GENERAL SIR JOHN KOTELAWALA DEFENCE UNIVERSITY

Communication Networks

ET 3102

https://classroom.google.com/u/1/c/NjUwODgwNjkwNTY3

Classroom > Communication Networks - ET 3102

- Photonic and Laser Engine...
- Next Generation Cellular N...
- Communication Technology
- Communication Theory
- Communication Systems
- Deep Learning
- Machine Learning
- 35th Intake : EE & ET
- Individual Design Project (...
- Random Signals and Proce...

Archived classes

Settings

Stream | Classwork | People | Grades

Class code

nhqhb2c

Announce something to your class

Upcoming

No work due soon

View all

This is where you can talk to your class

Use the stream to share announcements, post assignments, and respond to student questions

Stream settings

10:49 AM

Outline

Overview on ISO/OSI reference model for open systems, packet and distributed systems and Topologies.

Physical and Data Link Layers.

Network (IP) and Transport Layers (TCP/UDP).

Session Layer, Presentation and Application Layer.

Local Area Network and Wide Area Networks.

Submodule Outline

Transport Layers (TCP/UDP)

TCP: Transmission Control Protocol,

UDP: User Datagram Protocol,

segmentation and reassembling, Service point addressing,

Flow control and congestion control,

error control schemes.

Overview of Mobile IP and Mobility management

Review of Concepts Learned So far…

LAYER 1

- Repeater, hub An interconnection at layer 1 is realized by simply propagating the electrical signal further.
- This may require some signal amplification; a device realizing this is called a repeater.
- The name clarifies that it does nothing but send the signal forward, enabling its propagation to a longer distance.
- If the signal is digital, the repeater may also eliminate distortion effects introduced by the channel.
- Such an operation is called regeneration.
- A regenerative repeater tries to clean the signal from noise before amplifying it (which would result in amplifying the noise as well).
- Sometimes, multiple repeaters are integrated in a single device called hub; the difference between repeater and hub is in the number of interconnected lines, which is two for the former and higher for the latter.
- Actually, a hub does not even need to amplify the signal, the terminology just emphasizes its ability to connecting multiple inputs.
- Repeaters or hubs perform the simplest kind of interconnection.
- These devices are blind to the information content of what they are transmitting (repeating).
- They simply take the electrical signal and send it forward, improving its physical quality.
- Upper layer devices instead use information with higher level of abstraction.

LAYER 2

- Bridge, switch The devices operating on the data link layer are called bridges; a device with the same functionality as a bridge, but with the additional requirement of multiple input capability, is often given a different name: a switch.
- When layer 2 is considered, the requirement of connecting multiple lines is very common.
- Thus, the terms “bridge” and “switch” are often used as synonyms.
- The interconnection realized by a switch involves multiple access, which is no longer regulated by physical (electrical) characteristics of the signal, but rather by the MAC sublayer.
- A switch does not only interpret the bits of a packet, but also its content.
- In practice, the switch extracts the MAC address of the destination, which is contained in the packet, and uses this piece of information to determine how to connect the lines.
- However, bridges and switches cannot realize all kinds of interconnections.
- In particular, they do not have the capability to interconnect all kinds of medium-access protocols because the MAC addressing would be different.
- For example, a network operating on a hybrid wired/wireless physical layer – where some communication lines are cables and others are radio links, is clearly impossible to realize via repeaters, which do not interconnect different physical technologies. Switches can realize an interconnection only between certain kinds of differentMAC sublayers, for example between Ethernet and Fast Ethernet.
- However, to enable an entirely general interface of different medium access mechanisms, we must transfer to the
- network layer.

LAYER 3

- *Router* The purpose of the network layer is to allow communication between different kinds of technologies in a single network construct.
- This is achieved by means of a universal reference system, for example realized through IP addressing.
- The interconnection devices at layer 3 are called *routers*.
- In fact, as the name suggests, routers are in charge of directing the messages from source to destination, an operation called *routing*.
- For this reason, routers they are the places where routing decisions are made.
- Actually, hubs can also sometimes operate some simple routing procedures but these are very rough techniques that do not require interactions at the network layer and especially operate with a specific medium access control (MAC).
- Routers are instead capable of performing much more complex procedures, in an entirely general context.
- Note also that they always connect several lines at once, thus there is no need for a different name for single-input and multiple-input connecting devices.

LAYER 4 and Above

- Gateway Devices realizing connections at layers higher than 3 are often generically denoted as gateways.
- They includes transport gateways, which operate at layer 4 and are able to interconnect parts of the network using a different transport protocol, as well as application gateways, which are able to understand the content of the exchanged data and translate it into a different format.
- In general, the term is used to denote all interconnecting devices that have higher capability than simply managing routing of packets; for example, certain gateways allow the connection end points to be reached more efficiently, or filter the communication to improve security; these functions are sometimes called proxy and firewall.
- Other gateways are able to perform network address translation (NAT).

NETWORK LAYER ROUTING

- The term “routing” describes the procedures to select paths (also called routes) in a network.
- A common example of routing in everyday life involves the selection of roads when driving towards a destination.
- Information networks require routing decisions to guide the messages through the communication devices to the final destination.
- The routing problem exists even outside the field of telecommunication networks;
- it was identified and studied by mathematicians in general contexts, which has given it a rigorous and theoretically solid basis. However, the strong impulse that the task of finding efficient routes has received in the last decades is motivated by the growth of telecommunication networks.
- The routing problem will be regarded at many levels.

OSI Protocol Layer Functional Summary

Layer	Name	Functions
7	Application	User level data.
6	Presentation	Standardized data appearance, blocking, text compression.
5	Session	Sessions or logical connections between parts of an application; message sequencing, recovery.
4	Transport	Flow control, end-to-end error detection and correction, priority service.
3	Network	Routing, message blocking into uniformly sized packets.
2	Data Link	Reliable data delivery over physical medium, transmission error recovery, separating packets into uniformly sized frames.
1	Physical	Actual communication across physical medium, individual bit transmission.

Internet Communication Layers Functional Summary

Layer	Functions	Responsibilities
Application	Prepare messages from user interactions.	User interaction, addressing.
Transport	Convert messages to packets.	Sequencing, reliability (integrity), error correction.
Internet	Convert packets to datagrams.	Flow control, routing.
Physical	Transmit datagrams as individual bits.	Data communication

Internet Communication Layer Protocol Summary

Layer	TCP Protocols	UDP Protocols
Application	HTTP (Hypertext Transfer Protocol): Used for communicating webpages. SMTP (Simple Mail Transfer Protocol): Used for communicating e-mail. FTP (File Transfer Protocol): Used for receiving or sending files Telnet (Terminal Emulation Protocol): Used for performing remote operations as if directly connected to the host from a terminal and others.	SNMP (Simple Network Monitoring Protocol): Used for controlling network devices. Syslog(System Audit Log): Used for entering records in the system log. Time: Used for communication and synchronizing time among network devices and others.
Transport	TCP.	UDP.
Internet	IP.	IP.
Physical	Data Communication.	Data communication.

EXAMPLE

When we enter a URL in a browser :

- Browser checks cache for DNS entry to find the corresponding IP address of website.
- It looks for following cache. If not found in one, then continues checking to the next until found.
  - Browser Cache
  - Operating Systems Cache
  - Router Cache
  - ISP Cache
- If not found in cache, ISP's (Internet Service Provider) DNS server initiates a DNS query to find IP address of server that hosts the domain name.
- The requests are sent using small data packets that contain information content of request and IP address it is destined for.
- Browser initiates a TCP (Transfer Control Protocol) connection with the server using synchronize(SYN) and acknowledge(ACK) messages.
- Browser sends an HTTP request to the web server. GET or POST request.
- Server on the host computer handles that request and sends back a response. It assembles a response in some format like JSON, XML and HTML.
- Server sends out an HTTP response along with the status of response.
- Browser displays HTML content

Transport Layer in Summary Form

- At the highest level, the transport is concerned with how bytes are transmitted, while the protocol determines which bytes to transmit (and to some extent when).

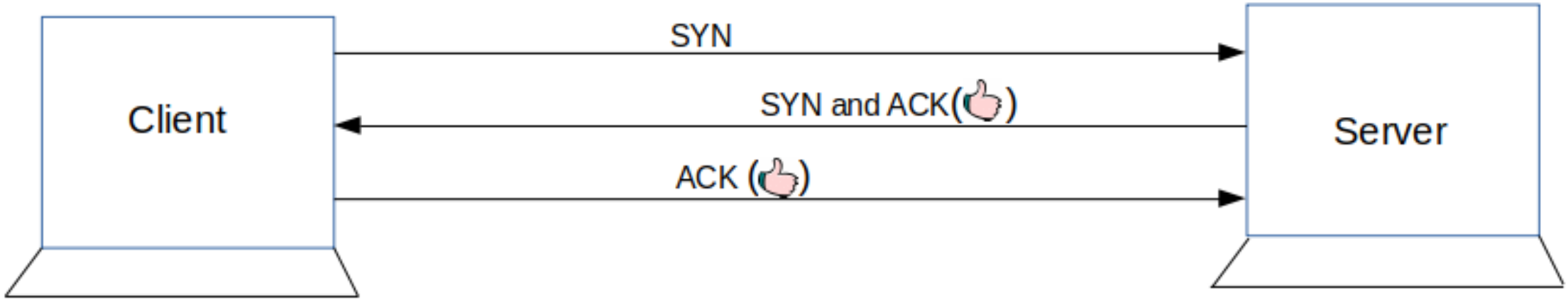
- A different way of saying the same thing: a transport is an abstraction for a socket (or similar I/O endpoint) while a protocol is an abstraction for an application, from the transport's point of view.

- Yet another view is the transport and protocol interfaces together define an abstract interface for using network I/O and interprocess I/O.

- There is always a 1:1 relationship between transport and protocol objects: the protocol calls transport methods to send data, while the transport calls protocol methods to pass it data that has been received.

