, in the late 1960s.

The OSI model is composed of seven layers, which are Application, Presentation, Session, Transport, Network, Data Link, and Physical layers from top to bottom. Each layer focuses on its own task that can be mapped onto one of the above six protocols and facilities. Typically, the application, presentation, and session layers undertake all the detailed tasks of application software. In detail, the application layer defines the medium with which application software communicates to other computers; the presentation layer defines file formats (such as binary, gif, jpeg, and text) in order to display a file in an appropriate way; the session layer is to initiate, control, and end communications. The transport layer takes the responsibility of the application data transportation via a network. The network layer realizes routing between hosts via a network. The data link layer carries out access to the transmission medium for the data transmission on a network. The physical layer combines the tasks of selection and layout of the types of network topology and physical communication medium.

The TCP/IP model consists of four layers, which are Application, Transport, Network, and Link layers, shown in Figure 11.1. Each layer in this model can be also mapped onto one or more items of the above six items of protocols and facilities. The application layer carries out the tasks of application protocols. The transport layer undertakes the tasks of transmission protocols. The network layer realizes the missions of the routing protocols. And the link layer settles all the issues including the access to the transmission medium and the construction of communication device and network infrastructure, which are mostly involved in hardware control on the data flow from the origin host to the destination host over a network. In this four-layer model, the application layer focuses on the details of the applications without considering the communication while the other three layers do not care about the application but concentrate on how to move the data across the network.

As it is applied in the Internet, the TCP/IP model will be focused on in the following sections.

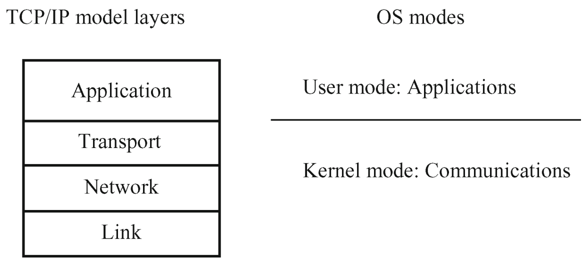


Fig. 11.1 TCP/IP model layers with process execution modes in OS.

11.2.5 TCP/IP Protocol Suite

Each layer of the TCP/IP model should fulfill the task that is defined by the protocols and facilities and in addition, there are usually several protocols that are associated with one of the layers in the TCP/IP model. These protocols make up the TCP/IP suite (commonly called the Internet Protocol Suite). Figure 11.2 shows the popular protocols in the four layers in the TCP/IP model. Some of them can implement the task of one layer, such as TCP and User Datagram Protocol (UDP), which can fulfill the mission in the transport layer; some of them can only practice part of the task of one layer, which means that they need some other protocol or facility to cooperate with to fulfill the task of their corresponding layer, such as Address Resolution

Protocol (ARP) and Reverse Address Resolution Protocol (RARP).

In this section, it will give general concepts about the protocols of the TCP/IP protocol suite. In the later sections, some important protocols, such as TCP, UCD, and IP protocols, will be discussed further. And some protocols of the application layer will be introduced along with the corresponding software in the UNIX operating system in the final several sections.

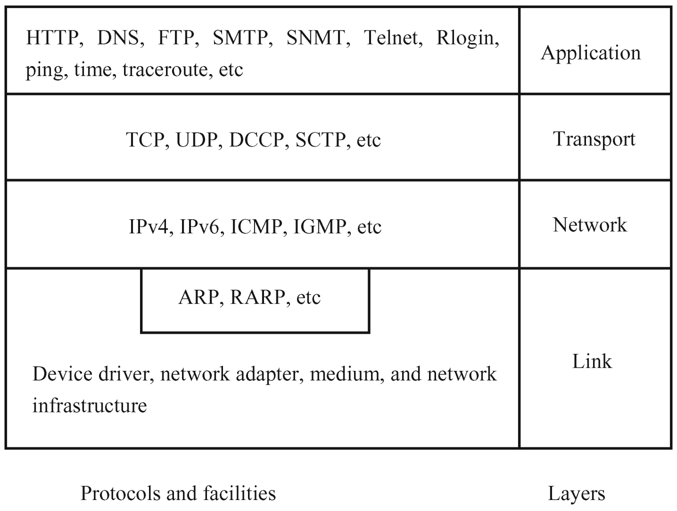

HTTP, DNS, FTP, SMTP, SNMT, Telnet, Rlogin, ping, time, traceroute, etc
Application
TCP, UDP, DCCP, SCTP, etc
Transport
IPv4, IPv6, ICMP, IGMP, etc
Network
ARP, RARP, etc
Device driver, network adapter, medium, and network infrastructure
Link
Protocols and facilities
Layers

Fig. 11.2 The TCP/IP protocol suite.

11.2.5.1 Protocols in Application Layer

In the application layer, there are some protocols that are really well-known for the Internet users, for example, Hypertext Transfer Protocol (HTTP), Telnet and Rlogin (remote login), File Transfer Protocol (FTP), Simple Mail Transfer Protocol (SMTP), Domain Name System (DNS), Ping (testing a host reachable), Trivial File Transfer Protocol (TFTP), and Simple Network Management Protocol (SNMP). Each of these protocols is involved in one application that can be utilized by the Internet users to invoke one Internet service, interact with the Internet and do some activity over the Internet, such as Web browsing, sending and receiving email, chatting on-line, transmitting files between different hosts over the Internet, logging remotely in a host, etc.

11.2.5.2 Protocols in Transport Layer

As the purpose of the transport layer is to provide a flow of data between two hosts for the application layer above, the transport layer includes protocols that provide various functions, such as: segmentation, reassembly, and error recovery. In the transport layer, TCP and UDP are two dominant protocols.

The Transmission Control Protocol (TCP) along with the Internet Protocol (IP) constitutes the original of the Internet Protocol Suite (the TCP/IP protocol suite). In their cooperation for the message transmission over the

Internet, the IP tackles the lower-lever communication between hosts and the TCP handles the task only at a higher level of the two end systems, such as a Web browser and a Web server for the Web browsing application. The TCP provides reliable and sequenced delivery of a stream of bytes from one computer to another. Besides the common tasks of connection establishment, data transfer, reliable transmission, error detection, and connection termination, the tasks of the TCP also include the management of maximum segment size, flow control, congestion control, selective acknowledgments, window scaling, TCP timestamps, out-of-band-data, forcing data delivery, etc. A typical process of data transmission from one host to another with the TCP is to divide the data up from the application layer into the proper-sized packets for the below network layer, to send the packets via the IP, to acknowledge received packets, and to set timeouts in order to make sure the other end acknowledges packets to be sent. Since the TCP provides the reliable flow of data, the application layer supported by the TCP can ignore this issue. The first specifications of the TCP were written in RFC 675 in 1974. Since then, the TCP has been developed in many ways and the enhancements in its different aspects are published in some of the subsequent RFCs.

The User Datagram Protocol (also called the Universal Datagram Protocol, abbreviated as UDP), on the other hand, provides a much simpler service to the application layer without implicit hand-shaking dialogues for guaranteeing reliability and data integrity. In other words, the UDP just sends packets of data — datagrams — from one host to the other, but it does not make sure that the datagrams reach the other end. With the UDP at the transport layer, any desired reliability must be added by the application layer. However, in some situations, the UDP is more useful than the TCP. For example, for the time-tight applications, such as the streaming media systems of the Internet Protocol television (IPTV) and Voice over IP (VoIP), the rate of data transmission is more important than the data integrity, but the connection establishing for the reliable, error-free, and in-order delivery of data can make a significant delay. UDP was defined in RFC 768 in 1980.

Therefore, the UDP loads different applications in the application layer from the TCP. UDP can be applied in broadcasting and multicasting, DNS, etc. while TCP applications can be Telnet and Rlogin, FTP, SMTP, etc.

There are also some other transport layer protocols, such as DCCP and SCTP. Different from the TCP, Datagram Congestion Control Protocol (DCCP) provides a way with which to access to congestion control mechanisms without their implementation at the application layer and to allow for flow-based semantics without the reliable in-order delivery. This scheme quite fits into the applications time-limited on the data delivery, in which the delay of the reliable sequenced delivery and congestion control can make the data delivery useless to the receiver. The first implementation of DCCP was released in Linux Kernel Version 2.6.14 in 2005. And as a proposed standard, DCCP has been published in RFC 4340 (Kohler et al 2006) by the IETF in 2006. Stream Control Transmission Protocol (SCTP) provides not only some of the same services of TCP and UDP, but also multiple streams (which mean the delivery of data chunks with independent streams to eliminate unnecessary delay of the TCP's head-of-line blocking) and multi-homing (which means that the system has multiple network interfaces to meet the need of highly-available data transfer) support. SCTP is introduced and defined in RFC 3286 (Ong et al 2002) and 4960 (Stewart 2007).

11.2.5.3 Protocols in Network Layer

The Internet Protocol (IP) is the foremost protocol in the network layer of the TCP/IP protocol suite and takes the job of delivering different protocol datagrams (or packets) from one host to another host just based on their addresses. For this objective, the IP describes addressing mechanics and datagram encapsulation structures. There are many versions of the IP, but among them, Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6) are officially adopted. IPv4 uses 32-bit addressing structure, which has a capacity of more than 4 billion addresses while the IPv6 employs 128-bit addressing structure that holds about 340 undecillion addresses. IPv4 was defined in RFC 791 (Postel 1981.1) by IETF in 1981 and IPv6 was described in RFC 2460 around 1998. Today, IP4 is still the most widely exploited network protocol, although its successor, IPv6 is being supported by the worldwide infrastructure.

With the datagram encapsulation, the IP can be used to connect networks with different link layer implementations, such as Ethernet, token ring, ATM, Fiber Distributed Data Interface (FDDI), Wi-Fi, etc. Since different link layer implementation may have distinguished addressing strategies, it can be the most complex for IP to address and route. IP addressing is about how to allocate an IP address to an end host and how to group sub-networks of IP host addresses. IP routing can be performed by all hosts, but mostly by dedicated routers, which adopt either interior gateway protocols or external gateway protocols relied on their responsibilities in their network in order to help make IP datagram forwarding decisions across IP networks.

The IP provides an unreliable (or best effort) delivery. With this strategy, the benefits are reducing the network complexity and having routers along the transmission path just forward packets to the next gateway that matches the routing prefix of the destination address. And it also meets the need of the surroundings in the Internet, that is, no central monitoring facility exists and availability of links and nodes dynamically changes.

As the main protocol performing routing at the network layer, the IP is used for routing by both TCP and UDP. Any chunk of TCP and UDP data transferred around the Internet needs to pass through the network layer with the IP at both end systems and at every passed router.

Usually, the IP is used for communicating data across packet-switched networks. The packet switching is a communications technology that divides all transmitted data into packets regardless of their content and type. Contrary to the circuit switching applied in public switched telephone networks, which sets up a dedicated connection in a constant bit rate and with constant delay between nodes for exclusive use during the communication session, the packet switching is to deliver sequences of packets through network adapters and routers over networks in a variable bit rate and with variable latency because of the dynamic of availability of links and nodes, network traffic load, and the packet buffering and queuing. The packet switching, partly, determines the unreliability of the Internet Protocol service.

