

Network Protocols (Ch 15 of S&G)

Why we need them?

- Enable sharing the hardware network links
- Overcome sources of unreliability in the network
 - lost packets
 - temporary failure of an intervening routing node
 - mangled packets
 - reflections on the media, soft errors in communication hardware buffers, etc.
 - out of order delivery
 - packets of the same message routed via different intervening nodes leading to different latencies

What is a minimal protocol

- Bridge application's notion of “message” to the network's notion of a packet
 - application layer
 - hands over application program's message to the transport layer
 - e.g. rtp_send and rtp_rcv of Project 5

– transport layer

- e.g. RTP layer in Project 5
- at sending end
 - takes a message from the application layer and breaks it into packets commensurate with the network characteristics
 - attaches headers to the packets that contain information for use at the destination
 - handles retransmissions if necessary for overcoming network errors
- at receiving end
 - use the header info to assemble a message destined for an application program at this node
 - keeps track of packets of message(s) being assembled
 - negotiate with the sender (using ACKS/NACKS) to complete the message assembly
 - hand over assembled message to the application layer

– network layer

- implements the network driver to deal with the physical characteristics of the network
 - e.g.
 - » CSMA/CD for Ethernet
 - » token re-generation for token ring
- routing packets on the available network links
- filtering packets on the network and snarfing those intended for this node

Protocol Layering

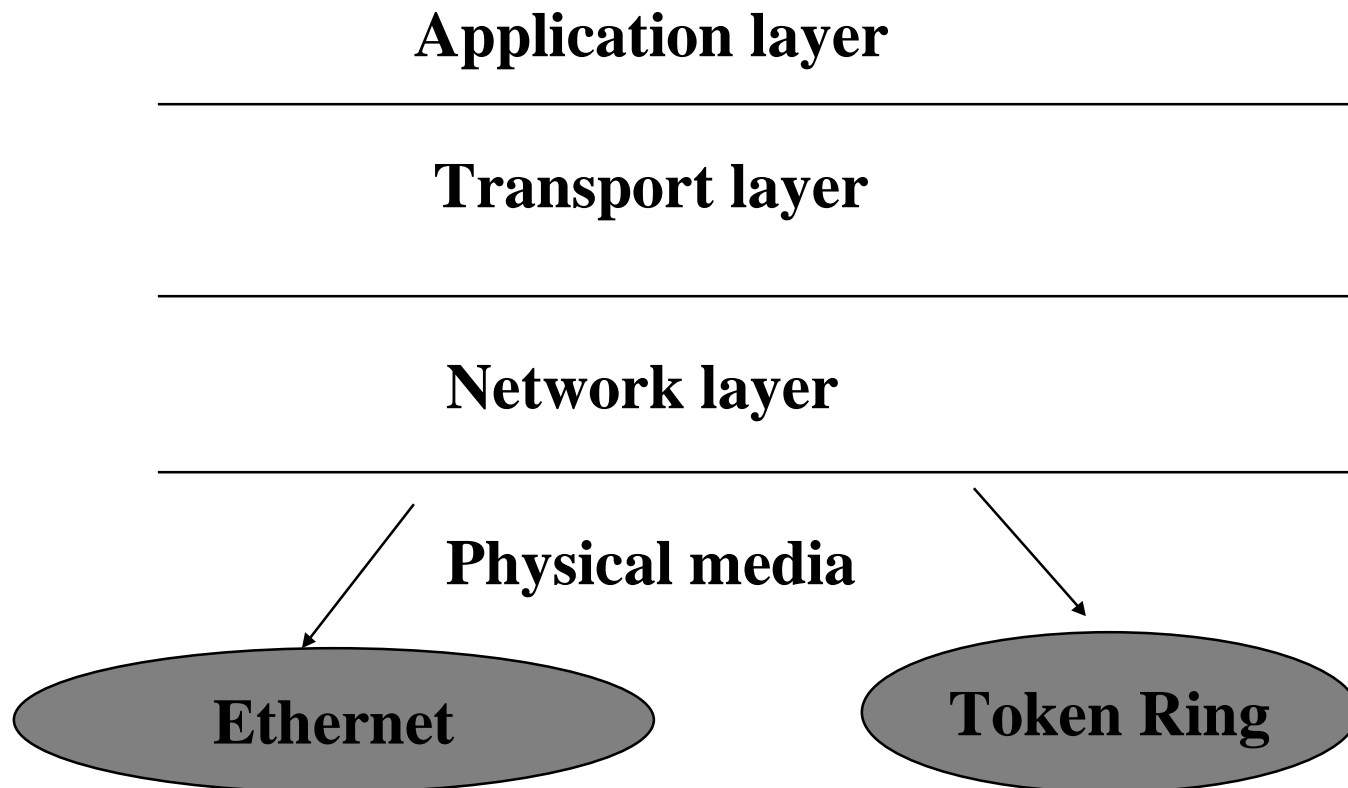
- Do we need these three layers?
- Do we need any more layers?
- What does the layering mean?
- How rigid is the layering?
- Does layering imply inefficiency?

Layering Advantages

- layering allows functionally partitioning the responsibilities (similar to having procedures for modularity in writing programs)
- allows easily integrating (plug and play) new modules at a particular layer without any changes to the other layers
- rigidity is only at the level of the interfaces between the layers, not in the implementation of these interfaces
- by specifying the interfaces judiciously inefficiencies can be avoided

Why not combine transport and network?

- Consider multiple physical media



- separation of transport and network is good
 - allows the networking layer to decide the best path to take from source to destination
 - addition/removal of physical media does not affect the transport layer code
- are these three layers sufficient?
 - consider a node that wants to talk with other nodes that may be on local LAN as well as other nodes in the “outside” world
- now we can generalize the layered protocol model...

ISO Model

7	Application	interact with user: e.g. mail, telnet, ftp
6	Presentation	char conv., echoing, format diffs: endian-ness
5	Session	Process to process comm.: e.g. Unix sockets
4	Transport	packetizing, seq-num, retrans.: e.g. TCP, UDP
3	Network	routing, routing tables: e.g. IP
2	Data Link	interface to physical media, error recovery: e.g. retrans on collision in Ethernet
1	Physical	Electrical and mechanical characteristics of physical media: e.g. Ethernet

Practical Aspects of Layering

- OSI model only a historic guide
 - arpanet days
- With the evolution of Internet
 - some layers got collapsed
- These days...
 - layers 7-5: ftp, telnet, smtp, ..
 - layer 4: TCP, UDP
 - layer 3 (with aspects of 2): IP
 - layer 2 and 1: Ethernet bridges/repeaters

RPC in Unix and Protocols

- User of RPC at the application level
 - e.g. rlogin, rsh, rexec,...
- RPC package is at the session layer
 - transport level
 - transparent to users of RPC
 - may use Shared Memory, UDP or TCP (this has to be specified at the time of creation of the socket)
 - network layer
 - IP
 - data link and physical layer
 - LAN/WAN