UNIT – 5

Transport Layer Services

Unit-05/Lecture-01/ Lecture-02

Transport Layer Services [RGPV/ Jun 2013, Dec 2013, Jun 2014]

Transport layer services are conveyed to an application via a programming interface to the transport layer protocols. The services may include the following features:

- **Connection-oriented communication:** It is normally easier for an application to interpret a connection as a data stream rather than having to deal with the underlying connection-less models, such as the datagram model of the User Datagram Protocol (UDP) and of the Internet Protocol (IP).
- **Same order delivery:** The network layer doesn't generally guarantee that packets of data will arrive in the same order that they were sent, but often this is a desirable feature. This is usually done through the use of segment numbering, with the receiver passing them to the application in order. This can cause head-of-line blocking.
- **Reliability:** Packets may be lost during transport due to network congestion and errors. By means of an error detection code, such as a checksum, the transport protocol may check that the data is not corrupted, and verify correct receipt by sending an ACK or NACK message to the sender. Automatic repeat request schemes may be used to retransmit lost or corrupted data.
- **Flow control:** The rate of data transmission between two nodes must sometimes be managed to prevent a fast sender from transmitting more data than can be supported by the receiving data buffer, causing a buffer overrun. This can also be used to improve efficiency by reducing buffer underrun.
- **Congestion avoidance:** Congestion control can control traffic entry into a telecommunications network, so as to avoid congestive collapse by attempting to avoid oversubscription of any of the processing or link capabilities of the intermediate nodes and networks and taking resource reducing steps, such as reducing the rate of sending packets. For example, automatic repeat requests may keep the network in a congested state; this situation can be avoided by adding congestion avoidance to the flow control, including slow-start. This keeps the bandwidth consumption at a low level in the beginning of the transmission, or after packet retransmission.
- **Multiplexing:** Ports can provide multiple endpoints on a single node. For example, the name on a postal address is a kind of multiplexing, and distinguishes between different recipients of the same location. Computer applications will each listen for information on their own ports, which enables the use of more than one network service at the same time. It is part of the transport layer in the TCP/IP model, but of the session layer in the OSI model.

TCP 3-WAY HANDSHAKE (SYN,SYN-ACK,ACK)

The TCP three-way handshake in Transmission Control Protocol (also called the TCP-handshake; three message handshake and/or SYN-SYN-ACK) is the method used by TCP set up a TCP/IP connection over

an Internet Protocol based network. TCP's three way handshaking technique is often referred to as "SYN-SYN-ACK" (or more accurately SYN, SYN-ACK, ACK) because there are three messages transmitted by TCP to negotiate and start a TCP session between two computers. The TCP handshaking mechanism is designed so that two computers attempting to communicate can negotiate the parameters of the network TCP socket connection before transmitting data such as SSH and HTTP web browser requests.

This 3-way handshake process is also designed so that both ends can initiate and negotiate separate TCP socket connections at the same time. Being able to negotiate multiple TCP socket connections in both directions at the same time allows a single physical network interface, such as ethernet, to be multiplexed to transfer multiple streams of TCP data simultaneously.

Below is a (very) simplified diagram of the TCP 3-way handshake process. Have a look at the diagram on the right as you examine the list of events on the left.

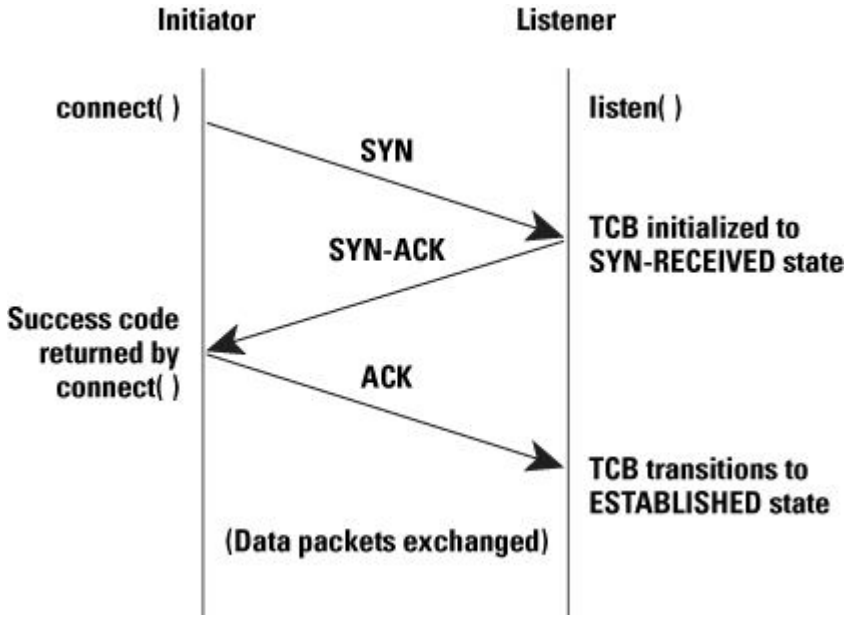


FIGURE: TCP 3-WAY HANDSHAKE DIAGRAM

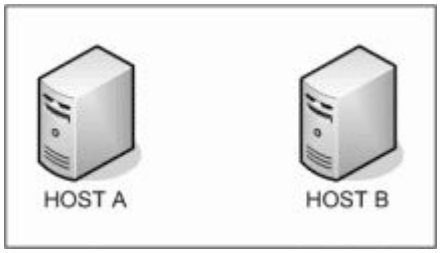


FIGURE: EVENT DIAGRAM

- Host A sends a TCP SYNchronize packet to Host B
- Host B receives A's SYN
- Host B sends a SYNchronize-ACKnowledgement
- Host A receives B's SYN-ACK
- Host A sends ACKnowledge
- Host B receives ACK.
- TCP socket connection is ESTABLISHED.
- TCP Three Way Handshake
- (SYN,SYN-ACK,ACK)

SYNchronize and ACKnowledge messages are indicated by a either the SYN bit, or the ACK bit inside the TCP header, and the SYN-ACK message has both the SYN and the ACK bits turned on (set to 1) in the TCP header.

TCP knows whether the network TCP socket connection is opening, synchronizing, established by using the SYNchronize and ACKnowledge messages when establishing a network TCP socket connection.

When the communication between two computers ends, another 3-way communication is performed to tear down the TCP socket connection. This setup and teardown of a TCP socket connection is part of what qualifies TCP a reliable protocol. TCP also acknowledges that data is successfully received and guarantees the data is reassembled in the correct order.

UDP is connectionless. That means UDP doesn't establish connections as TCP does, so UDP does not perform this 3-way handshake and for this reason, it is referred to as an unreliable protocol. That doesn't mean UDP can't transfer data, it just doesn't negotiate how the connection will work, UDP just transmits and hopes for the best.

PROTOCOLS ENCAPSULATED IN TCP

FTP, Telnet, HTTP, HTTPS, SMTP, POP3, IMAP, SSH and any other protocol that rides over TCP also has a three way handshake performed as connection is opened. HTTP web requests, SMTP emails, FTP file transfers all manage the messages they each send. TCP handles the transmission of those messages.

TCP 'rides' on top of Internet Protocol (IP) in the protocol stack, which is why the combined pair of

Internet protocols is called TCP/IP (TCP over IP). TCP segments are passed inside the payload section of the IP packets. IP handles IP addressing and routing and gets the packets from one place to another, but TCP manages the actual communication sockets between endpoints (computers at either end of the network or internet connection).

Process to Process Delivery

UDP and TCP are transport-layer protocols that create a process-to-process communication.