In the Internet, there are two major packet switching modes: connectionless packet switching (also called datagram switching) and connection-oriented packet switching (also known as virtual circuit switching). For the former, each packet holds address information, with which the packets are routed individually, sometime in different paths and out of order. For the latter, a connection must be first established before any packet is transferred, via which the packets can be delivered in order and reliably by acknowledging after successful delivery and automatically repeating request for missing data or detected bit-errors. Compared to the connection-oriented packet mode whose connections are always unicast (for one single destination host), the connectionless packet switching mode has broadcast function (for all hosts in a network) and multicast function (for a set of hosts belonging to a multicast group), which can save network resources when the same data have to be transmitted to several destinations. IP and UDP are connectionless protocols while TCP is connection-oriented.

The Internet Control Message Protocol (ICMP) is an auxiliary for the IP. It is mainly utilized by the network layer to exchange error messages (such as unavailability of a requested service and unreachability of a host) and other vital information with the network layer in another host or router. It is not used directly in user common network applications, except two popular diagnostic tools, ping and traceroute. The ICMP (also called the ICMPv4) is for the IPv4 while for the IPv6 is the ICMPv6. The ICMP is defined in RFC 792 (Postel 1981.2), and the ICMPv6 is in RFC 4443 (Conta et al 2006).

The Internet Group Management Protocol (IGMP) is used to send a UDP datagram to multiple hosts – multicast. The IGMP can be applied in the establishment and management of the multicast group memberships for IP hosts and neighboring multicast routers. The IGMP can also enhance resource utilization when used in the applications such as online streaming video and gaming. The IGMP is only used with the IPv4 because the IPv6 has a different solution to multicast. Three versions of the IGMP are defined respectively in RFC 1112 (Deering 1989), RFC 2236 (Fenner 1997), and RFC 3376 (Cain et al 2002).

11.2.5.4 Protocols in Link Layer

As the services of the lower layers have to be transparent to the higher layers, the implementation detail of services in the link layer is hidden from the applications in order to make the services easy to be called when some application is raised. However, since it consists of software and hardware simultaneously, the implementation of the link layer is different from and harder than those of the other layers. The implementation of services in the link layer of the TCP/IP protocol suite includes the interface between the network layer and link layer, the driver of network adaptor, and the network interface to the local network or router. Therefore, the protocols in the link layer are manifold and complex. For this reason, only some of them will be introduced in this book, and more information related to them can be referred to the references at the end of this chapter.

The Address Resolution Protocol (ARP) and Reverse Address Resolution Protocol (RARP) are dedicated protocols used to convert between the addresses executed by the network layer and the addresses by the hardware interface in the link layer for certain types of network interfaces (such as Ethernet and token ring). The ARP is described in RFC 826 (Plummer 1982) and the RARP in RFC 903 (Finlayson et al 1984). These two protocols are crucial for both the local network and the inter-network routing. For the latter, they can be used to make a decision on which router will be passed through next.

As mentioned above, the ARP and RARP have been implemented in many different types of networks, such as Ethernet, token ring, and many others. However, because of the prevailing of the IPv4 and Ethernet in networking, ARP is mostly used to map an IP address onto an Ethernet Media Access Control (MAC) address, which is a unique identifier assigned to most network adapters by the manufacturer for identification. The IP addresses are 32-bit, but the Ethernet MAC addresses (sometimes called as Ethernet Hardware Address, EHA) are 48-bit.

APR can have a function to test whether or not an IPv4 address is already in use, which is called APR probe. It is useful to make sure that an IPv4 address is available before using it.

In the link layer of the TCP/IP protocol suite, Ethernet protocol is also popular. Ethernet protocol, promoted by three companies (DEC, Intel, and Xerox) and published by the Institute of Electrical and Electronics Engineers (IEEE) around the early 1980s, uses 48-bit addresses and the bit rate of 10 Mbps. Its access method is called as the Carrier Sense, Multiple Access with Collision Detection (abbreviated as CSMA/CD). After that, the IEEE 802 Committee published a set of standards. Among these standards, 802.3 defines Ethernet networks, 802.4 defines token bus networks, and 802.5 defines token ring networks. Now Ethernet becomes one of the prevalent technologies in LANs. One of the most popular wired LAN implementations is the combination of the Ethernet over twisted pair (used in connection between end systems and the network) with the Ethernet over fiber optic (used in establishment of site backbones).

11.2.5.5 An Example for TCP/IP Protocol Suite

From the above sections, it is known that there are so many protocols in different layers of the TCP/IP protocol suite. Then, how do these protocols work together? Figure 11.3 gives a typical example to show how the protocols in different layers cooperate together to fulfill file transfer between two hosts with client-server model over the Internet.

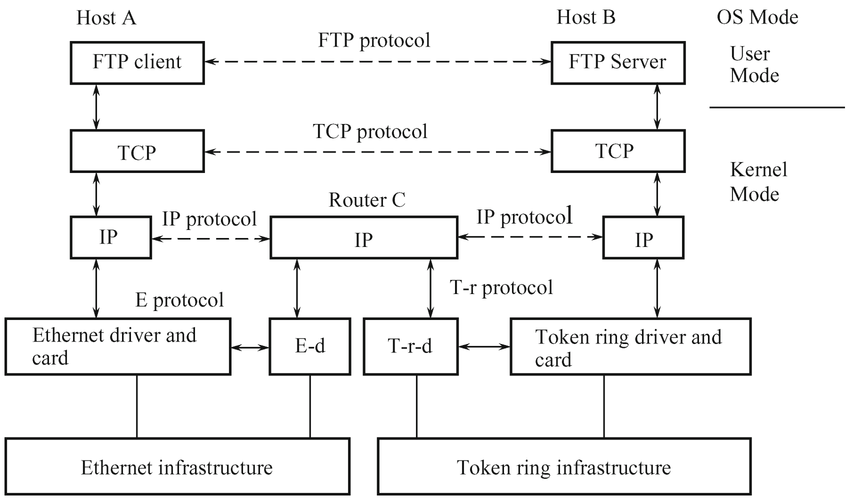


Fig. 11.3 Protocols used in file transmission between Host A and Host B via Router C (E-d replaces Ethernet driver and card, T-r-d replaces taken ring driver and card, E protocol means Ethernet protocol, and T-r protocol means token ring protocol in this figure).

In this case, the FTP protocol is executed in the application layer, and the TCP protocol is practiced in the transport layer of two hosts: A and B. As Host A is equipped with Ethernet adaptor and located in Ethernet LAN but Host B is within Token ring LAN, a router is necessary to connect these two different types of LANs. And the IP protocol is used in the network layer of Host A, Router C, and Host B. It is also seen that Ethernet protocol is running in the link layer of the Ethernet LAN and Token ring protocol is running within the same layer of the Token ring LAN. With the software modules that implement these protocols transmitting data down or up in sequence, the FTP client in Host A initiates the file transfer request and the FTP server in Host B responds and provides the corresponding service.

11.3 Encapsulation and Demultiplexing

The software implementation of the TCP/IP protocol suite is also called the protocol stack. As the TCP/IP protocol suite has several layers, the protocol stack has several modules, each of which implements the function of its corresponding layer. When a certain application sends data through each layer with some protocols, the data is delivered down via each module of the protocol stack until it is transmitted as a stream of bits across the network. During the delivery process, each layer adds headers in front of and sometimes information at the end of the data received from the upper layer. In this way, the higher level objects can be hidden from the underlying structures and functions can be logically separated between the layers. This technique is called encapsulation. Figure 11.4 shows the encapsulation of the application data down through the protocol stack.

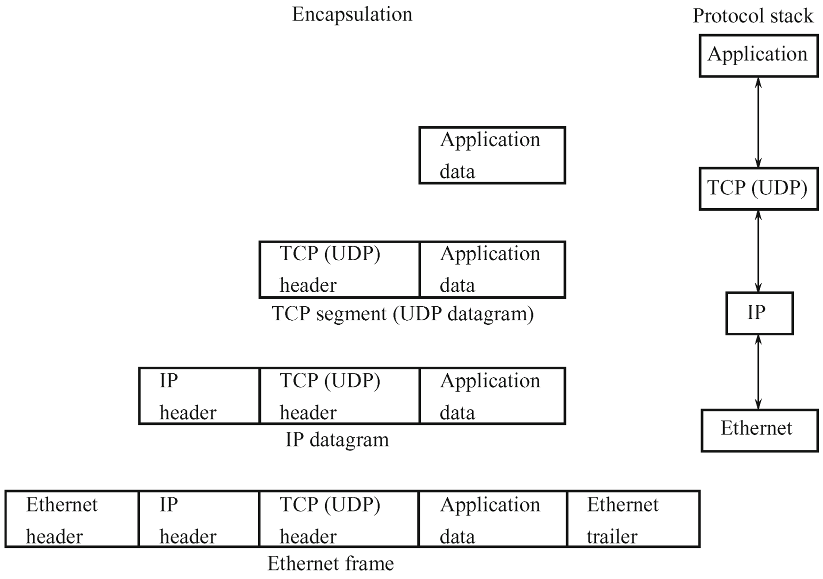


Fig. 11.4 Encapsulation of data.

If the TCP is used in the transport layer, the application data is encapsulated into TCP segments. A TCP segment has a 20-byte TCP header that is added to the application data. If the UDP is applied in the transport layer, the application data is encapsulated into UDP datagrams. A UDP datagram has an 8-byte UDP header. TCP segments (or UDP datagrams) are in turn encapsulated into IP datagrams via the network layer. The IP datagrams have 20-byte IP headers added to TCP segments (or UDP datagrams). When the Ethernet is used in the underlying layer, IP datagrams are then encapsulated into Ethernet frames before going across the network. The Ethernet frames have 14-byte Ethernet headers and 4-byte Ethernet trailers added to the IP datagrams. The size of an Ethernet frame is between 46 and 1500 bytes.

Known from Figure 11.2, there are several protocols in each layer of the TCP/IP protocol suite. These protocols need to be identified when the data travel up from layer to layer at the destination host. This issue can be tackled with the headers in different layers. In a header of one layer, there is one type of identifier to point out which protocol is used in the next upper layer. For example, to differentiate among the TCP, UDP, IMCP, and IGMP in the transport layer, the IP headers consist of an 8-bit protocol field, which is equal to 6 for the TCP, 17 for the UDP, 1 for the ICMP, and 2 for the IGMP. As some applications use the TCP and some other applications use the UDP, TCP or UDP headers have 16-bit port numbers to distinguish these applications. In the Ethernet header, there is a 16-bit frame type field to identify the protocols used in the layer or sublayer above it. When the data reaches its destination host, these identifiers are quite useful.

Opposite to the encapsulation at the origin host, at the destination host, an Ethernet frame needs to be demultiplexed the same number of times as its encapsulation layer by layer until it rises at the application layer. At each layer, the proper protocol is used to remove the corresponding header and look at the identifier in the header to make a decision about which protocol should be used in the next upper layer.

11.4 Networking Operating Systems

Networking operating systems came along with the computer networking and Internet. In other words, networking operating systems coexist with the computer networking and Internet. Therefore, networking operating systems should include the protocol stack in their architecture in order to provide the facilities for the computer networking and Internet. Also for the distributed computing, modern operating systems usually can not only support the independent local applications but also meet the needs of computer networking. These types of operating systems can be configured with different function modules, such as clients or servers, according to their real roles in the networking. Therefore, networking operating systems should be more scalable, flexible, and dynamical.

