Computer Networking Overview

Definition: Computer Network

A collection of computer systems or devices capable of exchanging data among elements of the collection.

Definition: Internet

A collection of computer networks under different adminstrative control and underlying technology capable of exchanging data among elements of the collection.

Physical organization of traditional computer networks

Hosts

The *end users* of the network.

Switching elements

Dedicated computing systems commonly known as *switches* and *routers*
These are responsible for routing of data flowing through the network
In the internet they also manage congestion by *throwing packets away!*

Transmission elements (called Channels or Links)

The physical media that interconnect hosts and switching elements

Wires	- low frequency electrical signaling (802.3 wired Ethernets)
Space	- Wireless radio frequency electromagnetic waves (802.11, WiFi), (802.16 WiMax)
Optical	- Super high frequency visible electomagnetic waves carried through optical fiber. 10^14 Hz

Media characteristics that impact network design

Band width (H) : The range of sine wave frequencies the medium supports without severe attenuation

Max theoretical bit rate = H log2(1 + (signal_power / noise_power))

Bit rate: The inverse of the time it takes to send 1 bit of information. On a 10 Mbps link it takes only 0.1 microseconds to send one bit.

Wireless – low bit rate: up to 100 Mbps
Wired – medium bit rate: up to 1 Gbps
Optical – high bit rate: up to 1 Tbps

There are a LOT more frequecies between 3x10^14 and 4x10^14 than between 3x10^9 and 4 *10^9

Error probability: The probability that a bit will be received in a state that is the inverse of the state in which it was sent (0 -> 1) or (1 -> 0)

Wireless – high error rate (so bad that FEC codes are typically used)
Wired – medium-low error rate (only error detection codes used)
Optical – very low error rate (only error detection codes used)

Latency: The time it takes a bit to travel from one network node to another (at approximately the speed of light)

Physical latency lies in the range *0.7c* to *1.0c* (*c = speed of light = 300Km / msec*)

In store and forward networks such as the Internet most latency is due to queueing delays in switching / routing.

Latency IS significant in networks using geosynchronous satellites flying at 36,000 km.

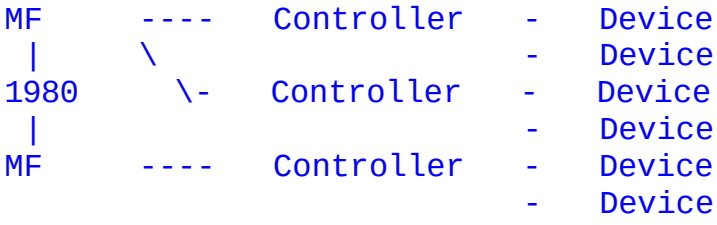
Starlink satellites fly at 550 km.

Historical development of networks and network architectures

Two major approaches (1960's and 70's)

Hierarchical

Mainframe (MF) hierarchies developed by IBM



Peer to Peer networks built around DEC minis

One common set of problems was based upon the fact that the early network protocols were built into the *application* instead of the *OS kernel.*

Lack of standardization in protocols inhibits interoperability
Application specific protocols requires links dedicated to specific apps.
Embedded networking code in application software

Resulting applications were:

Expensive to develop
Hard to maintain
Excessively large and complex
Non portable from one type of net (serial line) to another (LAN).

The solution

The solutions evolved in two phases

Phase I

Define *standard protocol (s).*
Divorce network code from apps, move it to the OS kernel and create *well defined nework API*

```
Applications
--- API ---
Network access code (in OS kernel)
```

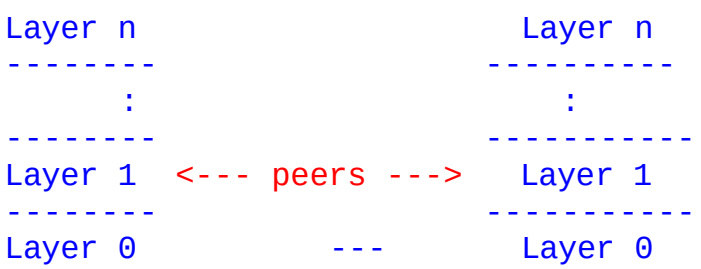
The remaining problem

Network access code (now in the O/S kernel) was

Large, complex, and difficult to maintain
Not well suited for use with *heterogeneous components* (LAN's and long haul links).

