

Communications and Computer Networks

The fundamental purpose of a communication network is the exchange of information between two parties. An example is the exchange of voice signals between two telephones on the same network. It is often impractical (if not outright impossible) to be directly or point-to-point connected. This is for one (or both) of the following contingencies.

1. The devices are very far apart and cannot share a dedicated link.
2. There is a set of devices, each of which requires a link to many others at the same time.

The solution to the problem is to attach each device to a communication network. There are two categories into which communication networks are organized: local area networks (LANs) and wide area networks (WANs).

Wide Area Networks

Wide area networks generally cover a large geographical area, typically multiple cities, countries, or continents. Typically, a WAN consists of a number of interconnected switching nodes called *routers*. A transmission from any device is routed through these internal nodes to the specified destination device. These switching nodes are not concerned with the content of the data; rather, their purpose is to provide a switching facility that will move the data from node until they reach their destination. The switching employed is normally:

1. **Circuit Switching:** In a circuit switching network, a dedicated communications path is established between two stations through the nodes of the network. Data generated by the source station are transmitted along the dedicated path as rapidly as possible. Since the path is established *a priori* there is no delay involved. The most common example of circuit switching is the telephone network. Disadvantages are the cost involved; for example one pays a fixed rate for a phone call even when the two parties do not talk.
2. **Packet Switching:** Data is sent out in a sequence of small chunks called *packets*. Each packet is passed through the network from node to node

alone some path leading from source to destination. At each switching node, the entire packet is received, stored briefly in a queue, and then transmitted to the next node. This is commonly used in computer to computer communications. An example of packet switching is the postal network. Whenever the network becomes overloaded, computer using the network must wait before they can send additional packets. However, since multiple computers can share the network bandwidth, fewer connections are required and the cost is kept low.

Usually, WANS range from 1.5Mbps to 155 Mbps (where Mbps is million bits per second). Examples of WAN technologies are: ARPANET (Section 2.7 of Comer [1]), X.25 (Section 2.11.1 of Comer [1]), Frame Relay, ISDN (Integrated Services Digital Network), ATM (Asynchronous Transfer Mode) (Section 2.6 of Comer [1]).

Local Area Networks

The scope of the LAN is small, typically a single building or a cluster of buildings. It is also usually the case that the LAN is owned by the same organization that owns the communicating computers. The difference in the geographical scope of LANs leads to technologies that are different from those of WANs.

Traditionally, LANs make use of a broadcast network approach rather than a switching network. With a broadcast network, there are no intermediate switching nodes. Rather there is a network interface card (NIC) that directly connects each computer to the network (medium) shared by the other computers. A transmission from any computer is broadcast to and received by all the other computers. Data is usually transmitted in packets. Because the medium is shared, only one computer can transmit a packet at a time.

Examples of LAN technologies include: Ethernet (IEEE 802.3) (Section 2.4 of Comer [1]), FDDI (token ring passing LAN) (Section 2.5 of Comer [1]).

Internetworking concept: Protocols and Architecture

The primary goal of internetworking is a system that hides the details of the underlying network hardware while providing universal communication services. There are two approaches to hiding network details:

1. **Application-level interconnection:** In such systems, an application program executing on each computer understands the details of the network connections and interoperates across those connections with application programs on other computers. Take for instance an electronic mail system, where an application mail program is configured to forward a message to another mail program on another computer. The path from the source and destination may involve different networks, but this does not matter as long as the mail systems on the various computers cooperate by forwarding the message. This, however, leads to cumbersome communication for the following reasons:
 - (a) Adding new functionality to the system means building a new application program for each computer.

- (b) Adding new network hardware means modifying existing application programs on the computer.
- (c) Finally, each application program (say electronic mail system, web browser) would need to understand the network connections for the computer, resulting in duplication of code.

2. **Network-level interconnection:** A network-level interconnection provides a mechanism that delivers small packets of data from the original source to destination without using intermediate application programs. This separates the data communication duties from application programs, and existing technologies can be modified while the application programs on computers remain unchanged.

The key to designing universal network-level interconnection is the concept of internetworking. The idea is to detach the notions of communications from the details of network technologies, and hide low-level details from the user. Thus, instead of a single module for performing communications, there is a *structured layer* of modules that implements the communication function. Each module/layer communicates with its peer module/layer on another computer via a series of *protocols*, which may be defined as a set of rules governing the exchange of data between two entities. The layers and protocols together define the *network architecture*.

