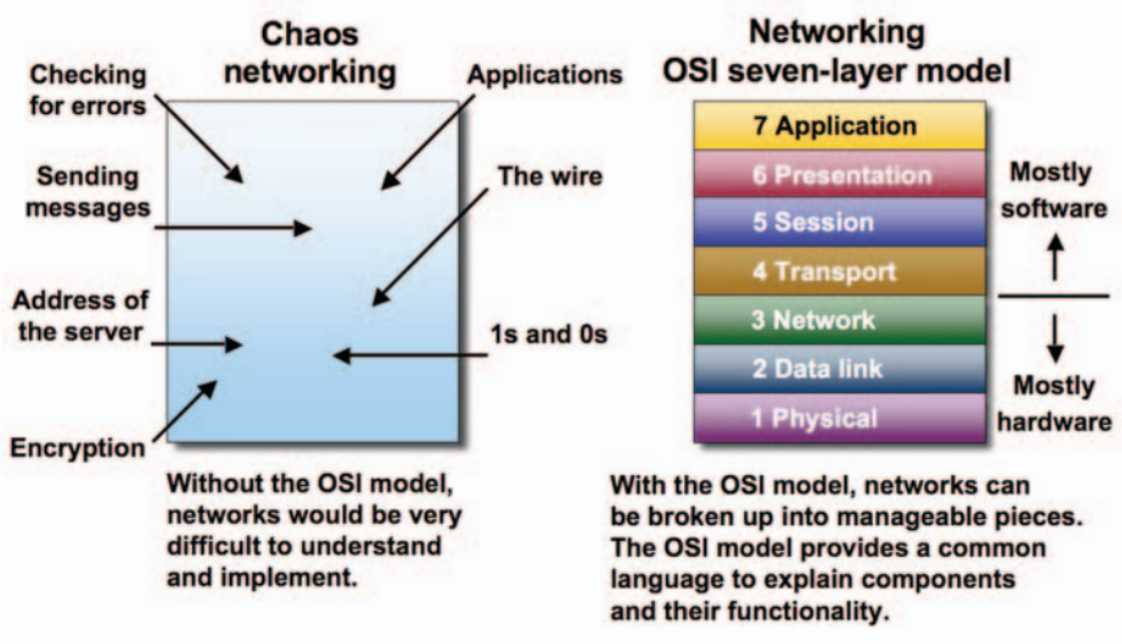
OSI MODEL

Introduction

The Open Systems Interconnection (OSI) model is a reference tool for understanding data communications between any two networked systems. It divides the communications processes into seven layers. Each layer both performs specific functions to support the layers above it and offers services to the layers below it. The three lowest layers focus on passing traffic through the network to an end system. The top four layers come into play in the end system to complete the process. Although TCP/IP has been used for network communications before the adoption of the OSI model, it supports the same functions and features in a differently layered arrangement.

An Overview of the OSI Model

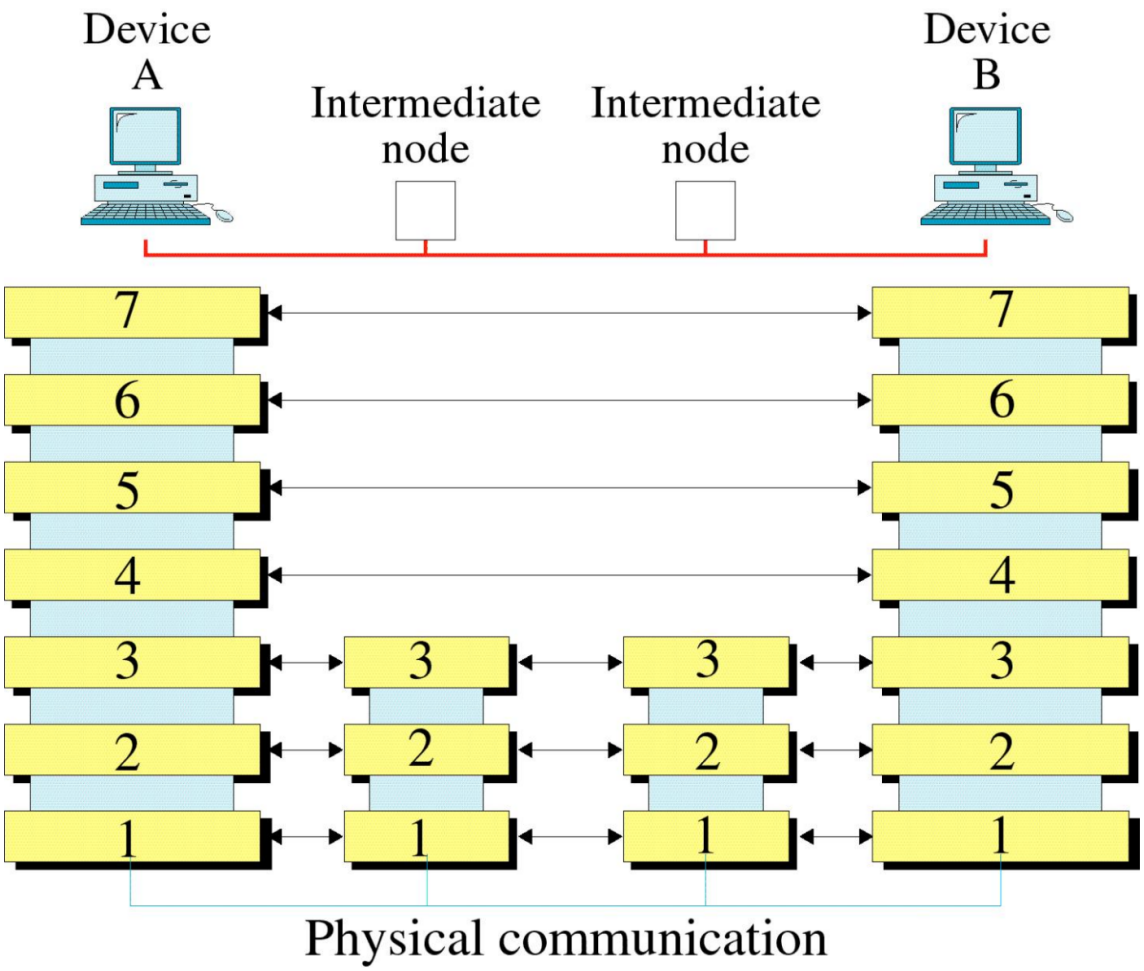


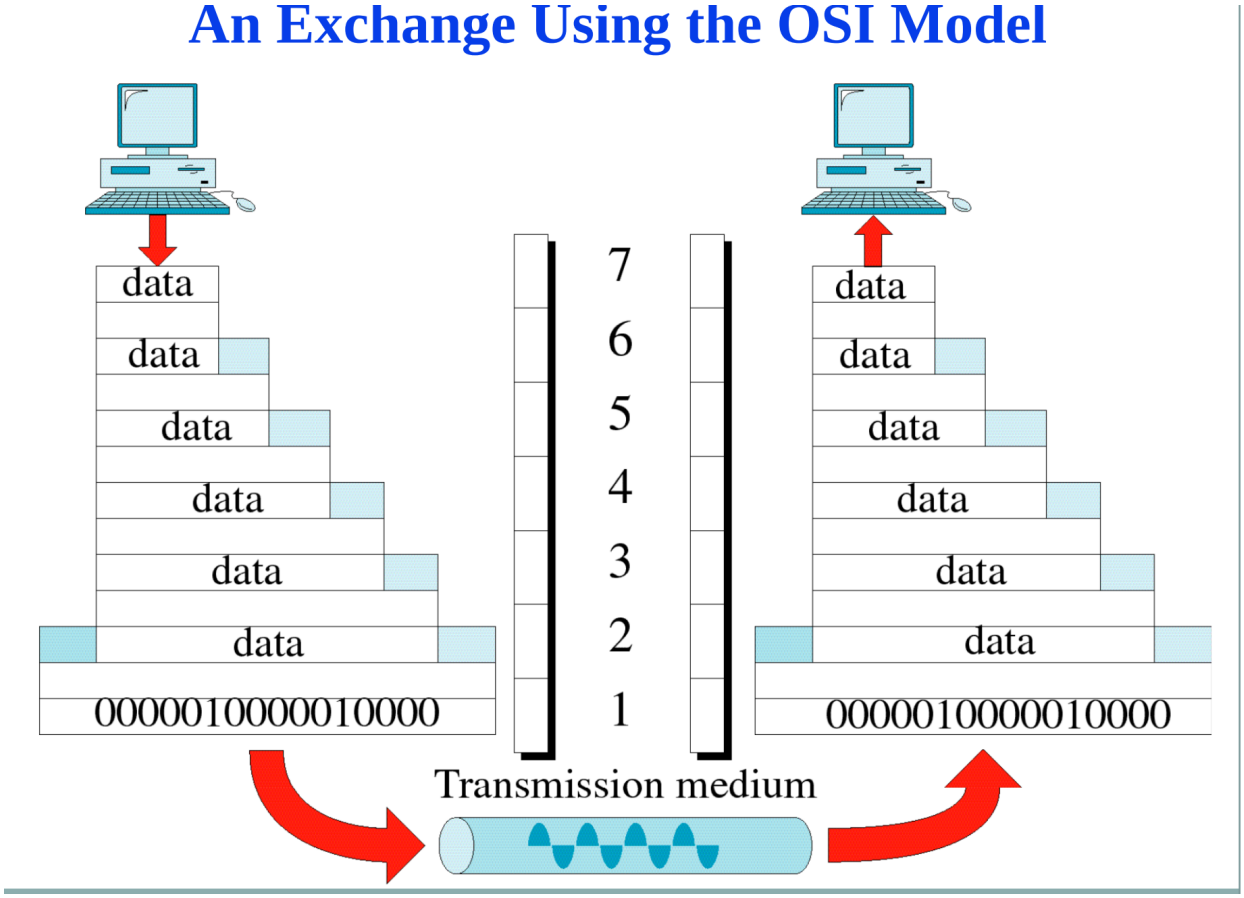
A networking model offers a generic means to separate computer networking functions into multiple layers. Each of these layers relies on the layers below it to provide supporting capabilities and performs support to the layers above it. Such a model of layered functionality is also called a “protocol stack” or “protocol suite”. Protocols, or rules, can do their work in either hardware or software or, as with most protocol stacks, in a combination of the two. The nature of these stacks is that the lower layers do their work in hardware or firmware (software that runs on specific hardware chips) while the higher layers work in software. The Open System Interconnection model is a seven-layer structure that specifies the requirements for communications between two computers. The ISO (International Organization for Standardization) standard 7498-1 defined this model. This model allows all network elements to

operate together, no matter who created the protocols and what computer vendor supports them.
The main benefits of the OSI model include the following:
• Helps users understand the big picture of networking
• Helps users understand how hardware and software elements function together
• Makes troubleshooting easier by separating networks into manageable pieces
• Defines terms that networking professionals can use to compare basic functional relationships on different networks
• Helps users understand new technologies as they are developed
• Aids in interpreting vendor explanations of product functionality

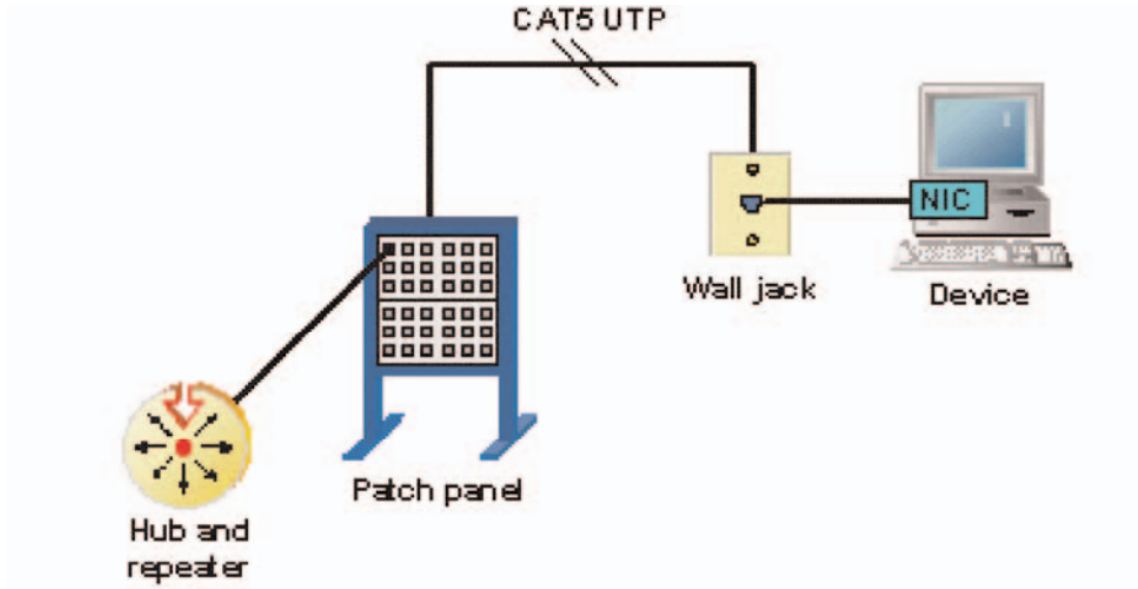
Principles on which OSI model was designed:

A layer should be created where a different level of abstraction is needed.
Each layer should perform a well-defined function.
The function of each layer should be chosen according to the internationally standardized protocols.
The number of layers should be large enough that distinct functions should not be put in the same layer and small enough that the architecture does not become very complex.


Device
A
Intermediate
node
Intermediate
node
Device
B
7
6
5
4
3
2
1
7
6
5
4
3
2
1
3
2
1
3
2
1
Physical communication



Layer 1 – The Physical Layer



The physical layer of the OSI model defines connector and interface specifications, as well as the medium (cable) requirements. Electrical, mechanical, functional, and procedural specifications are provided for sending a bit stream on a computer network.
It is the bottom layer of the OSI Model. It is responsible for the actual physical connection between the devices. The such physical connection may be made by using twisted pair cables. It is concerned with transmitting bits over a communication channel.