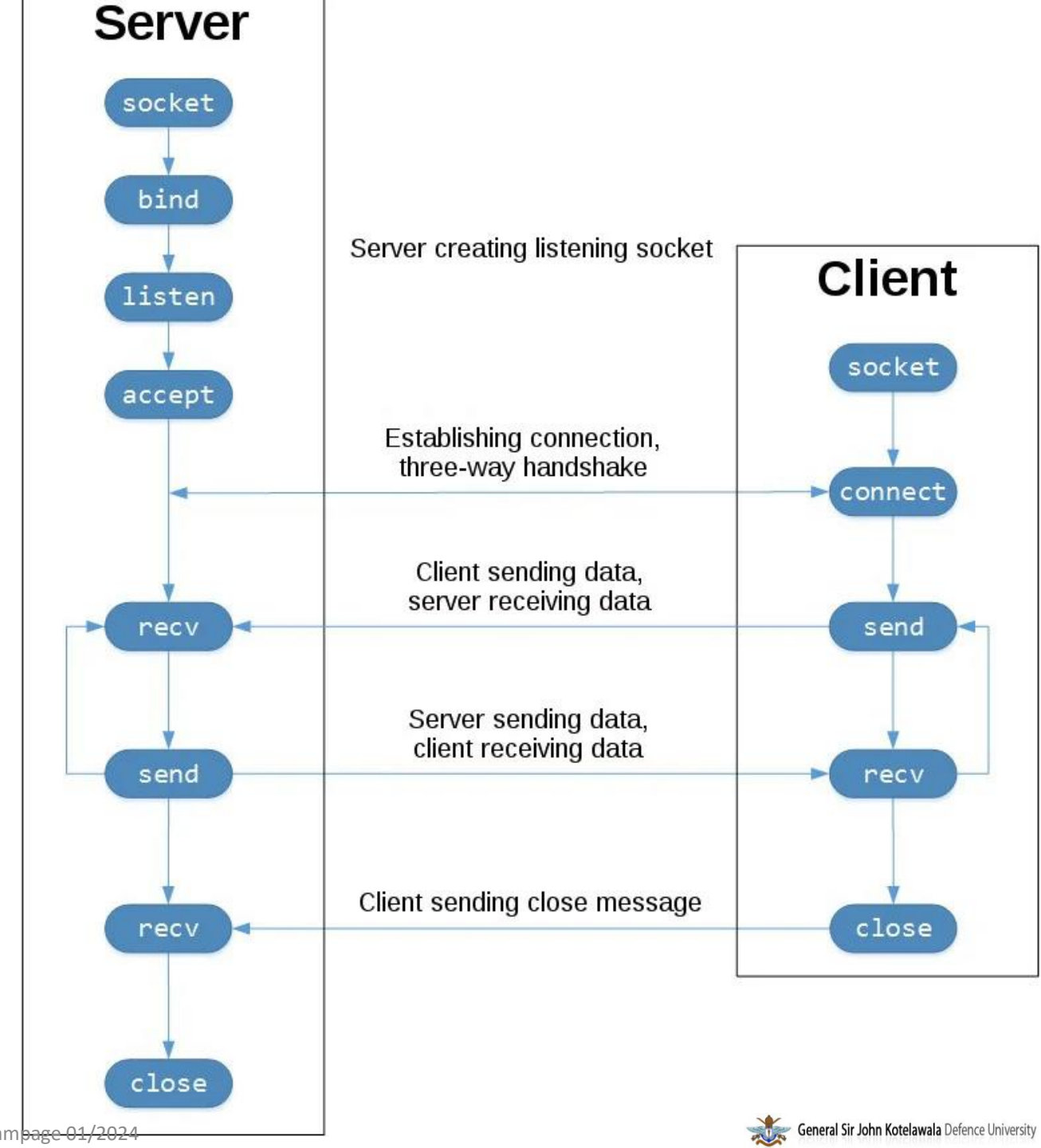
EXAMPLE

From Last Sub-Module



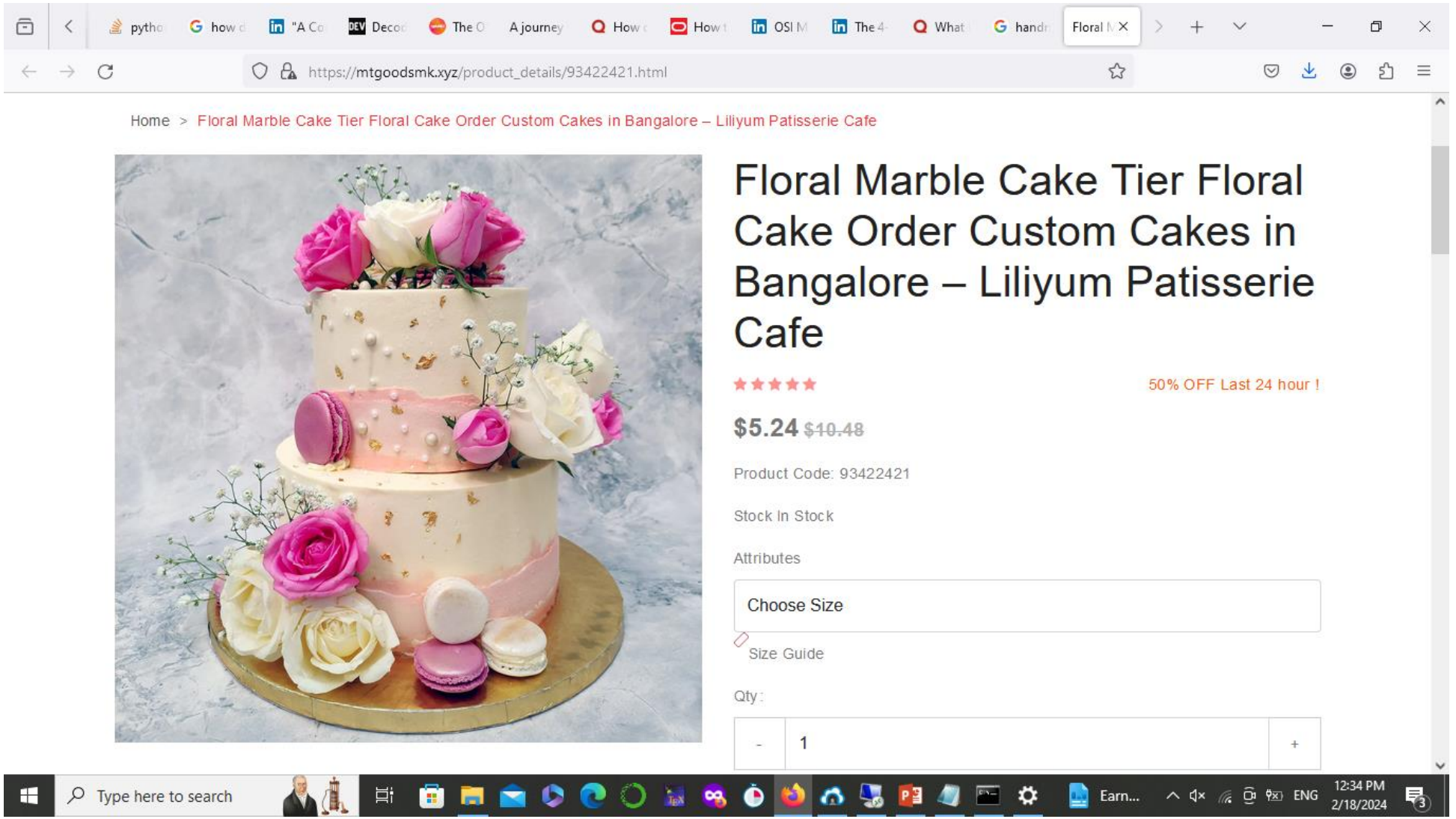
EXAMPLE

From Last Sub-Module



An Illustrative Example

Application Layer



Layer 7 - Application Layer - Your KDU batch mate has seen a handmade wedding cake on your wife's web page and she orders for it and informs you too. (HTTP).

Presentation Layer

- You can’t ship it whole because the DHL would charge you a huge amount for that size and batch mate will not be pleased.
- So, you take it apart, separate the layers of the cake and put then in to 4 different boxes with baseplates.
- The Top Layer box, Middle layer box, a Bottom Layer box, and a all Decoration Flowers box.
- You include an instruction book with photographs on how to reassemble, put plastic frame in each box to avoid damage to components. (HTML)

This is what it looks like disassembled in HTML:

Line wrap

```
<!doctype html>
<html lang="en">
<head>
    <!--Shell Type: thd-shell -->
<!--Shell Version: v31.14.1-->

    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
    <meta http-equiv="content-language" content="en" />
    <meta name="theme-color" content="#F96302" />
    <link href="https://assets.thdstatic.com" rel="preconnect"/>
    <link href="https://images.thdstatic.com" rel="preconnect"/>
    <link href="https://s.go-mpulse.net" rel="preconnect"/>
    <link href="https://c.go-mpulse.net" rel="preconnect"/>
    <link href="https://localization.thdws.com" rel="preconnect"/>
    <link href="https://bam.nr-data.net" rel="preconnect"/>
    <link href="https://lpcdn.lpsnmedia.net" rel="preconnect"/>
    <link href="https://homedepot.demdex.net" rel="preconnect"/>
    <link href="https://assets.thdstatic.com" rel="dns-prefetch"/>
    <link href="https://images.thdstatic.com" rel="dns-prefetch"/>
    <link href="https://s.go-mpulse.net" rel="dns-prefetch"/>
    <link href="https://c.go-mpulse.net" rel="dns-prefetch"/>
    <link href="https://localization.thdws.com" rel="dns-prefetch"/>
    <link href="https://bam.nr-data.net" rel="dns-prefetch"/>
    <link href="https://lpcdn.lpsnmedia.net" rel="dns-prefetch"/>
    <link href="https://homedepot.demdex.net" rel="dns-prefetch"/>
    <link href="https://clickstream-producer.hd-personalization-prod.gcp.homedepot.com" rel="dns-prefetch"/>
    <link href="https://collector-px04kmklxl.px-cloud.net" rel="dns-prefetch"/>
    <link href="https://dpm.demdex.net" rel="dns-prefetch"/>
    <link href="https://homedepot.tt.omtrdc.net" rel="dns-prefetch"/>
    <link href="https://siteintercept.qualtrics.com" rel="dns-prefetch"/>
    <link href="https://homedepot-app.quantummetric.com" rel="dns-prefetch"/>
    <link href="https://mr.homedepot.com" rel="dns-prefetch"/>
    <link href="https://www.google-analytics.com" rel="dns-prefetch"/>
    <link href="https://www.res-x.com" rel="dns-prefetch"/>
    <link href="https://va.v.liveperson.net/" rel="dns-prefetch"/>

    <style>
        .alert,.alert-inline,.bttn,.bttn--cta,.bttn--cta .bttn__content,.bttn--filter,.bttn--filter .bttn__con
    </style>

    <script>
    !function(){if('PerformanceLongTaskTiming' in window){var g=window.__tti={e:[]};
    g.o=new PerformanceObserver(function(l){g.e=g.e.concat(l.getEntries())});
    g.o.observe({entryTypes:['longtask']})}}();
    </script>


<script type="text/javascript">
    try {
        if(document.cookie.indexOf('THD_NR=')===-1){
          var expDate=new Date();
          var newValue=(+75 > 0 && Math.random() < +75) ? 1 : 0;
          expDate.setDate(expDate.getDate()+1);
          document.cookie='THD_NR='+newValue+';expires='+expDate+';domain=homedepot.com;path=/';
          document.cookie='THD_NR='+newValue+';expires='+expDate+';domain=homedepotdev.com;path=/';
        }
        var parts=('; '+document.cookie).split('; THD_NR=');
        var value;
        if(parts.length===2){
```

Session Layer

- You and She both have to be on line when you send him the cake, so you need to set up a session like a phone call.
- Her number is (IP Address): 10.100.111.101 (like a phone number).

```
Ethernet adapter Ethernet 2:

   Connection-specific DNS Suffix  . : ecpivab.edu
   Link-local IPv6 Address . . . . . : fe80::81a3:2d14:f1e6:aa4b%4
   IPv4 Address. . . . . . . . . . . : 10.100.111.101
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.100.111.1
```

- Your number is: 23.212.145.75

```
P:\users\peklatt.ECPIVAB>nslookup www.hom
Server:  dc03.ecpivab.edu
Address:  10.142.4.2

Non-authoritative answer:
Name:    e14801.x.akamaiedge.net
Address:  23.212.145.75
Aliases:  www.homedepot.com
          www.homedepot.com.edgekey.net
```

Session Layer

- So now we have the connection between the two PCs, but we also need to know what loading dock your friend will be coming to (from below, port 443) and
- which room in her house she would like all the layers of the cake stored until she gets them all and can start to build it (in this case her back porch is port 56,851).
- So we have now created a Socket, or Windows Socket, or Winsock of:

```
TCP    10.100.111.101:56851   23.212.145.75:443      ESTABLISHED
```

- and both sides are listening while all the packages are being delivered.
- This your (last) Session or WINSOCK.
- Everything up to this point is considered DATA

Transport Layer

- Transport Layer, TCP is responsible for assuring reliable delivery of the cake to your friend's porch.
- To do this, he first labelled the boxes (SEGMENTS) as 1 of 4, 2 of 4, 3 of 4, and 4 of 4, (Sequencing) and
- put a return post card on each box that
- your friend need to sign and date and send back once he receive each box (ACK or Acknowledgement of Receipt)

Network Layer

- The network layer is just responsible for getting the packages to the correct Zip Code (LAN - Network).
- The packages have to travel from your Zip Code (IP Network 23.0.0.0) to her (IP Network 10.100.111.0) using tractor trailers, trains, and planes. (Routers on the network).
- This layer puts a destination and return IP Network and places it on the SEGMENTS turning them into PACKETS (sometimes called IP Datagrams or just Datagrams).