- UDP is an unreliable and connectionless protocol that requires little overhead and offers fast delivery.
- In the client-server paradigm, an application program on the local host, called the client, needs services from an application program on the remote host, called a server.
- Each application program has a unique port number that distinguishes it from other programs running at the same time on the same machine.
- The client program is assigned a random port number called the ephemeral port number.
- The server program is assigned a universal port number called a well-known port number.
- The combination of the IP address and the port number, called the socket address, uniquely defines a process and a host.
- The UDP packet is called a user datagram.
- UDP has no flow control mechanism.
- Transmission Control Protocol (TCP) is a connection-oriented, reliable, stream transport-layer protocol in the Internet model.
- The unit of data transfer between two devices using TCP software is called a segment; it has 20 to 60 bytes of header, followed by data from the application program.
- TCP uses a sliding window mechanism for flow control.
- Error detection is handled in TCP by the checksum, acknowledgment, and time-out.
- Corrupted and lost segments are retransmitted, and duplicate segments are discarded.
- TCP uses four timers—retransmission, persistence, keep-alive, and time-waited—in its operation.
- Connection establishment requires three steps; connection termination normally requires four steps.
- TCP software is implemented as a finite state machine.
- The TCP window size is determined by the receiver.

S.NO	RGPV QUESTIONS	Year	Marks
Q.1	With neat sketch explain connection establishment and release using 3-way handshaking in transport layer.	Dec 2007	7
Q.2	Explain the issue of addressing, multiplexing and flow control in connection-oriented transport protocol mechanisms.	Dec 2011	7

Q.3	Write short notes on transport services.	Dec 2013	5
Q.4	Explain how the connection is released in a TCP.	Jun 2013	7
Q.5	Explain the following terms in the context of transport layer: Three way handshake What is the purpose of sequence number in TCP packet?	Jun 2013	7
Q.6	How flow control is managed in TCP? Explain in detail.	Jun 2013	7
Q.7	Enlist various services provided by transport layer to its upper layer and explain why they are required.	Jun 2014	7

Unit-05/Lecture-03

UDP

UDP [RGPV/Dec 2009/ Jun 2010/ Dec 2010/ Jun 2011/Dec 2011 / Dec 2012/ Dec 2013]

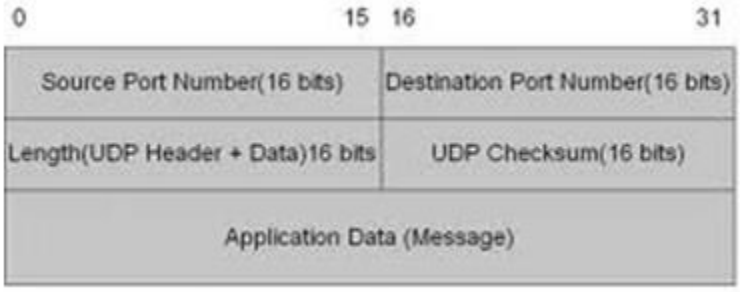


FIGURE : UDP HEADER

General Header

The general header (packet header) defines the endpoints of each association to which the packet belongs, guarantees that the packet belongs to a particular association, and preserves the integrity of the contents of the packet including the header itself. There are four fields in the general header:

- **Source port address** This is a 16-bit field that defines the port number of the process sending the packet.
- **Destination port address** This is a 16-bit field that defines the port number of the process receiving the packet.
- **Verification tag** This is a number that matches a packet to an association. This prevents a packet from a previous association from being mistaken as a packet in this association. It serves as an identifier for the association; it is repeated in every packet during the association. There is a separate verification used for each direction in the association.
- **Checksum** This 32-bit field contains a CRC-32 checksum. Note that the size of the checksum is increased from 16 (in UDP, TCP, and IP) to 32 bits to allow the use of the CRC-32 checksum.

S.NO	RGPV QUESTIONS	Year	Marks
Q.1	Why UDP does exist? Would it not have been enough to just let user processes send raw IP packets?	Jun 2011	7
Q.2	Why is UDP needed? why can't user program directly access IP?	Dec 2011 June 2015	7

Q.3	Give the format of UDP datagram and explain the semantics of every field. How is the checksum in the header computed?	Dec 2010	7	
Q.4	What is UDP? In case where reliability is not a primary importance, UDP would make a good transport protocol. Give example of specific case.	Dec 2009	7	
Q.5	What does UDP provide that is not provided by IP?	Dec 2012	7	

TCP

Unit-05/Lecture-04

TCP SEGMENT[RGPV/Dec 2009/ Jun 2010/ Dec 2011/Dec 2012/ Dec 2013]

The segment consists of a 20- to 60-byte header, followed by data from the application program. The header is 20 bytes if there are no options and up to 60 bytes if it contains options.

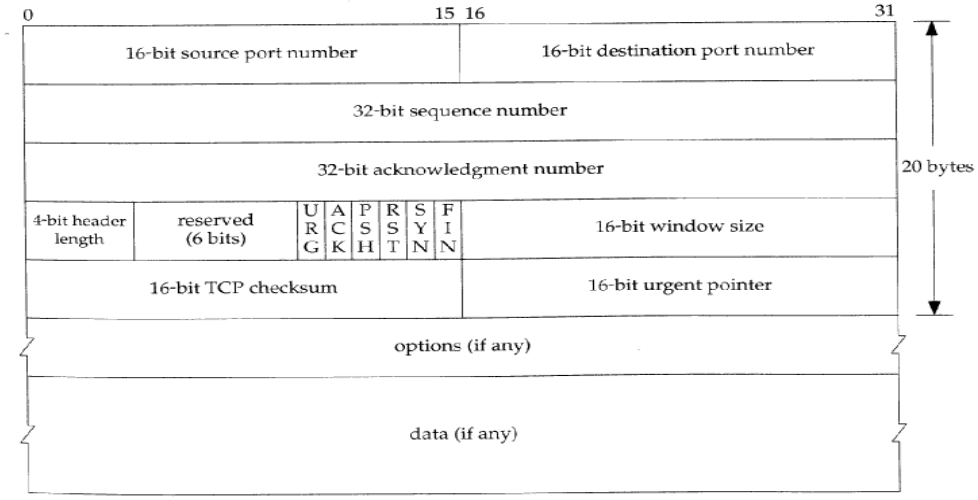


FIGURE : TCP HEADER

- **Source port address** This is a 16-bit field that defines the port number of the application program in the host that is sending the segment. This serves the same purpose as the source port address in the UDP header.
- **Destination port address** This is a 16-bit field that defines the port number of the application program in the host that is receiving the segment. This serves the same purpose as the destination port address in the UDP header.
- **Sequence number** This 32-bit field defines the number assigned to the first byte of data contained in this segment. As we said before, TCP is a stream transport protocol. To ensure connectivity, each byte to be transmitted is numbered. The sequence number tells the destination which byte in this sequence comprises the first byte in the segment. During connection establishment, each party uses a random number generator to create an initial sequence number (ISN), which is usually different in each direction.
- **Acknowledgment number** This 32-bit field defines the byte number that the receiver of the segment is expecting to receive from the other party. If the receiver of the segment has successfully received byte number x from the other party, it defines x + I as the acknowledgment number. Acknowledgment and data can be piggybacked together.

- **Header length** This 4-bit field indicates the number of 4-byte words in the TCP header. The length of the header can be between 20 and 60 bytes. Therefore, the value of this field can be between 5 (5 x 4 =20) and 15 (15 x 4 =60).
- **Reserved** This is a 6-bit field reserved for future use.
- **Control** This field defines 6 different control bits or flags. One or more of these bits can be set at a time.

Control field

- **RST:** Reset the connection
- **SYN:** Synchronize sequence numbers
- **FIN:** Terminate the connection
- **URG**: Urgent pointer is valid
- **ACK:** Acknowledgment is valid
- **PSH:** Request for push

These bits enable flow control, connection establishment and termination, connection abortion, and the mode of data transfer in TCP.