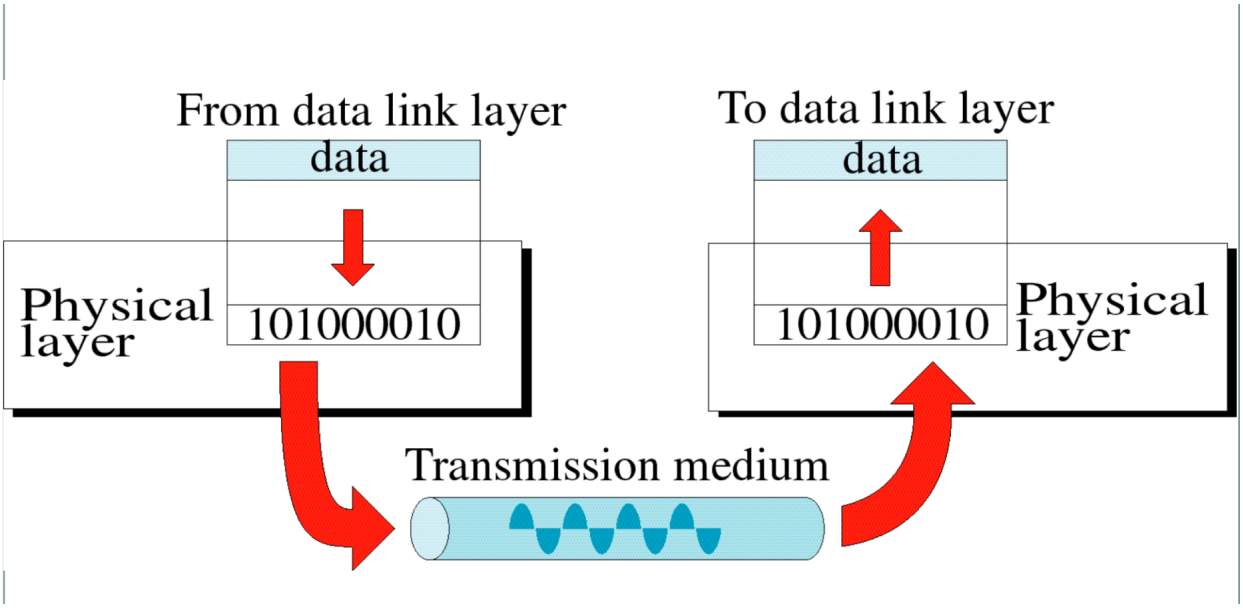
Components of the physical layer include

- Cabling system components
- Adapters that connect media to physical interfaces
- Connector design and pin assignments
- Hub, repeater and patch panel specifications
- Wireless system components
- Parallel SCSI (Small Computer System Interface)
- Network Interface Card (NIC)

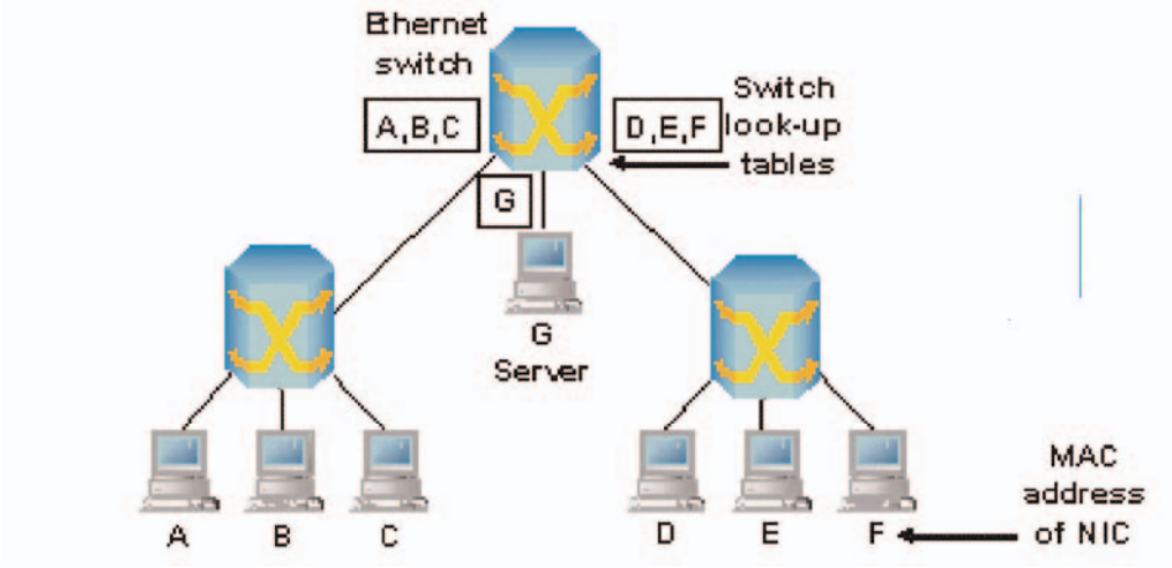
In a LAN environment, Category 5e UTP (Unshielded Twisted Pair) cable is generally used for the physical layer for individual device connections. Fiber optic cabling is often used for the physical layer in a vertical or riser backbone link. The IEEE, EIA/TIA, ANSI, and other similar standards bodies developed standards for this layer.

Functions of Physical Layer

- Transforming bits into signals Provides synchronization of bits by a clock.
- Physical layer manages the way a device connects to network media.
- It defines the transmission rate.
- It defines the way in which the devices are connected to the medium.
- It provides physical topologies
- It can use different techniques of multiplexing.

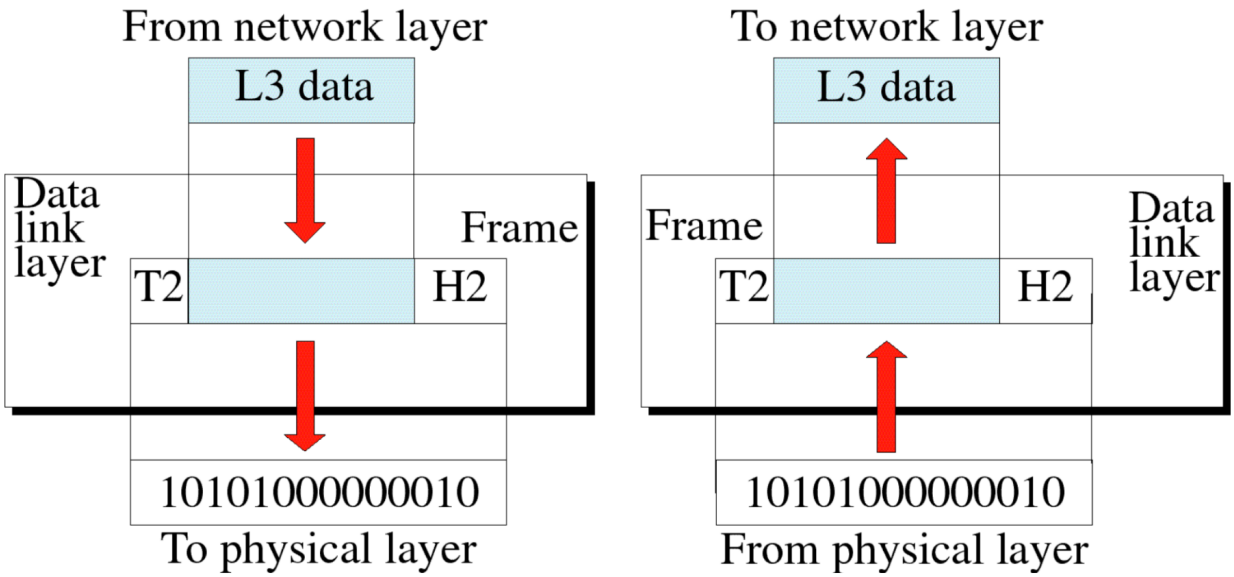


Layer 2 – The Data Link Layer



Layer 2 of the OSI model provides the following functions:
• Allows a device to access the network to send and receive messages
• Offers a physical address so a device's data can be sent on the network
• Works with a device's networking software when sending and receiving messages
• Provides error-detection capability
Common networking components that function at layer 2 include:
• Network interface cards
• Ethernet and Token Ring switches
• Bridges NICs have a layer 2 or MAC address.

A switch uses this address to filter and forward traffic, helping relieve congestion and collisions on a network segment. Bridges and switches function in a similar fashion; however, bridging is normally a software program on a CPU, while switches use Application-Specific Integrated Circuits (ASICs) to perform the task in dedicated hardware, which is much faster.

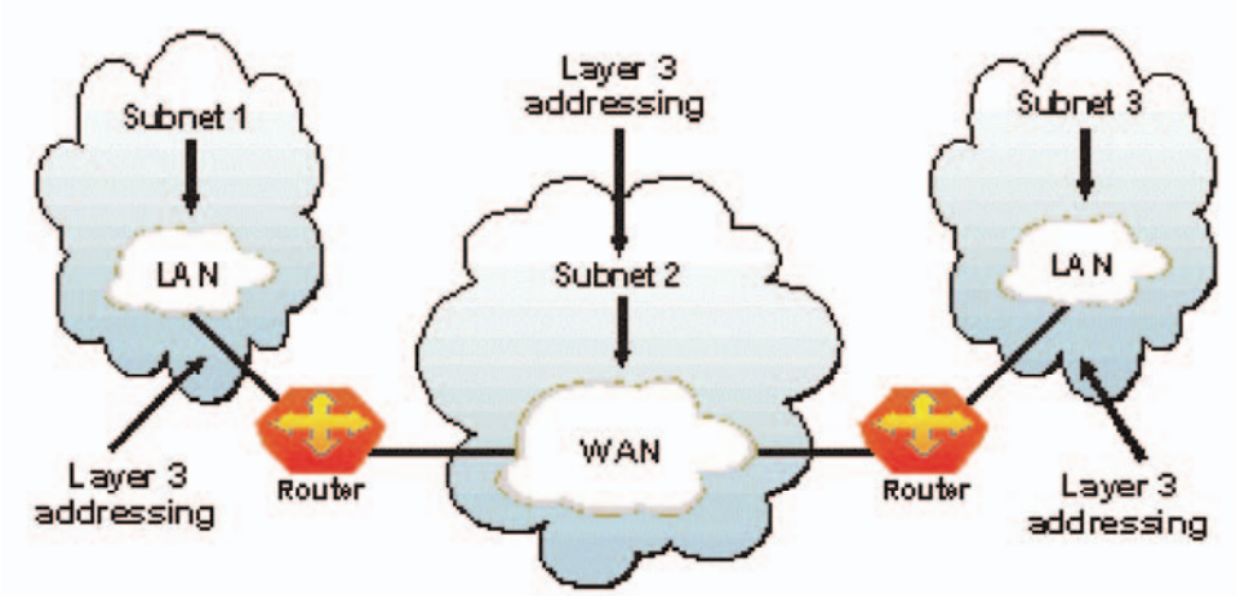


It is responsible for the node-to-node delivery of data.
It receives the data from the network layer and creates FRAMES, adds the physical address to these frames & pas them to the physical layer It consists of 2 layers:
Logical Link Layer (LLC): Defines the methods and provides addressing information for communication between network devices.
Medium Access Control (MAC): establishes and maintains links between communicating devices.

Functions of Data Link Layer

Framing: DLL divides the bits received from the N/W layer into frames. (Frame contains all the addressing information necessary to travel from S to D).
Physical addressing: After creating frames, DLL adds the physical address of the sender/receiver (MAC address) in the header of each frame.
Flow Control: DLL prevents the fast sender from drowning the slow receiver.
Error Control: It provides the mechanism of error control in which it detects & retransmits damaged or lost frames.
Access Control: When a single communication Channel is shared by multiple devices, the MAC layer of DLL provides help to determine which device has control over the channel.

Layer 3 – The Network Layer



Layer 3, the network layer of the OSI model, provides an end-to-end logical addressing system so that a packet of data can be routed across several layer 2 networks (Ethernet, Token Ring, Frame Relay, etc.). Note that network layer addresses can also be referred to as logical addresses.

Initially, software manufacturers, such as Novell, developed proprietary layer 3 addressing. However, the networking industry has evolved to the point that it requires a common layer 3 addressing system. Internet Protocol (IP) addresses make networks easier to both setup and connect with one another. The Internet uses IP addresses to provide connectivity to millions of networks around the world.

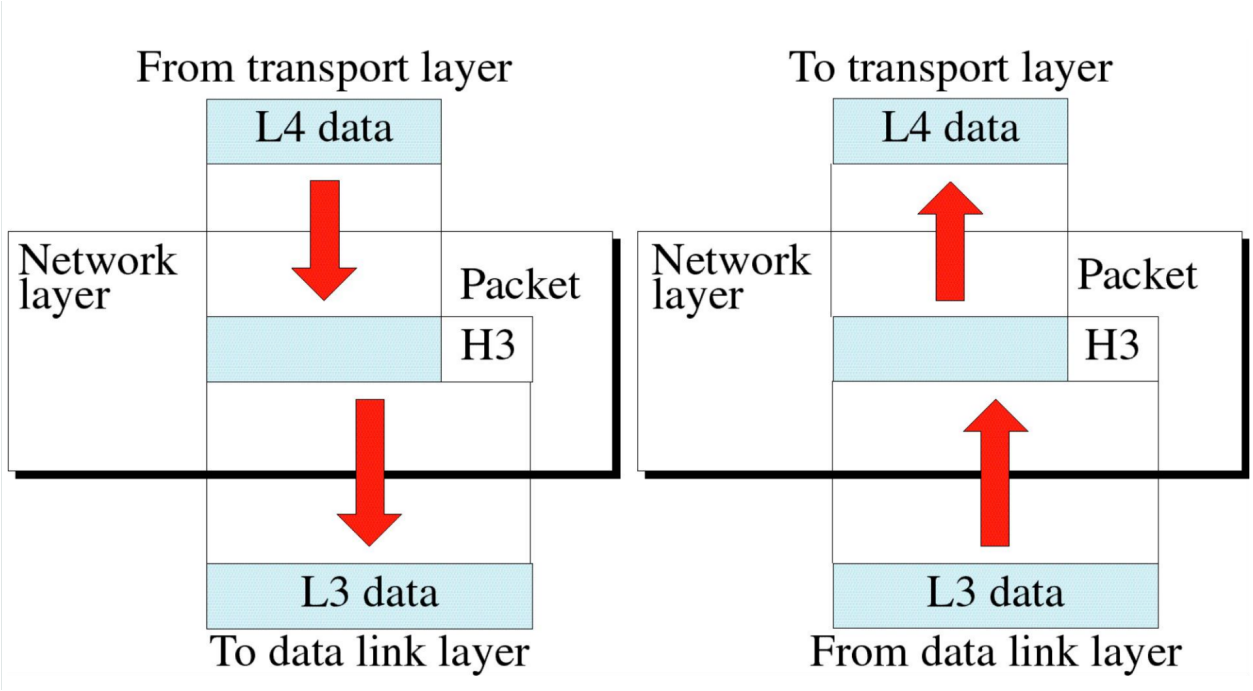
To make it easier to manage the network and control the flow of packets, many organizations separate their network layer addressing into smaller parts known as subnets. Routers use the network or subnet portion of the IP addressing to route traffic between different networks. Each router must be configured specifically for the networks or subnets that will be connected to its interfaces.