Data Link Layer

- Once the delivery has arrived at her network (10.100.111.0), the Post Office will call her at his host number (0.0.0.101) and ask for his Street Address (MAC) and she will reply with the Physical Address. (a protocol called ARP).
- The Router will forward the package to the Switch (Post office puts the package in the box of the mailman who is responsible for delivering that address by Jeep).
- The mailman then takes the package and puts it in his Jeep).

```
Description . . . . . . . . . . . : Realtek USB GbE Family Controller
Physical Address. . . . . . . . . : 80-CE-62-78-1C-26
```

- The mailman will put her address above 80-CE-62–78–1C-26 on the destination label and post offices return street address 00–18–0A-CA-06–36 (from below)(the Header):

```
Internet Address      Physical Address      Type
10.100.111.1          00-18-0a-ca-06-36     dynamic
```

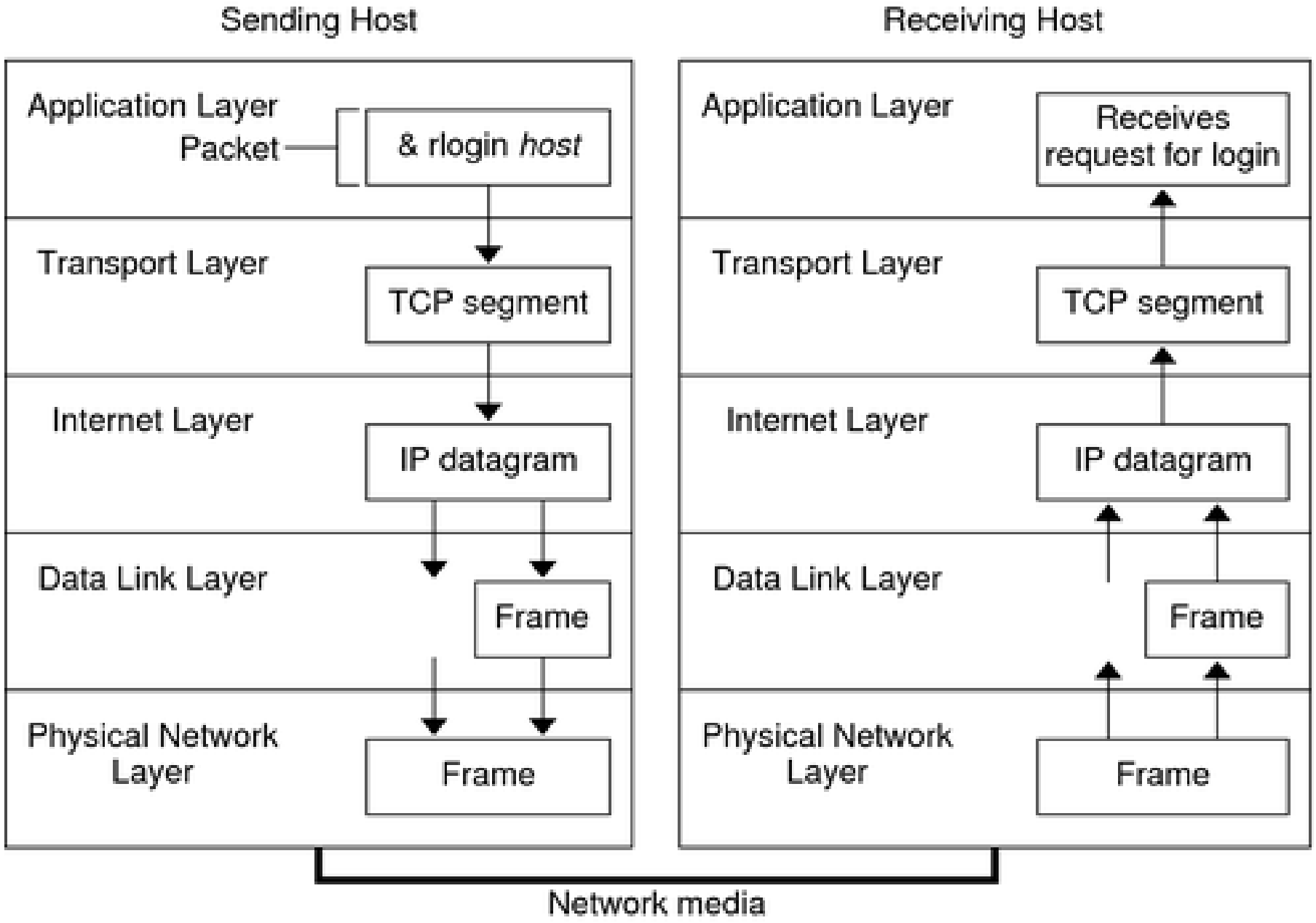
- The mailman will then put an error detection code on the package (a Packing slip saying what should be inside).
- (The Trailer or Checksum). This is now called a FRAME.

Physical Layer

- Physical Layer, This is then put into the jeep which goes out on the road with the FRAMEs but they are all converted to just ones and zeros, called BITS.
- When the package arrives at your friend's destination, she will check the packing slip and ensure that all the parts are accounted for and if they're all there (her check matches the Checksum),
- She RECEIVES it and passes it up stack removing each envelope(covers and baseplates) until she gets to the Application layer (cake layers and decoration flowers).

- If anything is missing (Her check does not match the checksum) she discards the package and
- awaits a replacement that will automatically be sent if the sender does not receive her TCP Acknowledgement within a few milliseconds.

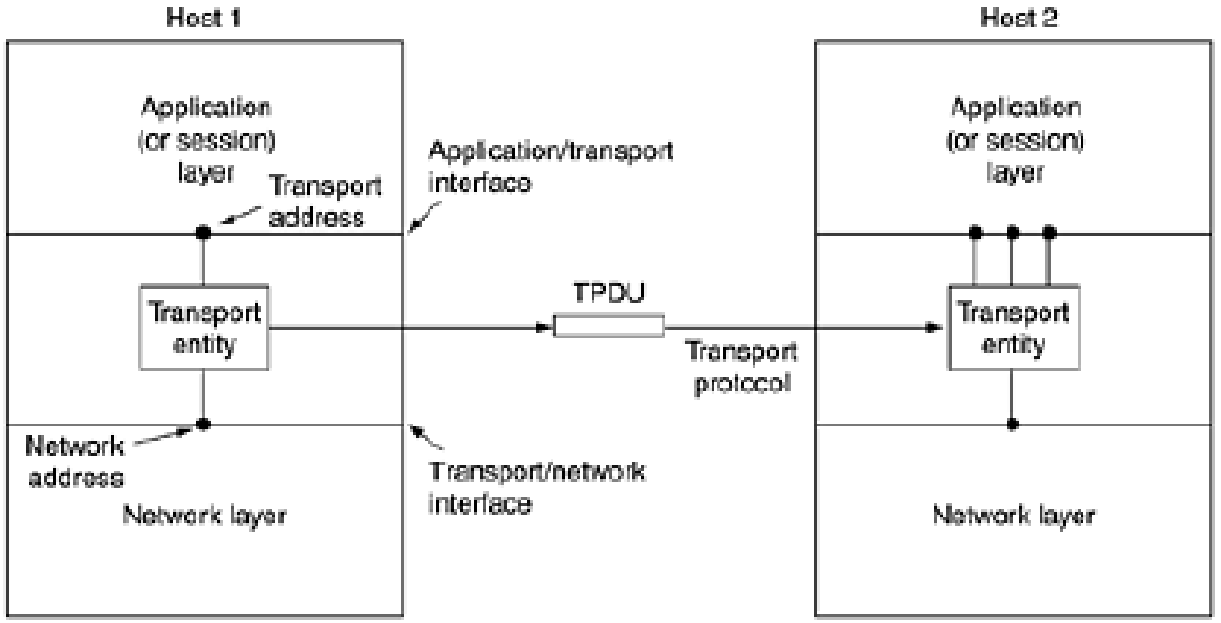
Review of Data Structures



Transport Layer

Primitives

- The ultimate goal of the transport layer is to provide efficient, reliable, and cost-effective service to its users, normally processes in the application layer.
- To achieve this goal, the transport layer makes use of the services provided by the network layer.
- The hardware and/or software within the transport layer that does the work is called the transport entity.
- The transport entity can be located in the operating system kernel, in a separate user process, in a library package bound into network applications, or conceivably on the network interface card.
- The (logical) relationship of the network, transport, and application layers is illustrated in following Figure.



Why Network Layer ?

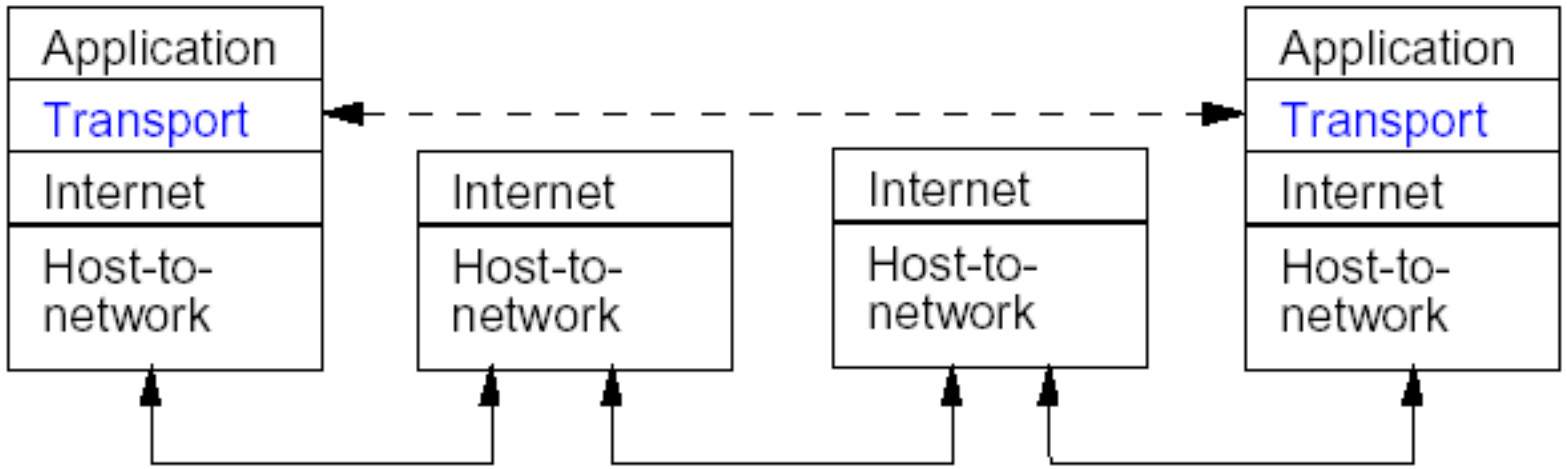
- The transport code runs entirely on the users‘ machines, but the network layer mostly runs on the routers, which are operated by the carrier (at least for a wide area network). What happens if the network layer offers inadequate service? Suppose that it frequently loses packets? What happens if routers crash from time to time?.
- The existence of the transport layer makes it possible for the transport service to be more reliable than the underlying network service. Lost packets and mangled data can be detected and compensated for by the transport layer.
- The transport service primitives can be implemented as calls to library procedures in order to make them independent of the network service primitives.
- If all real networks were flawless and all had the same service primitives and were guaranteed never, ever to change, the transport layer might not be needed.
- In the real world it fulfills the key function of isolating the upper layers from the technology, design, and imperfections of the subnet. The bottom four layers can be seen as the transport service provider, whereas the upper layer(s) are the transport service user.