Consider the situation in Figure 1, where a French and English philosopher wish to communicate, but neither speaks the others language. However, both philosophers have Dutch translators on their respective staffs. Each translator has a secretary who communicates with the other translator's secretary via fax. Let us suppose the English philosopher has a message for his French counterpart, and his message in English reads *I like rabbits*. The philosopher conveys this message to his translator, who translates the message into Dutch which reads *Ik vind konijnen leuk*. The translator also adds header to this message for his counterpart, which indicates that the language employed is Dutch. The translator gives his information to the secretary who adds a fax number to the message and faxes this information to her counterpart. The other secretary on receipt of this message, removes the fax number from the message and passes on the remaining information to the French translator who, in turn, translates the Dutch message into an equivalent French message that reads *J'aime bien les lapins*. Finally, he passes this translated message to the French philosopher.

Here, both the source and the destination have three layers of communication. Layers 1,2, and 3 represent the secretaries, translators, and philosophers respectively. The protocol employed between the peer to peer translator layers is the Dutch language, while fax is the protocol employed by the secretary layers. Now suppose, each philosopher does not have Dutch translators on their staffs but have Hindi translators instead. The communication between the two philosophers will take place and the only change is that layer 2 represents Hindi translators with the peer to peer protocol being the Hindi language. Furthermore, this change does not affect the philosopher and the secretary layers.

In very general terms communication involves three agents: applications, computers, and networks. One example of an application is a file transfer (ftp) operation. These applications execute on computers, that can often support multiple simultaneous applications. Computers, in turn, are connected to networks and the data to be exchanged are transferred by the network from one

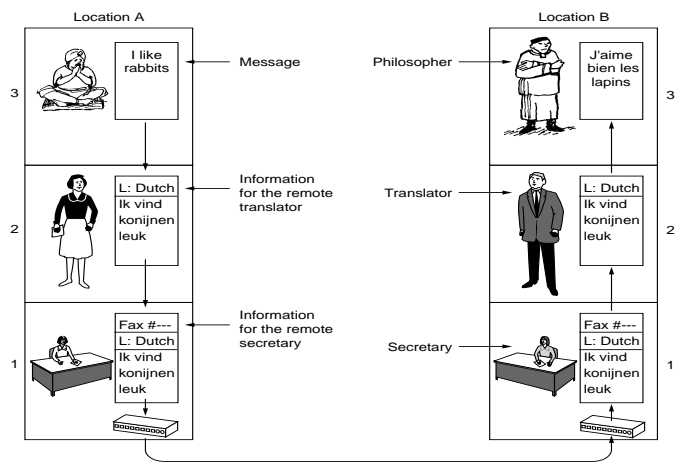


Figure 1: The philosopher-translator-secretary architecture

computer to another. With these three agents in mind, it is natural to organize the communication task into three relatively independent layers.

1. **Network Access Layer:** This layer is concerned with the exchange of data between a computer and the network to which it is attached. The sending computer must provide the network with the address of the destination computer, so that the network will route the data to the appropriate destination. The sending computer might also like to invoke services such as priority, that might be provided by the network. The specific software employed at this layer depends on the type of network used, and different standards developed for circuit/packet switching etc. Thus, the layers above the network access layer need not be concerned about the specifics of the network to be used. Thus, two computers attached to a physical network can now communicate.
2. **Transport Layer:** Regardless of the nature of the applications that are exchanging data (ftp,http), there is usually a requirement that data be exchanged reliably. Thus, it makes sense to collect those mechanisms in a common layer called the transport layer.
3. **Application Layer:** The application layer contains the logic needed to support various user applications. For each type of application, a separate module is needed that is characteristic to that application.

Moreover, these three layers communicate with their peers on other computers via a set of protocols analogous to the communication between the two philosophers mentioned above. These protocols require the exchange of various control information, including the network address of the destination computer, and an address that is specific to the particular application (the latter addresses are called *ports* or *service access points* (SAPs)). Suppose an application, associated with SAP 1 on computer X wishes to send a message to another application associates with SAP 2 on computer Y. The application at X hands over its message to the transport layer. The transport layer may break this message into smaller pieces to make it manageable. To each of these pieces, the transport layer appends a transport header containing protocol control information. This process is known as *encapsulation* and the combination of data from the next higher layer and control information is known as a *protocol data unit* (PDU). For example, the header may contain the following information:

1. **Destination SAP:** When the destination transport layer receives the transport PDU, it must know the application to which the data is to be delivered.
2. **Sequence Number:** Since the transport layer is sending a sequence of PDUs, it numbers them sequentially, so that if they arrive out of order the destination transport layer may reorder them.
3. **Error Detection Code:** The sending transport layer may also include a code that is a function of the data portion of the transport PDU. The receiving transport layer protocol performs the same calculation. If there is any discrepancy, the receiver can discard the transport PDU and take appropriate corrective action.