Routers communicate with one another using routing protocols, such as Routing Information Protocol (RIP) and the Open version of Shortest Path First (OSPF), to learn of other networks that are present and to calculate the best way to reach each network based on a variety of criteria (such as the path with the fewest routers). Routers and other networked systems make these routing decisions at the network layer.

When passing packets between different networks, it may become necessary to adjust their outbound size to one that is compatible with the layer 2 protocol that is being used. The network layer accomplishes this via a process known as fragmentation. A router's network layer is usually responsible for fragmentation. All reassembly of fragmented packets happens at the network layer of the final destination system.

Two of the additional functions of the network layer are diagnostics and the reporting of logical variations in normal network operation. While the network layer diagnostics may be initiated by

any networked system, the system discovering the variation reports it to the original sender of the packet that is found to be outside normal network operation.
The variation reporting exception is content validation calculations. If the calculation done by the receiving system does not match the value sent by the originating system, the receiver discards the related packet with no report to the sender. Retransmission is left to a higher layer's protocol. Some basic security functionality can also be set up by filtering traffic using layer 3 addressing routers or other similar devices.
It is responsible for the source-to-destination delivery of a packet across multiple networks. If two systems are attached to different networks with devices like routers, then the N/W layer is used. Thus DLL oversees the delivery of the packet between the two systems on the same network and the network layer ensures that the packet gets its point of origin to its final destination.



Functions of Network Layer

Internetworking: It provides Internetworking.
Logical Addressing: When a packet is sent outside the network, N/W layer adds the Logical (network) address of the sender & receiver to each packet. Network addresses are assigned to local devices by n/w administrator and assigned dynamically by a special server called DHCP (Dynamic Host Configuration Protocol)
Routing: When independent n/w are connected to create internetwork several routes are available to send the data from S to D. this n/w is interconnected by routers & gateways that route the packet to the final destination.

Layer 4 – The Transport Layer

Layer 4, the transport layer of the OSI model, offers end-to-end communication between end devices through a network. Depending on the application, the transport layer either offers reliable, connection-oriented, or connectionless, best-effort communications.
It is responsible for the process-to-process delivery of the entire message. TL looks after the delivery of the entire message considering all its packets & makes sure that all packets are in order. On the other hand n/w layer treated each packet independently. At the receiver side, TL provides services to the application layer & takes services from n/w layer. On the source side, TL receives messages from the upper layer into packets and reassembles these packets again into messages at the destination.
Transport Layer provides two types of services:

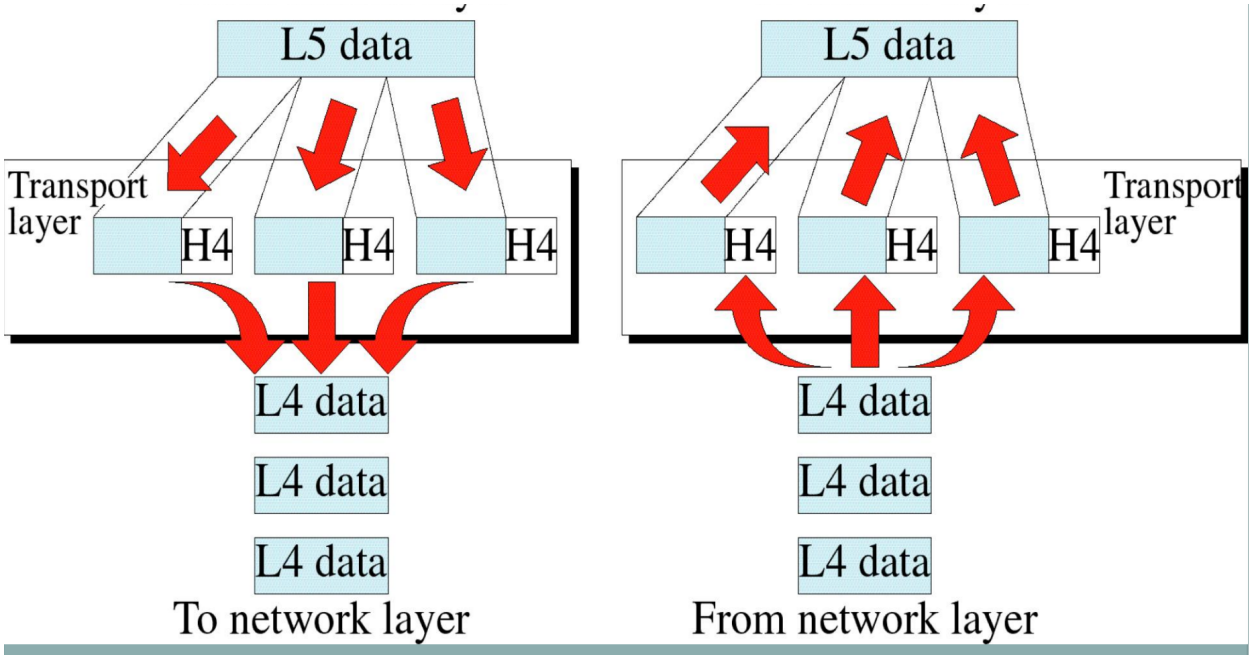
1. Connection-Oriented Transmission: In this type of transmission the receiving devices send an acknowledgment back to the source after a packet or group of the packet is received. It is a slower transmission method.
2. Connectionless Transmission: In this type of transmission the receiving device does not send an acknowledgment back to the source. It is a faster transmission method.

Functions of Transport Layer

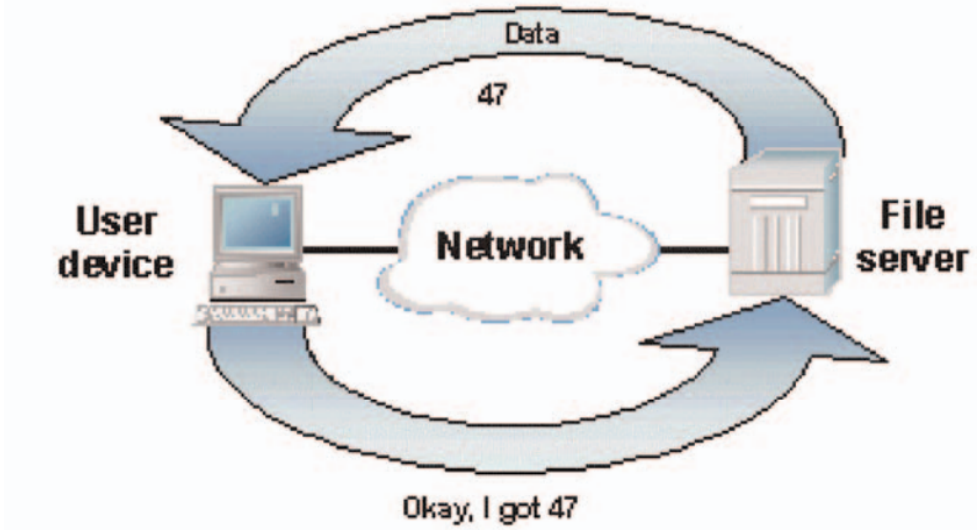
- Application identification
- Client-side entity identification
- Confirmation that the entire message arrived intact
- Segmentation of data for network transport
- Control of data flow to prevent memory overruns
- Establishment and maintenance of both ends of virtual circuits
- Transmission-error detection
- Realignment of segmented data in the correct order on the receiving side
- Multiplexing or sharing multiple sessions over a single physical link
- Segmentation of message into packet & reassembly of packets into the message.
- Port addressing: Computers run several processes. TL header includes a port address with each process.
- Flow Control: The flow control facility prevents the source from sending data packets faster than the destination can handle.

- Error control: TL ensures that the entire message arrives at the receiving TL without error.

The most common transport layer protocols are the connection-oriented TCP Transmission Control Protocol (TCP) and the connectionless UDP User Datagram Protocol (UDP).



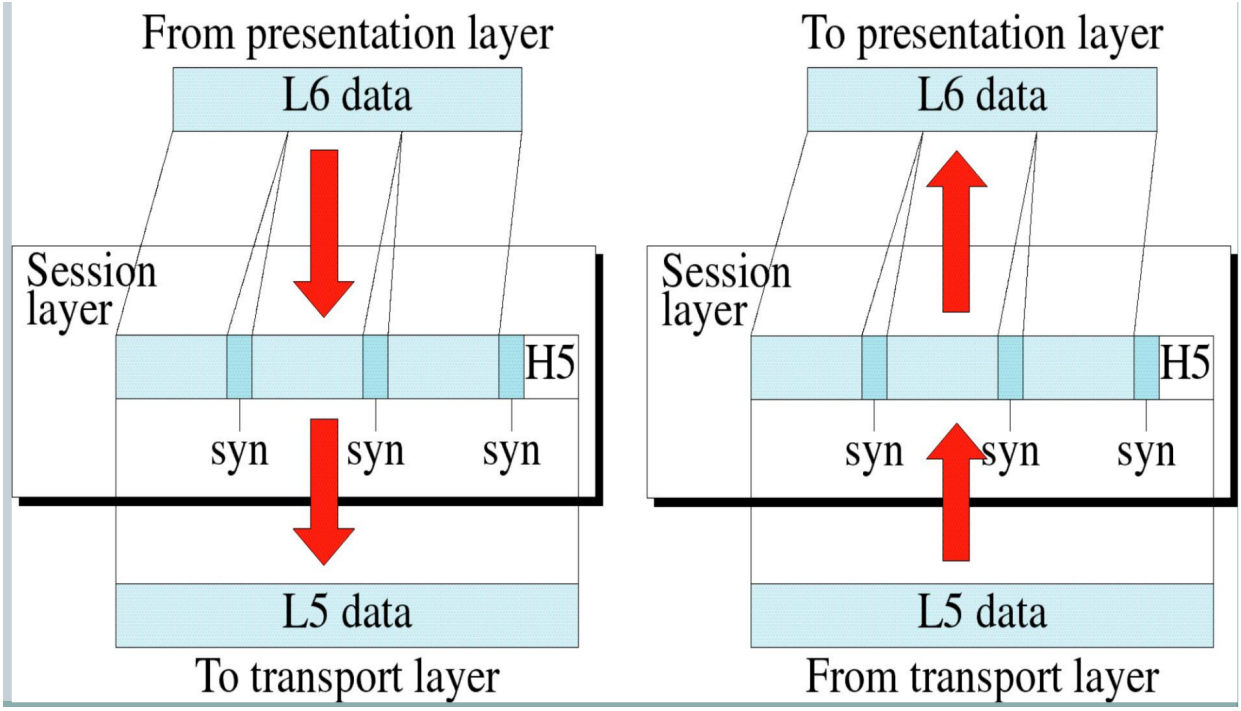
Layer 5 – The Session Layer



Layer 5, the session layer, provides various services, including tracking the number of bytes that each end of the session has acknowledged receiving from the other end of the session.
This session layer allows applications functioning on devices to establish, manage, and terminate a dialog through a network.

Functions of Session Layer

1. The virtual connection between application entities
2. Synchronization of data flow
3. Creation of dialog units
4. Connection parameter negotiations
5. Partitioning of services into functional groups
6. Acknowledgments of data received during a session
7. Retransmission of data if it is not received by a device
8. Establishing, Maintaining, and ending a session: When sending device first contact with receiving device, it sends syn (synchronization) packet to establish a connection & determines the order in which information will be sent. The receiver sends ack (acknowledgment). So the session can be set & end.
9. Dialog Control: This function determines which device will communicate first and the amount of data that will be sent.
10. Dialog separation: The process of adding checkpoints & markers to the stream of data is called dialog separation.



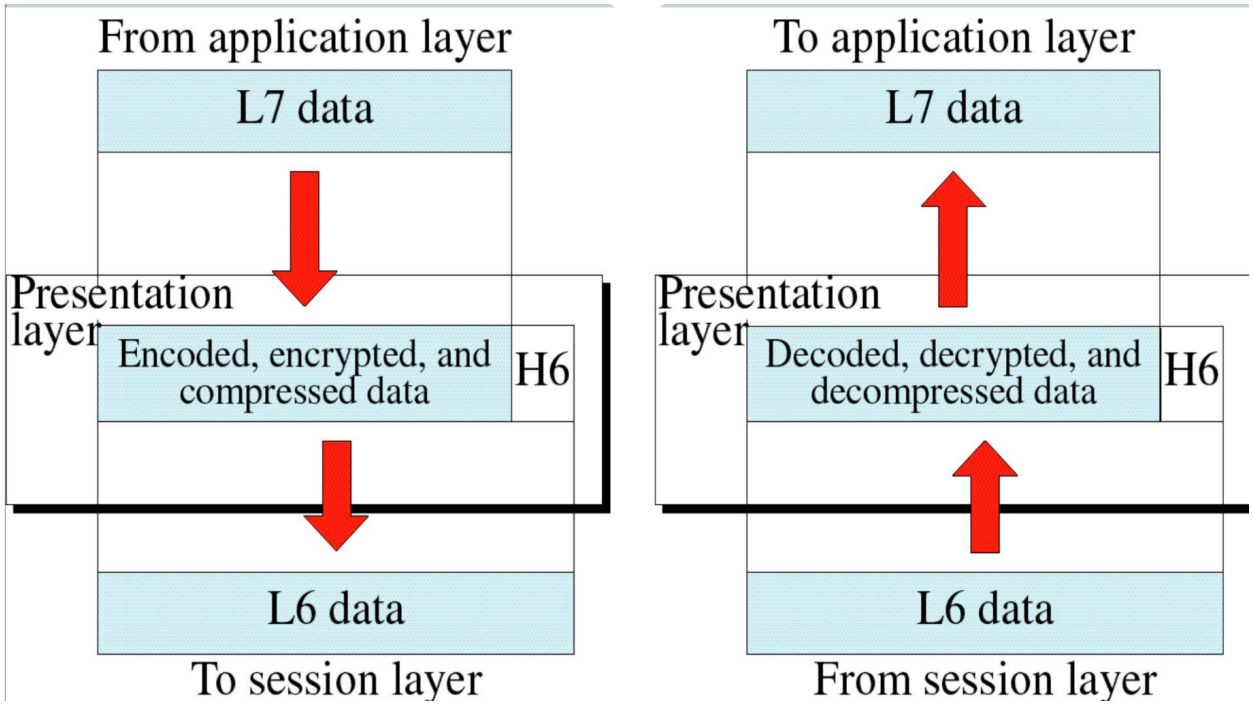
Layer 6 – The Presentation Layer

Layer 6, the presentation layer, is responsible for how an application formats the data to be sent out onto the network. The presentation layer basically allows an application to read (or understand) the message.