Introduction

- The transport layer is an end-to-end layer – this means that nodes within the subnet do not participate in transport layer protocols – only the end hosts.

- As with other layers, transport layer protocols send data as a sequence of packets (segments).

- The purpose of the transport layer—to provide a reliable service on top of an unreliable network.



Multiplexing

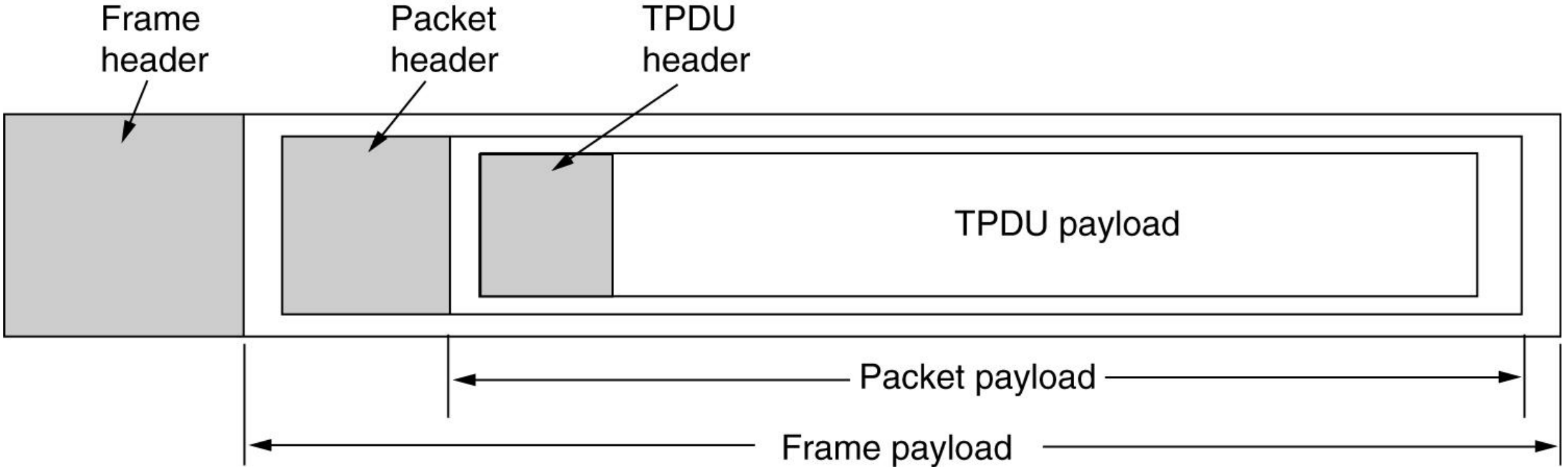
- The **network layer** provides communication between two hosts.
- The **transport layer** provides communication between two ***processes*** running on different hosts.
- A process is an instance of a program that is running on a host.
- There may be multiple processes communicating between two hosts – for example, there could be a FTP session and a Telnet session between the same two hosts.

Multiplexing

- The transport layer provides a way to multiplex / demultiplex communication between various processes.
- To provide multiplexing, the transport layer adds an address to each segment indicating the source and destination processes.
- Note these addresses need only be unique locally on a given host.
- In TCP/IP these transport layer addresses are called ***port-numbers***.
- The 5-tuple: (sending port, sending IP address, destination port, destination IP address, transport layer protocol) uniquely identifies a process-to-process connection in the Internet.

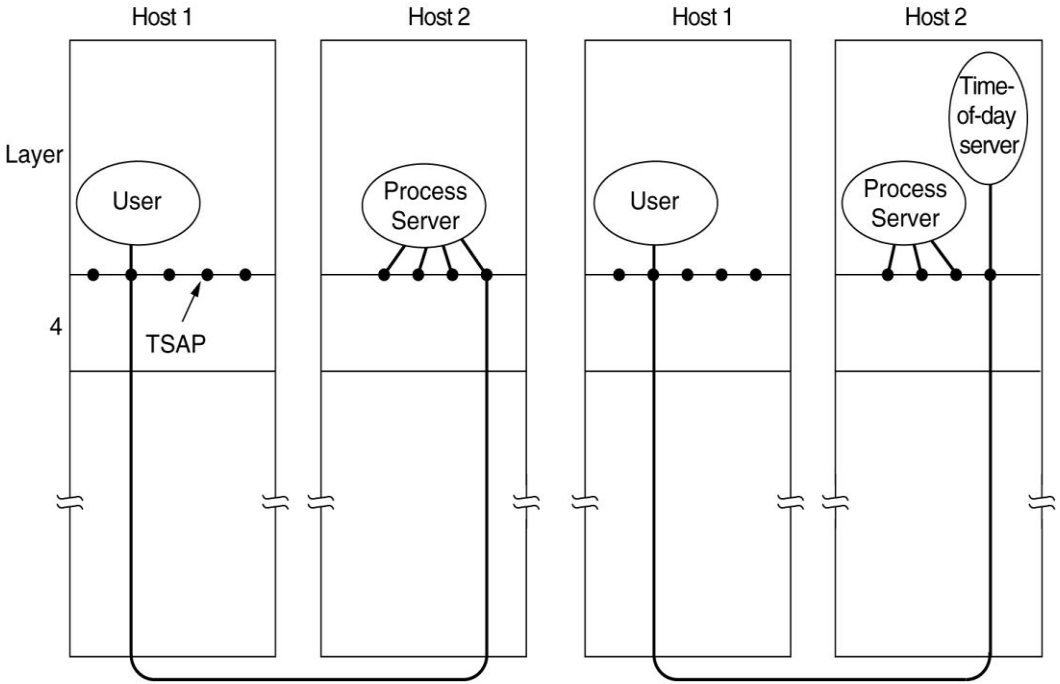
Transmission Control Protocol (UDP)

Transport Control Data Unit (TPDU)



- TPDU (Transport Protocol Data Unit) for messages sent from transport entity to transport entity.
- TPDUs (exchanged by the transport layer) are contained in packets (exchanged by the network layer).
- The packets are contained in frames (exchanged by the data link layer).
- When a frame arrives, the data link layer processes the frame header and passes the contents of the frame payload field up to the network entity.
- The network entity processes the packet header and passes the contents of the packet payload up to the transport entity.

Connection Establishment



- When an application (e.g., a user) process wishes to set up a connection to a remote application process, it must specify which one to connect to. (Connectionless transport has the same problem: To whom should each message be sent?) The method normally used is to define transport addresses to which processes can listen for connection requests.
- In the Internet, these end points are called port.
- Typically use the generic term TSAP, (Transport Service Access Point) for a port.
- The analogous end points in the network layer (i.e., network layer addresses) are then called NSAPs. IP addresses are examples of NSAPs.
- Figure above illustrates the relationship between the NSAP, TSAP and transport connection.
- Application processes, both clients and servers, can attach themselves to a TSAP to establish a connection to a remote TSAP. These connections run through NSAPs on each host, as shown.
- The purpose of having TSAPs is that in some networks, each computer has a single NSAP, so some way is needed to distinguish multiple transport end points that share that NSAP.

Transport Service Primitives

Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	This side wants to release the connection

- Consider an application with a server and a number of remote clients.
- To start with, the server executes a LISTEN primitive, typically by calling a library procedure that makes a system call to block the server until a client turns up.
- When a client wants to talk to the server, it executes a CONNECT primitive. The transport entity carries out this primitive by blocking the caller and sending a packet to the server.
- Encapsulated in the payload of this packet is a transport layer message for the server's transport entity.

Transport Service Primitives

Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	This side wants to release the connection

- The client's CONNECT call causes a CONNECTION REQUEST TPDU (Transport Protocol Data Unit) to be sent to the server.
- When it arrives, the transport entity checks to see that the server is blocked on a LISTEN (i.e., is interested in handling requests). It then unblocks the server and sends a CONNECTION ACCEPTED TPDU back to the client.
- When this TPDU arrives, the client is unblocked and the connection is established.
- Data can now be exchanged using the SEND and RECEIVE primitives.
- In the simplest form, either party can do a (blocking) RECEIVE to wait for the other party to do a SEND. When the TPDU arrives, the receiver is unblocked. It can then process the TPDU and send a reply.

Transport Service Primitives

Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	This side wants to release the connection

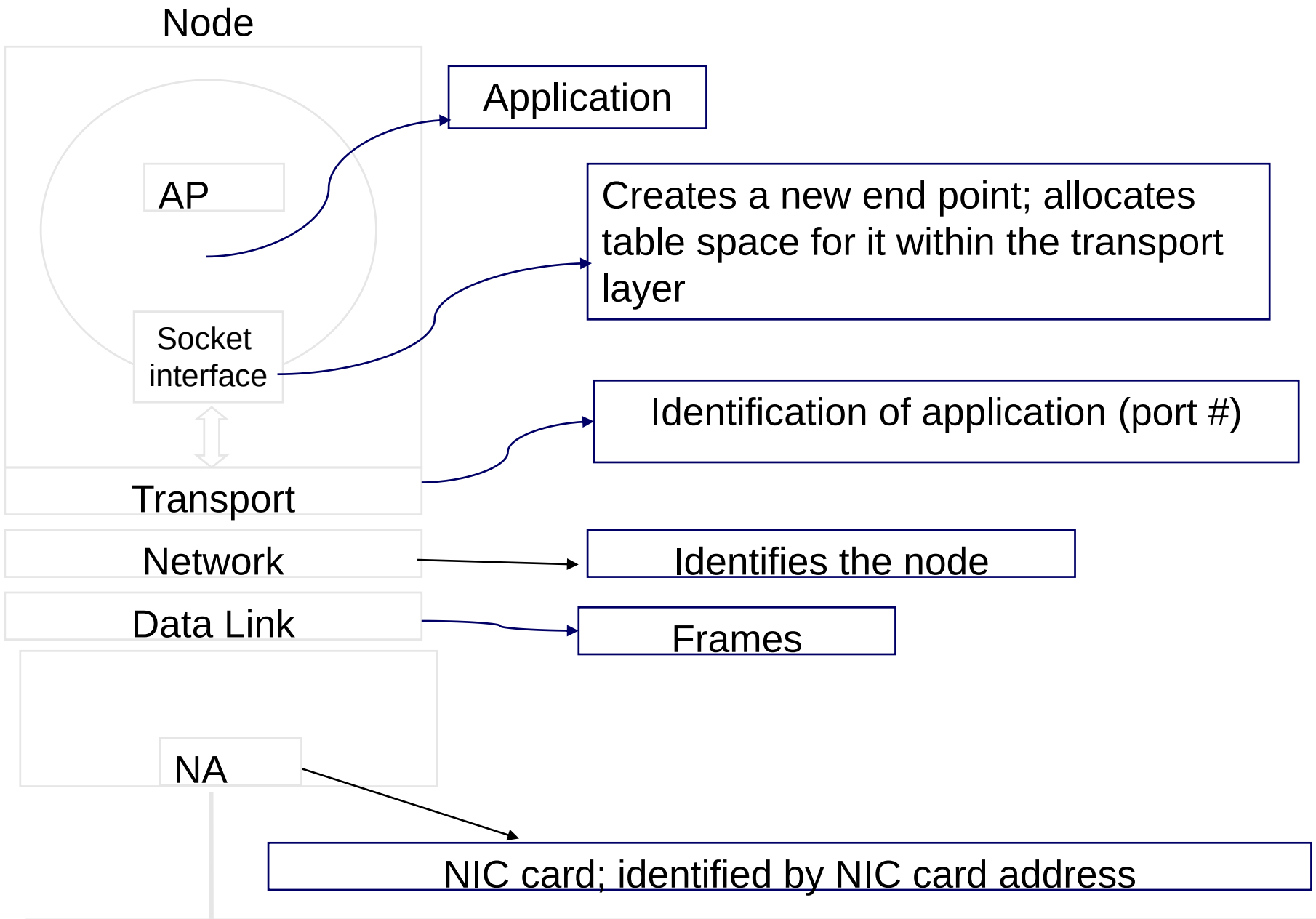
- Every data packet sent will also be acknowledged (eventually). The packets bearing control TPDUs are also acknowledged, implicitly or explicitly.
- These acknowledgements are managed by the transport entities, using the network layer protocol, and are not visible to the transport users.
- When a connection is no longer needed, it must be released to free up table space within the two transport entities. Disconnection has two variants: asymmetric and symmetric.
  - In the asymmetric variant, either transport user can issue a DISCONNECT primitive, which results in a DISCONNECT TPDU being sent to the remote transport entity. Upon arrival, the connection is released.
  - In the symmetric variant, each direction is closed separately, independently of the other one.
- When one side does a DISCONNECT, that means it has no more data to send but it is still willing to accept data from its partner. In this model, a connection is released when both sides have done a DISCONNECT.