The transport layer hands over each PDU to the network access layer. This layer appends further control information to the transport PDU (which is regarded as the data portion) creating a network access PDU. For example, the header may contain the following information:

1. **Destination Computer Address:** The network must know which computer the data is to be delivered.
2. **Facilities request:** The network access protocol might want to make use of certain network resources, such as priority.

The network access layer is not concerned with the details of the transport header. The network accepts the network access PDU from computer X and delivers it to computer Y. The network access module in Y receives the PDU, strips off the header, and transfers the enclosed transport PDU to the transport layer module. The transport layer examines the transport PDU header, and, on the basis of the SAP field delivers the enclosed data to the appropriate application; in our case this was the file transfer module in computer Y. The overall process is illustrated in Figure 2.

Common protocol architectures

Two protocol architectures have served as the basis for the development of communication standards:

1. The TCP/IP protocol suite.
2. The OSI reference model.

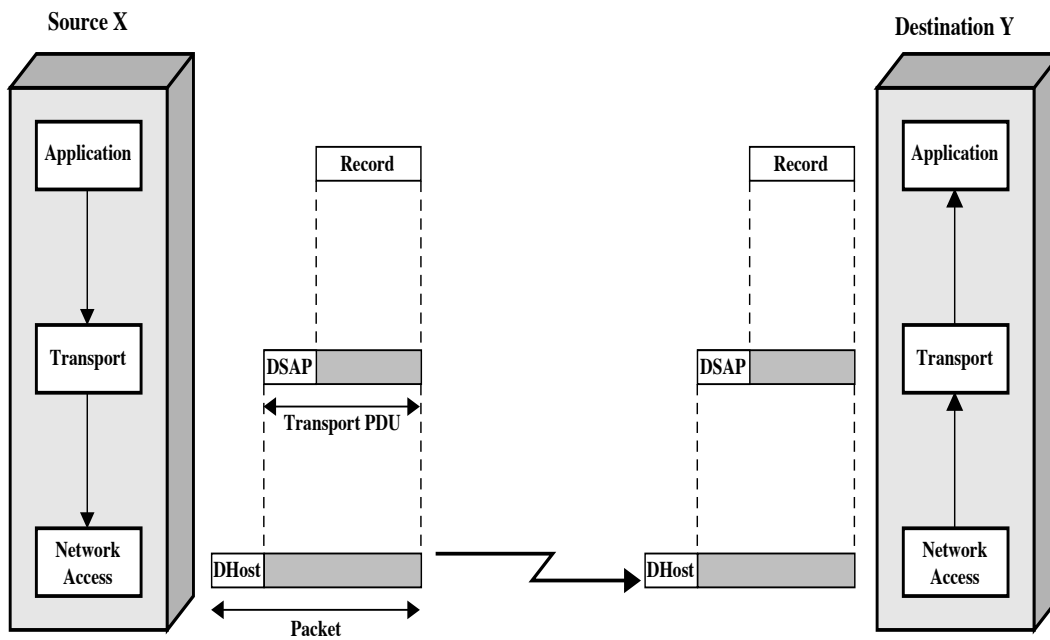
We provide a brief overview of these two architectures in this section. Finally, we highlight the important differences between these two models.

The TCP/IP protocol suite

TCP/IP is the result of protocol research and development conducted on the experimental packet-switched network ARPANET, funded by the Defense Advanced Research Projects Agency (DARPA) and is generally referred to as the TCP/IP protocol suite.