It is concerned with the syntax & semantics of the information exchanged between the two devices. It was designed for data encryption, decryption and compression.

Functions of Presentation Layer

1. Encryption and decryption of a message for security
2. Compression and expansion of a message so that it travels efficiently
3. Graphics formatting
4. Content translation
5. System-specific translation
6. Data Presentation or Translation: Different computers use different encoding systems. It ensures that the data being sent is in the format that the recipient can process.
7. Data Encryption: PL provides this facility by which hides the information from everyone except the person who originally sent the information & the intended recipient. When encrypted data arrives at the destination, PL decrypts the message.
8. Data Compression: PL shrinks large amounts of data into smaller pieces i.e. it reduces the size of data.



Layer 7 – The Application Layer

Layer 7, the application layer, provides an interface for the end user operating a device connected to a network. This layer is what the user sees, in terms of loading an application (such as a Web browser or e-mail); that is, this application layer is the data the user views while using these applications.
It enables the user to access the network. It provides a user interface & supports for services such as e-mail, file transfer, and access to the world wide web. So it provides services to different user applications.

Functions of the Application Layer

1. Support for file transfers
2. Ability to print on a network
3. Electronic mail
4. Electronic messaging
5. Browsing the World Wide Web
6. Mail Services: This application provides various email services.
7. File transfer & Access: It allows users to access files in a remote host, retrieve files from remote computers for use etc.
8. Remote log-in: A user can log into a remote computer and access the resources of that computer. Accessing the World Wide Web: Most common application today is access of the World Wide Web.

Layers 8, 9, and 10

Whether designed to be a humorous extension or a secret technician code, layers 8, 9, and 10 are not officially part of the OSI model. They refer to the non-technical aspects of computer networking that often interfere with the smooth design and operation of the network.

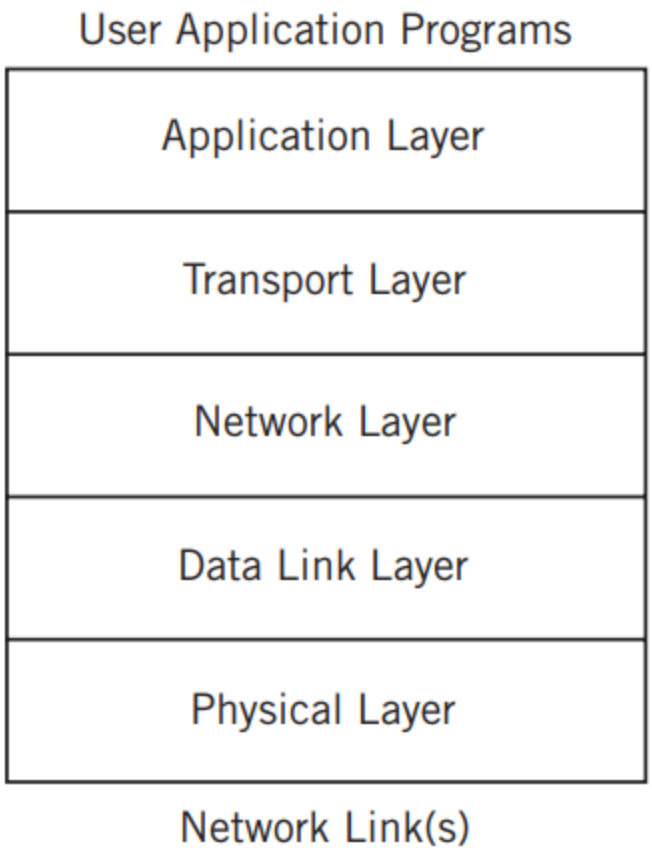
1. Layer 8 is usually considered the “office politics” layer. In most organizations, there is at least one group that is favored, at least temporarily, by management and receives “special” treatment. When it comes to networking, this may mean that this group always has the latest and/or fastest equipment and highest-speed network links.
2. Layer 9 is generally referred to as the “blinders” layer. This layer applies to organizational managers who have already decided, usually with little or no current information, to dictate a previously successful network plan. They may say things such as: “It worked in my last company, so we will use it here.” “Everybody says this is the right solution.” “I read in an airline magazine that this was the best way to do it so that is what we will do.” What these managers seem to forget is that they are paying a highly qualified staff to provide them with useful information. These managers bypass planning in order to make a quick decision.
3. Layer 10, the “user” layer, is in every organization. But users are much more than a layer. While they are one of the reasons the network exists, users can also be a big part of the need for troubleshooting. This is especially true when the users have computers at home and have decided to “help” the network administrator or manager by making changes to the network without consulting the network staff. Equally challenging is the user who “didn’t do anything” when the network segment in his/her immediate vicinity suddenly stopped working. In these cases, the layer 10 identification coincides with layer 10 troubles (and the “ID10T” label some technicians have used).

THE TCP/IP PROTOCOL SUITE

The protocol stack used on the Internet is the Internet Protocol Suite. It is usually called TCP/IP after two of its most prominent protocols, but there are other protocols as well. The TCP/IP model is based on a five-layer model for networking. From bottom (the link) to top (the user application), these are the physical, data link, network, transport, and application layers. Not all layers are completely defined by the model, so these layers are “filled in” by external standards and protocols. The layers have names but no numbers, and although sometimes people speak of “Layer 2” or “Layer 3,” these are not TCP/IP terms. Terms like these are actually from the OSI Reference Model. The TCP/IP stack is open, which means that there are no “secrets” as to how it works. (There are “open systems” too, but with TCP/IP, the systems do not have to be “open” and often are not.) Two compatible end-system applications can communicate regardless of their underlying architectures, although the connections between layers are not defined.
The TCP/IP stack is comprised of modules. Each module provides a specific function, but the modules are fairly independent. The TCP/IP layers contain relatively independent protocols that can be used depending on the needs of the system to provide whatever function is desired. In TCP/IP, each higher layer protocol is supported by lower layer protocols. The whole collection of

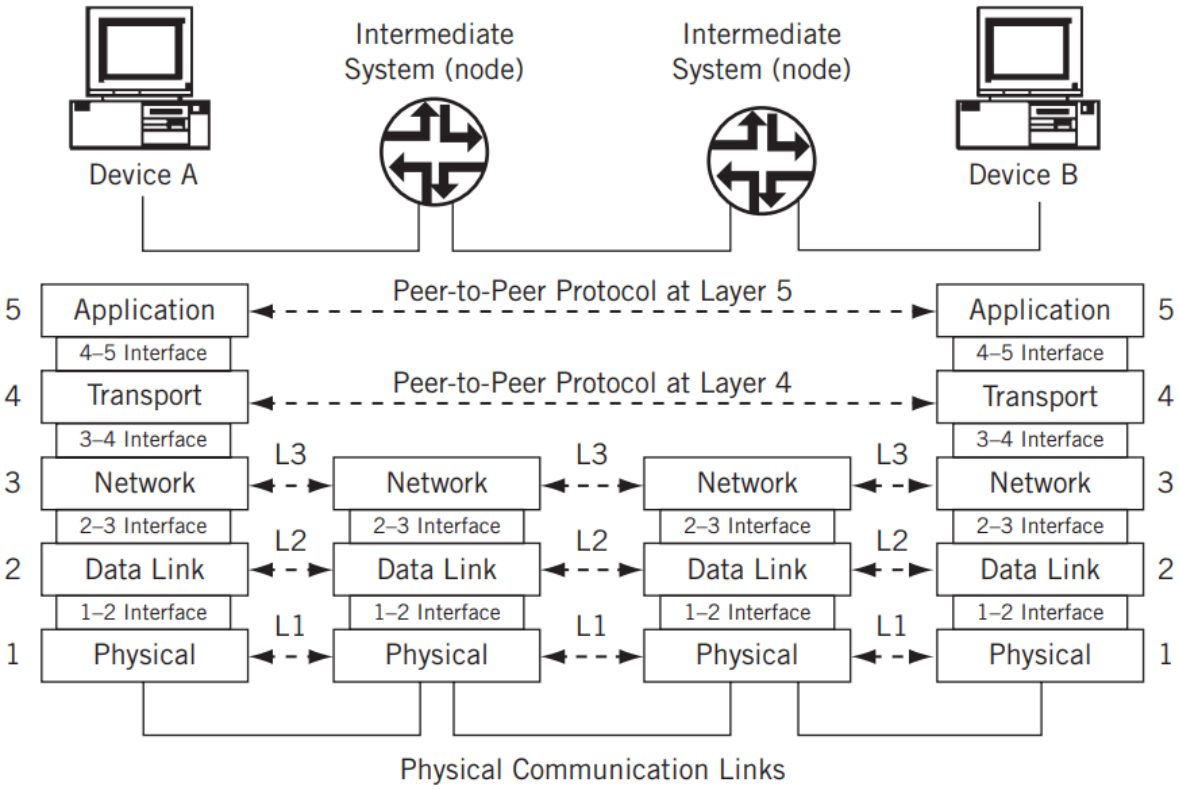
protocols forms a type of hourglass shape, with IP in the middle, and more and more protocols up or down from there.

The TCP/IP Layers The TCP/IP protocol stack models a series of protocol layers for networks and systems that allows communications between any types of devices. The model consists of five separate but related layers, as shown below. The Internet protocol suite is based on these five layers. TCP/IP says most about the network and transport layers, and a lot about the application layer. TCP/IP also defines how to interface the network layer with the data link and physical layers, but is not directly concerned with these two layers themselves. The Internet protocol suite assumes that a layer is there and available, so TCP/IP does not defi ne the layers themselves. The stack consist of protocols, not implementations, so describing a layer or protocols says almost nothing about how these things should actually be built. Not all systems on a network need to implement all fi ve layers of TCP/IP. Devices using the TCP/IP protocol stack fall into two general categories: a host or end system (ES) and an intermediate node (often a router) or an intermediate system (IS). The intermediate nodes usually only involve the fi rst three layers of TCP/IP (although many of them still have all fi ve layers for other reasons, as we have seen). In TCP/IP, as with most layered protocols, the most fundamental elements of the process of sending and receiving data are collected into the groups that become the layers. Each layer's major functions are distinct from all the others, but layers can be combined for performance reasons. Each implemented layer has an interface with the layers above and below it (except for the application and physical layers, of course) and provides its defi ned service to the layer above and obtains services from the layer below. In other words, there is a service interface between each layer, but these are not standardized and vary widely by operating system. TCP/IP is designed to be comprehensive and fl exible. It can be extended to meet new requirements, and has been. Individual layers can be combined for implementation purposes, as long as the service interfaces to the layers remain intact. Layers can even be split when necessary, and new service interfaces defi ned. Services are provided to the layer above after the higher layer provides the lower layer with the command, data, and necessary parameters for the lower layer to carry out the task. Layers on the same system provide and obtain services to and from adjacent layers. However, a peer-to-peer protocol process allows the same layers on different systems to communicate. The term peer means every implementation of some layer is essentially equal to all others. There is no "master" system at the protocol level. Communications between peer layers on different systems use the defi ned protocols appropriate to the given layer. In other words, services refer to communications between layers within the same process, and protocols refer to communications between processes.



Protocols and Interfaces

It is important to note that when the layers of TCP/IP are on different systems, they are only connected at the physical layer. Direct peer-to-peer communication between all other layers is impossible. This means that all data from an application have to flow “down” through all five layers at the sender, and “up” all five layers at the receiver to reach the correct process on the other system. These data are sometimes called a service data unit (SDU). Each layer on the sending system adds information to the data it receives from the layer above and passes it all to the layer below (except for the physical layer, which has no lower layers to rely on in the model and actually has to send the bits in a form appropriate for the communications link used). Likewise, each layer on the receiving system unwraps the received message, often called a protocol data unit (PDU), with each layer examining, using, and stripping off the information it needs to complete its task, and passing the remainder up to the next layer (except for the application layer, which passes what’s left off to the application program itself). For example, the data link layer removes the wrapper meant for it, uses it to decide what it should do with this data unit, and then passes the remainder up to the network layer.

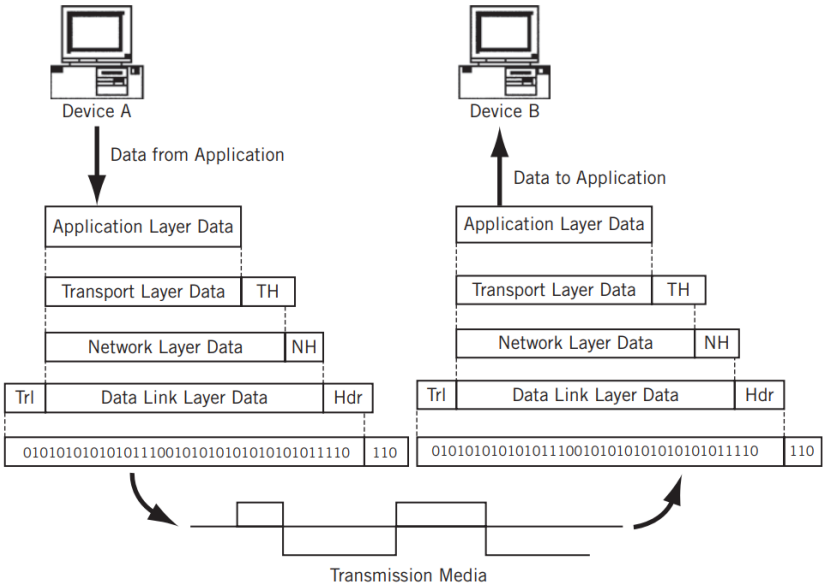


As shown in the figure, there is a natural grouping of the five-layer protocol stack at the network layer and the transport layer. The lower three layers of TCP/IP, sometimes called the network support layers, must be present and functional on all systems, regardless of the end system or intermediate node role. The transport layer links the upper and lower layers together. This layer can be used to make sure that what was sent was received, and what was sent is useful to the receiver (and not, for example, a stray PDU misdirected to the host or unreasonably delayed). The process of encapsulation makes the whole architecture workable. Encapsulation of one layer's information inside another layer is a key part of how TCP/IP works.