Flag Description

- **URG** The value of the urgent pointer field is valid.
- **ACK** The value of the acknowledgment field is valid.
- **PSH** Push the data.
- **RST** Reset the connection.
- **SYN** Synchronize sequence numbers during connection.
- **FIN** Terminate the connection.
- **Window size** This field defines the size of the window, in bytes, that the other party must maintain. Note that the length of this field is 16 bits, which means that the maximum size of the window is 65,535 bytes. This value is normally referred to as the receiving window (rwnd) and is determined by the receiver. The sender must obey the dictation of the receiver in this case.
- **Checksum** This 16-bit field contains the checksum. The calculation of the checksum for TCP follows the same procedure as the one described for UDP. However, the inclusion of the checksum in the UDP datagram is optional, whereas the inclusion of the checksum for TCP is mandatory. The same pseudoheader, serving the same purpose, is added to the segment. For the TCP pseudoheader, the value for the protocol field is 6.
- **Urgent pointer** This l6-bit field, which is valid only if the urgent flag is set, is used when the segment contains urgent data. It defines the number that must be added to the sequence number to obtain the number of the last urgent byte in the data section of the segment.
- **Options** There can be up to 40 bytes of optional information in the TCP header. We will not discuss these options here; please refer to the reference list for more information.

A TCP Connection

TCP is connection-oriented. A connection-oriented transport protocol establishes a virtual path between the source and destination. All the segments belonging to a message are then sent over this virtual path. Using a single virtual pathway for the entire message facilitates the acknowledgment process as well as retransmission of damaged or lost frames. You may wonder how TCP, which uses the services of IP, a connectionless protocol, can be connection-oriented. The point is that a TCP connection is virtual, not physical. TCP operates at a higher level. TCP uses the services of IP to deliver individual segments to the receiver, but it controls the connection itself. If a segment is lost or corrupted, it is retransmitted. Unlike TCP, IP is unaware of this retransmission. If a segment arrives out of order, TCP holds it until the missing segments arrive; IP is unaware of this reordering.

In TCP, connection-oriented transmission requires three phases: connection establishment, data transfer, and connection termination.

Connection Establishment

TCP transmits data in full-duplex mode. When two TCPs in two machines are connected, they are able to send segments to each other simultaneously. This implies that each party must initialize communication and get approval from the other party before any data are transferred.

Three-Way Handshaking The connection establishment in TCP is called three way handshaking. In our example, an application program, called the client, wants to make a connection with another application program, called the server, using TCP as the transport layer protocol.

The process starts with the server. The server program tells its TCP that it is ready to accept a connection. This is called a request for a passive open. Although the server TCP is ready to accept any connection from any machine in the world, it cannot make the connection itself. The client program issues a request for an active open. A client that wishes to connect to an open server tells its TCP that it needs to be connected to that particular server. TCP can now start the three-way handshaking process. To show the process, we use two time lines: one at each site. Each segment has values for all its header fields and perhaps for some of its option fields, too.

The three steps in this phase are as follows.

1. The client sends the first segment, a SYN segment, in which only the SYN flag is set. This segment is for synchronization of sequence numbers. It consumes one sequence number. When the data transfer starts, the sequence number is incremented by 1. We can say that the SYN segment carries no real data, but we can think of it as containing 1 imaginary byte.

A SYN segment cannot carry data, but it consumes one sequence number.

2. The server sends the second segment, a SYN +ACK segment, with 2 flag bits set: SYN and ACK. This segment has a dual purpose. It is a SYN segment for communication in the other direction and serves

as the acknowledgment for the SYN segment. It consumes one sequence number.

A SYN +ACK segment cannot carry data, but does consume one sequence number.

3. The client sends the third segment. This is just an ACK segment. It acknowledges the receipt of the second segment with the ACK flag and acknowledgment number field. Note that the sequence number in this segment is the same as the one in the SYN segment; the ACK segment does not consume any sequence numbers. An ACK segment, if carrying no data, consumes no sequence number. Simultaneous Open A rare situation, called a simultaneous open, may occur when both processes issue an active open. In this case, both TCPs transmit a SYN + ACK segment to each other, and one single connection is established between them. SYN Flooding Attack The connection establishment procedure in TCP is susceptible to a serious security problem called the SYN flooding attack. This happens when a malicious attacker sends a large number of SYN segments to a server, pretending that each of them is corning from a different client by faking the source IP addresses in the datagrams. The server, assuming that the clients are issuing an active open, allocates the necessary resources, such as creating communication tables and setting timers. The TCP server then sends the SYN +ACK segments to the fake clients, which are lost. During this time, however, a lot of resources are occupied without being used. If, during this short time, the number of SYN segments is large, the server eventually runs out of resources and may crash. This SYN flooding attack belongs to a type of security attack known as a denial-of-service attack, in which an attacker monopolizes a system with so many service requests that the system collapses and denies service to every request.

Some implementations of TCP have strategies to alleviate the effects of a SYN attack. Some have imposed a limit on connection requests during a specified period of time. Others filter out datagrams coming from unwanted source addresses. One recent strategy is to postpone resource allocation until the entire connection is set up, using what is called a cookie. SCTP, the new transport layer protocol that we discuss in the next section, uses this strategy.

Data Transfer

After connection is established, bidirectional data transfer can take place. The client and server can both send data and acknowledgments. We will study the rules of acknowledgment later in the chapter; for the moment, it is enough to know that data traveling in the same direction as an acknowledgment are carried on the same segment. The acknowledgment is piggybacked with the data.

Urgent Data TCP is a stream-oriented protocol. This means that the data are presented from the application program to TCP as a stream of bytes. Each byte of data has a position in the stream. However, on occasion an application program needs to send urgent bytes. This means that the sending application program wants a piece of data to be read out of order by the receiving application program. As an example, suppose that the sending application program is sending data to be processed by the

receiving application program. When the result of processing comes back, the sending application program finds that everything is wrong. It wants to abort the process, but it has already sent a huge amount of data. If it issues an abort command (control +C), these two characters will be stored at the end of the receiving TCP buffer. It will be delivered to the receiving application program after all the data have been processed. The solution is to send a segment with the URG bit set. The sending application program tells the sending TCP that the piece of data is urgent. The sending TCP creates a segment and inserts the urgent data at the beginning of the segment. The rest of the segment can contain normal data from the buffer. The urgent pointer field in the header defines the end of the urgent data and the start of normal data. When the receiving TCP receives a segment with the URG bit set, it extracts the urgent data from the segment, using the value of the urgent pointer, and delivers them, out of order, to the receiving application program.

Connection Termination

Any of the two parties involved in exchanging data (client or server) can close the connection, although it is usually initiated by the client. Most implementations today allow two options for connection termination: three-way handshaking and four-way handshaking with a half-close option.

Three-Way Handshaking Most implementations today allow three-way handshaking for connection termination. In a normal situation, the client TCP, after receiving a close command from the client process, sends the first segment, a FIN segment in which the FIN flag is set. Note that a FIN segment can include the last chunk of data sent by the client. Connection termination using three-way handshaking can be just a control segment as shown in Figure 23.20. If it is only a control segment, it consumes only one sequence number.

The FIN segment consumes one sequence number ifit does not carry data. 2. The server TCP, after receiving the FIN segment, informs its process of the situation and sends the second segment, a FIN +ACK segment, to confirm the receipt of the FIN segment from the client and at the same time to announce the closing of the connection in the other direction. This segment can also contain the last chunk of data from the server. If it does not carry data, it consumes only one sequence number.

The FIN +ACK segment consumes one sequence number if it does not carry data.

3. The client TCP sends the last segment, an ACK segment, to confirm the receipt of the FIN segment from the TCP server. This segment contains the acknowledgment number, which is 1 plus the sequence number received in the FIN segment from the server. This segment cannot carry data and consumes no sequence numbers. Half-Close In TCP, one end can stop sending data while still receiving data. This is called a half-close. Although either end can issue a half-close, it is normally initiated by the client. It can occur when the server needs all the data before processing can begin.