Transport Service Primitives

- connum = LISTEN (local)
- connum = CONNECT(local, remote)
- status = SEND (connum, buffer, bytes)
- status = RECEIVE(connum, buffer,bytes)
- status = DISCONNECT(connum)

- The primitives for a simple transport service.

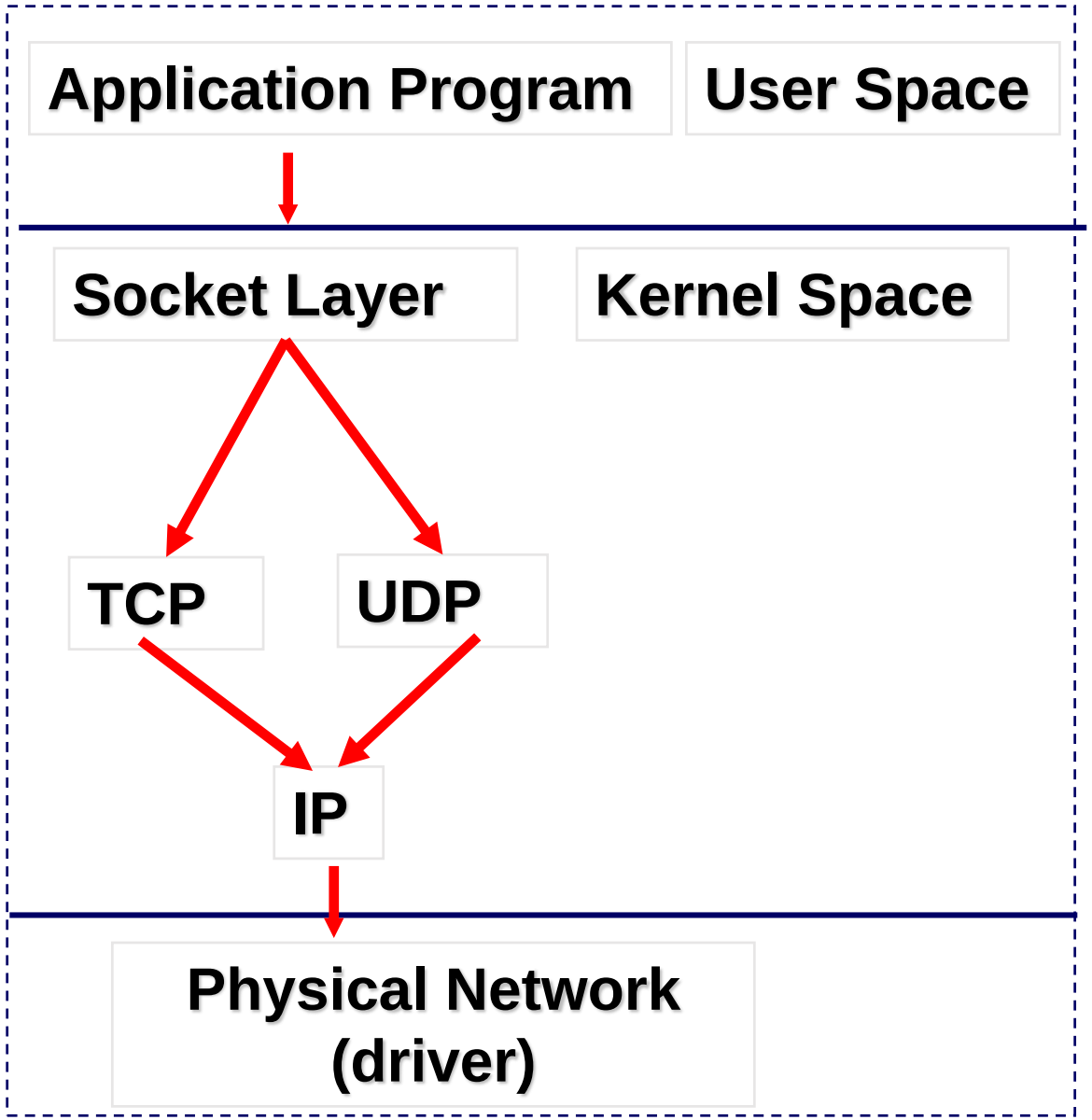
Implementation of Protocol Stack



Socket

- **Software interface designed to communicate between the user program and TCP/IP protocol stack**
- **Implemented by a set of library function calls**
- **Socket is a data structure inside the program**
- **Both client and server programs communicate via a pair of sockets**

Socket Framework



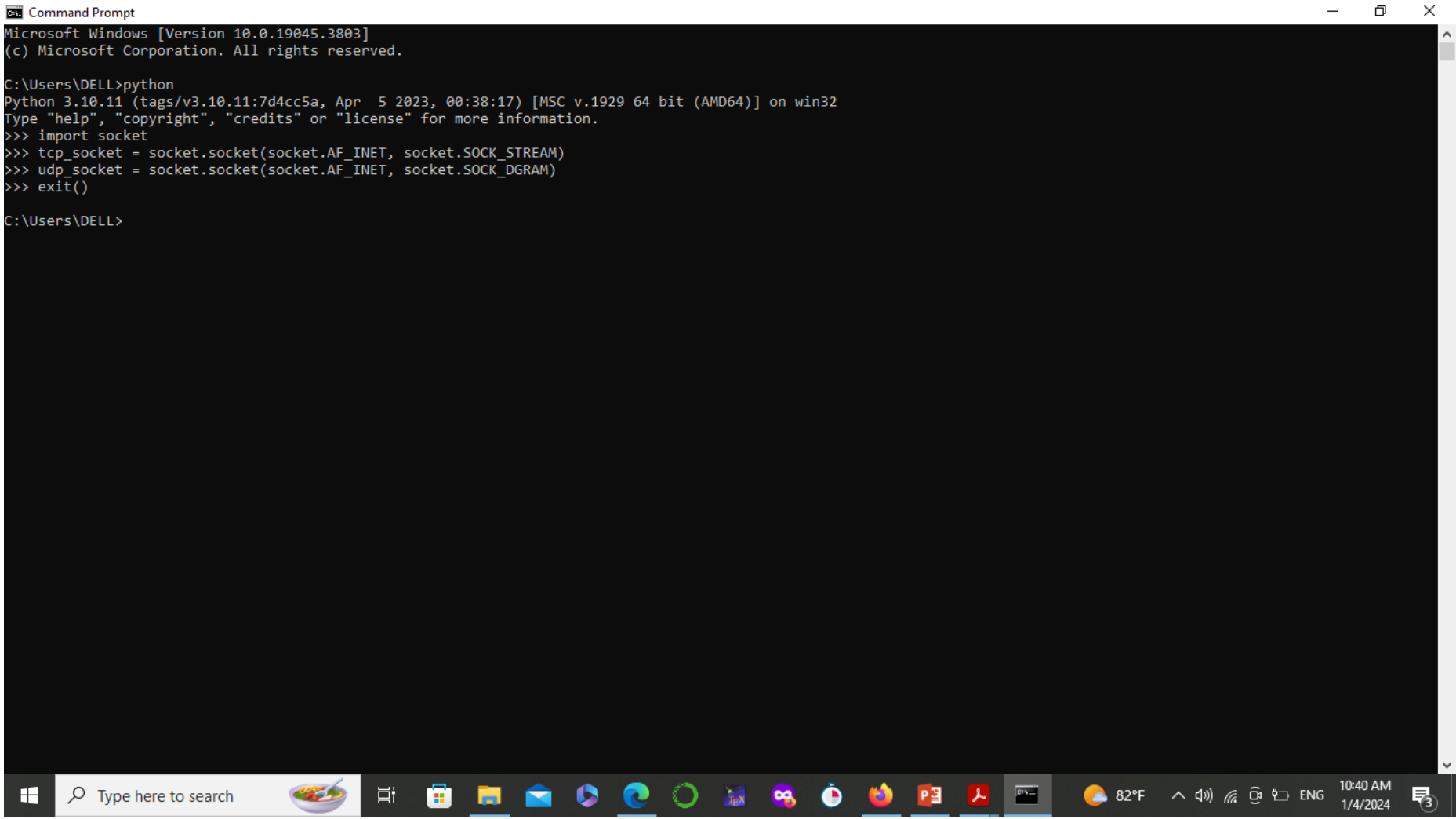
Socket Families

- There are several significant socket domain families:
  - Internet Domain Sockets (AF_INET)
    - implemented via IP addresses and port numbers
  - Unix Domain Sockets (AF_UNIX)
    - implemented via filenames (think “named pipe”)
  - Novell IPX (AF_IPX)
  - AppleTalk DDS (AF_APPLETALK)

Type of Sockets

- Stream (SOCK_STREAM) - Uses TCP protocol. Connection-oriented service
- Datagram (SOCK_DGRAM) - Uses UDP protocol. Connectionless service
- Raw (SOCK_RAW) - Used for testing