Encapsulation

Each layer uses encapsulation to add the information its peer needs on the receiving system. The network layer adds a header to the information it receives from the transport at the sender and passes the whole unit down to the data link layer. At the receiver,
the network layer looks at the control information, usually in a header, in the data it receives from the data link layer and passes the remainder up to the transport layer for further processing. This is called encapsulation because one layer has no idea what the structure or meaning of the PDU is at other layers. The PDU has several more or less official names for the structure at each layer. The exception to this general rule is the data link layer, which adds both a header and a trailer to the data it receives from the network layer. The general flow of encapsulation in TCP/IP is shown in Figure below. Note that on the transmission media itself (or communications link), there are only bits, and that some "extra" bits are added by the

communication link for its own purposes. Each PDU at the other layers is labeled as data for its layer, and the headers are abbreviated by layer name. The exception is the second layer, the data link layer, which shows a header and trailer added at that level of encapsulation. Although the intermediate nodes are not shown, these network devices will only process the data (at most) through the first three layers. In other words, there is no transport layer to which to pass network-layer PDUs on these systems for data communications (management is another issue).



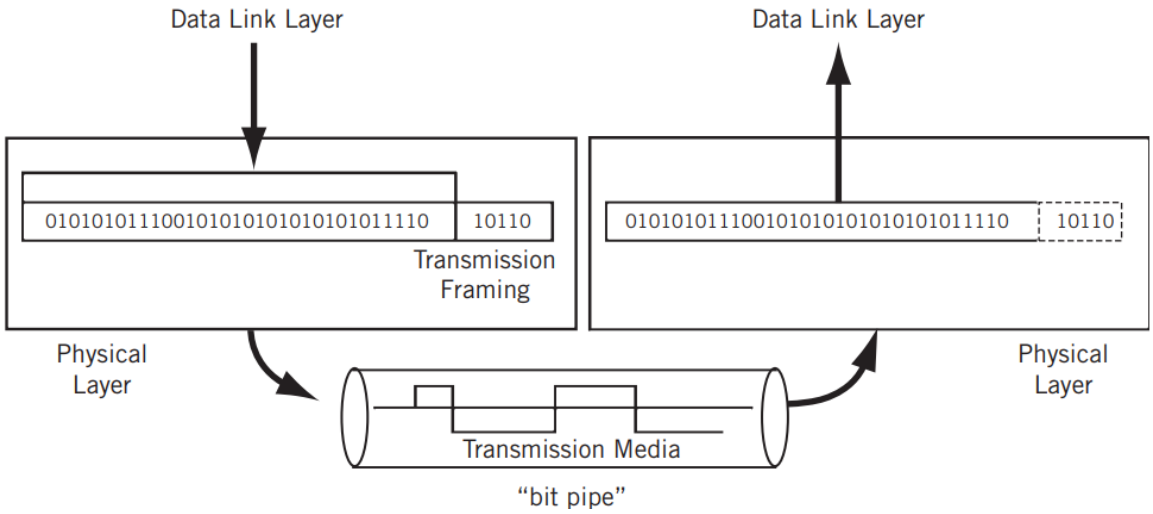
THE LAYERS OF TCP/IP

TCP/IP is mature and stable, and is the only protocol stack used on the Internet. This is all about networking with TCP/IP, but it is easy to get lost in the particulars of TCP/IP if some discussion of the general tasks that TCP/IP is intended to accomplish is not included. This section takes a closer look at the TCP/IP layers, but only as a general guide to how the layers work.

The Physical Layer

The physical layer contains all the functions needed to carry the bit stream over a physical medium to another system. The figure below shows the position of the physical layer to the data link layer and the transmission medium. The transmission medium forms a pure “bit pipe” and should not change the bits sent in any way. Now, transmission “on the wire” might send bits through an extremely complex transform, but the goal is to enable the receiver to reconstruct the bit stream exactly as sent. Some information in the form of transmission framing can be added to the data link layer data, but this is only used by the physical layer and the transmission medium itself. In some cases, the transmission medium sends a constant idle bit pattern until interrupted by data. Physical layer specifications have four parts: mechanical, electrical or optical, functional, and procedural. The mechanical part specifies the physical size and shape of the connector itself so that components will plug into each other easily. The electrical/ optical

specification determines what value of voltage or line condition determines whether a pin is active or what exactly represents a 0 or 1 bit. The functional specification specifies the function of each pin or lead on the connector (first lead is send, second is receive, and so on). The procedural specification details the sequence of actions that must take place to send or receive bits on the interface. (For Ethernet, the send pair is activated, then a “preamble” is sent, and so forth.) The Ethernet twisted pair interfaces from the IEEE are common implementations of the physical layer that includes all these elements.



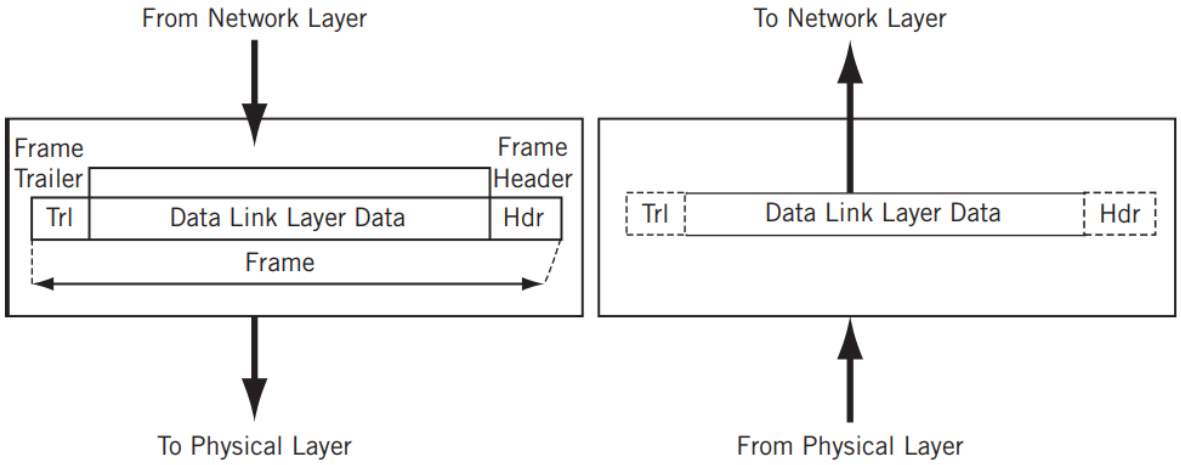
There are other things that the physical layer must determine, or be configured to expect.

- Data rate—This transmission rate is the number of bits per second that can be sent. It also defines the duration of a symbol on the wire. Symbols usually represent one or more bits, although there are schemes in which one bit is represented by multiple symbols.
- Bit synchronization—The sender and receiver must be synchronized at the symbol level so that the number of bits expected per unit time is the same. In other words, the sender and receiver clocks must be synchronized (timing is in the millisecond or microsecond range). On modern links, the timing information is often “recovered” from the received data stream.
- Configuration—So far we’ve assumed simple point-to-point links, but this is not the only way that systems are connected. In a multipoint configuration, a link connects more than two devices, and in a multisystem bus/broadcast topology such as a LAN, the number of systems can be very high.
- Topology—The devices can be arranged in a number of ways. In a full mesh topology, all devices are directly connected and one hop away, but this requires a staggering amount of links for even a modest network. Systems can also be arranged as a star topology, with all systems reachable through a central system. There is also the bus (all devices are on a common link) and the ring (devices are chained together, and the last is linked to the first, forming a ring).
- Mode—So far, we’ve only talked about one of the systems as the sender and the other as the receiver. This is operation in simplex mode, where a device can only send or receive, such as with weather sensors reporting to a remote weather station. More

realistic devices use duplex mode, where all systems can send or receive with equal facility. This is often further distinguished as half-duplex (the system can send and receive, but not at the same time) and full-duplex (simultaneous sending and receiving).

The Data Link

Layer Bits are just bits. With only a physical layer, System A has no way to tell System B, “Get ready some bits,” “Here are the bits,” and “Did you get those bits okay?” The data link layer solves this problem by organizing the bit stream into a data unit called a frame. It is important to note that frames are the data link layer PDUs, and these are not the same as the physical layer transmission frames mentioned in the previous section. For example, network engineers often speak about T1 frames or SONET frames, but these are distinct from the data link layer frames that are carried inside the T1 or SONET frames. Transmission frames have control information used to manage the physical link itself and has little to do directly with process-to-process communications. This “double-frame” arrangement might sound redundant, but many transmission frames originated with voice because digitized voice has no framing at the “data link” layer. The data link layer moves bits across the link and can add reliability to the raw communications link. The data link layer can be very simple, or make the link appear error free to the layer above, the network layer. The data link layer usually adds both a header and trailer to the data presented by the network layer. This is shown in Figure 1.13. The frame header typically contains a source and destination address (known as the “physical address” since it refers to the physical communication port) and some control information. The control information is data passed from one data link layer to the



other data link layer, and not user data. The body of the frame contains the sequence of bits being transferred across the network. The trailer usually contains information used in detecting bit errors (such as cyclical redundancy check [CRC]). A maximum size is associated with the frame that cannot be exceeded because all systems must allocate memory space (buffers) for the data. In a networking context, a buffer is just special memory allocated for communications. The data link layer performs framing, physical addressing, and error detection (error correction is another matter entirely, and can be handled in many ways, such as by resending a copy of

the frame that had the errors). However, when it comes to frame error detection and correction in the real world, error detection bits are sometimes ignored and frames that defy processing due to errors are simply discarded. This does not mean that error detection and correction are not part of the data link layer standards: It means that in these cases, ignoring and discarding are the chosen methods of implementation. In discard cases, the chore of handling the error condition is “pushed up the stack” to a higher layer protocol.

This layer also performs access control (this determines whose turn it is to send over or control the link, an issue that becomes more and more interesting as the number of devices sharing the link grows). In LANs, this media access control (MAC) forms a sublayer of the data link layer and has its own addressing scheme known (not surprisingly) as the MAC layer address or MAC address. For now, it is enough to note that LANs such as Ethernet do not have “real” physical layer addresses and that the MAC address performs this addressing function.

In addition, the data link layer can perform some type of fl ow control. Flow control makes sure senders do not overwhelm receivers: a receiver must have adequate time to process the data arriving in its buffers. At this layer, the fl ow control, if provided, is link-by-link. (We’ll see shortly that end-to-end—host-to-host—fl ow control is provided by the transport layer.) LANs do not usually provide fl ow control at the data link layer, although they can. Not all destination systems are directly reachable by the sender. This means that when bits at the data link layer are sent from an originating system, the bits do not arrive at the destination system as the “next hop” along the way. Directly reachable systems are called adjacent systems, and adjacent systems are always “one hop away” from the sender. When the destination system is not directly reachable by the sender, one or more intermediate nodes are needed. Now the sender (System A) is not directly connected to the receiver (System B). Another system, System 3, receives the frame and must forward it toward the destination. This system is usually called a switch or router (there are even other names), depending on internal architecture and network role. On a WAN (but not on a LAN), this second frame is a different frame because there is no guarantee that the second link is identical to the first. Different links need different frames. Identical frames are only delivered to systems that are directly reachable, or adjacent, to the sender, such as by an Ethernet switch on a LAN.

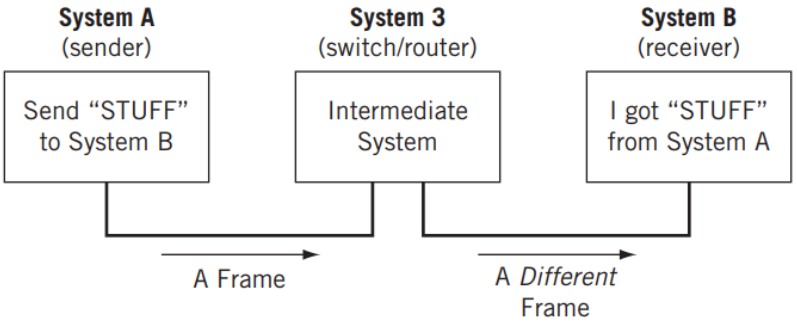


FIGURE 1.14

A more complex network. Note that the frames are technically different even if the same medium is used on both links.

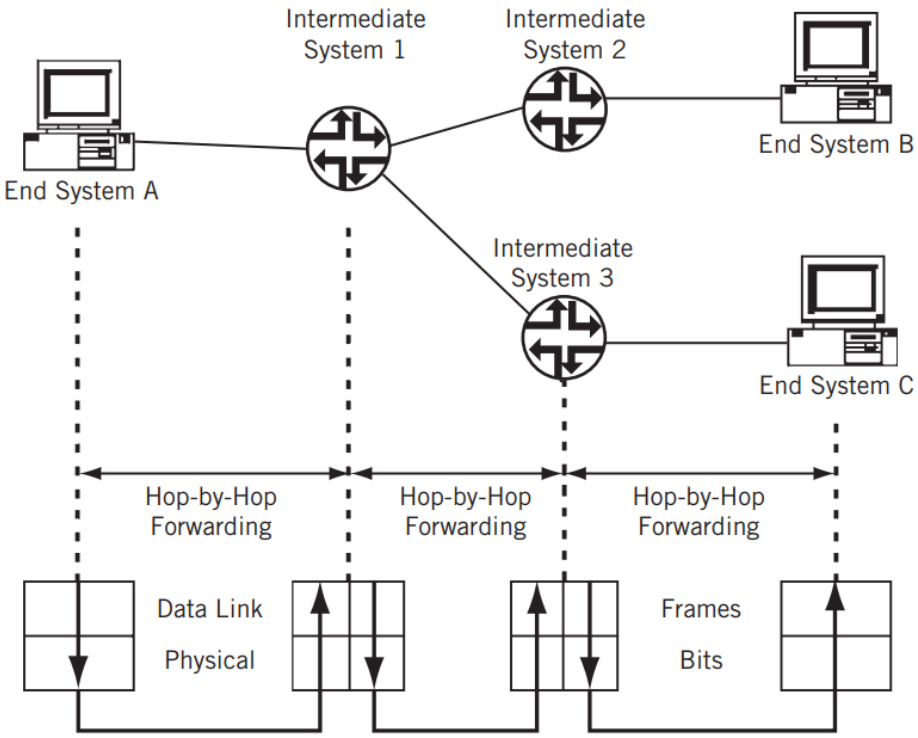


FIGURE 1.15

Networking with intermediate systems is called hop-by-hop delivery. A "hop" is the usual term used on the Internet or a router network to indicate the forwarding of a packet between one router or another (or between a host and router). Frames can "hop" between Layer 2 switches, but the term is most commonly used for Layer 3 router hops (which can consist of multiple switch-to-switch frame "hops"). There can be more than one intermediate system between the source and destination end systems, of course, as shown in Figure 1.15. Consider the case where End System A is sending a bit stream to End System C.