After half-closing of the connection, data can travel from the server to the client and acknowledgments

can travel from the client to the server. The client cannot send any more data to the server. Note the sequence numbers we have used. The second segment (ACK) consumes no sequence number. Although the client has received sequence number y - 1 and is expecting y, the server sequence number is still y - 1. When the connection finally closes, the sequence number of the last ACK segment is still x, because no sequence numbers are consumed during data transfer in that direction.

Flow Control

TCP uses a sliding window to handle flow control. The sliding window protocol used by TCP, however, is something between the Go-Back-N and Selective Repeat sliding window. There are two big differences between this sliding window and the one we used at the data link layer. First, the sliding window ofTCP is byte-oriented; the one we discussed in the data link layer is frame-oriented. Second, the TCP's sliding window is of variable size; the one we discussed in the data link layer was of fixed size.

The bytes inside the window are the bytes that can be in transit; they can be sent without worrying about acknowledgment. The imaginary window has two walls: one left and one right. The window is opened, closed, or shrunk. These three activities, as we will see, are in the control of the receiver (and depend on congestion in the network), not the sender.

The sender must obey the commands of the receiver in this matter. Opening a window means moving the right wall to the right. This allows more new bytes in the buffer that are eligible for sending. Closing the window means moving the left wall to the right. This means that some bytes have been acknowledged and the sender need not worry about them anymore. Slinking the window means moving the right wall to the left. This is strongly discouraged and not allowed in some implementations because it means revoking the eligibility of some bytes for sending. This is a problem if the sender has already sent these bytes. Note that the left wall cannot move to the left because this would revoke some of the previously sent acknowledgments. A sliding window is used to make transmission more efficient as weD as to control the flow of data so that the destination does not become overwhelmed with data. TCP sliding windows are byte-oriented. The size of the window at one end is determined by the lesser of two values: receiver window (rwnd) or congestion window (cwnd). The receiver window is the value advertised by the opposite end in a segment containing acknowledgment. It is the number of bytes the other end can accept before its buffer overflows and data are discarded. The congestion window is a value determined by the network to avoid congestion.

S.NO	RGPV QUESTIONS	Year	Marks
Q.1	Compare TCP and OSI class 4 transport protocol	Dec 2011	7
Q.2	Differentiate between TCP & UDP?	Jun 2010	7

	Enumerate on the various TCP congestion control scheme with their merits and demerits.	Dec 2013 Dec 2014	
Q.3	How does reliability in data transmission? explain	Dec 2012	7
Q.4	How does fast retransmission improve TCP's overall utilization of network resources?	Dec 2010	7
Q.5	Draw and explain the TCP header. Explain each field in detail.	Dec 2012	7
Q.6	What is window management in TCP?	Dec 2012	7

Unit-05/Lecture-05/Lecturer- 06

Integrated Services & Differentiated Services

INTEGRATED SERVICES [RGPV/Dec 2009/Dec 2013/Jun 2014]

Two models have been designed to provide quality of service in the Internet:

- Integrated Services
- Differentiated Services

Both models emphasize the use of quality of service at the network layer (IP), although the model can also be used in other layers such as the data link layer. This means that every user receives the same level of services. This type of delivery does not guarantee the minimum of a service, such as bandwidth, to applications such as real-time audio and video. If such an application accidentally gets extra bandwidth, it may be detrimental to other applications, resulting in congestion. Integrated Services, sometimes called IntServ, is a flow-based QoS model, which means that a user needs to create a flow, a kind of virtual circuit, from the source to the destination and inform all routers of the resource requirement. Integrated Services is a flow based QoS model designed for IP.

Signaling

The reader may remember that IP is a connectionless, datagram, packet-switching protocol. How can we implement a flow-based model over a connectionless protocol? The solution is a signaling protocol to run over IP that provides the signaling mechanism for making a reservation. This protocol is called Resource Reservation Protocol (RSVP) and will be discussed shortly.

Flow Specification

When a source makes a reservation, it needs to define a flow specification.

A flow specification has two parts:

- Rspec (resource specification)
- Tspec (traffic specification)

Rspec defines the resource that the flow needs to reserve (buffer, bandwidth, etc.).

Tspec defines the traffic characterization of the flow.

Admission

After a router receives the flow specification from an application, it decides to admit or deny the service. The decision is based on the previous commitments of the router and the current availability of the resource.

Service Classes

Two classes of services have been defined for Integrated Services:

- guaranteed service
- controlled-load service

Guaranteed Service Class

This type of service is designed for real-time traffic that needs a guaranteed minimum end-to-end delay. The end-to-end delay is the sum of the delays in the routers, the propagation delay in the media, and the setup mechanism. Only the first, the sum of the delays in the routers, can be guaranteed by the router. This type of service guarantees that the packets will arrive within a certain delivery time and are not discarded if flow traffic stays within the boundary of Tspec. We can say that guaranteed services are quantitative services, in which the amount of end-to-end delay and the data rate must be defined by the application.

Controlled-Load Service Class

This type of service is designed for applications that can accept some delays, but are sensitive to an overloaded network and to the danger of losing packets. Good examples of these types of applications are file transfer, e-mail, and Internet access. The controlled load service is a qualitative type of service in that the application requests the possibility of low-loss or no-loss packets.

RSVP

In the Integrated Services model, an application program needs resource reservation. As we learned in the discussion of the IntServ model, the resource reservation is for a flow. This means that if we want to use IntServ at the IP level, we need to create a flow, a kind of virtual-circuit network, out of the IP, which was originally designed as a datagram packet-switched network. A virtual-circuit network needs a signaling system to set up the virtual circuit before data traffic can start. The Resource Reservation Protocol (RSVP) is a signaling protocol to help IP create a flow and consequently make a resource

reservation. Before discussing RSVP, we need to mention that it is an independent protocol separate from the Integrated Services model. It may be used in other models in the future.

Multicast Trees

RSVP is different from some other signaling systems we have seen before in that it is a signaling system designed for multicasting. However, RSVP can be also used for unicasting because unicasting is just a special case of multicasting with only one member in the multicast group. The reason for this design is to enable RSVP to provide resource reservations for all kinds of traffic including multimedia which often uses multicasting.

Receiver-Based Reservation

In RSVP, the receivers, not the sender, make the reservation. This strategy matches the other multicasting protocols. For example, in multicast routing protocols, the receivers, not the sender, make a decision to join or leave a multicast group.

RSVP Messages

RSVP has several types of messages. However, for our purposes, we discuss only two of them:

- Path
- Resv

Path Messages Recall that the receivers in a flow make the reservation in RSVP. However, the receivers do not know the path travelled by packets before the reservation is made. The path is needed for the reservation. To solve the problem, RSVP uses Path messages. A Path message travels from the sender and reaches all receivers in the multicast path. On the way, a Path message stores the necessary information for the receivers. A Path message is sent in a multicast environment; a new message is created when the path diverges.

Resv Messages After a receiver has received a Path message, it sends a Resv message. The Resv message travels toward the sender (upstream) and makes a resource reservation on the routers that support RSVP. If a router does not support RSVP on the path, it routes the packet based on the best-effort delivery methods we discussed before.

Reservation Merging

In RSVP, the resources are not reserved for each receiver in a flow; the reservation is merged. Rc3 requests a 2-Mbps bandwidth while Rc2 requests a I-Mbps bandwidth. Router R3, which needs to make a bandwidth reservation, merges the two requests. The reservation is made for 2 Mbps, the larger of the two, because a 2-Mbps input reservation can handle both requests. The same situation is true for

R2. The reader may ask why Rc2 and Rc3, both belonging to one single flow, request different amounts of bandwidth. The answer is that, in a multimedia environment, different receivers may handle different grades of quality. For example, Rc2 may be able' to receive video only at 1 Mbps (lower quality), while Rc3 may be able to receive video at 2 Mbps (higher quality).