The communication task in TCP/IP is organized into five relatively independent layers:

1. **Physical Layer:** The physical layer is the network hardware layer, and is concerned with the characteristics of the transmission medium, the nature of the signals, data rate, and related matters.
2. **Network Interface Layer:** This is the lowest software layer, responsible for accepting IP datagrams, and transmitting them over a specific network in frames. This layer is responsible for communication between two computers attached to the same computer network.
3. **Internet Layer:** The internet layer handles communication between the source and the destination computers. Both these computers may be attached to two different computer networks. The internet layer accepts a



DSAP = destination service access point
 DHost = destination host

Figure 2.5 Operation of a Protocol Architecture

Figure 2:

request to send a packet from the transport layer along with an identification of the machine to which the packet is to be sent. It encapsulates the packet in an IP datagram, uses a routing algorithm to determine whether to deliver the packet directly or sent it to a router, and passes the datagram to the network interface layer for transmission. It also receives datagrams, checks their validity, and uses the routing datagram to check whether the datagram is to be processed locally or sent to another router. For datagrams addressed to the local machine, the internet layer deletes the datagram header and delivers the packet to the transport layer. The internet layer adopts the internet layer (IP) protocol. The aim of the IP protocol is to provide *best-effort* delivery. It does not attempt to correct errors, although it sends ICMP (Internet Control Message Protocol) error and control messages as needed.

4. **Transport Layer:** It's duty is to primarily provide communication between the application program on the source computer with that on the destination computer. It divides the stream of data being transmitted by the application program into small packets, and passes each packet along with the destination computer address to the internet layer for transmission. It also provides reliable transport at the receiving end ensuring that the data arrives without error and in sequence.
5. **Application Layer:** This contains the necessary logic for a particular application. The application layer interacts with the transport layer to send and receive data.

The application and transport layers are called *end-to-end* since they are only implemented on the source and destination computers. A router, for instance, need not have a transport and application layers. Typically, the transport, internet, and network interface layers are implemented in software and in the operating system of the computer. The application layer is a software layer that resides in the user disc space. Common examples of TCP/IP protocols include:

1. **Application Layer:** Telnet, FTP, e-mail etc.
2. **Transport Layer:** TCP (Transmission Control Protocol), UDP (User Datagram Protocol).
3. **Internet Layer:** IP (Internet Protocol), ICMP (Internet Control Message Protocol), IGMP (Internet Group Management Protocol).
4. **Network Interface Layer:** Device Driver and interface card.

The OSI reference model

The OSI (Open Systems Interconnect) proposed a standard known as the OSI reference model. It contains 7 conceptual layers organized as follows:

1. Physical Hardware Connection.
2. Data Link Layer.
3. Network Layer.
4. Transport Layer.

5. Session Layer.
6. Presentation Layer.
7. Application Layer.

The OSI reference model was designed as a framework for developing protocol standards. The designers of OSI assumed that this model and the protocols developed within this model would dominate computer communications, eventually replacing rival models such as TCP/IP. However, this is not happened. It is the TCP/IP architecture that has come to dominate. Thus, our emphasis in this course will be on the TCP/IP protocol suite. The book by Tanenbaum [3] is a good introduction to the OSI 7 layer model.

Differences between TCP/IP and OSI models:

There are two subtle and important differences between the TCP/IP layering and OSI reference model. This section will provide a short overview of some of the important differences; more details can be found in Section 11.6 of Comer [1].

1. **Link-Level vs End-To-End Reliability:** In the OSI model, protocol software detects and handles errors at all layers. In contrast to such a scheme, TCP/IP assumes that reliability is an end-to-end problem. There is little or no reliability in most TCP/IP network interface software; rather this is the duty of the transport and internet layers. The resulting freedom from interface layer verification makes TCP/IP software much easier to understand and implement correctly.
2. **Locus of Intelligence and Decision Making:** The OSI layer model assumes that the network is a utility that provides transport service. The network vendor also handles problems like routing, flow control, acknowledgements internally making transfers reliable. In short, the network is a complex independent system to which one may attach relatively simple computers; the hosts themselves participate minimally in network operations. In contrast, TCP/IP requires hosts to participate in almost all of the network protocols such as end-to-end error detection and recovery. Thus, a TCP/IP internet can be viewed as a simple packet delivery system to which intelligent hosts attach.

Suggested Readings

1. Chapters 2,3, and 11 of Comer [1].
2. Chapter 1 and 2 of Stallings [2].
3. Chapter 1 of Tanenbaum [3].

References

References

- [1] D.E. COMER, *Internetworking with TCP/IP: Principles, Protocols, and Architectures*, 4th edition, Prentice Hall, NJ, 2000.

- [2] W. STALLINGS, *Data and Computer Communications*, 6th edition, Prentice Hall, NJ, 2002.
- [3] A. TANENBAUM, *Computer Networks*, 4th edition, Prentice Hall, NJ, 2003.