Practical 4 - Recap



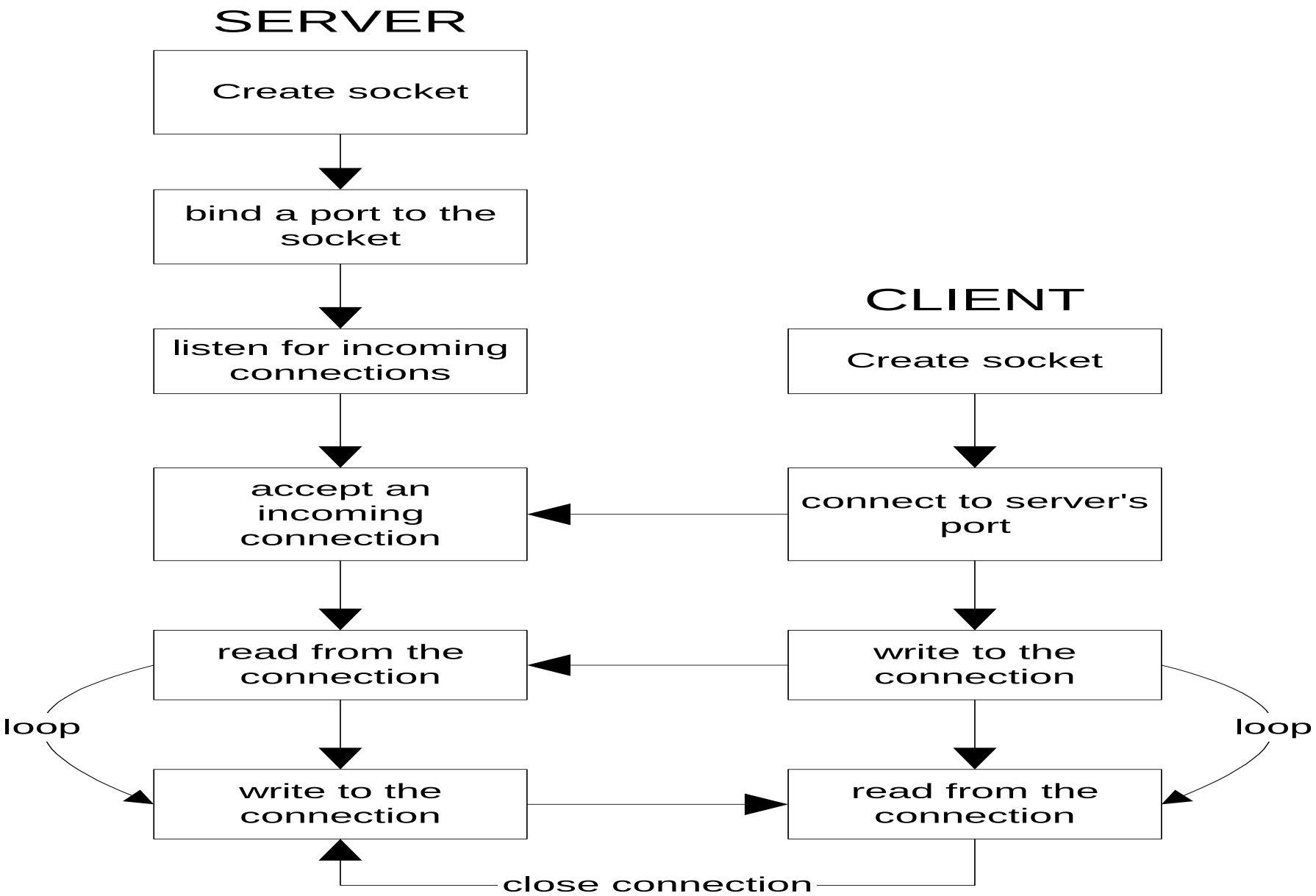
To create a TCP socket using IPv4, you would call: socket.socket() function as follows:
tcp_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

To create a UDP socket using IPv4, you would use the following code:
udp_socket = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)

To close a socket:

tcp_socket.close()

TCP Client Server

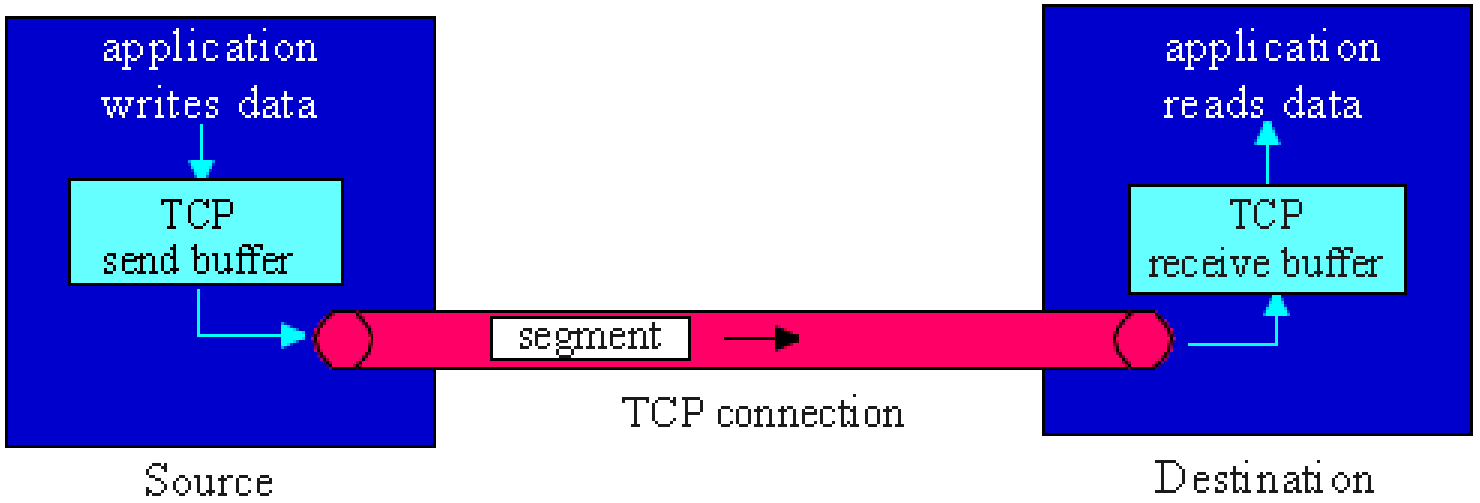


Transport Control Protocol

- connection oriented.
- full duplex.

- **TCP Services:**
  - Reliable transport
  - Flow control
  - Congestion control
  - UDP does not provide any of these services

Flow of TCP Segments



TCP Services

- TCP converts the unreliable, best effort service of IP into a reliable service, i.e., it ensures that each segment is delivered correctly, only once, and in order.

- Converting an unreliable connection into a reliable connection is basically the same problem we have considered at the data link layer, and essentially the same solution is used:

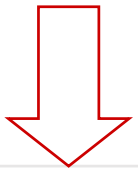
- TCP numbers each segment and uses an ARQ protocol to recover lost segments.

- →Some versions of TCP implement Go Back N and other versions implement Selective Repeat.

TCP Services

- However, there are a few important differences between the transport layer and the data link layer.

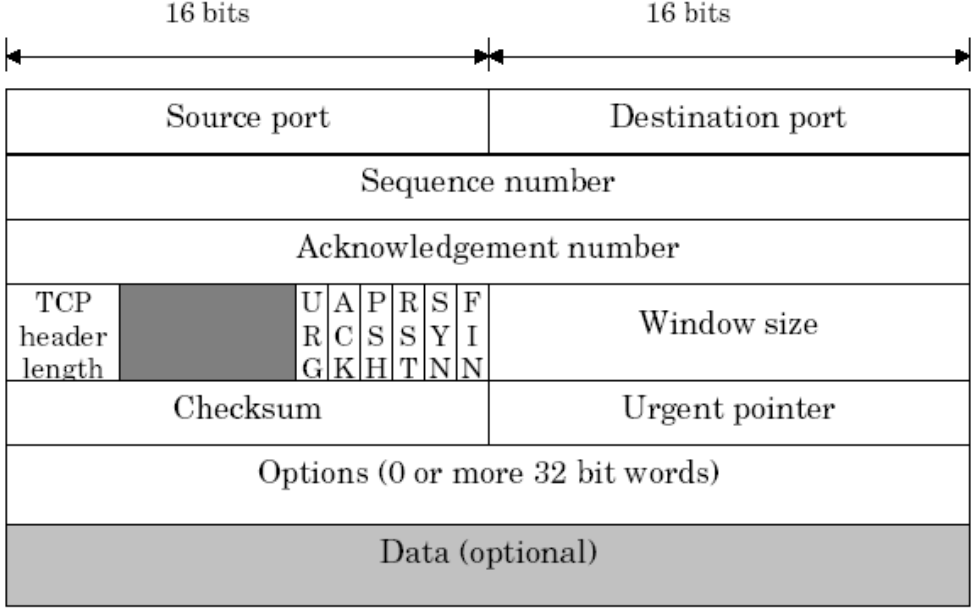
- → At the data link layer, we viewed an ARQ protocol as being operated between two nodes connected by a point-to-point link.



- → Packets can arrive out-of-order, and packets may also be stored in buffers within the network and then arrive at much later times.

- → The round-trip time will change with different connections and connection establishment is more complicated.

TCP Header



The Source port and Destination port fields identify the local end points of the connection. A port plus its host's IP address forms a 48-bit unique end point.

The source and destination end points together identify the connection.

The Sequence number and Acknowledgement number fields perform their usual functions. Note that the latter specifies the next byte expected, not the last byte correctly received.

Both are 32 bits long because every byte of data is numbered in a TCP stream.

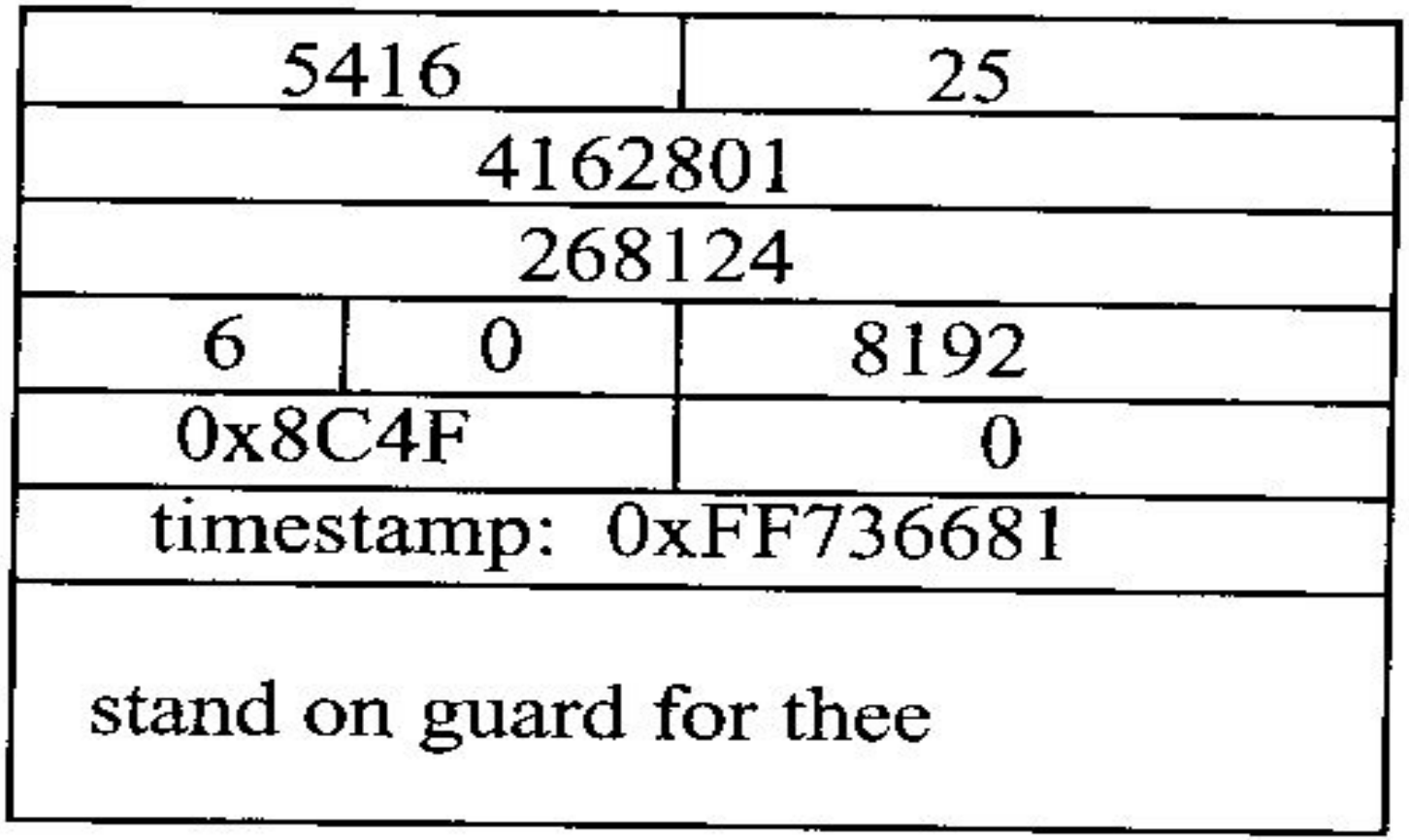
The TCP header length tells how many 32-bit words are contained in the TCP header. This information is needed because the Options field is of variable length, so the header is, too.