Reservation Styles

When there is more than one flow, the router needs to make a reservation to accommodate all of them. RSVP defines three types of reservation styles.

Wild Card Filter Style In this style, the router creates a single reservation for all senders. The reservation is based on the largest request. This type of style is used when the flows from different senders do not occur at the same time. Fixed Filter Style In this style, the router creates a distinct reservation for each flow. This means that if there are n flows, n different reservations are made. This type of style is used when there is a high probability that flows from different senders will occur at the same time.

Shared Explicit Style In this style, the router creates a single reservation which can be shared by a set of flows.

Soft State

The reservation information (state) stored in every node for a flow needs to be refreshed periodically. This is referred to as a soft state as compared to the hard state used in other virtual-circuit protocols such as ATM or Frame Relay, where the information about the flow is maintained until it is erased. The default interval for refreshing is currently 30 s.

Problems with Integrated Services

There are at least two problems with Integrated Services that may prevent its full implementation in the Internet:

- scalability
- service-type limitation

Scalability

The Integrated Services model requires that each router keep information for each flow. As the Internet is growing every day, this is a serious problem.

Service-Type Limitation

The Integrated Services model provides only two types of services, guaranteed and control-load. Those

opposing this model argue that applications may need more than these two types of services.

DIFFERENTIATED SERVICES [RGPV/Dec 2009/ Jun 2014]

Differentiated Services (DS or Diffserv) was introduced by the IETF (Internet Engineering Task Force) to handle the shortcomings of Integrated Services. Two fundamental changes were made:

- The main processing was moved from the core of the network to the edge of the network. This solves the scalability problem. The routers do not have to store information about flows. The applications, or hosts, define the type of service they need each time they send a packet.
- The per-flow service is changed to per-class service. The router routes the packet based on the class of service defined in the packet, not the flow. This solves the service-type limitation problem. We can define different types of classes based on the needs of applications.

DS Field

In Diffserv, each packet contains a field called the DS field. The value of this field is set at the boundary of the network by the host or the first router designated as the boundary router. IETF proposes to replace the existing TOS (type of service) field in IPv4 or the class field in IPv6 by the DS field

The DS field contains two subfields: DSCP and CU. The DSCP (Differentiated Services Code Point) is a 6-bit subfield that defines the per-hop behavior (PHB). The 2-bit CU (currently unused) subfield is not currently used.

The Diffserv capable node (router) uses the DSCP 6 bits as an index to a table defining the packet-handling mechanism for the current packet being processed.

Per-Hop Behavior

The Diffserv model defines per-hop behaviors (PHBs) for each node that receives a packet. So far three PHBs are defined: DE PHB, EF PHB, and AF PHB. DE PHB The DE PHB (default PHB) is the same as best-effort delivery, which is compatible with TOS.

- Low latency
- Ensured bandwidth

This is the same as having a virtual connection between the source and destination. AF PHB The AF PHB (assured forwarding PHB) delivers the packet with a high assurance as long as the class traffic does not exceed the traffic profile of the node. The users of the network need to be aware that some packets may be discarded.

Traffic Conditioner

To implement Oiffserv, the OS node uses traffic conditioners such as meters, markers, shapers, and

droppers

Meters

The meter checks to see if the incoming flow matches the negotiated traffic profile. The meter also sends this result to other components. The meter can use several tools such as a token bucket to check the profile.

Marker

A marker can remark a packet that is using best-effort delivery (OSCP: 000000) or down-mark a packet based on information received from the meter. Downmarking (lowering the class of the flow) occurs if the flow does not match the profile. A marker does not up-mark (promote the class) a packet.

Shaper

A shaper uses the information received from the meter to reshape the traffic if it is not compliant with the negotiated profile.

Dropper

A dropper, which works as a shaper with no buffer, discards packets if the flow severely violates the negotiated profile.

S.NO	RGPV QUESTIONS	Year	Marks
Q.1	Explain the following- (i) Integrated services (ii) Differentiated services	Dec 2009	7
Q.2	Write short notes on integrated service.	Dec 2013	5
Q.3	Compare integrated service and differentiated service.	Jun 2014	7
Q.4	Discuss all services of transport layers.	Jun 2015	7

Unit-05/Lecture-07

INTERNETWORKING DEVICE

Layered Communication

Network communication models are generally organized into layers. The OSI model specifically consists of seven layers, with each layer representing a specific networking function. These functions are controlled by protocols, which govern end-to-end communication between devices. As data is passed from the user application down the virtual layers of the OSI model, each of the lower layers adds a header (and sometimes a trailer) containing protocol information specific to that layer. These headers are called Protocol Data Units (PDUs), and the process of adding these headers is referred to as encapsulation.

The PDU of each lower layer is identified with a unique term:

#	*Layer*	*PDU Name*
7	Application	-
6	Presentation	-
5	Session	-
4	Transport	**Segments**
3	Network	**Packets**
2	Data-link	**Frames**
1	Physical	**Bits**

Commonly, network devices are identified by the OSI layer they operate at (or, more specifically, what header or PDU the device processes). For example, switches are generally identified as Layer-2 devices, as switches process information stored in the Data-Link header of a frame (such as MAC addresses in Ethernet). Similarly, routers are identified as

Layer-3 devices, as routers process logical addressing information in the Network header of a packet (such as IP addresses).

However, the strict definitions of the terms switch and router have blurred over time, which can result in confusion. For example, the term switch can now refer to devices that operate at layers higher than

Layer-2. This will be explained in greater detail in this guide.

Icons for Network Devices

The following icons will be used to represent network devices

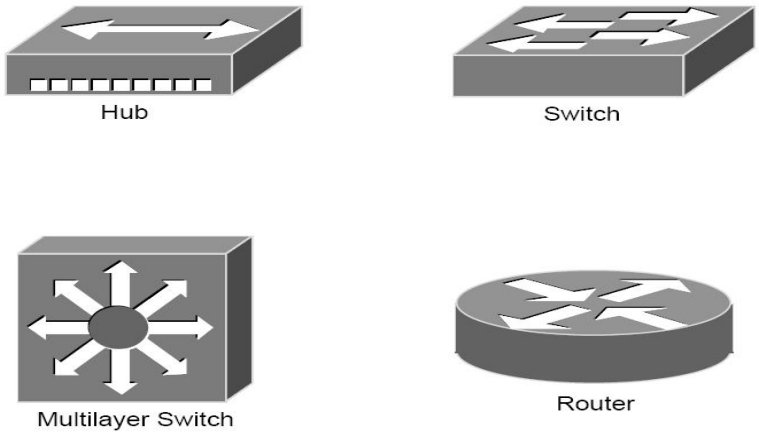


FIGURE: NETWORK DEVICES

Layer-1 Hubs [RGPV/Jun 2011]

Hubs are Layer-1 devices that physically connect network devices together for communication. Hubs can also be referred to as repeaters. Hubs provide no intelligent forwarding whatsoever. Hubs are incapable of processing either Layer-2 or Layer-3 information, and thus cannot make decisions based on hardware or logical addressing.

Thus, hubs will always forward every frame out every port, excluding the port originating the frame. Hubs do not differentiate between frame types, and thus will always forward unicasts, multicasts, and broadcasts out every port but the originating port.

Ethernet hubs operate at half-duplex, which allows a device to either transmit or receive data, but not simultaneously. Ethernet utilizes Carrier Sense Multiple Access with Collision Detect (CSMA/CD) to control media access. Host devices monitor the physical link, and will only transmit a frame if the link is idle.