Note that the intermediate systems (routers) have two distinct physical and data link layers, refl ecting the fact that the systems have two (and often more) communication links, which can differ in many ways. (The fi gure shows a typical WAN confi guration with point-to-point links, but

routers on LANs, and on some types of public data service WANs, can be deployed in more complicated ways.) However, there is something obviously missing from this fi gure. There is no connection between the data link layers on the intermediate systems! How does the router know to which output port and link to forward the data in order to ultimately reach the destination? (In the fi gure, note that Intermediate System 1 can send data to either Intermediate System 2 or Intermediate System 3, but only through Intermediate System 3, which forwards the data, is the destination reachable.) These forwarding decisions are made at the TCP/IP network layer.

The Network Layer

The network layer delivers data in the form of a packet from source to destination, across as many links as necessary. The biggest difference between the network layer and the data link layer is that the data link layer is in charge of data delivery between adjacent systems (directly connected systems one hop away), while the network layer delivers data to systems that are not directly connected to the source. There can be many different types of data link and physical layers on the network, depending on the variety of the link types, but the network layer is essentially the same on all systems, end systems, and intermediate systems alike. Figure 1.16 shows the relationship between the network layer and the transport layer above and the data link layer below. A packet header is put in place at the sender and interpreted by the receiver. A router simply looks at the packet header and makes a forwarding decision based on this information. The transport layer does not play a role in the forwarding decision.

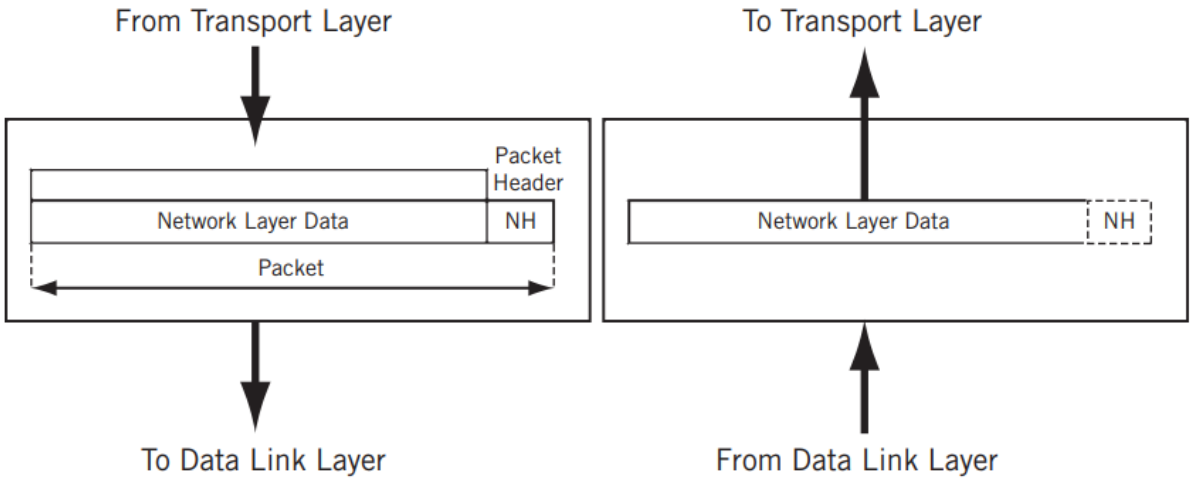


FIGURE 1.16

How does the network layer know where the packet came from (so the sender can reply)? The key concept at the network layer is the network address, which provides this information. In TCP/IP, the network address is the IP address. Every system in the network receives a network address, whether an end system or intermediate system. Systems require at least one network address (and sometimes many more). It is important to realize that this network address is different from, and independent of, the physical address used by the frames that carry the packets between adjacent systems. Why should the systems need two addresses for the two layers? Why can't they just both use either the data link ("physical") address or the network

address at both layers? There are actually several reasons. First, LAN addresses like those used in Ethernet come from one group (the IEEE), while those used in TCP/IP come from another group (ICANN). Also, the IP address is universally used on the Internet, while there are many types of physical addresses. Finally, there is no systematic assignment of physical addresses (and many addresses on WANs can be duplicated and so have “local significance only”). On the other hand, IP network addresses are globally administered, unique, and have a portion under which many devices are grouped. Therefore, many devices can be addressed concisely by this network portion of the IP address. A key issue is how the network addresses “map” to physical addresses, a process known generally as address resolution. In TCP/IP, a special family of address resolution protocols takes care of this process.

The network address is a logical address. Network addresses should be organized so that devices can be grouped under a part of that address. In other words, the network address should be organized in a fashion similar to a telephone number, for example, 212-555-1212 in the North American public switched telephone network (PSTN). The sender need only look at the area code or “network” portion of this address (212) to determine if the destination is local (area codes are the same) or needs to be sent to an intermediate system to reach the 212 area code (source and destination area codes differ). For this scheme to work effectively, however, all telephones that share the 212 area code should be grouped together. The whole telephone number beginning with 212 therefore means “this telephone in the 212 area code.” In TCP/IP, the network address is the beginning of the device’s complete IP address. A group of hosts is gathered under the network portion of the IP address. IP network addresses, like area codes, are globally administered to prevent duplication, while the rest of the IP address, like the rest of the telephone number, is locally administered, often independently. In some cases, the packet that arrives at an intermediate system inside a frame is too large to fi t inside the frame that must be sent out. This is not uncommon: different link and LAN types have different maximum frame sizes. The network layer must be able to fragment a data unit across multiple frames and reassemble the fragments at the destination. We’ll say more about fragmentation in a later chapter.

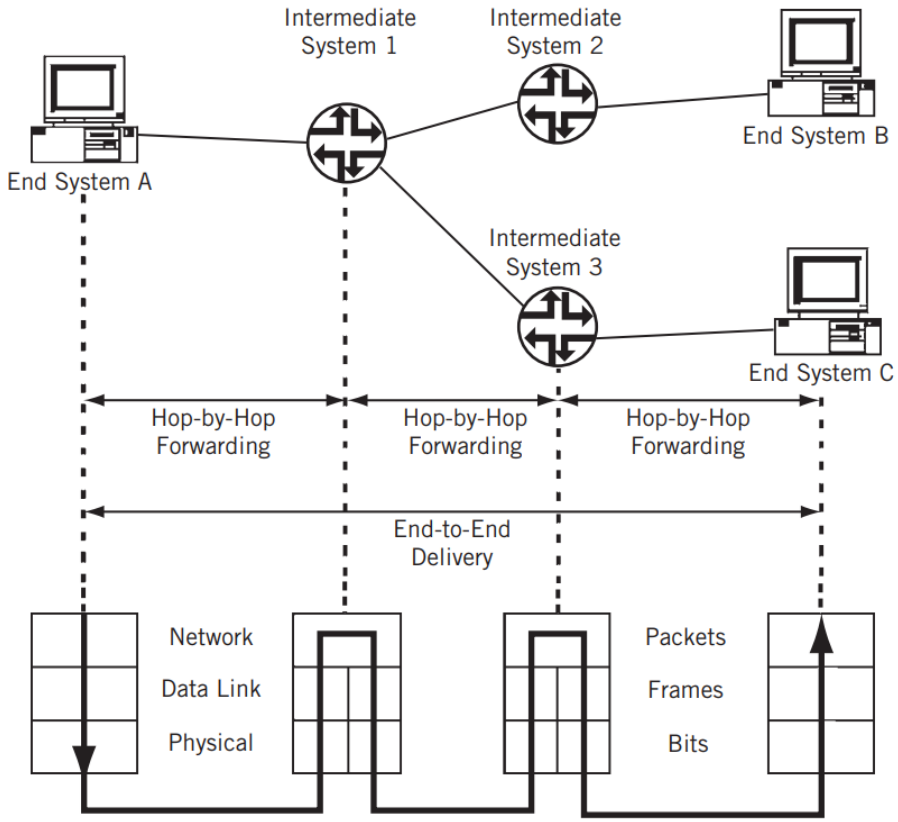


FIGURE 1.17

The network layer uses one or more routing tables to store information about reachable systems. The routing tables must be created, maintained, and purged of old information as the network changes due to failures, the addition or deletion of systems and links, or other configuration changes. This whole process of building tables to pass data from source to destination is called routing, and the use of these tables for packet delivery is called forwarding. The forwarding of packets inside frames always takes place hop by hop. This is shown in Figure 1.17, which adds the network layer to the data link layers already present and distinguishes between hop-by-hop forwarding and end-to-end delivery. On the Internet, the intermediate systems that act at the packet level (Layer 3) are called routers. Devices that act on frames (Layer 2) are called switches, and some older telephony-based WAN architectures use switches as intermediate network nodes. Whether a node is called a switch or router depends on how they function internally.

In a very real sense, the network layer is at the very heart of any protocol stack, and TCP/IP is no exception. The protocol at this layer is IP, either IPv4 or IPv6 (some think that IPv6 is distinct enough to be known as TCPv6/IPv6).

The Transport Layer

Process-to-process delivery is the task of the transport layer. Getting a packet to the destination system is not quite the same thing as determining which process should receive the packet's

content. A system can be running file transfer, email, and other network processes all at the same time, and all over a single physical interface. Naturally, the destination process has to know on which process the sender originated the bits inside the packet in order to reply. Also, systems cannot simply transfer a huge multimegabit file all in one packet. Many data units exceed the maximum allowable size of a packet. This process of dividing message content into packets is known as segmentation. The network layer forwards each and every packet independently, and does not recognize any relationship between the packets. (Is this a file transfer or an email packet? The network layer does not care.) The transport layer, in contrast, can make sure the whole message, often strung out in a sequence of packets, arrives in order (packets can be delivered out of sequence) and intact (there are no errors in the entire message). This function of the transport layer involves some method of flow control and error control (error detection and error correction) at the transport layer, functions which are absent at the network layer. The transport-layer protocol that performs all of these functions is TCP. The transport-layer protocol does not have to do any of this, of course. In many cases, the content of the packet forms a complete unit all by itself, called a datagram. (The term “datagram” is often used to refer to the whole IP packet, but not in this book.) Self-contained datagrams are not concerned with sequencing or flow control, and these functions are absent in the User Datagram Protocol (UDP) at the transport layer. So there are two very popular protocol packages at the transport layer:

- TCP—This is a connection-oriented, “reliable” service that provides ordered delivery of packet contents.
- UDP—This is a connectionless, “unreliable” service that does not provide ordered delivery of packet contents.

In addition to UDP and TCP, there are other transport-layer protocols that can be used in TCP/IP, all of which differ in terms of how they handle transport-layer tasks. Developers are not limited to the standard choices for applications. If neither TCP nor UDP nor any other defined transport-layer service is appropriate for your application, you can write your own transport-layer protocols and get others to adopt it (or use your application package exclusively).

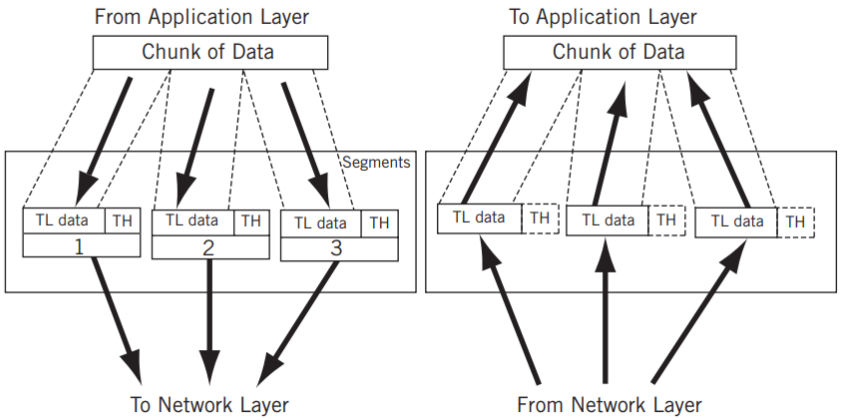


FIGURE 1.18

In TCP/IP, it is often said that the network layer (IP itself) offers an “unreliable” or “best effort” service, while the transport layer adds “reliability” in the form of flow and error control. Later in

this book, we'll see why these terms are unfortunate and what they really mean. The network layer gets a single packet to the right system, and the transport layer gets the entire message to the right process. Figure 1.18 shows the transport layer breaking up a message at the sender into three pieces (each labeled "TL data" for transport-layer data and "TH" for transport-layer header). The figure then shows the transport layer reassembling the message at the receiver from the various segments that make up a message. In TCP/IP, there are also data units known as datagrams, which are always handled as self-contained units. There are profound differences between how the transport layer treats segments and datagrams, but this figure is just a general illustration of segment handling. The functions that the transport layer, which in some protocols is called the end-to-end layer, might have to include the following:

- Process addressing and multiplexing—Also known as "service-point addressing," the transport layer has to decide which process originated the message and to which process the message must be delivered. These are also known as port addresses in TCP/IP. Port addresses are an important portion of the application socket in TCP/IP.
- Segment handling—In cases where each message is divided into segments, each segment has a sequence number used to put the message back together at the destination. When datagrams are used, each data unit is handled independently and sequencing is not necessary

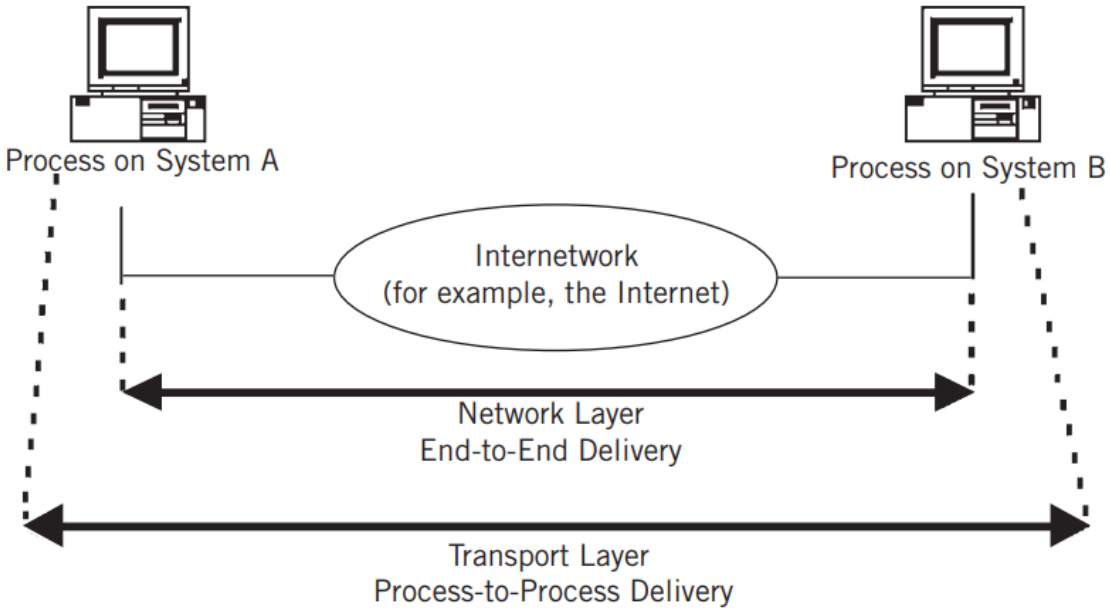


FIGURE 1.19

- Connection control—The transport layer can be connectionless or connection-oriented (in fact, several layers can operate in either one of these ways). Connectionless (CL) layers treat every data unit as a self-contained, independent unit. Connection-oriented (CO) layers go through a three-phase process every time there is data to send to a destination after an idle period (connection durations can vary). First, some control messages establish the connection, then the data are sent (and exchanged if replies are necessary), and finally the connection is closed. Many times, a comparison is made between a telephone conversation ("dial, talk, hang up") with connections and an

intercom (“push and talk any time”) for connectionless communications, but this is not precise. Generally, segments are connection-oriented data units, and datagrams are connectionless data units.