Technically, this field really indicates the start of the data within the segment, measured in 32-bit words, but that number is just the header length in words, so the effect is the same.

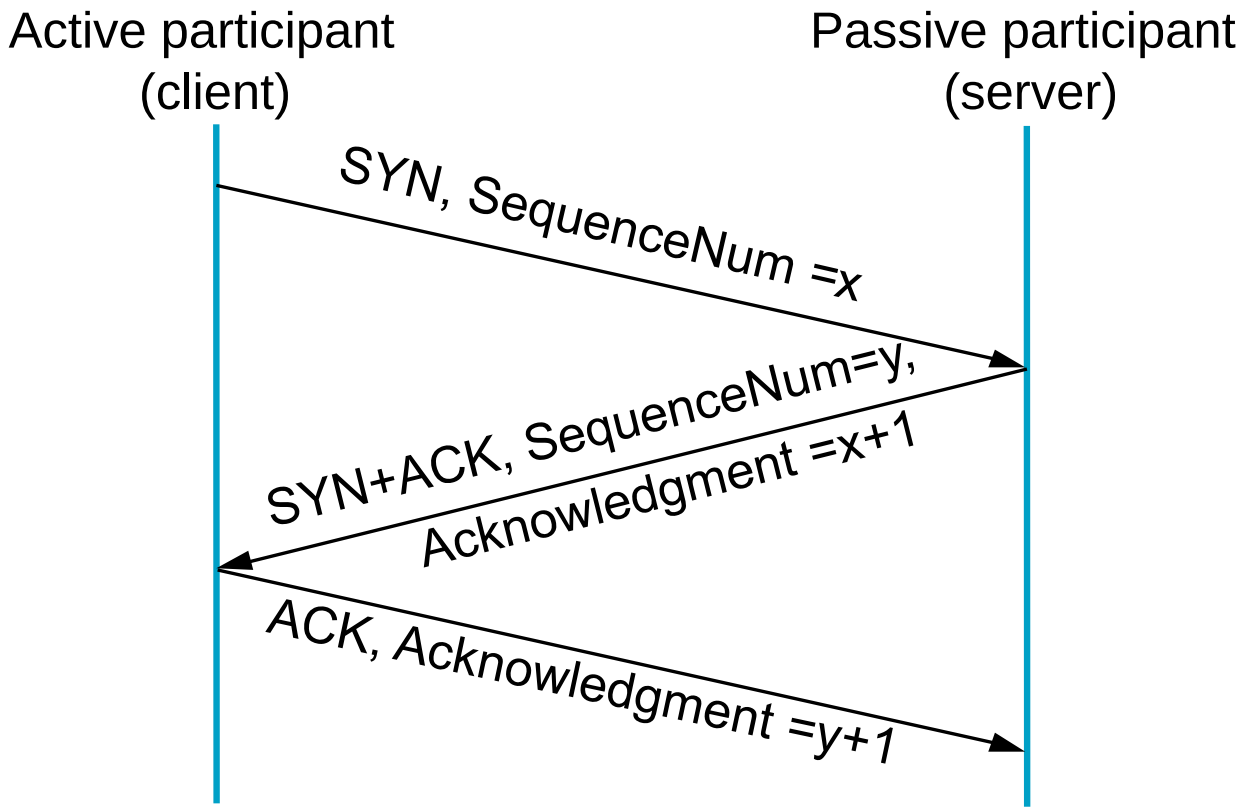
TCP Header

- The six 1-bit flags. *URG* is set to 1 if the *Urgent pointer* is in use. The *Urgent pointer* is used to indicate a byte offset from the current sequence number at which urgent data are to be found. This facility is in lieu of interrupt messages.
- The *ACK* bit is set to 1 to indicate that the *Acknowledgement number* is valid. If *ACK* is 0, the segment does not contain an acknowledgement so the *Acknowledgement number* field is ignored.
- The *PSH* bit indicates PUSHed data. The receiver is hereby kindly requested to deliver the data to the application upon arrival and not buffer it until a full buffer has been received (which it might otherwise do for efficiency).
- The *RST* bit is used to reset a connection that has become confused due to a host crash or some other reason. It is also used to reject an invalid segment or refuse an attempt to open a connection. In general, if you get a segment with the *RST* bit on, you have a problem on your hands.
- The *SYN* bit is used to establish connections. The connection request has *SYN* = 1 and *ACK* = 0 to indicate that the piggyback acknowledgement field is not in use. The connection reply does bear an acknowledgement, so it has *SYN* = 1 and *ACK* = 1. In essence the *SYN* bit is used to denote CONNECTION REQUEST and CONNECTION ACCEPTED, with the *ACK* bit used to distinguish between those two possibilities.
- The *FIN* bit is used to release a connection. It specifies that the sender has no more data to transmit. However, after closing a connection, the closing process may continue to receive data indefinitely. Both *SYN* and *FIN* segments have sequence numbers and are thus guaranteed to be processed in the correct order.

Sample TCP Packet



TCP Connection Establishment



Closing a TCP Connection

- client closes socket:

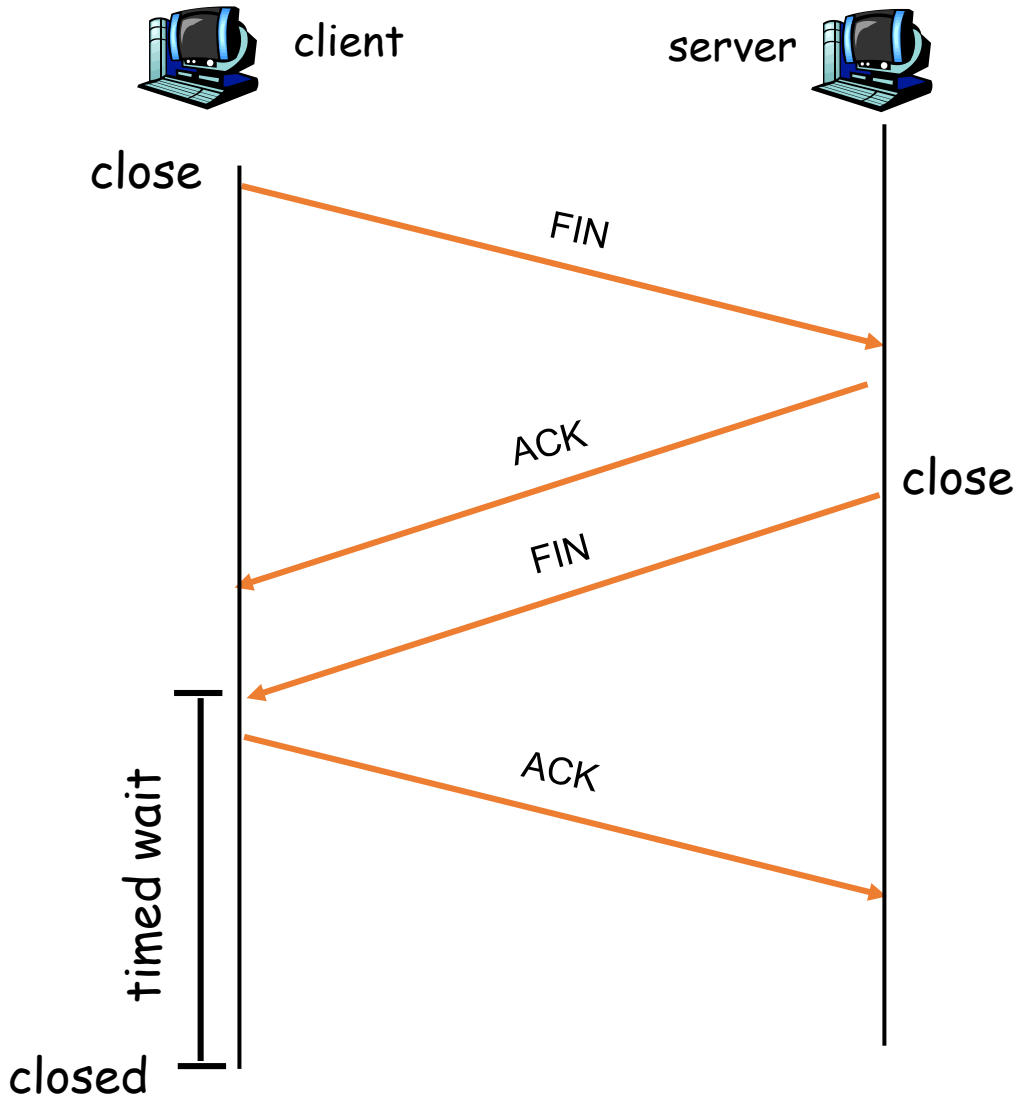
  `clientSocket.close();`

- Step 1: client end system sends TCP FIN

  control segment to server

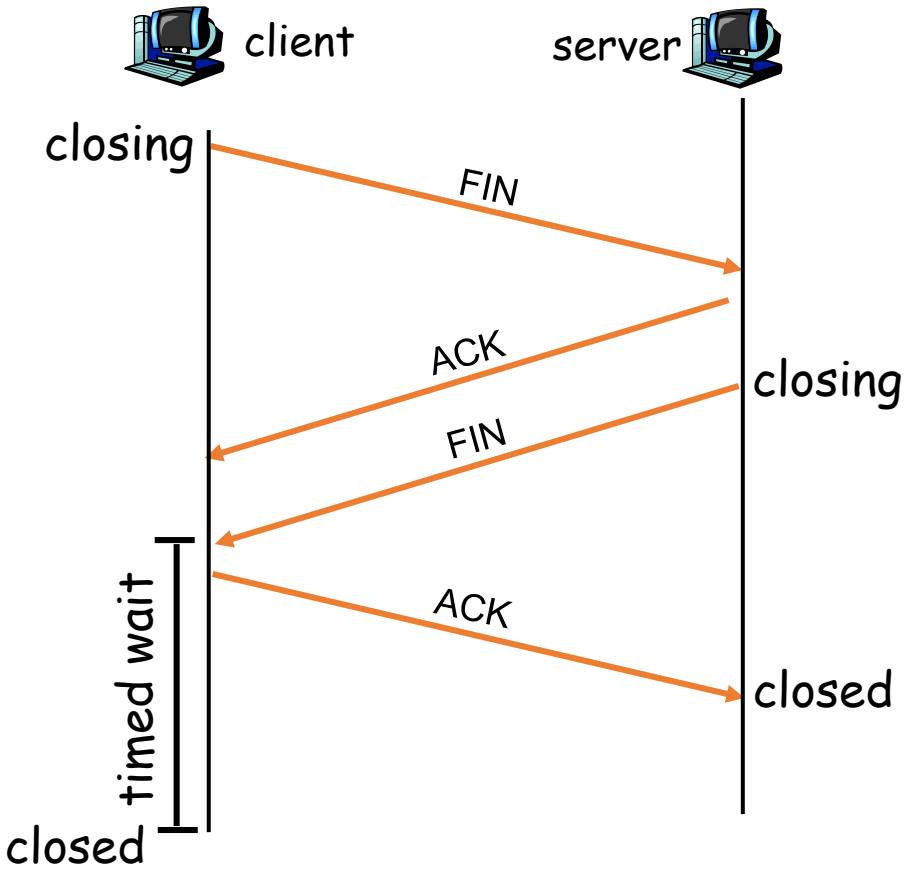
- Step 2: server receives FIN, replies with ACK.