However, if two devices transmit a frame simultaneously, a collision will occur. If a collision is detected, the hub will discard the frames and signal the host devices. Both devices will wait a random amount of time before resending their respective frames.

Remember, if any two devices connected to a hub send a frame simultaneously, a collision will occur.

Thus, all ports on a hub belong to the same collision domain. A collision domain is simply defined as any physical segment where a collision can occur.

Multiple hubs that are uplinked together still all belong to one collision domain. Increasing the number of host devices in a single collision domain will increase the number of collisions, which can significantly degrade performance.

Hubs also belong to only one broadcast domain – a hub will forward both broadcasts and multicasts out every port but the originating port. A broadcast domain is a logical segmentation of a network, dictating how far a broadcast (or multicast) frame can propagate. Only a Layer-3 device, such as a router, can separate broadcast domains.

Layer-2 Switching[RGPV/ Jun 2014]

Layer-2 devices build hardware address tables, which will contain the following at a minimum:

• Hardware addresses for host devices

• The port each hardware address is associated with

Using this information, Layer-2 devices will make intelligent forwarding decisions based on frame (Data-Link) headers. A frame can then be forwarded out only the appropriate destination port, instead of all ports. Layer-2 forwarding was originally referred to as bridging. Bridging is a largely deprecated term (mostly for marketing purposes), and Layer-2

forwarding is now commonly referred to as switching.

There are some subtle technological differences between bridging and switching. Switches usually have a higher port-density, and can perform forwarding decisions at wire speed, due to specialized hardware circuits called ASICs (Application-Specific Integrated Circuits). Otherwise, bridges and switches are nearly identical in function.

Ethernet switches build MAC-address tables through a dynamic learning process. A switch behaves much like a hub when first powered on. The switch will flood every frame, including unicasts, out every port but the originating port.

The switch will then build the MAC-address table by examining the source MAC address of each frame. Consider the following diagram:

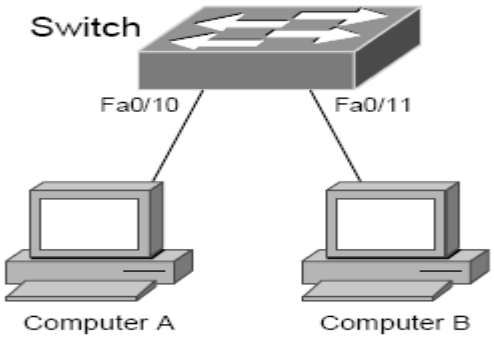


FIGURE: SWITCH

Switches always learn from the source MAC address.

A switch is in a perpetual state of learning. However, as the MAC-address table becomes populated, the flooding of frames will decrease, allowing the switch to perform more efficient forwarding decisions.

While hubs were limited to half-duplex communication, switches can operate in full duplex. Each individual port on a switch belongs to its own collision domain. Thus, switches create more collision domains, which results in fewer collisions.

Like hubs though, switches belong to only one broadcast domain. A Layer- 2 switch will forward both broadcasts and multicasts out every port but the originating port. Only Layer-3 devices separate broadcast domains. Because of this, Layer-2 switches are poorly suited for large, scalable networks. The Layer-2 header provides no mechanism to differentiate one

network from another, only one host from another. This poses significant difficulties. If only hardware addressing existed, all devices would technically be on the same network. Modern 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 Layer-2 switched environment. Switches, as a rule, will forward a broadcast out every port.

Even with a conservative estimate of a billion devices on the Internet, the resulting broadcast storms would be devastating. The Internet would simply collapse.

Both hubs and switches are susceptible to switching loops, which result in destructive broadcast storms. Switches utilize the Spanning Tree Protocol (STP) to maintain a loop-free environment.

Remember, there are three things that switches do that hubs do not:

• Hardware address learning

• Intelligent forwarding of frames

• Loop avoidance

Hubs are almost entirely deprecated – there is no advantage to using a hub over a switch. At one time, switches were more expensive and introduced more latency (due to processing overhead) than hubs, but this is no longer the case.

Layer-2 Forwarding Methods

Switches support three methods of forwarding frames. Each method copies all or part of the frame into memory, providing different levels of latency and reliability. Latency is delay - less latency results in quicker forwarding. The Store-and-Forward method copies the entire frame into memory, and performs a Cycle Redundancy Check (CRC) to completely ensure the integrity of the frame. However, this level of error-checking introduces the highest latency of any of the switching methods. The Cut-Through (Real Time) method copies only enough of a frame's header to determine its destination address. This is generally the first 6 bytes following the preamble. This method allows frames to be transferred at wire speed, and has the least latency of any of the three methods. No error checking is attempted when using the cut-through method. The Fragment-Free (Modified Cut-Through) method copies only the first 64 bytes of a frame for error-checking purposes. Most collisions or corruption occur in the first 64 bytes of a frame. Fragment-Free represents a compromise between reliability (store-and-forward) and speed (cut-through).

Layer-3 Routing

Layer-3 routing is the process of forwarding a packet from one network to another network, based on the Network-layer header. Routers build routing tables to perform forwarding decisions, which contain the following:

• The destination network and subnet mask

• The next hop router to get to the destination network

• Routing metrics and Administrative Distance

Note that Layer-3 forwarding is based on the destination network, and not the destination host. It is possible to have host routes, but this is less common.

The routing table is concerned with two types of Layer-3 protocols:

• Routed protocols - assigns logical addressing to devices, and routes packets between networks.

Examples include IP and IPX.

• Routing protocols - dynamically builds the information in routing tables. Examples include RIP, EIGRP, and OSPF.

Each individual interface on a router belongs to its own collision domain.

Thus, like switches, routers create more collision domains, which results in fewer collisions.

Unlike Layer-2 switches, Layer-3 routers also separate broadcast domains. As a rule, a router will never forward broadcasts from one network to another network.

Routers will not forward multicasts either, unless configured to participate in a multicast tree.

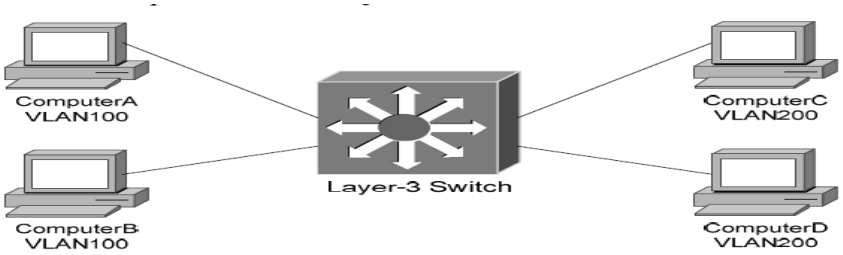
Traditionally, a router was required to copy each individual packet to its buffers, and perform a route-table lookup. Each packet consumed CPU cycles as it was forwarded by the router, resulting in latency. Thus, routing was generally considered slower than switching.

It is now possible for routers to cache network-layer flows in hardware, greatly reducing latency. This has blurred the line between routing and switching, from both a technological and marketing standpoint.

VLANs – A Layer-2 or Layer-3 Function?

a switch will forward both broadcasts and multicasts out every port but the originating port.

However, a switch can be logically segmented into multiple broadcast domains, using Virtual LANs (or VLANs).



Each VLAN represents a unique broadcast domain:

• Traffic between devices within the same VLAN is switched (forwarded at Layer-2).

• Traffic between devices in different VLANs requires a Layer-3 device to communicate.

Broadcasts from one VLAN will not be forwarded to another VLAN. This separation provided by VLANs is not a Layer-3 function. VLAN tags are inserted into the Layer-2 header.

Thus, a switch that supports VLANs is not necessarily a Layer-3 switch. However, a purely Layer-2 switch cannot route between VLANs. Remember, though VLANs provide separation for Layer-3 broadcast domains, and are often associated with IP subnets, they are still a Layer-2 function.