As known in Section 11.2.4, when the networking is seen in the logical way, its functions can be divided into different layers of the TCP/IP or OSI models. From the view of the operating system, it is a good way to see networks in a logical perspective. In this way, we can put different modules associated with the protocols in different layers into corresponding layers of the operating system structure. As described in Figure 2.1, operating systems have their own hierarchy from hardware (just like physical layer in the OSI model) up to applications (just like application layer in the TCP/IP model). These natural relations can help the reader know in which level of the operating system an implementation module of one protocol should be put. It is important for the reader to have this perspective because this book is discussing UNIX operating systems now. For an operating system administrator or designer, it is meaningful and effective to make the networking modules fit into the operating system architecture.

As an operating system usually makes the physical detail hidden from the users, it is natural to put it in the kernel of the operating system how to make the communication between two hosts. Therefore, when a user initiates a networking application, a user process is created for the user and then the operating system makes the corresponding system call to invoke the appropriate protocol modules in sequence according to the user application and the protocol stack. On the right side of Figures 11.1 and 11.3, at the application layer is running a user process while the functions of the lower three layers are usually executed in the kernel of the operating system. It is typical the way done in UNIX.

Since networking operating systems are a much younger and flourishing member of operating system family, there are still new emerging characters and potential for networking operating systems. However, as the benefits of sharing computer resources among users, providing a development cooperation platform for variedly located researchers, and enhancing the reliability of the whole computer system have been brought into the reality by the computer networking, some certain features of networking operating systems can be listed as follows.

Inside operating systems, in addition to the common services that operating systems usually have and have been introduced in previous chapters, some other facilities should be included in networking operating systems. No matter what role in a network is played by the computer with the networking operating system — a client, server, or router, the following are must-be:

- The modules are implemented for the protocols that carry out routing the application data over a network.
- The modules are practiced for the protocols that execute access to the physical transmission medium over a network.

For a client or server, the following is also necessary:

- The modules are realized for the protocols that fulfill the application data transportation between two processes on two hosts over a network.

While the above should be implemented in the kernel of operating systems, some network applications, attached to the interface between users and operating systems, can be provided. These applications can, typically, be:

- Network system administration, which is used to manage the resources over the network with high availability and fault tolerance.
- Network user administration, which is used to add, remove, and manage users who are expected to use resources on the network.
- Remote login, which is to allow users to access resources over the network.
- File transmission, which is used to transmit files between different hosts over a network.
- Email service, which is used for users to send and receive emails.
- Web browsing, which provides websites browse service.
- Data network storage and backup services, which can enlarge the storage space and provide expending data backup, besides data sharing.
- On-line chatting that provides a public or private platform for users to communicate with each other at real time.
- Network printer service, which can let several users share a number of

printers via a network.
- Network resource security, which can give users the proper authorization and authentication for login restriction and access control.
- Network database management system and database access logic, which can let differently-located users share some database via a network.
- Name service and directory service, which can help direct users to the hosts and directories over a network of interest.
- And other emerging network applications.

As mentioned in Sections 11.2.3 and 11.2.5, the software implementation of these applications is usually divided into two parts — client and server, and located in different computers, which relies on the roles played by the computers in a network. And according to the inner relationship between applications and lower-layer protocols, some of applications will be discussed along with their corresponding protocols in the following sections.

11.5 IP Protocol

The Internet Protocol (IP) is quite popular in the computer networking world. All the TCP, UDP, ICMP, and IGMP datagrams are transmitted with the IP. However, the IP provides an unreliable and connectionless delivery service.

Being unreliable, the IP just makes a best effort for its delivery service. If some mistakes happen in the delivery, the IP can send back an ICMP message to the source host. Thus, if a delivery needs reliability, it requires the support of the upper-layer protocol, such as the TCP. For being connectionless, the IP does not have the function of real connection between the source and destination hosts. The IP lets its datagrams deliver independently no matter whether or not the datagrams are linked with their context. Therefore, when they arrive at the destination host, the datagrams may be out of order. This issue can be handled by some fields in the IP header.

There are two UNIX commands: ping and traceroute, which have an inherent relationship with the IP and will be discussed in the following sections.

11.5.1 IP Header

As known, there are two official versions of IP, IPv4 and IPv6. They have different IP headers. Without any option, the regular size of the IPv4 header is 20 bytes (160 bits), shown in Figure 11.5, while the size of the IPv6 header is 40 bytes (320 bits), shown in Figure 11.6.

Here gives the explanation of the fields in the IPv4 header.
- Version: for IPv4, it is 4.
- Header length (H-length): it is the number of 32-bit words in an IP header

	0-3 bits	4-7 bits	8-15 bits	16-31 bits	
0-31	Version	H-length	Type of service	Total length in bytes	
32-63	Identification			Flags	Fragment offset (51-63 bits)
64-95	Time to live		Protocol	Header checksum	
96-127	Source IP address				
128-159	Destination IP address				
	Options (bytes)				
	Data				

Fig. 11.5 IPv4 header and IPv4 datagram (The numbers in the blocks of the first row are the bit numbers that the blocks cover in one 32-bit word; Flags field and Fragment offset field in the third row cover three bits and 13 bits, respectively. The numbers in the blocks at the left column are the numbers of bits composing a regular IPv4 header of 20 bytes).

with any options. For a regular IPv4 header without any option, it is 5. As a 4-bit binary number is limited up to 15 in decimal system, the greatest length of an IPv4 header is 60 bytes.

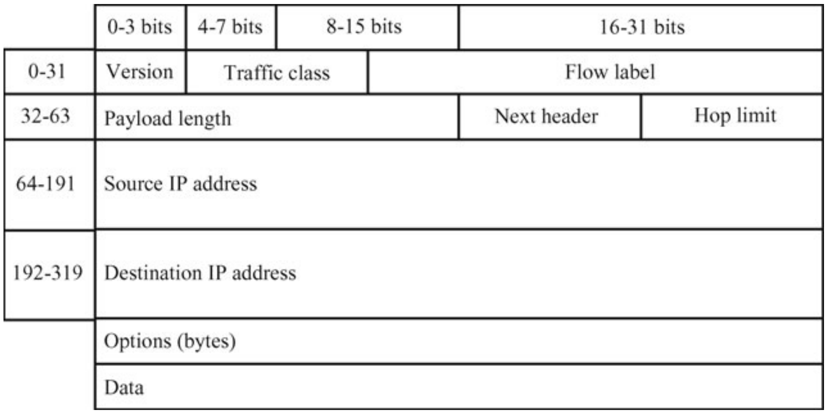
- Type of service (TOS): its first 3 bits are usually ignored, the middle 4 bits are TOS bits, and the final bit is always 0. Among four bits of TOS, the first bit is minimize delay bit; the second one is maximize throughput bit; the third one is maximize reliability bit; the final one is minimize monetary cost bit. The names of these four bits can tell their functions. All the four bits can be 0 simultaneously, but only one of them can be 1. The minimum delay bit should be set 1 when used in remote login applications such as Telnet and Rlogin while the maximum throughput bit must be 1 when used in file transfer with FTP.
- Total length: it is the number of total bytes of an IPv4 datagram. Since it covers 16 bits, the greatest length of an IPv4 datagram is 65 535 bytes.
- Identification: it is a unique identifier for each datagram. It is added by one each time a datagram is sent between hosts or routers via a network.
- Flags: from high order to lower order, its first bit is reserved and always 0; the second bit is Don't Fragment (DF); and the third bit is More Fragments (MF).
- Fragmentation offset: it is the offset in 8-byte units of a particular fragment from the beginning of the original IP datagram.
- Time to live (TTL): it is an upper limit of the number of routers through which a datagram can travel. It is set to a certain value by the sender and decremented by one by each router via which the datagram passes. When the number is equal to 0 and the destination host is not reached, the datagram is discarded. In this way, it can prevent packets' forever routing over a network with limiting the lifetime of the datagram.

- Protocol: it is an identifier for the upper protocols that send the application data to the IP. It allocates 1 to the ICMP, 2 to the IGMP, 6 to the TCP, and 17 to the UDP.
- Header checksum: it is used to calculate the sum of an IP header when the datagram arrives at the destination in order to check whether or not the delivery is correct. When calculated, the IP header is considered as a sequence of 16-bit words. It only checks the IP header, and the upper protocol headers should be checked in their own layers.
- Source IP address: it is a unique 32-bit address for the source host. The IP address will be discussed in the following section.
- Destination IP address: it is a unique 32-bit address for the destination host.
- Options: it is the extended datagram information, such as security and handling restrictions, record route, timestamp, loose source routing, or strict source routing. Dependent on the need of the extended information, the length of the option list is variable.

As the link layer has an upper limit on the size of the frame that can be transmitted, the IP performs fragmentation if the datagram size is larger than this frame size. The fragments of an IP datagram are reassembled until they reach the destination. For fragmentation, the value of the identification field of an IP datagram is copied into the identification field of each of its fragments. The More Fragments of the flags field of all its fragments, except the final fragment, is set to 1. The fragment offset field of each fragment holds its offset from the beginning of the original IP datagram. The total length field of each fragment is the size of this fragment, not the whole IP datagram. This technique makes it possible for the receiver to reassemble the fragments in the correct way with the information in the IP header even though the fragments of a datagram arrive at the final destination out of order.

In Figure 11.5, the most significant bit is on the left and the least significant bit is on the right. IP datagrams are transmitted in the big-endian byte order. That means, the transmitting order is: $0-7$ bits are transmitted at the first, $8-15$ bits at the second, $16-23$ bits at the third and $24-31$ bits at the forth.

As shown in Figure 11.6, the IPv6 datagram has a total different header. The most significant feature of the IPv6 is that the IP address is extended from 32 bits of IPv4 to 128 bits. As the focus of this book is put on the IPv4, the details of other fields are beyond this book and can be referred in RFC 2460 (Deering et al 1989).



	0-3 bits	4-7 bits	8-15 bits	16-31 bits
0-31	Version	Traffic class		Flow label
32-63	Payload length		Next header	Hop limit
64-191	Source IP address			
192-319	Destination IP address			
	Options (bytes)			
	Data			

Fig. 11.6 IPv6 header and IPv6 datagram.

11.5.2 IPv4 Addressing

Every interface in the Internet must have a unique Internet address (IP address). For IPv4, the IP addresses are 32-bit. Instead of using a linear address space, an IPv4 address is composed of four decimal parts, each of which covers one byte of the address, for instance, 192.252.124.231. This is also called dotted-decimal notation. There are five classes in the Internet addresses. The class difference of an address can be identified in the first number of its dotted-decimal address, shown in Figure 11.7.