- Flow control—Just as with the data link layer, the transport layer can include flow control mechanisms to prevent senders from overwhelming receivers. In this case, however, the flow control is end-to-end rather than link-by-link. Datagrams do not require this service.
- Error control—This is another function that can be performed at the data link layer, but again end-to-end at the transport layer rather than link-by-link. Communication links are not the only source of errors, which can occur inside a system as well. Again, datagrams do not require this service.

Figure 1.19 shows the relationship between the network layer and the transport layer more clearly. The network layer operates from the network interface to the network interface, while the transport layer is more specific and operates from process to process.

The Application Layer

It might seem that once data are transferred from end-system process to end-system process, the networking task is pretty much complete. There is a lot that still needs to be done at the application level itself. In models of protocol stacks, it is common to place another layer between the transport layer and the user, the application layer. However, the TCP/IP protocol stack really stops at the transport layer (where TCP and UDP are). It is up to the application programmer to decide what should happen at the client and server level at that point, although there are individual RFCs for guidance, such as for FTP. Although it is common to gather these TCP/IP applications into their own layer, there really is no such thing in TCP/IP as an application layer to act as some kind of “glue” between the application’s user and the network. In nearly all TCP/IP stacks, the application layer is part of the application process. In spite of the lack of a defi ned layer, a TCP/IP application might still have a lot to do, and in some ways the application layer is the most complex “layer” of all. There are two major tasks that the application often needs to accomplish: session support and conversion of internal representation. Not all applications need both, of course, and some applications might not need either, but this overview includes both major functions.

Session

Support A session is a type of dialog controller between two processes that establish maintains and synchronizes (controls) the interaction (dialog). A session decides if the communication can be half-duplex (both ends take turns sending) or full-duplex (both ends can send whenever they want). It also keeps a kind of “history” of the interaction between endpoints, so that when things go wrong or when the two communicate again, some information does not have to be resent. In practical terms, the session consists of all “state variables” necessary to construct the history of the connection between the two devices. It is more difficult, but not impossible, to implement sessions in a connectionless environment because there is no easy way to associate the variables with a convenient label.

Internal Representation Conversion

The role of internal representation conversion is to make sure that the data exchange over the network is useful to the receivers. If the internal representation of data differs on the two systems (integer size, bit order in memory, etc.), the application layer translates between the formats so the application program does not have to. This layer can also provide encryption and compression functions, although it is more common to implement these last two functions separately from the network. Standard protocol specifi cations can use the Abstract Syntax Notation 1 (ASN.1) defi nitions for translation purposes. ASN.1 can be used in programming, network management, and other places. ASN.1 defi nes various things such as which bit is “fi rst on the wire” regardless of how it is stored internally, how many bits are to be sent for the numbers 0 through 255 (8), and so on. Everything can be translated into ASN.1, sent across the network, and translated back to whatever internal format is required at the destination. The role of internal representation conversion is shown in Figure 1.20. The figure shows four sequential memory locations, each storing the letter “a” followed by the integer 259. Note that not only are there differences between the amount of memory addressed at once, but also in the order of the bits for numerics. In some protocol stacks, the application program can rely on the services of a fully functional conversion for internal representation to perform these services. However, in TCP/IP, every network application program must do these things for itself.

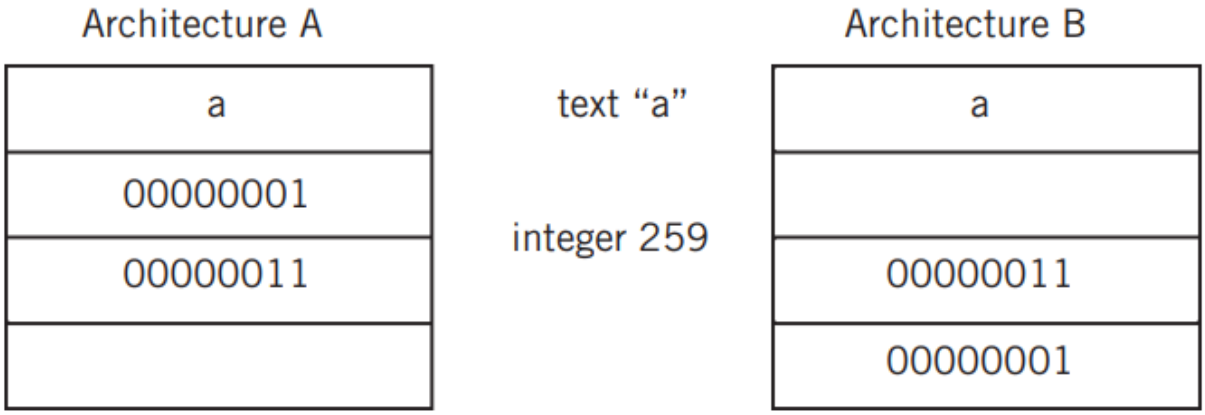


FIGURE 1.20

Applications in TCP/IP

TCP/IP does not provide session or presentation services directly to an application. Programmers are on their own, but this does not mean they have to create everything from scratch. For example, applications can use a character-based presentation service called the Network Virtual Terminal (NVT), part of the Internet’s telnet remote access specification. Other applications can use Sun’s External Data Representation (XDR) or IBM’s (and Microsoft’s) NetBIOS programming libraries for presentation services. In this respect, there are many presentation layer services that TCP/IP can use, but there is no formal presentation service standard in TCP/IP that all applications must use. Host TCP/IP implementations typically provide a range of applications that provide users with access to the data handled by the

transport-layer protocols. These applications use a number of protocols that are not part of TCP/IP proper, but are used with TCP/IP. These protocols include the Hyper-Text Transfer Protocol (HTTP) used by Web browsers, the Simple Message Transfer Protocol (SMTP) used for email, and many others.

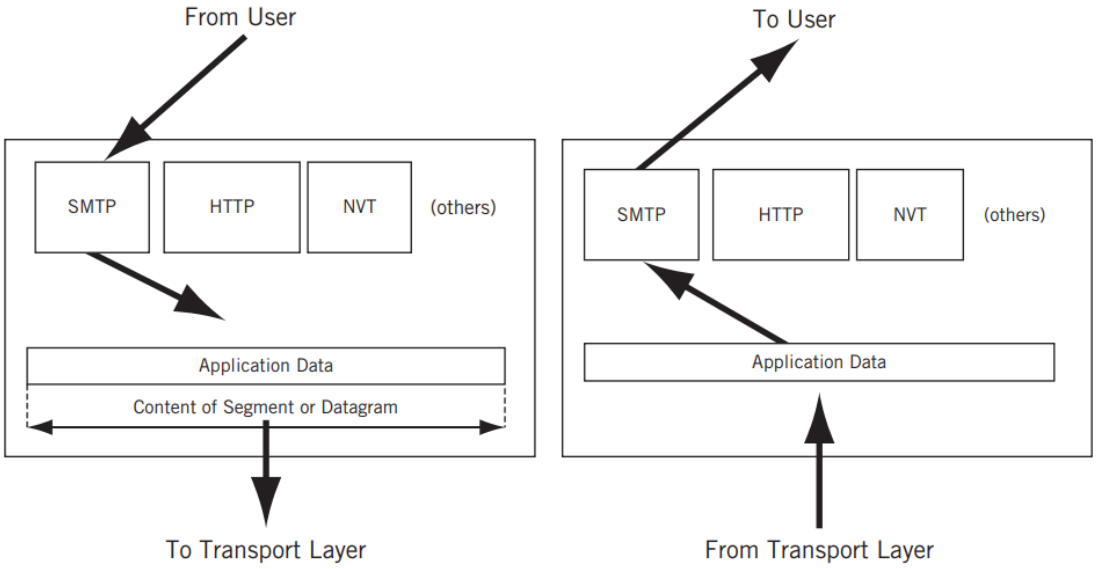


FIGURE 1.21

In TCP/IP, the application protocol, the application service, and the user application itself often share the same name. The file transfer protocol in TCP/IP, called FTP, is at once an application protocol, an application service, and an application run by a user. It can sometimes be confusing as to just which aspect of FTP is under discussion. The role of TCP/IP applications is shown in Figure 1.21. Note that this “layer” sits on top of the TCP/IP protocol stack and interfaces with programs or users directly. Some protocols provide separate layers for sessions, internal representation conversion, and application services. In practice, these are seldom implemented independently. It just makes more sense to bundle them together by major application, as in TCP/IP

THE TCP/IP PROTOCOL SUITE

To sum up, the five layers of TCP/IP are physical, data link, network, transport, and application. The TCP/IP stack is a hierarchical model made up of interactive modules. Each module provides a specific function. In TCP/IP, the layers contain relatively independent protocols that can be “mixed and matched” depending on the needs of the system to provide whatever function is desired. TCP/IP is hierarchical in the sense that each higher-layer protocol is supported by one or more lower-layer protocols. Figure 1.22 maps some of the protocols used in TCP/IP to the various layers of TCP/IP.

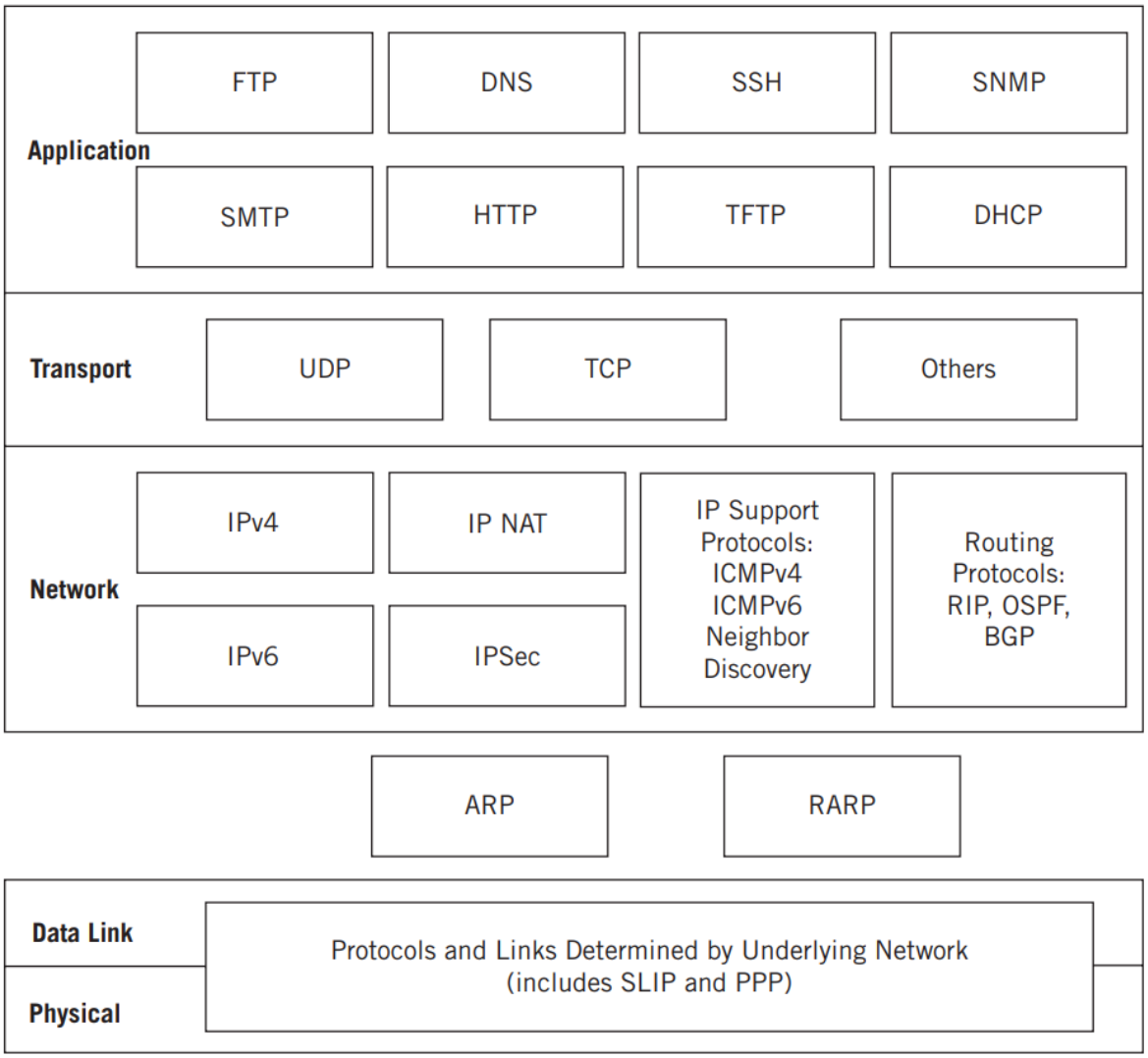


FIGURE 1.22

User Datagram Protocol

The TCP/IP transport layer has another major protocol. UDP is as connectionless as IP. When applications use UDP instead of TCP, there is no need to establish, maintain, or tear down a connection between a source and a destination before sending data. Connection management adds overhead and some initial delay to the network. UDP is a way to send data quickly and simply. However, UDP offers none of the reliability services that TCP does. UDP applications cannot rely on TCP to ensure error-free, guaranteed (via acknowledgments), and in-sequence delivery of data to the destination. For some simple applications, purely connectionless data delivery is good enough. Single request-response message pairs between applications are sent more efficiently with UDP because there is no need to exchange a flurry of initial TCP segments to establish a connection. Many applications will not be satisfied with this mode of operation,

however, because it puts the burden of reliability on the application itself. UDP is often used for short transactions that fit into one datagram and packet. Real-time applications often use UDP with another header inside called the real-time transport protocol (RTP). RTP borrows what it needs from the TCP header, such as a sequence number to detect (but not to resend) missing packets of audio and video, and uses these desirable features in UDP.