Layer-3 Switching

In addition to performing Layer-2 switching functions, a Layer-3 switch must also meet the following criteria:

• The switch must be capable of making Layer-3 forwarding decisions (traditionally referred to as routing).

• The switch must cache network traffic flows, so that Layer-3 forwarding can occur in hardware.

Many older modular switches support Layer-3 route processors – this alone does not qualify as Layer-3 switching. Layer-2 and Layer-3 processors can act independently within a single switch chassis, with each packet requiring a route-table lookup on the route processor.

Layer-3 switches leverage ASICs to perform Layer-3 forwarding in hardware. For the first packet of a particular traffic flow, the Layer-3 switch will perform a standard route-table lookup. This flow is then cached in hardware – which preserves required routing information, such as the destination network and the MAC address of the corresponding next-hop.

Traffic between devices within the same VLAN, such as Computer A and Computer B, is switched at Layer-2 as normal. The first packet between devices in different VLANs, such as Computer A and Computer D is routed.

The switch will then cache that IP traffic flow, and subsequent packets in that flow will be switched in hardware.

Layer-3 Switching vs. Routing – End the Confusion!

The evolution of network technologies has led to considerable confusion over the terms switch and router. Remember the following:

• The traditional definition of a switch is a device that performs Layer-2 forwarding decisions.

• The traditional definition of a router is a device that performs Layer-3 forwarding decisions.

switching functions were typically performed in hardware, and routing functions were typically performed in software. This resulted in a widespread perception that switching was fast, and routing

was slow (and expensive).

Once Layer-3 forwarding became available in hardware, marketing gurus muddied the waters by distancing themselves from the term router. Though Layer-3 forwarding in hardware is still routing in every technical sense, such devices were rebranded as Layer-3 switches.

Ignore the marketing noise. A Layer-3 switch is still a router.

Compounding matters further, most devices still currently referred to as routers can perform Layer-3 forwarding in hardware as well. Thus, both Layer-3 switches and Layer-3 routers perform nearly identical functions at the same performance.

There are some differences in implementation between Layer-3 switches and routers, including (but not limited to):

• Layer-3 switches are optimized for Ethernet, and are predominantly used for inter-VLAN routing. Layer-3 switches can also provide Layer-2 functionality for intra-VLAN traffic.

• Switches generally have higher port densities than routers, and are considerably cheaper per port than routers (for Ethernet, at least).

• Routers support a large number of WAN technologies, while Layer-3 switches generally do not.

• Routers generally support more advanced feature sets.

Layer-3 switches are often deployed as the backbone of LAN or campus networks. Routers are predominantly used on network perimeters, connecting to WAN environments.

Multilayer Switching

Multilayer switching is a generic term, referring to any switch that forwards traffic at layers higher than Layer-2. Thus, a Layer-3 switch is considered a multilayer switch, as it forwards frames at Layer-2 and packets at Layer-3.

A Layer-4 switch provides the same functionality as a Layer-3 switch, but will additionally examine and

cache Transport-layer application flow information, such as the TCP or UDP port. By caching application flows, QoS (Quality of Service) functions can be applied to preferred applications.

S.NO	RGPV QUESTIONS	Year	Marks
Q.1	What is hub? Explain various types of hub? How hub is different from switch?	Jun 2011	7
Q.2	What do you mean by bridges?	Jun 2007	7
Q.3	List and discuss advantages & disadvantages of bridges relative to a repeater.	Jun 2005	7
Q.4	Distinguish between the functions of the following networking devices: hub, l2 switch, routers and gateways. Explain giving neat sketches.	Jun 2014 June 2015	7

Unit-05/Lecture-08

Configuring TCP/IP

Configuring TCP/IP

This chapter describes configuring TCP/IP for the Microsoft Windows platforms that are supported by the Oracle Transparent Gateway for DRDA. TCP/IP is a communications facility that is already part of the operating system. No third-party protocol software is required. Read this chapter to learn more about configuring TCP/IP.

Port Number

The DRDA standard specifies that port 446 be used for DRDA services. However, if several DRDA Servers are operating on the same system, then they will need to provide service on different ports. Therefore, the port number that is used by each DRDA Server will need to be extracted from the configuration of each individual DRDA Server. DB2 for OS/390 and DB2/400 typically use the DRDA standard port number, 446, whereas DB2/UDB typically uses 50000 as the port number. Refer to IBM DB2 Administrator and Installation guides for locating and changing these port numbers for your DRDA Server. For additional information, consult your DB2 DBA or System Administrator.

Configuring TCP/IP

The following configuration example is for Microsoft Windows NT 4.0. Other Microsoft Windows operating systems may have these panels in a different location or may present them differently, but the required contents will be essentially the same.

You configure TCP/IP from the network configuration tool in the Microsoft Windows Control Panel.

Click the Protocol tab and select TCP/IP Protocol. Then click the Properties button to display the Properties panel.

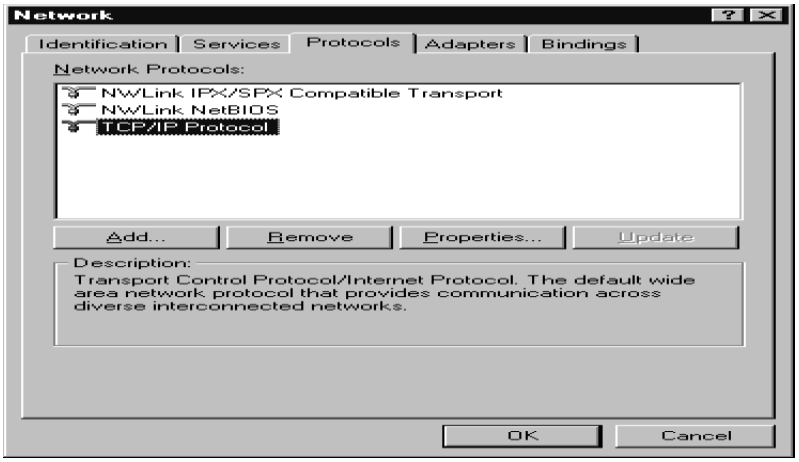


FIGURE: NETWORK CONFIGURATION TOOL

If the TCP/IP Protocol is not already installed, click Add and then select the TCP/IP Protocol.

Configuration consists of assigning a Hostname, an IP Address, and a Network Mask to a given network interface.

In the IP Address tab, use the pull-down list to choose the Adapter you will use. Your network administrator can tell you whether you will be using DHCP or a static IP address. If using a static IP, then you must enter the appropriate values for IP Address, Subnet Mask, and Default Gateway.