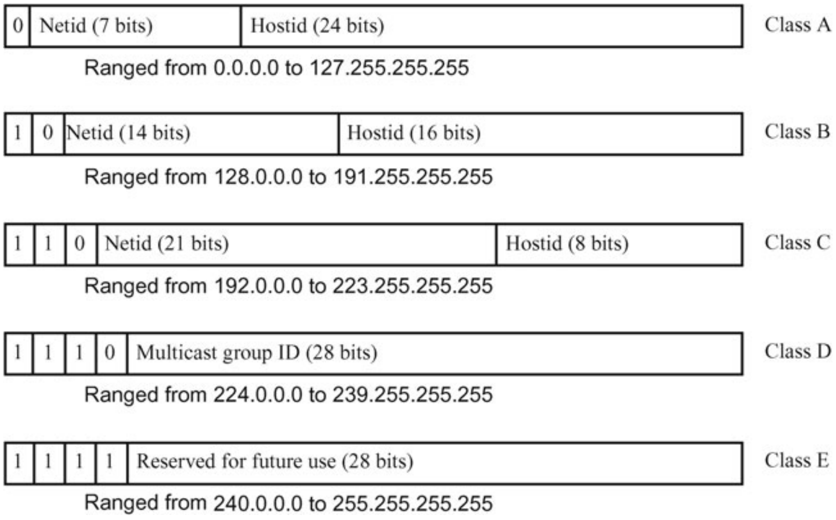

0 | Netid (7 bits) | Hostid (24 bits)
Class A
Ranged from 0.0.0.0 to 127.255.255.255
1 | 0 | Netid (14 bits) | Hostid (16 bits)
Class B
Ranged from 128.0.0.0 to 191.255.255.255
1 | 1 | 0 | Netid (21 bits) | Hostid (8 bits)
Class C
Ranged from 192.0.0.0 to 223.255.255.255
1 | 1 | 1 | 0 | Multicast group ID (28 bits)
Class D
Ranged from 224.0.0.0 to 239.255.255.255
1 | 1 | 1 | 1 | Reserved for future use (28 bits)
Class E
Ranged from 240.0.0.0 to 255.255.255.255

Fig. 11.7 Five classes of Internet addresses and their ranges.

11.5.3 IPv4 Routing

Mentioned in Section 11.2.2, when computer networking, a computer can take the role of a host or router. In the network layer, a computer with a multi-user operating system such as UNIX can be configured to act not only as a host but also as a router. Being just a host in a network, a computer cannot forward a datagram; as a router, a computer can do datagram forward.

In general, the network layer can receive a datagram from the upper layer that may perform one of the protocols, including TCP, UPD, ICMP, and IGMP, and send the datagram down to the link layer. Also, it can receive a datagram from a network interface. And then it sends the datagram to the upper layer if the computer IP address does match the datagram destination IP address. Or it forwards the datagram if the computer does not match the datagram destination IP address and has been configured as a router. Or it discards the datagram if the computer has not been configured as a router and the IP address does not match.

In the network layer, there is a routing table to support the IP routing. The data structure of each entry of the routing table typically consists of the following fields:

- IP address for the destination, which can be a host address or a network address. If it is a host address, the host flag bit is set. If it is a network address, it has a zero hostid (shown in Figure 11.7) and maps into all the hosts in the same network, and the host flag bit is zero.
- IP address in the networking that can be a next-hop router or a directly connected network, which is used to forward a datagram. If it is a next-hop router, the router flag bit is set.
- Flags, two of which are the host flag bit and router (or gateway) flag bit.
- Reference count, which displays the reference count of the route entry. For a connection-oriented protocol such as TCP, the route is held when the connection is established. Known that Telnet is an application protocol of TCP in Section 11.2.5, a Telnet execution can increase the reference count of the route entry by one.
- Use count, which shows the number of packets passed through the route. When a datagram-sending protocol such as ping is used, the use count is added by one for each action of the protocol performance.
- Name of the local network interface, which can be used by the kernel to map it into a socket device that is usually corresponding to an attached Ethernet card or Token ring card.

In the routing table, there are two special entries: default route and loop-back interface.

The default route allows any other network reachable through a single router. For default route entry in the routing table, the field of the destination IP address is identified as default, its router flag is set, and the IP address of the next-hop router is usually the gateway through which the user can access the Internet.

The loop-back interface is used by the host to communicate to itself. It lets a client and server on the same host to communicate each other by using TCP/IP. For loop-back interface in the routing table, two fields of the destination IP address and the directly connected network have the same value: 127.0.0.1. By convention, the Class A network ID 127 (as Netid shown in Figure 11.7) is reserved for the loop-back interface.

The IP routing algorithm is shown in Figure 11.8. Typically, the IP routing algorithm makes at most three iterations of match searching in the routing table.

The first iteration is to search the routing table for the entry that the host flag is set and the IP address of the destination host matches both the network ID (the class number plus Netid in Figure 11.7) and the host ID (Hostid in Figure 11.7). If the matched entry is found and its route flag is set, the packet is sent to the specified next-hop router. If the route flag of the matched entry is reset, the packet is sent to the directly connected interface.

If the first iteration does not find the matched entry, the second iteration starts to search. The second iteration is to search the routing table for the entry that its host flag is reset and its IP address of destination network is matched. If the matched entry is found and its route flag is set, the packet is sent to the specified next-hop router. If the route flag of the matched entry is reset, the packet is sent to the directly connected interface.

If the second iteration does not find the matched entry, either, the third iteration starts to search. This time, it searches the routing table for the default entry. Usually, the default entry can be found and the packet is sent to the specified next-hop router. If even the default entry cannot be found, the error message, which can show “host unreachable” or “network unreachable”, will be displayed.

As IP routing is done from hop to hop, IP protocol usually does not know the complete route to a destination if the destination is not connected directly to the sending host. All that IP routing does is to provide the IP address of the next-hop router to which the datagram is sent. It is assumed that the next-hop router is closer to the destination than the sending host is, and that the next-hop router is directly connected to the sending host. Another feature of IP routing is that the routing is specified to a network. That is, the routing table is just a subset of all the options of possibly routing for a datagram. In this way, it simplifies the routing for any datagram.

In UNIX, there is a routing daemon, called routed, to do initialize the routing table and decide which routes are selected into the routing table. Known in Section 4.5.6, daemons are usually started when the system is booting. The routing daemon is also started at the system booting and does updating the routing table periodically (typically once each 30 seconds). Except the routing daemon, the ifconfig command can be used to set an interface’s address, and the route command can do the similar task. In some UNIX operating systems, the /etc/defaultrouter file holds a default router,

```
while (the end of routing table is not reached)
  {           /* the first search for a matching host address */
  if (the host flag is set and IP address of destination host is completely matched)    {
    if (the route flag is set)      {
          send the packet to the next-hop router;
          return;
          }
          else
          {
            send the directly connected interface;
            return;
          }
    }
  }
while (the end of routing table is not reached)
  {              /* the second search for a matching net work address */
  if (the host flag is reset and IP address of destination network is matched)  {
    if (the route flag is set)      {
          send the packet to the next-hop router;
          return;
          }
          else
          {
            send the directly connected interface;
            return;
          }
    }
  }
while (the end of routing table is not reached)
  {              /* the third search for a default route */
     if (it is not the default entry)  {
         continue;
        }
     send the packet to the next-hop router;
      return;
  }
Display error me ssage to show "host unr eachable" or "net work un re achable";
return;
```

Fig. 11.8 The C-style illumination of IP routing algorithm.

which is added to the routing table when the system is booting.

11.5.4 Two commands, ping and traceroute

Here, we will introduce two commands, ping and traceroute, which can be used to check the status of the network connected to our computers.

11.5.4.1 Testing a Host Connection

Ping command is usually used to test a network connection. A user can execute the ping program on a host, which sends the echo requests with a client to a server on the tested host. And the client waits for the response from the server. When the response comes back to the user host, the client displays the testing result on the user's screen.

In UNIX, if a process invokes the ping program, the kernel sets the identifier field in the ICMP (shown in Section 11.2.5.3) message to the process ID of the process, which makes the ping program distinguish the returned responses if there are several processes of ping executing simultaneously on the same host.

The syntax and function of ping command are as follows.

```
$ ping [-options] hostname
```

Function: to send an IP datagram to the host with the 'hostname' name to test whether it is connected to the network (or Internet) and display the tested result on the screen.

Common options:

-c n: to set the number of packets that will be sent and received to 'n';

-s p_size: to set the size of packets that will be sent to 'p_size' bytes.

Note: without any option, the ping program just displays "hostname is alive" if the hostname is available or "no answer" if the hostname is not found. By default, the size of packets to send is 56 bytes.

For example:

```
$ ping hebust.edu.cn
hebust.edu.cn is alive
$
```

The above ping command without any option and it displays just a simple sentence.

Another example:

```
$ ping -c 3 beds.ac.uk
PING beds.ac.uk (194.80.218.20):  56 data bytes
64 bytes from 194.80.218.20:  icmp_seq=0 ttl=255 time=347ms
64 bytes from 194.80.218.20:  icmp_seq=1 ttl=255 time=348ms
64 bytes from 194.80.218.20:  icmp_seq=2 ttl=255 time=348ms
--- beds.ac.uk PING Statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 347/348/348 ms
$
```

The second ping command with the option of -c 3 sends and receives three messages. The first line of the ping output displays the IP address of the tested host. The DSN (Domain Name System) resolver has mapped the given hostname into the IP address, which will be discussed simply in Section 11.7.2.

We can see that the ICMP sequence number, TTL, and the round-trip time are following. The ICMP sequence number begins with 0 and is added by one each time a new echo request is sent. As the ICMP header is 8-byte, the whole size of each package is 64 bytes. As known, IP is a best effort datagram delivery service and its packets can be lost, reordered, or duplicated. The ping program displays the sequence number of each returned packet, having us check if packets are lost, reordered, or duplicated. If the echo reply for a certain sequence number does not appear in the displayed list, it is lost. If sequence number M is shown before sequence number M-1, they are reordered. When sequence number M appears twice or more in the list, it is duplicated.

Shown in Section 11.5.1, TTL is the time-to-live field of the IP header, which is an upper limit of the number of routers that a datagram can go through.

The ping program can also calculate the round-trip time with subtracting the current time when the reply returns by the sending time that has been stored in the ICMP message when the echo request was sent.

The third example:

```
$ ping -c 3 -s 500 beds.ac.uk
PING beds.ac.uk (194.80.218.20):  500 data bytes
508 bytes from 194.80.218.20:  icmp_seq=0 ttl=255 time=348ms
508 bytes from 194.80.218.20:  icmp_seq=0 ttl=255 time=347ms
508 bytes from 194.80.218.20:  icmp_seq=0 ttl=255 time=348ms
--- beds.ac.uk PING Statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 347/348/348 ms
$
```

The third ping command with the -c and -s options sends and receives three messages of the 508-byte size. Compared to the second ping command, since the packet size increases not too much, the round-trip time does not change a lot, either.

11.5.4.2 Tracing the Route from one Host to Another

Traceroute command can be used to trace the route from one host to another. It also uses ICMP and the TTL filed in the IP header. As the TTL field in the IP header occupies 8 bits, the maximum TTL value is 255. Each time the datagram is passed to a router, the TTL value is subtracted by one or the number of seconds that the router holds the datagram, which is dependent on the router. Mostly, the TTL value is subtracted by one by each passed router.

Therefore, the traceroute program takes the mechanism: firstly, it sends an IP datagram with a TTL of 1 to the destination host. The first encountered router subtracts the TTL value by 1 and makes it zero. Explained in Section

11.5.1, when the value of TTL is equal to 0 and the destination host is not reached, the datagram is discarded. And the ICMP shows time out. The first round-trip can identify the IP address of the first router in the path to the destination host. Then the traceroute program does the second test. It sends a datagram with a TTL of 2, and the IP address of the second router can be found. In this way, many times of round trips with increasing TTL values are taken until the destination host is reached. At the final round trip when the destination host is reached, as the TTL is equal to 1, the time is not exceeded and the datagram is not discarded by the destination host. Hence, the last reply returned to the sending host is different from the previous ones.