  Closes connection, sends FIN.



Closing a TCP Connection

- Step 3: client receives FIN, replies with ACK.
  - Enters “timed wait” - will respond with ACK to received FINs
- Step 4: server, receives ACK. Connection closed.
- Note: with small modification, can handle simultaneous FINs.



Flow Control and Congestion Control

- In a network, it is often desirable to limit the rate at which a source can send traffic into the subnet.

- If this is not done and sources send at too high of a rate, then buffers within the network will fill-up resulting in long delays and eventually packets being dropped.

- Moreover as packets gets dropped, retransmission may occur, leading to even more traffic.

- When sources are not regulated this can lead to ***congestion collapse*** of the network, where very little traffic is delivered to the destination.

Flow Control and Congestion Control

- Two different factors can limit the rate at which a source sends data.

- → the inability of the destination to accept new data.
- Techniques that address this are referred to as ***flow control***.

- → the number of packets within the subnet.
- Techniques that address this are referred to as ***congestion control***.

Flow Control and Congestion Control

- Flow control and congestion control can be addressed at the transport layer, but may also be addressed at other layers.

- For example, some DLL protocols perform flow control on each link. And some congestion control approaches are done at the network layer.

- Both flow control and congestion control are part of TCP.

Approaches to Congestion Control

- Congestion control may be addressed at both the network level and the transport layer.

- At the network layer possible approaches include

- Packet dropping → when a buffer becomes full a router can drop waiting packets - if not coupled with some other technique, this can lead to greater congestion through retransmissions.

- Packet scheduling → certain scheduling policies may help in avoiding congestion - in particular scheduling can help to isolate users that are transmitting at a high rate.

Approaches to Congestion Control

- Dynamic routing → when a link becomes congested, change the routing to avoid this link - this only helps up to a point (eventually all links become congested) and can lead to instabilities

- Admission control/Traffic policing - Only allow connections in if the network can handle them and make sure that admitted sessions do not send at too high of a rate - only useful for connection-oriented networks.

Approaches to Congestion Control

- An approach that can be used at either the network or transport layers is

- Rate control → this refers to techniques where the source rate is explicitly controlled based on feedback from either the network and/or the receiver.

- For example, routers in the network may send a source a "choke packet" upon becoming congested. When receiving such a packet, the source should lower it rate.

Approaches to Congestion Control

- These approaches can be classified as either "congestion avoidance" approaches, if they try to prevent congestion from ever occurring, or as "congestion recovery" approaches, if they wait until congestion occurs and then react to it. In general, “better to prevent than to recover."

- Different networks have used various combinations of all these approaches.

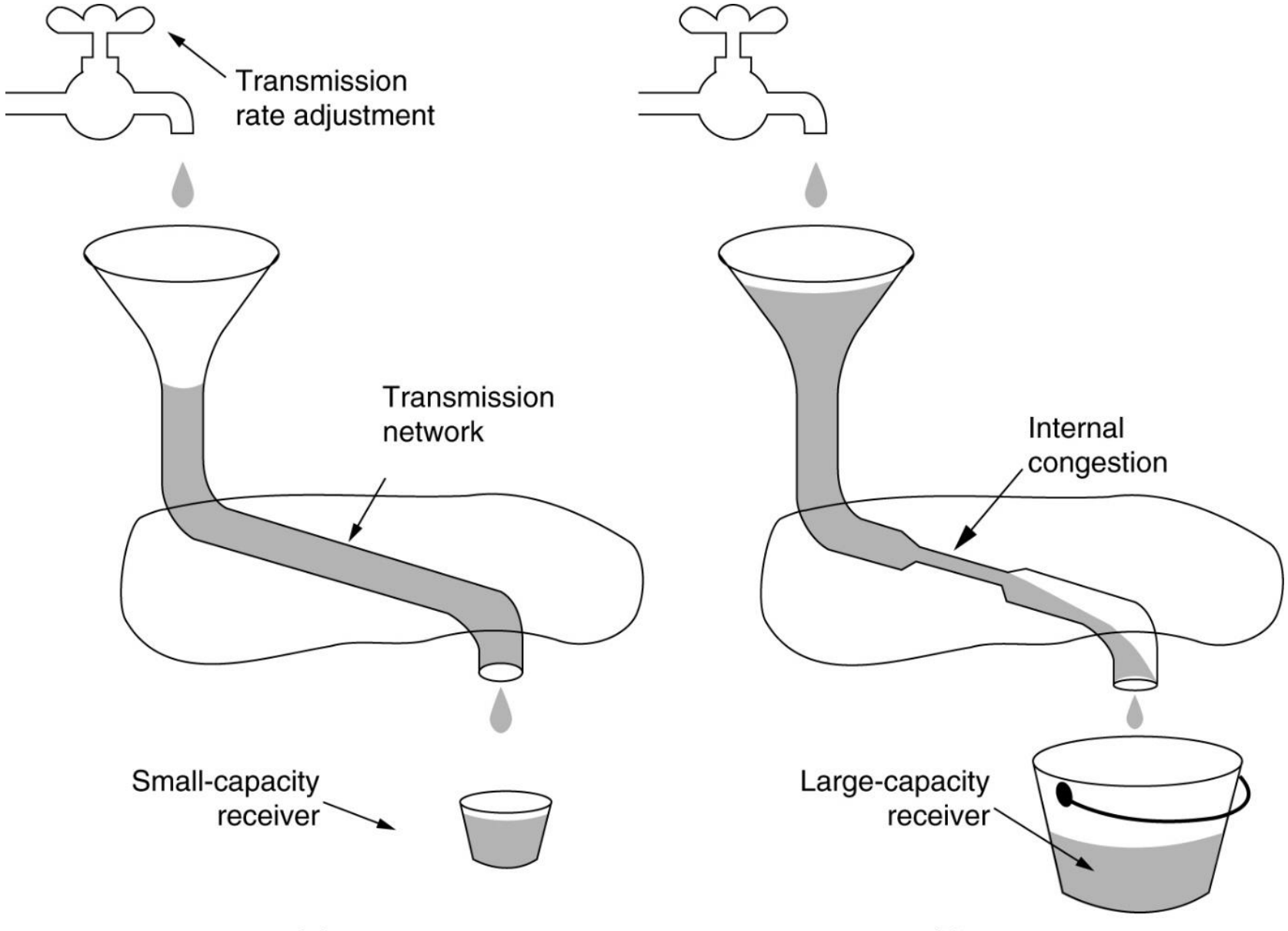
- Traditionally, rate control at the transport layer has been used in the Internet, but new approaches are beginning to be used that incorporate some of the network layer techniques discussed above.

Congestion Control in TCP

- TCP implements end-to-end congestion control. TCP detects congestion via the ACK's from the sliding-window ARQ algorithm used for providing reliable service.
- When the source times out before receiving an ACK, the most likely reason is because a link became congested. TCP uses this as an indication of congestion. In this case, TCP will slow down the transmission rate.

- TCP controls the transmission rate of a source by varying the window size used in the sliding window protocol.

Flow Control vs Congestion Control



User Datagram Protocol (UDP)

User Datagram Protocol

- UDP (User Datagram Protocol) → unreliable, connectionless; No TCP's flow control;
- applications where prompt delivery more important than accurate delivery (speech, video, …)
- (*Metaphor: postal service*)

User Datagram Protocol (UDP)

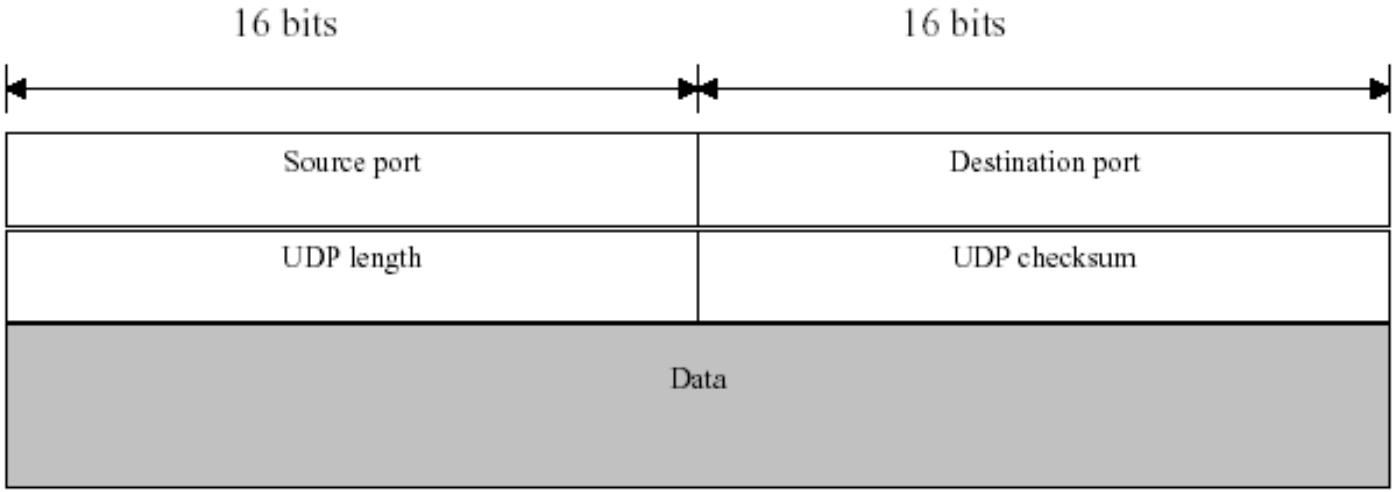
- Datagram protocol → it does not have to establish a connection to another machine before sending data.

- Takes the data an application provides,
- Packs it into a UDP packet
- Hands it to the IP layer.
- The packet is then put on the wire, and that is where it ends.

- There is no way to guarantee that the packet will reach its destination.

User Datagram Protocol

- Basically all UDP provides is a multiplexing capability on top of IP.
- The length field gives the length of the entire packet including the header.
- The checksum is calculated on the entire packet (and also on part of the IP header).
- Calculating the checksum is optional, and if an error is detected the packet may be discarded or may be passed on to the application with an error flag.



References

1. Nevio Benvenuto and Michele Zorzi, (2011). Principles of Communications Networks and Systems, John Wiley.

2. Thomas Robertazzi, (2011). Basics of Computer Networking (Springer Briefs in Electrical and Computer Engineering), Springer.

3. Andrew S. Tanenbaum and David J. Wetherall. 2010. Computer Networks (5th. ed.). Prentice Hall Press, USA.


©udayadamnage

Submodule Outline

Network (IP) and Transport Layers (TCP/UDP)

Routing, IP addressing: IPv4 and IPv6, IP sub-networking,

Internet Protocol,

TCP: Transmission Control Protocol,

UDP: User Datagram Protocol,

segmentation and reassembling, Service point addressing,

Flow control and congestion control,

error control schemes.

Overview of Mobile IP and Mobility management

Exercise One