IPv4 Addressing and Subnetting

Hardware Addressing

A hardware address is used to uniquely identify a host within a local network. Hardware addressing is a function of the Data-Link layer of the OSI model (Layer-2). Ethernet utilizes the 48-bit MAC address as its hardware address. The MAC address is often hardcoded on physical network interfaces, though some interfaces support changing the MAC address using special utilities. In virtualization environments, dynamically assigning MAC addresses is very common.
A MAC address is most often represented in hexadecimal, using one of two accepted formats:
00:43:AB:F2:32:13
0043.ABF2.3213
The first six hexadecimal digits of a MAC address identify the manufacturer of the physical network interface. This is referred to as the OUI (Organizational Unique Identifier). The last six digits uniquely identify the host itself, and are referred to as the host ID.
The MAC address has one shortcoming – it contains no hierarchy. MAC addresses provide no mechanism to create boundaries between networks. There is no method to distinguish one network from another. This lack of hierarchy poses significant difficulties to network scalability. If only Layer-2 hardware addressing existed, all hosts would technically exist on the same network. Internetworks like the Internet could not exist, as it would be impossible to separate my network from your network. Imagine if the entire Internet existed purely as a single Layer-2 switched network. Switches, as a rule, will forward a broadcast out every port. With billions of hosts on the Internet, the resulting broadcast storms would be devastating. The Internet would simply collapse.
The scalability limitations of Layer-2 hardware addresses are mitigated using logical addresses, covered in great detail in this guide.

Logical Addressing

Logical addressing is a function of the Network layer of the OSI Model (Layer-3), and provides a hierarchical structure to separate networks. Logical addresses are never hardcoded on physical network interfaces, and can be dynamically assigned and changed freely. A logical address contains two components:

- Network ID – identifies which network a host belongs to.
- Host ID – uniquely identifies the host on that network.

Examples of logical addressing protocols include Internetwork Packet Exchange (IPX) and Internet Protocol (IP). IPX was predominantly used on Novell networks, but is now almost entirely deprecated. IP is the most widely-used logical address, and is the backbone protocol of the Internet.

Internet Protocol (IP)

In the 1970's, the Department of Defense developed the Transmission Control Protocol (TCP), to provide both Network and Transport layer functions. When this proved to be an inflexible solution, those functions were separated - with the Internet Protocol (IP) providing Network layer services, and TCP providing Transport layer services. Together, TCP and IP provide the core functionality for the TCP/IP or Internet protocol suite. IP provides two fundamental Network layer services:

- Logical addressing – provides a unique address that identifies both the host, and the network that host exists on.
- Routing – determines the best path to a particular destination network, and then routes data accordingly.

IP was originally defined in RFC 760, and has been revised several times. IP Version 4 (IPv4) was the first version to experience widespread deployment, and is defined in RFC 791.
IPv4 employs a 32-bit address, which limits the number of possible addresses to 4,294,967,296. IPv4 will eventually be replaced by IP Version 6 (IPv6), due to a shortage of available IPv4 addresses.
IPv4 Addressing
A core function of IP is to provide logical addressing for hosts. An IP address provides a hierarchical structure to both uniquely identify a host, and what network that host exists on. An IP address is most often represented in decimal, in the following format:

158.80.164.3

An IP address is comprised of four octets, separated by periods:
First Octet: 158
Second Octet: 80
Third Octet: 164
Fourth Octet: 3
Each octet is an 8-bit number, resulting in a 32-bit IP address. The smallest possible value of an octet is **0, or 00000000** in binary. The largest possible value of an octet is **255, or 11111111** in binary. The above IP address represented in binary would look as follows:
First Octet: 10011110
Second Octet: 01010000
Third Octet: 10100100
Fourth Octet: 00000011

The Subnet Mask

Part of an IP address identifies the network. The other part of the address identifies the host. A subnet mask is required to provide this distinction:

158.80.164.3 255.255.0.0

The above IP address has a subnet mask of 255.255.0.0. The subnet mask follows two rules:

- If a binary bit is set to a 1 (or on) in a subnet mask, the corresponding bit in the address identifies the network.
- If a binary bit is set to a 0 (or off) in a subnet mask, the corresponding bit in the address identifies the host.

Looking at the above address and subnet mask in binary:

IP Address: 10011110.01010000.10100100.00000011
Subnet Mask: 11111111.11111111.00000000.00000000

The first 16 bits of the subnet mask are set to 1. Thus, the first 16 bits of the address (158.80) identify the network. The last 16 bits of the subnet mask are set to 0. Thus, the last 16 bits of the address (164.3) identify the unique host on that network.
The network portion of the subnet mask must be contiguous. For example, a subnet mask of 255.0.0.255 is not valid.

Hosts on the same logical network will have identical network addresses, and can communicate freely. For example, the following two hosts are on the same network:
Host A: 158.80.164.100 255.255.0.0
Host B: 158.80.164.101 255.255.0.0
Both share the same network address (158.80), which is determined by the 255.255.0.0 subnet mask. Hosts that are on different networks cannot communicate without an intermediating device. For example:
Host A: 158.80.164.100 255.255.0.0
Host B: 158.85.164.101 255.255.0.0
The subnet mask has remained the same, but the network addresses are now different (158.80 and 158.85 respectively). Thus, the two hosts are not on the same network, and cannot communicate without a router between them. Routing is the process of forwarding packets from one network to another. Consider the following, trickier example:
Host A: 158.80.1.1 255.248.0.0
Host B: 158.79.1.1 255.248.0.0
The specified subnet mask is now 255.248.0.0, which doesn't fall cleanly on an octet boundary. To determine if these hosts are on separate networks, first convert everything to binary:
Host A Address: 10011110.01010000.00000001.00000001
Host B Address: 10011110.01001111.00000001.00000001
Subnet Mask: 11111111.11111000.00000000.00000000
Remember, the 1 (or on) bits in the subnet mask identify the network portion of the address. In this example, the first 13 bits (the 8 bits of the first octet, and the first 5 bits of the second octet) identify the network. Looking at only the first 13 bits of each address:
Host A Address: 10011110.01010
Host B Address: 10011110.01001

Clearly, the network addresses are not identical. Thus, these two hosts are on separate networks, and require a router to communicate.

IP Address Classes

The IPv4 address space has been structured into several classes. The value of the first octet of an address determines the class of the network:

Class	First Octet Range	Default Subnet Mask
Class A	1 - 127	255.0.0.0
Class B	128 - 191	255.255.0.0
Class C	192 - 223	255.255.255.0
Class D	224 - 239	-

Class A networks range from 1 to 127. The default subnet mask is 255.0.0.0. Thus, by default, the first octet defines the network, and the last three octets define the host. This results in a maximum of 127 Class A networks, with 16,777,214 hosts per network!
Example of a Class A address:
Address: 64.32.254.100 Subnet Mask: 255.0.0.0
Class B networks range from 128 to 191. The default subnet mask is 255.255.0.0. Thus, by default, the first two octets define the network, and the last two octets define the host. This results in a maximum of 16,384 Class B networks, with 65,534 hosts per network.
Example of a Class B address:
Address: 152.41.12.195 Subnet Mask: 255.255.0.0
Class C networks range from 192 to 223. The default subnet mask is 255.255.255.0. Thus, by default, the first three octets define the network, and the last octet defines the host. This results in a maximum of 2,097,152 Class C networks, with 254 hosts per network.
Example of a Class C address:
Address: 207.79.233.6 Subnet Mask: 255.255.255.0
Class D networks are reserved for multicast traffic. Class D addresses do not use a subnet mask.

CIDR (Classless Inter-Domain Routing)

Classless Inter-Domain Routing (CIDR) is a simplified method of representing a subnet mask. CIDR identifies the number of binary bits set to a 1 (or on) in a subnet mask, preceded by a slash.
For example, a subnet mask of 255.255.255.240 would be represented as follows in binary:
11111111.11111111.11111111.11110000
The first 28 bits of the above subnet mask are set to 1. The CIDR notation for this subnet mask would thus be /28.
The CIDR mask is often appended to the IP address. For example, an IP address of
192.168.1.1
and a subnet mask of
255.255.255.0

would be represented as follows using CIDR notation:
192.168.1.1 /24

Address Classes vs. Subnet Mask

Remember the following three rules:

- The first octet on an address dictates the class of that address.
- The subnet mask determines what part of an address identifies the network, and what part identifies the host.
- Each class has a default subnet mask. A network using its default subnet mask is referred to as a classful network.

For example, 10.1.1.1 is a Class A address, and its default subnet mask is 255.0.0.0 (/8 in CIDR).
It is entirely possible to use subnet masks other than the default. For example, a Class B subnet mask can be applied to a Class A address:
10.1.1.1 /16
However, this does not change the class of the above address. It remains a Class A address, which has been subnetted using a Class B mask. Remember, the only thing that determines the class of an IP address is the first octet of that address.
Likewise, the subnet mask is the only thing that determines what part of an address identifies the network, and what part identifies the host.

Subnet and Broadcast Addresses

On each IP network, two host addresses are reserved for special use:

- The subnet (or network) address
- The broadcast address

Neither of these addresses can be assigned to an actual host. The subnet address is used to identify the network itself. A routing table contains a list of known networks, and each network is identified by its subnet address. Subnet addresses contain all 0 bits in the host portion of the address.
For example, 192.168.1.0/24 is a subnet address. This can be determined by looking at the address and subnet mask in binary:
IP Address: 11000000.10101000.00000001.00000000
Subnet Mask: 11111111.11111111.11111111.00000000
Note that all host bits in the address are set to 0. The broadcast address identifies all hosts on a particular network. A packet sent to the broadcast address will be received and processed by every host on that network. Broadcast addresses contain all 1 bits in the host portion of the address.
For example, 192.168.1.255/24 is a broadcast address.
Note that all host bits are set to 1:
IP Address: 11000000.10101000.00000001.11111111
Subnet Mask: 11111111.11111111.11111111.00000000

Broadcasts are one of three types of IP packets:

- Unicasts are packets sent from one host to one other host
- Multicasts are packets sent from one host to a group of hosts
- Broadcasts are packets sent from one host to all other hosts on the local network

A router, by default, will never forward a multicast or broadcast packet from one interface to another.
A switch, by default, will forward a multicast or broadcast packet out every port, except for the port that originated the multicast or broadcast.

Subnetting

Subnetting is the process of creating new networks (or subnets) by stealing bits from the host portion of a subnet mask.
There is one caveat: stealing bits from hosts creates more networks but fewer hosts per network.
Consider the following Class C network:
192.168.254.0
The default subnet mask for this network is
255.255.255.0.
This single network can be segmented, or subnetted, into multiple networks.
For example, assume a minimum of 10 new networks are required. Resolving this is possible using the following magical formula:
2 power n
The exponent ‘n’ identifies the number of bits to steal from the host portion of the subnet mask.
The default Class C mask (255.255.255.0) looks as follows in binary:
11111111.1111111.1111111.00000000
There are a total of 24 bits set to 1, which are used to identify the network. There are a total of 8 bits set to 0, which are used to identify the host, and these host bits can be stolen.
Stealing bits essentially involves changing host bits (set to 0 or off) in the subnet mask to network bits (set to 1 or on). Remember, network bits in a subnet mask must always be contiguous - skipping bits is not allowed. Consider the result if three bits are stolen. Using the above formula:
2 n = 2 3 = 8 = 8 new networks created
However, a total of 8 new networks does not meet the original requirement of at least 10 networks.
Consider the result if four bits are stolen:
2power n = 2 power 4 = 16 = 16 new networks created
A total of 16 new networks does meet the original requirement. Stealing four host bits results in the following new subnet mask:
11111111.11111111.11111111.11110000 = 255.255.255.240

In the previous example, a Class C network was subnetted to create 16 new networks, using a subnet mask of 255.255.255.240 (or /28 in CIDR). Four bits were stolen in the subnet mask,

leaving only four bits for hosts. To determine the number of hosts this results in, for each of the new 16 networks, a slightly modified formula is required:
2 power n – 2
Consider the result if four bits are available for hosts:
2power n – 2 = 2 4 – 2 = 16 – 2 = 14
usable hosts per network Thus, subnetting a Class C network with a /28 mask creates 16 new networks, with 14 usable hosts per network.
Why is the formula for calculating usable hosts 2 power n – 2? Because it is never possible to assign a host an address with all 0 or all 1 bits in the host portion of the address. These are reserved for the subnet and broadcast addresses, respectively. Thus, every time a network is subnetted, useable host addresses are lost.

The 2 power n -2 Rule and Subnetted Networks

To avoid confusion, it was historically unacceptable to use the first and last new networks created when subnetting, as it is possible for a classful network to have the same subnet and broadcast address as its subnetted networks. This required the 2 power n – 2 formula to also be used when calculating the number of new networks created while subnetting. However, this is no longer a restriction for modern equipment and routing protocols. Specifically, on Cisco IOS devices, the following command is now enabled by default:
Router(config)# ip subnet-zero
The ip subnet-zero commands allows for the use of networks with all 0 or all 1 bits in the stolen network portion of the address. Thus, the formula for calculating the number of new networks created is simply 2 n . Remember though, the formula for calculating usable hosts is always 2 power n – 2.

Network Administration

Network administration involves a wide array of operational tasks that help a network to run smoothly and efficiently. Without network administration, it would be difficult for all but the smallest networks to maintain network operations.
The exact definition of "network administration" is hard to pin down. In a larger enterprise, it would more often be strictly related to the actual network. Specifically, this would include the management and maintenance of switches, routers, firewalls, VPN gateways, etc. In smaller companies, the network admin is often a jack-of-all-trades and involved in the configuration of databases, installation, maintenance, and upgrading of software, management of user accounts and security groups, desktop support, and sometimes even basic software development.

The main tasks associated with network administration include:

- Design, installation, and evaluation of the network
- Execution and administration of regular backups

- Creation of precise technical documentation, such as network diagrams, network cabling documents, etc.
- Provision for precise authentication to access network resources
- Provision for troubleshooting assistance
- Administration of network security, including intrusion detection