The syntax and function of traceroute command are as follows.

```
$ traceroute [-options] hostname
```

Function: to trace the host with the 'hostname' name and to display the routers in the path from the sending host to the destination host with the 'hostname' name on the screen.

Common options:

-n: to display the IP addresses rather than the hostnames.

Note: By default, the size of packets to send is 32 bytes.

For example:

```
$ traceroute hebust.edu.cn
traceroute to hebust.edu.cn (222.223.188.179), 30 hops max, 40 byte
packets
1 pc2010bvh.campus (192.168.189.2) 20 ms 10 ms 10 ms
2 hebust.edu.cn (222.223.188.179) 30 ms 30 ms 40 ms
$
```

The first line of the traceroute output displays the name and IP address of the destination and the maximum value of TTL that the traceroute can set is 30. The datagram size is 40 bytes with the 20-byte IP header and 8-byte UDP header. The rest 12-byte data include a sequent number, the time at which the datagram was sent, and a copy of the initial TTL value.

The following lines contain the TTL value, the name and IP address of the router or host, and the times taken of three packets sent by traceroute as they go from one router to the next. For any sent datagram, an asterisk is displayed if no reply is received within a set slice of time.

Because the traceroute command can threaten the network security by displaying a route to a host on the Internet and letting out the internal structure of the network to which the host is connected and the IP addresses of some routers on the network, its execution is usually disabled in most systems.

11.6 TCP Protocol and Applications

As a connection-oriented, reliable, and byte-stream protocol, Transmission control protocol (TCP) must establish a connection between two applications before providing other services. The two applications usually are at two different end points on a network.

11.6.1 TCP Segment, Header and Services

As shown in Figure 11.4, the unit of data that TCP transfers is called a TCP segment, and each segment has a 20-byte TCP header. Figure 11.9 gives the detail of a TCP header and segment.

	0-15 bits	16-31 bits
0-31	Source port number	Destination port number
32-63	Sequence number	
64-95	Acknowledgment number	
96-127	D-offset | Reserved | C | E | U | A | P | R | S | F	Window size
128-159	TCP checksum	Urgent pointer
	Options (bytes)	
	Data	

Fig. 11.9 TCP header and segment (as shownin Figure 11.5, the numbers in the blocks of the first row are the bit numbers that the blocks cover in one 32-bit word; D-offset in the fifth row covers four bits and represents Data offset, followed by four reserved bits; each Letter among C, E, U, A, P, R, S, and F in the fifth row covers one bit and represents CWR, ECE, URG, ACK, PSH, RST, SYN, and FIN, respectively. The numbers in the blocks at the left column are the numbers of bits composing a regular TCP header of 20 bytes).

The following is the description for each field in a TCP header.

- Source port number: it is to specify the sending application. Combined with the source IP address in the IP header for a datagram, the source port number can be used to call a socket (shown in Section 7.9.4), which is an interface provided by UNIX for the user to communicate with the network. The source couple of IP address and port number make up a client identifier that has two portions: a client IP address and a client port number, both of which can be used by the kernel when further networking services are provided.
- Destination port number: it is to specify the receiving application. Like the source port number, along with the destination IP address in the IP

header for a datagram, the destination port number can be used to call a socket at the destination host. And the destination couple of IP address and port number can compose a server identifier that has two portions: a server IP address and a server port number, both of which can be used by the kernel when further networking services are provided at the receiving host. A socket pair (including the source couple and destination couple) uniquely identifies a TCP connection between two end points in a network and supports inter-process communication on two machines in the network via the client-server model.

- Sequence number: it is a byte index number in the data stream transferred from the source host to the destination host. For the sending host, the TCP transfers segments with a unidirectional byte stream and marks each byte in sequence. If the SYN flag is set (which means a new connection being established), the segment number is an initial segment number and the actual first byte is with the segment number that is one more than the initial number, in which the acknowledged segment is considered. If the SYN flag is reset, it is the accumulated sequence number of the first byte of this segment for this data transference. As sequence number field covers 32 bits, it is reset to 0 when it is bigger than $2^{32} - 1$. As TCP is a full-duplex service to the application layer, each end of a TCP connection has to maintain two sequence numbers for the data flowing in two directions, sending and receiving, respectively.
- Acknowledgment number: it is a byte index number used by the receiving end. If the ACK flag is set, the acknowledgment number is the next sequence number that the receiving end is expecting. It is one more than the sequence number of the last successfully received byte by the receiver and acknowledges receipt of all the previous bytes. However, the first ACK segment acknowledges just the initial sequence number itself received by the receiver, but not data bytes.
- Data offset: it is the size of the TCP header (including options) in 32-bit words and also indicates the offset of the actual data from the start of the TCP segment. Without any option, the minimum header size is 20 bytes. Because the data offset field covers four bits, the maximum header size is 60 bytes, allowing for up to 40 bytes of options in the header.
- Eight flags: there are eight flag bits, CWR, ECE, URG, ACK, PSH, RST, SYN, and FIN. Congestion Window Reduced (CWR) is set if the sending end has received a TCP segment with the ECE flag set and has responded with congestion control mechanism. ECE is set for Explicit Congestion Notification-Echo (ECN-Echo). URG is set if the urgent pointer is valid. ACK is set if the acknowledgment number is valid, and ACK after the initial SYN segment should be always set. Push function (PSH) is set if the receiver should push the buffered data to the receiving application as soon as possible. RST is set to reset the connection. Synchronize sequence numbers (SYN) is set to initiate a new connection and only the first segment sent from each end has SYN set. FIN is set if the sender has

finished data sending.

- Window size: it gives the size of the receiving window, which specifies the number of bytes (starting from the sequence number in the acknowledgment filed) that the receiving end is currently willing to receive. As this field covers 16 bits, the window size is limited to 65 535 bytes.
- TCP checksum: it is used to make error-checking of the TCP header and data. TCP checksum of a TCP segment is calculated and stored by the sending end, and verified by the receiving end.
- Urgent pointer: it is an offset of the sequence number of the last byte of urgent data from the sequence number that is stored in the sequence number field of the segment. The urgent pointer is valid only if the URG flag is set.
- Options: it is variable from 0 to 320 bits. Different portions of options have different functions. For example, the first 8 bits are used for end of option list, the second 8 bits are used for padding, the closely following 32 bits are used for Maximum segment size (MSS), etc. For the detail, readers should refer to the references at the end of this chapter.

TCP's first specification was described in RFC 675 in 1974 (Cerf 1974). Later on RFC 793 (Postel 1981.3), 1122, 2581, and 3168 have given some clarifying and supplementary information for TCP.

TCP typically provides services including segmenting, sending, receiving, checking, and rearranging data. When some upper layer application does request for TCP's services, TCP firstly makes a TCP connection between the two ends according to the socket pair. It breaks application data into segments. Then it sends segments in a stream of bytes across the TCP connection between the two ends. When sending a segment, the sending TCP uses a timer, which is set how long the sender will wait for an acknowledgement receipt from the other end. If the acknowledgement receipt is not received in time, the segment is retransmitted. At the other end of the connection, the receiving TCP receives the segment, and makes a checksum on its header and data. If the segment is examined invalid, TCP discards it. If the segment is verified, TCP makes acknowledgment for this segment. Because TCP segments are transferred as IP datagrams and IP datagrams can be out of order or duplicated when being transmitted, the received TCP segments can be out of order or duplicated. The receiving TCP has to rearrange the data if necessary. If IP datagrams are duplicated, the receiving TCP has to discard the duplicated data. Then TCP passes the received data in the correct order to the appropriate application according to the destination port number in the TCP header.

Since each end of a TCP connection provides a finite amount of buffer space, the receiving TCP only allows the sending end to transmit as much data as the receiver's buffers can contain. In this way, TCP makes a flow control.

However, like UNIX's treating the contents of a file as a stream, TCP transfers a stream of bytes between two ends of a TCP connection and does not know the contents of the transmitted bytes at all. It relies on the applications on each end of the connection to interpret the contents of the byte stream.

11.6.2 TCP Applications

In this section, we will discuss some popular applications that perform their network services for users via TCP. We will introduce telnet and rlogin, ftp, and email.

11.6.2.1 Remote Login: Telnet and Rlogin

Remote login is a fundamental application for some other applications: such as remote command execution. When we log in a remote host with telnet or rlogin, we can do some tasks just like what we do on a local host, including searching for some files or directories, copying files, execute a program, etc.

Remote login provides a facility that a user can log on a remote host across a network or the Internet, which may be thousands of miles far away, without having a terminal directly connected to the remote host. Remote login adopts the client-server model. There are two popular remote login application protocols: telnet and rlogin. Telnet is a standard application protocol that exists in almost every TCP/IP implementation and almost all the operating systems. The specification of telnet protocol can be found in RFC 854 (Postel et al 1983) that was published in 1983. Rlogin was originally designed for 4.2BSD and to work between UNIX operating systems only. But in some systems that support BSD, rlogin can also work well. The specification of rlogin protocol was given in RFC 1282 (Kantor 1991) that was available from 1991.

Telnet

The syntax and function of telnet are as follows.

```
$ telnet [-options] hostname
```

Function: to connect and log in a remote host with the 'hostname' name via a network.

Common options:

-a: to try to login automatically;

-l: to login with a specified user name.

Note: The hostname can be the name of the destination host or its IP address in dotted decimal notation. Usually a user has to have a valid account on the remote host if the user wants to log in the host, but some remote hosts allow a user to log in without an authorized account.

Since the telnet works in the client-server model, the user who is logging in the remote host with telnet interacts with the telnet client. The telnet client provides two modes, command mode and input mode.

If a user types in the telnet command with a hostname as an argument, the telnet client prompts the 'login:' to let the user type in user name and password for the account on the remote host.

When a user logs in the remote host via the telnet command without an argument (a hostname), the telnet client brings the user into the command mode and prompts the 'telnet>' on the user's terminal. Then the user can type in some telnet commands after the prompt, such as to make a telnet connect to the remote host with hostname by entering 'open hostname', to close the telnet connection by entering 'close' or 'quit', to get a telnet command list by entering '?' or 'help', etc.

After a connection is built up between the client at the local host and the server at the remote host, every keystroke on the local terminal is sent to the remote host and the telnet goes into the input mode. In the input mode, there are also two options of entering style: one is a character at a time, which is the default option; the other is a line at a time. To go back to the command mode, the user can type the telnet escape key (ˆ]).

In the input mode, as the user is interacting with the telnet client, what the user types in will be treated as commands for the operating system on the remote host, and the telnet server on the remote host will interpret them to take the corresponding actions.

Since most of systems should be accessed with authority for security issues, it is not wise to make an account known to public. However, for academic studies, we build up a temporary network of two hosts: one is a client host, called chost; the other is a server host, called shost. And we have a user name, wang, on both hosts. The following command is used on the client host (chost) to log on the server host (shost).

```
$ telnet
telnet> mode line
telnet> open shost
Trying 192.168.11.104...
Connected to shost.
Escape character is '^]'
...
Login:  wang
Password:
...           (typing in some command lines to be sent to the shost)
$ date        (here $ is the login shell prompt of shost)
^]
telnet> z
$ fg
telnet shost
^]
telnet>quit
$
```

In the above example, we first enter the telnet command. The telnet> prompt is displayed on the screen, which means it goes into the command mode of the telnet client. When we type in mode line after telnet>, we switch the entering style to line-a-time mode. We type in open shost to try to make a connection between the chost and shost. Then it lets us enter the username and password for the account on the shost. When the connection is established, the telnet client enters the insert mode and sends whatever we type in after that point. We can type in some command lines just like what we do on the local shell because we now interact with the login shell on the shost. We type in date command and the system time of the shost is displayed on the screen.