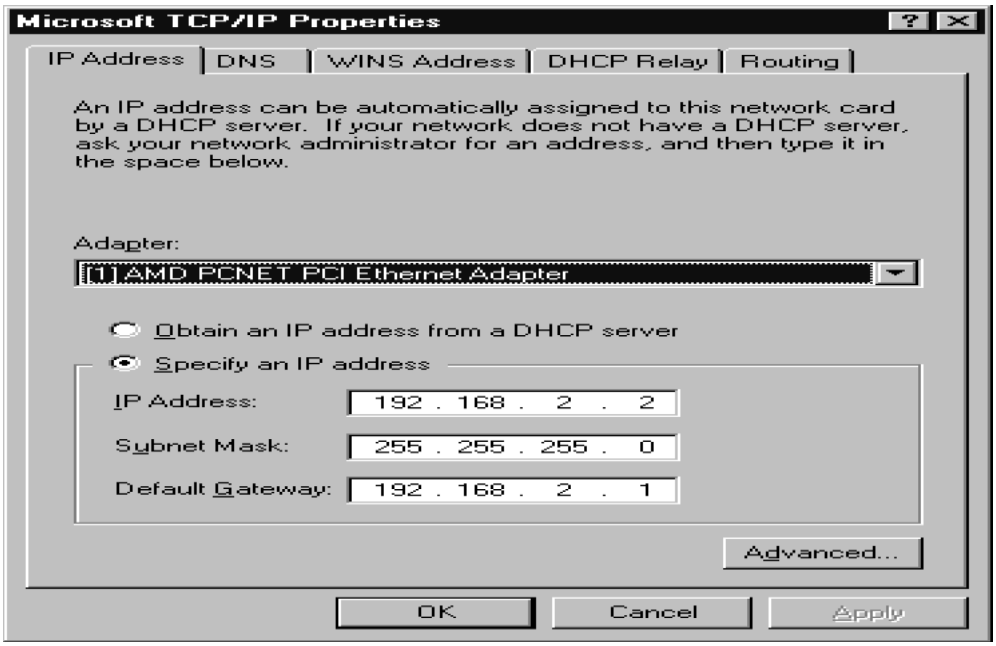


FIGURE: TCP/IP PROPERTIES PANEL

Additional configuration consists of defining a Name Server IP Address or creating entries in the Hosts file on the local machine. Name Servers translate host names into IP Addresses when queried on a particular host name. The Hosts file provides this same functionality, but in a non-network participating manner.

The Hosts file may be edited with a text editor of your choosing. For example, in Microsoft Windows NT, the file is located in:

C:\winnt\system32\drivers\etc\hosts

where C:\winnt is your Windows NT system root.

In order to use a Name Server, you must configure the TCP/IP to use DNS. Click on the DNS tab and enter a Host Name and Domain Name. Your network administrator will provide these values. Click the Add button below the Domain Suffix Search Order box and enter the IP Address of the Name Server. You may enter up to three name servers. Click [OK].

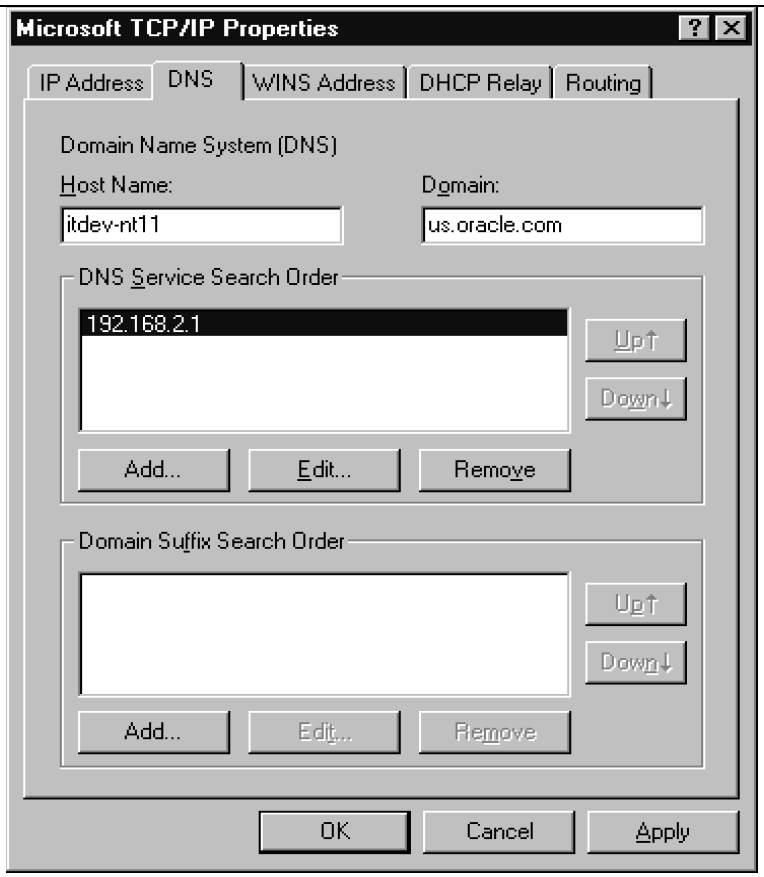


FIGURE :DEFINE A NAME SERVER

For local configuration (in other words, the gateway and the DRDA Server are on the same machine), it may be desirable to use the loop-back address. The IP address is 127.0.0.1 and is typically given the local name ("localhost" or "loopback") in the Hosts file. Using the loop-back address reduces the amount of network overhead by handling the traffic internally without actually talking to the network.
The gateway is configured for TCP/IP using the DRDA_CONNECT_PARM initialization file parameter. In an SNA configuration, this parameter would be set to the Side Information Profile name (name set). In a TCP/IP configuration, this parameter should be set to the IP address or Host name of the DRDA Server, which should be followed by the Service Port number of that server.
The rest of the DRDA-specific parameters are unrelated to the communications protocol and may be set the same for either SNA or TCP/IP installations.
Example #1: Configuration for a DRDA Server on a host named 'mvs01.domain.com' (or IP address of 192.168.1.2) with a Service Port number of 446.
DRDA_CONNECT_PARM=mvs01.domain.com:446
or
DRDA_CONNECT_PARM=192.168.1.2:446

Example #2: Configuration for a DRDA Server on the same host as the gateway with a Service Port number of 446.

DRDA_CONNECT_PARM=localhost:446

or
DRDA_CONNECT_PARM=127.0.0.1:446

S.NO	RGPV QUESTIONS	Year	Marks
Q.1	Write a short note on IP configuration.	June 2015	7
Q.2	Differentiate between TCP & UDP?	Jun 2010 Dec 2013	7

UNIT 5/LECTURE 09

ipconfig, ping command

Commands [RGPV/Jun 2010/ Jun 2011/ Dec 2013]

To test a TCP/IP configuration by using the ping command

1. To quickly obtain the TCP/IP configuration of a computer, open Command Prompt, and then type ipconfig. From the display of the ipconfig command, ensure that the network adapter for the TCP/IP configuration you are testing is not in a Media disconnected state.

At the command prompt, pings the loopback address by typing ping 127.0.0.1.

3. Ping the IP address of the computer.
4. Ping the IP address of the default gateway.

If the ping command fails, verify that the default gateway IP address is correct and that the gateway (router) is operational.

5. Ping the IP address of a remote host (a host that is on a different subnet).

If the ping command fails, verify that the remote host IP address is correct, that the remote host is operational, and that all of the gateways (routers) between this computer and the remote host are operational.

6. Ping the IP address of the DNS server

If the ping command fails, verify that the DNS server IP address is correct, that the DNS server is operational, and that all of the gateways (routers) between this computer and the DNS server are operational.

To open command prompt, click Start, point to All Programs, point to Accessories, and then click Command Prompt.

If the ping command is not found or the command fails, you can use Event Viewer to check the System Log and look for problems reported by Setup or the Internet Protocol (TCP/IP) service.

The ping command uses Internet Control Message Protocol (ICMP) Echo Request and Echo Reply messages. Packet filtering policies on routers, firewalls, or other types of security gateways might prevent the forwarding of this traffic.

The ipconfig command is the command-line equivalent to the winipcfg command, which is available in Windows Millennium Edition, Windows 98, and Windows 95. Windows XP does not include a graphical equivalent to the winipcfg command; however, you can get the equivalent functionality for viewing and renewing an IP address by opening Network Connections, right-clicking a network connection, clicking Status, and then clicking the Support tab. Used without parameters, ipconfig displays the IP address, subnet mask, and default gateway for all adapters.

To run ipconfig, open the command prompt, and then type ipconfig.

To open Network Connections, click Start, click Control Panel, click Network and Internet Connections, and then click Network Connections.

S.NO	RGPV QUESTIONS	Year	Marks
Q.1	Write a short note on IP configuration.	June 2015	3