After a while, we press CTRL-] back to the command mode of the telnet mode. Then we enter z that is a telnet client command for suspending the telnet execution and returning to the local host. The fg command can bring the telnet execution back to the foreground, just like what we do here.

Then we press CTRL-] back to the command mode of the telnet client. Finally, we enter quit to close the connection and quit the telnet operation.

Next example gives some experience of how to make a connection to some server via the Internet with the telnet client on the local host.

```
$ telnet mit.edu 79
Trying 18.9.22.69...
Connected to mit.edu.
Escape character is '^]'
guest
Student data loaded as of Oct 6, Staff data loaded as of Oct 6.
Notify Personnel or use WebSIS as appropriate to change your information.
Our on-line help system describes
  How to change data, how the directory works, where to get more info.
  For a listing of help topics, enter finger help@mit.edu. Try finger
  help_about@mit.edu to read about how the directory works.
 Directory bluepages may be found at http://mit.edu/communications/bp.
There were 3 matches to your request.
Complete information will be shown only when one individual matches
your query. Resubmit your query with more information.
For example, use both firstname and lastname or use the alias field.
        name:  Guest, Arthur Norman
  department:  Aeronautics And Astronautics
        year:  G
       alias:  A-guest
        name:  Guest, Iestyn W.
  department:  Lincoln Laboratory
       title:  Administrative Staff
       alias:  I-guest
        name:  Guest, Lindsay
  department:  Medical Department
       title:  Mental Health Resources Coordinator
       alias:  L-guest
Connection closed by foreign host.
$
```

In this example, we use a well-known port number (79) as an optical argument. 79 is the port number of finger server. As shown in Figure 11.9, every application has a unique port number in the TCP header, sometimes called well-known port number. Finger is also a TCP/IP application protocol (which was defined in RFC 1288 in 1991) (Zimmerman 1991) and has its corresponding command with the same name. Finger can be used to check the user information on a local or remote host. Here, we use guest as the user name, and the information shows that there are three matches.

The port number of the telnet server is 23. For readers to understand well the telnet protocol, it is necessary to know how the telnet client and server work. Figure 11.10 shows the function diagram of telnet client and server.

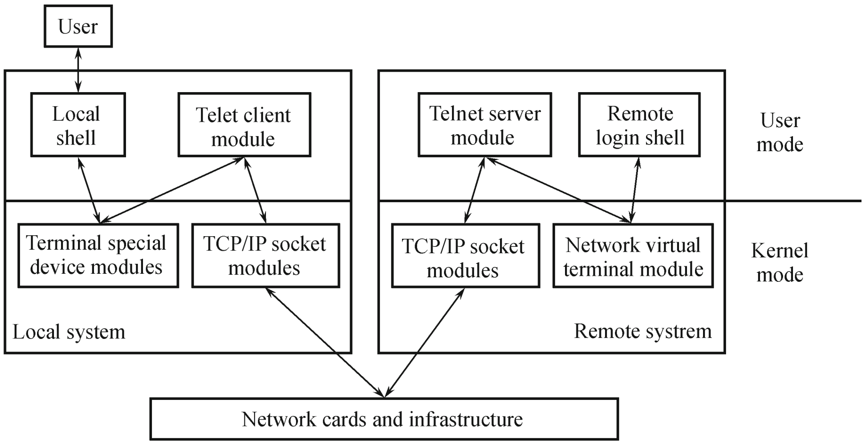


Fig. 11.10 The function diagram of telnet client and server.

As shown in Figure 11.10, when a connection is established between the local host and the remote host, the telnet client begins to interact with the user at the local terminal through a local shell. Via the TCP/IP protocols, the telnet client sends what the user typing in from the local terminal to the remote system and receives the response from the telnet server on the remote system. Then the reply from the telnet server is displayed on the local screen via the local shell.

The network virtual terminal (NVT), which was first defined in RFC 854, is an imaginary character device with an input device like keyboard and output device like a printer. The telnet client maps the user's input from the local terminal into the NVT, and the telnet server maps NVT into its input and its output. The client echoes what the user types to the output device by default. And data typed by the user on the local keyboard is sent to the server, and data received from the server is printed on the local screen.

The telnet server makes the NVT attached to the login shell on the remote host, where some remote interaction applications, such as text editors, can be used by the local user.

Rlogin

The syntax and function of rlogin are as follows.

```
$ rlogin [-options] hostname
```

Function: to connect and log in a remote UNIX host with the 'hostname' name via a network.

Common options:

-l: to login with a specified user name.

Note: The hostname can be the name of the destination host or its IP address in dotted decimal notation. If having a valid account on the remote host and logging in the remote host successfully, the user is put in the home directory and the login shell is executed. And all the hidden files for a local login are also executed for the remote login. If the user wants to interact just with the rlogin client rather than sending what entering to the rlogin server, the user can type a tilde key ($\sim$) as the first character of the line and followed by CTRL-Y (to suspend the client input and execute next commands on the local host), or CTRL-Z (to suspend the rlogin client), or CTRL-D (to terminate the rlogin). When the remote login is no use any more, typing in logout (or $\sim$ CTRL-D) can make the remote login close and return to the local system.

We can rewrite the start of the first example of the telnet command as the following.

```
$ rlogin shost
Login:  wang
Password:  ******
...
$
```

11.6.2.2 File Transfer: FTP

FTP is another popular TCP/IP application protocol, which is the Internet standard for file transfer or a complete file copied from one host to another. FTP also needs an account to log in the file server, but anonymous FTP allows a user to access a server.

FTP occupies a well-known port, numbered 21. FTP also uses the client-server model, but needs two types of TCP connections: control connection and data connection. Firstly, a FTP server on the remote host makes an open on the port 21, and waits for a connection from a FTP client on some local host. The FTP client creates a control connection by making an open to the port 21. The FTP holds the control connection for the whole time when the client and server are communicating to each other. Each time a file is transmitted between the client and server, a data connection is just produced. IP does different treatments for the control connection and data connection. Shown in Figure 11.5, there is a type-of-service field in the IP header. For a control connection, IP sets the minimize delay bit of its type-of-service field. For a data connection, IP sets the maximize throughput bit of the type-of-service field.

FTP supports a number of file types, such as ASCII or binary, and file structures, like byte-stream or record-oriented. Specification of FTP was published in RFC 959 (Postel et al 1985) in 1985.

In UNIX and other operating systems, there is a command, named ftp, corresponding to FTP application.

The syntax and function of ftp are as follows.

```
$ ftp [-options] hostname
```

Function: to log in and transfer files from or to a remote UNIX host with the 'hostname' name via a network.

Common options:

-i: to disable prompting during multiple file transfers;

-v: to display all the responses from the remote host.

Note: Because of the way that the FTP makes a connection between the ftp client and server, if the ftp server on the remote host has not done an open on the 21 port before the ftp client on the local host does a connection request, the connection cannot be established and an error message is displayed by ftp command. Also because of security issues, a user should have a verified account and access permission to transfer files from or to a remote host, and mostly, ftp servers allow a user more likely to download files from than to upload files to them.

Table 11.1 lists some of the ftp commands.

Table 11.1 Some of ftp commands

Command	Function
ascii	To set the ftp to ASCII mode in order to transfer ASCII-type files like text files
binary	To set the ftp to binary mode in order to transfer non-ASCII files including executable files and image files
cd	To change the working directory on the remote host
close	To close the ftp connection to the remote host but stay in the ftp command
dir rdir lfile	To store the listing of rdir directory on the remote host into lfile file on the local host
get rfile [lfile]	To download rfile file from the remote host to the lfile file in the current directory on the local host, or to the rfile file in the current directory on the local host without lfile argument
help [command]	To show the information of command, or a summary list of all ftp commands without command argument
ls [rdir]	To list the files in the rdir directory on the remote host, or in the current directory on the remote host without rdir argument
mget rfiles	To download multiple files listed in rfiles from the remote host to the current directory on the local host
mput lfiles	To upload multiple files listed in lfiles from the local host to the current directory on the remote host
open [hostname]	To try to open a connection to the remote host
put lfile [rfile]	To upload lfile file from the local host to the rfile file in the current directory on the remote host, or to the lfile file in the current directory on the remote host without rfile argument
quit	To end the ftp command
user [login_name]	To log in the remote host with a user name, login_name
! command	To execute the command on the local host

To try some ftp commands, we can use anonymous FTP. The site of ftp.pku.edu.cn is an anonymous FTP site, so you can use anonymous as a user name and your e-mail address as the password to access it, like the following example.

For example:

```
$ ftp ftp.pku.edu.cn
Connected to vineyard.pku.edu.cn.
220-Welcome to public FTP service in Peking University
220
User < vineyard.pku.edu.cn:<none>>:  anonymous
331 Please specify the password.
Password:
230-
230- Max 3 connections an IP
230-
230- Only downloading permitted
230-
230- @ Peking University
230-
230 Login successful
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
Linux
lost+found
mnt
open
welcome.msg
226 Directory send OK.
ftp:  received 43 bytes, time 0.00seconds 43000.00Kbytes/sec.
ftp>cd open
250 PORT command successful. Consider using PASV.
ftp>ls -l
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x 2 ftp    ftp  4096 Mar 04 2010 389ds
...
drwxr-xr-x 2 ftp    ftp  4096 Feb 25 2010 OpenGTS
...
drwxr-xr-x 5 ftp    ftp  4096 Apr 20 05:50 ftp
...
drwxr-xr-x 2 ftp    ftp  4096 Jul 19 2010 openca
...
226 Directory send OK.
ftp:  received 5243 bytes, time 0.02seconds 327.69Kbytes/sec.
ftp> cd openca
250 Directory successfully changed.
ftp>ls -l
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-r- - r-- 1 ftp ftp 9267407 Feb 25 2010 openca-base-1.0.2.tar.gz
-rw-r- - r-- 1 ftp ftp 397801 Feb 25 2010 openca-tools-1.1.0.tar.gz
226 Directory send OK.
ftp:  received 165 bytes, time 0.00seconds 165000.00Kbytes/sec.
ftp>get openca-tools-1.1.0.tar.gz
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for openca-tools-1.1.0.tar.gz
<397801 bytes>.
226 Transfer complete.
ftp:  received 397801 bytes, time 2.27seconds 175.63Kbytes/sec.
```

```
ftp>quit
221 Goodbye.
$
```

In this example, we use the ftp command to connect to ftp.pku.edu.cn with the user name and password of anonymous and the author's email address, respectively. After making a connection successfully, we use the ls command to list the home directory on the remote host, and move the directory to the open directory. We use the ls -l command to make a long list of the open directory. Then we move the current directory to the subdirectory of open – openca. We find a compressed file – openca-tools-1.1.0.tar.gz – over there. Then we download the file to the current directory on the local system. Finally we quit the fit command.

11.6.2.3 Email Protocols and Programs

Email (electronic mail) is a popular application in the Internet. Involved in email are some protocols and the email user and transfer agent programs, which will be discussed in this section. Before we discuss the protocols and programs involved in email, we give a whole picture about how the email technology to operate over a network or the Internet. Figure 11.11 displays the function diagram of email technology.

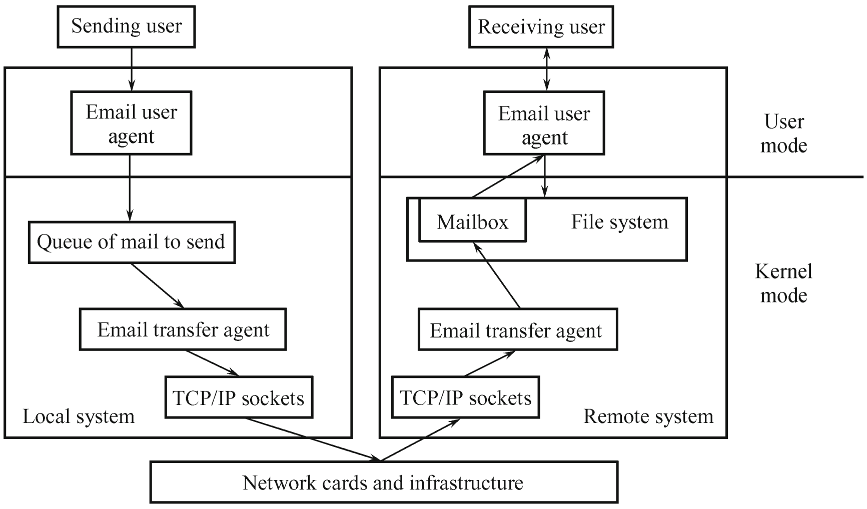


Fig. 11.11 The function diagram of email technology.

Typically, the process of sending an email from the sending host to receiving host is started from an email sender's launching an email user agent on a local host. The email user agent on the local host puts the email on the queue of mail to be sent. The email transfer agent (also called message transfer agent, or abbreviated MTA) on the local host manages the queue of email to be sent and sends the email through the TCP/IP sockets and a network (or the Internet) to the remote host. On the remote host, another email transfer agent receives the email and puts it in the mailbox that is usually a directory in the remote file system, and sends a message "you have mail" to the email receiver. The receiver uses the email user agent to dispose the email, or to store the email in a specified directory, or forward it, or reply to its sender, or delete it according to his own wishes. The email is simply a file stored in the directory.

Email Protocols

There are several protocols that support email application. The most important one is Simple Mail Transfer Protocol (SMTP). RFC 821 (Postel 1982) published in 1982 gives the detail of SMTP protocol, which describes how to transfer the message between the local and remote hosts, or how two email transfer agents communicate each other via a TCP connection. RFC 822 (Crocker 1982) in 1982 is also about the email technology, and defines the format of an email message, including an envelope, a header, and a body, which is transferred by using RFC 821. SMTP protocol also adopts the client-server model.

There are two common fashions to access emails: one is from the specified host that stores emails; the other is from different hosts that can access the specified email host that contains emails. The former uses the Post Office Protocol (POP). The latter adopts the Internet Message Access Protocol (IMAP, or called Interactive Mail Access Protocol). Nowadays, the latter is the more popular and convenient fashion, which allows a user, no matter who is at home, or school, or office, using different computers at different locations, on which different email user agents may be available, to access the email transfer agent on an email server that adopts the IMAP protocol. This is the most common fashion that the Internet Service Providers (ISPs) adopt.

POP is described in RFC 918 of 1984 (Reynolds 1984), and IMAP protocol is in RFC 1064 of 1988 (Crispin 1988). And both of them are updated in several later RFCs, respectively.

RFC 1521 of 1993 describes Multipurpose Internet Mail Extensions (MIME) or Multimedia Internet Mail Standard (MIMS) (Borenstein et al 1993). For email application, MIME is another important protocol that defines the extensions to the email body structure. It adds some new headers to what the RFC 822 defines, which tell the recipient the structure of the body. In fact, it makes multimedia – files including images, sounds, movies, and compute programs – to be able to attach to an email. The body can still be transmitted as SMTP, regardless of the mail contents. Because of MIME, the size of an email can become quite large, maybe several megabytes. It also needs the email user agent to adopt the extensions on the email body structure.

Email Message Structure

Email messages consist of three parts: envelope, header, and body.

The envelope includes two fields:

- To: field, which specifies the destination address of the email;
- From: field, which identifies the source address of the email.

The envelope is used by the email transfer agent to transfer and deliver the email via a TCP connection. In UNIX, each user has a user name and the user name is also the email address of the user, which can be used in the destination address or source address of an email. When sending an email across the Internet, the email address should follow the rules of the Internet Domain Name System.

The above two fields also belong to the header. Except them, the header includes several other fields:

- Attach: field, which contains a list of attachments (files) that will be sent along with the message;
- CC: field, which means carbon copies and identifies other destination addresses of the email;
- Date: field, which holds the date when the email is sent;
- Reply-to: field, which holds the address where an original email is received from and this reply-to email will be sent to;
- Subject: field, which specifies the subject of the message.

Each field contains a name with a colon and the field value. There are some more header fields that are added automatically by the email agents. They are used by the agents rather than the user and are not explicitly given here. The attachment files can be ASCII text files or multimedia types and subtypes of files, including image files (such as jpeq and gif files), audio files (such as basic and mp3 files), and video files (such as mpeg files). The multimedia types and subtypes of files are supported by MIME.

The body is the main content of the message.

When the email user agent on the local host gets the body of the message typed by the user, it adds some header fields to the body, and passes the whole to the email transfer agent. Then, the email transfer agent adds the envelop and some more header fields to what the email user agent sends to it, and sends the total result to the email transfer agent on the remote host.

Email Programs

Shown in Figure 11.11, protocols involved in the email technology are implemented with two parts, that is, two types of email programs: one is email user agent; the other is email transfer agent. The former is used to edit, read, and dispose the messages; the latter is applied to transfer messages between the local host and remote host via a network or the Internet, and deliver the messages. There are many email user agents available in UNIX, such as mail command, pine, and KMail.

The pine program is developed for UNIX by the University of Washington.

Like the pico editor, introduced in Section 3.2, pine has the really similar user environment as the pico does, where a user can compose, send or read messages, and manage the address book. If the pine program is available in the system, users can get familiar to it quickly after using the pico.

KMail is a GUI program and executes under the KDE desktop environment (see in Section 2.5). Having the experience of Microsoft Windows, it is easy for readers to learn how to use KMail by themselves. In this book, we will discuss just the mail command.

In UNIX, the most popular email transfer agent is Sendmail. The system administrator sets up the local email transfer agent, and users can make a choice on which email user agent they use. The email transferring usually occupies the port 25.

Figure 11.11 shows a mailbox in the file system. In UNIX, the mailbox is usually the directory /usr/mail or some other directory, and emails are files in the directory. Each time a user logs in the system, if there is new email for him, the system gives notice "you have mail" on the screen. Users can use the mail command or pine program to read the mail in the mailbox.

Mail Command

In BSD, the mail program is called Mail and the following mail command can invoke Mail. In UNIX System V, the mail program is mailx. Here introduces the mail command. The syntax and function of mail are as follows.

```
$ mail [-options] [destination_address]
```

Function: to send or receive email.

Common options:

-b ad1: to specify the address ad1 where a blind carbon copy is sent to;

-c ad2: to specify the address ad2 where a carbon copy is sent to;

-P: to make all messages displayed with full headers;

-s: to specify the subject for email to be sent.

When sending email, just type the mail command with the destination addresses as arguments and in the way of entering other UNIX commands, like the following:

```
$ mail wangwei@hostname
```

After entering the mail command, it goes into the mail operation. You can type in the main body of the email message line by line. When finished, you press CTRL-D at a beginning of a new line to end the mail command and make it send the email.

When reading email, you enter mail without argument, just like:

```
$ mail
```

If there are some new emails waiting for your disposition, a list of message headers is displayed on the screen.

11.7 UDP Protocol and Applications

Different from TCP protocol that is connection-oriented, reliable, and byte-stream, UDP is a transport layer protocol that is simple, unreliable, and datagram-oriented. With the features of UDP, the applications based on UDP are different from ones of TCP.

11.7.1 UDP Protocol

RFC 768 (Postel 1980) published in 1980 gives the detailed description of the specification of UDP. From the studies in Section 11.6, we have known that TCP has to segment the application data. However, each UDP datagram is usually just one operation output of a process. And each UDP datagram is often sent with one IP datagram. UDP passes the application datagrams to the network layer without guarantee that all the datagrams can reach their destination.

Figure 11.12 displays the UDP header and datagram.

	0−15 bits	16−31 bits
0−31	Source port number	Destination port number
32−63	UDP length	UDP checksum
	Data	

Fig. 11.12 UDP header and segment (as shown in Figure 11.5, the numbers in the blocks of the first row are the bit numbers that the blocks cover in one 32-bit word; the numbers in the blocks at the left column are the numbers of bits composing a regular UDP header of 8 bytes).

Compared to TCP header, UDP header is quite simple. In UDP header, the source and destination port numbers have the same meaning as in TCP header. Since the protocol field in IP header specifies the protocol (shown in Figure 11.5), the source and destination port numbers can identify UDP applications or TCP applications independently.

The UDP length field gives the datagram length in bytes, including the UDP header and UDP data. Similar to TCP checksum, the UDP checksum makes error-checking on both the UDP header and UDP data. However, the UDP checksum is optional rather than mandatory such as in TCP.

11.7.2 UDP Applications

Discussed in Section 11.2.5.3, unicast is always a connection-oriented pocket mode. The connectionless packet switching mode has broadcast and multicast functions, which can save network resources if the same data have to be transmitted to several destinations. TCP is connection-oriented, which means an explicit connection established between two application ports on two hosts. Therefore, broadcasting and multicasting can only be based on UDP, which is connectionless.

Considered within a LAN or MAN, many hosts share a network. When a host in the network wants to deliver a datagram to every other host on the network, the broadcasting can do this task. When a host on the network wants to send a datagram just to a group of the hosts on the network, multicasting can deal with it. A set of hosts on the network can be divided into a multicast group.

In fact, broadcasting or multicasting is relying on the filtering function of the TCP/IP protocol stack (shown in Figure 11.4). When a datagram reaches a host via the network or the Internet, it will go through all the modules in the protocol stack until up to the application layer. Let's take the right column in Figure 11.4 as an example. Firstly, when a datagram reaches a host, the network card (here an Ethernet card) on the host and Ethernet protocol check if the destination address of the datagram matches the hardware address of the card, the broadcast address or multicast address, and makes the first filtering. If the datagram passes the first filtering, it goes up to the IP layer. The IP examines the datagram according to the IP address in the IP header and does the second filtering. If the datagram also passes the second filtering, it goes up to the transport layer and accepts the examination of UDP in terms of destination port number and checksum in the UDP header. It is the third filtering. At any filtering, if the datagram does not match, it will be discarded.

Finally, we discuss another important protocol in the TCP/IP suite, Domain Name System (DNS), even though it is not dedicated to UDP applications. DNS is usually used to map between hostnames and IP addresses, and to provide routing information for other TCP/IP applications. And DNS is usually done first by an application because the application must convert a hostname given by a user to an IP address before it can request UDP to send its data or ask TCP to make a connection. DNS also adopts the client-server model.

DNS provides a distributed database whose data are scattered in different web sites on the Internet. As the Internet is giant, dynamic and distributed, any single site can not contain all the IP addresses of all the hosts in the Internet. Typically, each site manages its own DNS database and provides a server program to support the query from the clients on other hosts via the Internet.

BSD has its solution to DNS, called Berkeley Internet Name Domain

(BIND), including resolver and BIND server (also called named). The resolver is an interface for users provided as a DNS client. TCP/IP applications can access DNS through the resolver. The resolver invokes gethostbyname program to map a hostname into an IP address and invokes getnamebyaddr program to map an IP address into a hostname.

11.8 Summary

In this chapter, we have discussed UNIX in the Internet and computer networking. UNIX has many original contributions to the development history of the computer networking and Internet. Most of the networking protocols were initially implemented on UNIX and most of the Internet services are provided by server processes running on the UNIX operating system.

Since the late 1960s, in computer network engineering, the Request for Comments (RFCs) have become the official publications that can help exchange information among the global computer network researchers.

According to their scales, computer networks can be divided into three types: LAN, MAN, and WAN. In a network with the client-server model, every client is connected to the server and also each other. The resources in a server can be shared with clients, but a client does not share any of its resources with servers and other clients.

Logically, a network can be mapped into the TCP/IP model. The TCP/IP model consists of four layers, which are Application, Transport, Network, and Link layers. The application layer carries out the tasks of application protocols. The transport layer undertakes the tasks of transmission protocols. The network layer realizes the missions of the routing protocols. And the link layer settles all the issues including the access to the transmission medium and the construction of communication device and network infrastructure.

Each layer of the TCP/IP model should fulfill the task that is defined by the protocols and facilities. These protocols make up the TCP/IP suite. The software implementation of the TCP/IP protocol suite is also called the protocol stack. As the TCP/IP protocol suite has several layers, the protocol stack has several modules, each of which implements the function of its corresponding layer. When a certain application sends data through each layer with some protocols, the data is delivered down via each module of the protocol stack until it is transmitted as a stream of bits across the network.

Since networking operating systems are a much younger and flourishing member of operating system family, there are still new emerging characters and potential for them. However, networking operating systems basically should have the benefits of sharing computer resources among users, providing a development cooperation platform for variedly located researchers, and enhancing the reliability of the whole computer system, except the common services that operating systems usually have.

The Internet Protocol (IP) is quite popular protocol. All the TCP, UDP, ICMP, and IGMP datagrams are transmitted with the IP. However, the IP provides an unreliable and connectionless delivery service.

Along with IP protocol, two commands, ping and traceroute, have been discussed. Ping is usually used to test a network connection. Traceroute can be used to trace the route form one host to another.

The Transmission Control Protocol (TCP) is a connection-oriented, reliable, and byte-stream protocol, which must establish a connection between two applications at two different end points on a network before providing other services. We also have discussed some popular applications: telnet and rlogin, ftp, and email. Telnet and rlogin are two popular remote login application protocols. Telnet is a standard application protocol that exists in almost every TCP/IP implementation and almost all the operating systems. Rlogin was originally designed for 4.2BSD and to work between UNIX operating systems only. The File Transfer Protocol (FTP) is another popular TCP/IP application protocol, which is the Internet standard for file transfer. FTP needs an account to log in the file server, but anonymous FTP allows a user to access a server.

There are several protocols that support email application. The most important one is SMTP. There are two common fashions to access emails: one is from the specified host that stores emails; the other is from different hosts that can access the specified email host that contains emails. The former uses the Post Office Protocol (POP). The latter adopts IMAP. MIME is another important protocol that defines the extensions to the email body structure.

The User Datagram Protocol (UDP) is a transport layer protocol that is simple, unreliable, and datagram-oriented. Broadcasting and multicasting can only be based on UDP because it is connectionless.

The Domain Name System (DNS) is usually used to map between hostnames and IP addresses, and to provide routing information for other TCP/IP applications.

Problems

Problem 11.1 Try to refer to RFC 681 by yourself via the Internet and read its contents.

Problem 11.2 List some more benefits that the computer networking brings to us, except ones that have been given in Section 11.2.2.

Problem 11.3 Try to inquire into the computer network that your campus or company adopts. What operating system, model, and protocols are used in your computer network?

Problem 11.4 Give some application protocols that adopt the client-server model. Explain how a client and server cooperate to work with each other.

Problem 11.5 Try to find an application that uses a peer-to-peer model and explain how it works.

Problem 11.6 Try to find RFCs for TCP and UDP protocols via the web site given in the references and do some studies on them. Compare them in detail. Give some examples about where they are applied to, respectively?

Problem 11.7 Try to find Stream control transmission protocol (SCTP) via the web site in the references and do some researches on how SCTP works.

Problem 11.8 Please analyze the reasons of the unreliability of IP delivery. How can make a reliable delivery?

Problem 11.9 What are ARP and RARP used for? Try to do some research on ARP and RARP protocols via the web site given in the references.

Problem 11.10 Please analyze how the telnet protocol does remote login with the assistance of TCP, IP, and Ethernet protocols by referring to Figure 11.3.

Problem 11.11 Try to create a program that can implement the encapsulation of data from the application layer to transport layer.

Problem 11.12 How can you make the protocol stack fit into the hierarchy structure of the operating system? Please depict the hierarchy structure of your version of networking operating system.

Problem 11.13 Explain how the IP handles the datagram fragmentation after doing some research on the IPv4. Try to create a program to implement the IP datagram fragmentation.

Problem 11.14 Give a description on how IP does routing.

Problem 11.15 Try to program to accomplish some main functions of TCP, including segmenting, sending, receiving, checking, and rearranging data, after doing a detailed research on TCP protocol.

Problem 11.16 Refer to RFC 854 and do researches on how NVT (network virtual terminal) works.

Problem 11.17 Try to search some anonymous FTP sites over the Internet. Log in one of them with the anonymous user name and try some harmless ftp commands on the site.

Problem 11.18 Try to do researches on RFCs 821, 822, and 1521, and analyze the difference between two body structures of SMTP and MIME.

Problem 11.19 Try to study TCP and UDP protocols and analyze why UDP is unreliable when compared to TCP.

Problem 11.20 How can DNS handle information of the hosts existing in the Internet?

References

Borenstein N, Freed N (1993) MIME (Multipurpose Internet Mail Extensions) Part One: Mechanisms for specifying and describing the format of Internet message bodies. RFC 1521.http://tools.ietf.org/html/rfc1521. Accessed 10 Oct 2010

Cain B, Deering S, Kouvelas I et al (2002) Internet Group Management Protocol, Version 3. RFC 3376. http://tools.ietf.org/html/rfc3376. Accessed 10 Oct 2010

Cerf V, Dalal Y (1974) Specification of Internet Transmission Control Program. RFC 675. http://tools.ietf.org/html/rfc675. Accessed 10 Oct 2010

Comer DE (1998) Computer networks and Internets. Prentice Hall, Upper Saddle River, New Jersey

Comer DE, Stevens DL (1998) Internetworking with TCP/IP vol III: Client-server programming and applications, windows sockets version. Prentice Hall, Upper Saddle River, New Jersey

Conta A, Deering S, Gupta M (ed) (2006) Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. RFC 4443. http://tools.ietf.org/html/rfc4443. Accessed 10 Oct 2010

Crispin M (1988) Interactive mail access protocol, 2nd edn. RFC 1064. http://tools.ietf.org/html/rfc1064. Accessed 13 Oct 2010

Crocker DH (1982) Standard for the format of ARPA Internet text messages. RFC 822. http://tools.ietf.org/html/rfc822. Accessed 13 Oct 2010

Deering S (1989) Host Extensions for IP Multicasting. RFC 1112. http://tools.ietf.org/html/rfc1112. Accessed 10 Oct 2010

Deering S, Hinden R (1998) Internet Protocol, Version 6 (IPv6) Specification. RFC 2460. http://tools.ietf.org/html/rfc2460. Accessed 10 Oct 2010

Fenner W (1997) Internet Group Management Protocol, 2nd edn. RFC 2236. http://tools.ietf.org/html/rfc2236. Accessed 10 Oct 2010

Finlayson R, Mann T, Mogul JC at el (1984) A Reverse Address Resolution Protocol. RFC 903. http://tools.ietf.org/html/rfc903. Accessed 10 Oct 2010

Holmgren S (1975) Network UNIX. RFC 681. http://tools.ietf.org/html/rfc681. Accessed 10 Oct 2010

Kantor B (1991) BSD rlogin. RFC 1282. http://tools.ietf.org/html/rfc1282. Accessed 13 Oct 2010

Kohler E, Handley M, Floyd S (2006) Datagram congestion control protocol (DCCP). RFC 4340. http://tools.ietf.org/html/rfc4340. Accessed 10 Oct 2010

Ong L, Yoakum J (2002) An introduction to the stream control transmission protocol (SCTP). RFC 3286. http://tools.ietf.org/html/rfc3286. Accessed 10 Oct 2010

Plummer DC (1982) An Ethernet Address Resolution Protocol, or converting network protocol address to 48 bit Ethernet address for transmission on Ethernet hardware. RFC 826. http://tools.ietf.org/html/rfc826. Accessed 10 Oct 2010

Postel JB (1980) User datagram protocol. RFC 768. http://tools.ietf.org/html/rfc768. Accessed 10 Oct 2010

Postel JB (ed) (1981) Internet Protocol. RFC 791. http://tools.ietf.org/html/rfc791. Accessed 10 Oct 2010

Postel JB (1981) Internet Control Message Protocol. RFC 792. http://tools.ietf.org/html/rfc792. Accessed 10 Oct 2010

Postel JB (1981) Transmission Control Protocol. RFC 793. http://tools.ietf.org/html/rfc793. Accessed 10 Oct 2010

Postel JB (1982) Simple Mail Transfer Protocol. RFC 821. http://tools.ietf.org/html/rfc821. Accessed 13 Oct 2010

Postel JB, Peynolds JK (1983) Telnet Protocol specification. RFC 854. http://tools.ietf.org/html/rfc854. Accessed 13 Oct 2010

Postel JB, Reynolds JK (1985) File Transfer Protocol (FTP). RFC 959. http://tools.ietf.org/html/rfc959. Accessed 13 Oct 2010

Reynolds JK (1984) Post Office Protocol. RFC 918. http://tools.ietf.org/html/rfc918. Accessed 13 Oct 2010

Stevens WR (2002) TCP/IP illustrated, volume 1: The protocols. China Machine Press, Beijing

Stevens WR (2002) TCP/IP illustrated, volume 3: TCP for transactions, HTTP, NNTP and UNIX ® domain protocols. China Machine Press, Beijing

Stewart R (ed) (2007) Stream control transmission protocol. RFC 4960. http://tools.ietf.org/html/rfc4960. Accessed 10 Oct 2010

Wright GR, Stevens WR (2002) TCP/IP illustrated, volume 2: The implementation. China Machine Press, Beijing

Zimmerman D (1991) The finger user information protocol. RFC 1288. http://tools.ietf.org/html/rfc1288. Accessed 13 Oct 2010