Phase II of the solution (borrowed from OS developers of late 1960's)

Organize network software as a *hierarchy of separate layers*



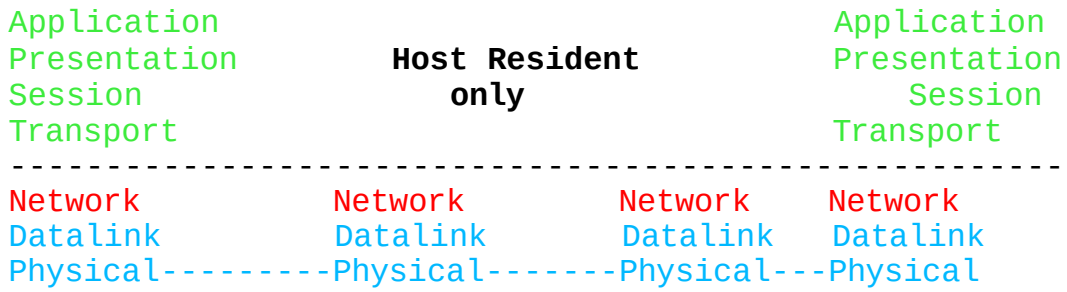
Principles of layered design

Each layer only knows about the layer above and below it.
Well defined layer functions and layer interface(s)
Minimized flow of information across the interfaces
Not too many and not too few layers.

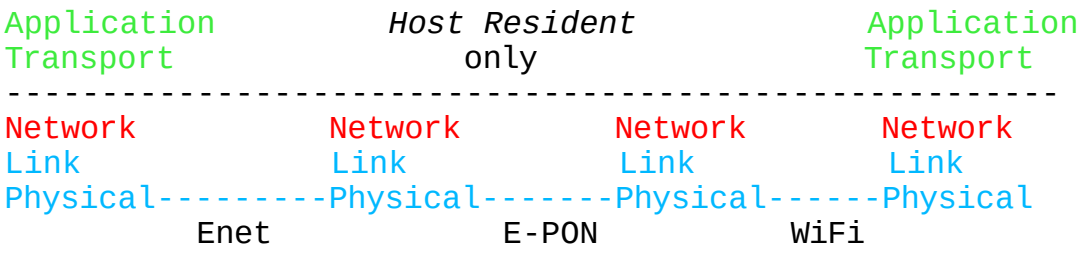
Network protocols are the rules governing the exchange of info between *peer (not adjacent)* layers

The ISO/OSI network architecture

The ISO (International Standards Organization) developed a reference model for OSI (Open Systems Interconnection). This model was somewhat pirated from IBM's Systems Network Architecture (SNA). A collection of real protocols were developed during the 1980's and for a time threatened to supplant TCP/IP.



Modern internet architecture



The network layer is the ONLY layer in which the same protocol must be implemented in every node ==> Keep it simple.

Headers and peer protocols:

As data travels down the stack each layer passes the data to be transmitted (to the next lower layer.

Next lower layer may add a header

As data travels back up the stack. Each layer removes the header created by its peer at the other end.

Layer N *never sees* Layer i headers for i < n.

Layer N treats Layer j headers as "user data" for j > n.

Elements of network Architectures

Service characteristics (applicable to services at *all* layers)

Connection orientation

Connection oriented *(the telephone model)*
- Requires a connection protocol before data exchange
- May provide
  - Message stream (preserves message boundaries)
  - Byte stream (does not guarantee to preserve message boundaries)
- Adv: Easier to provide QoS guarantees and to bill for service

Connection less *(the mail model)*
- Each packet transmitted as a standalone entity
- Also known as *datagram service*
- Adv: Simple

Reliability

Unreliable
- Loses, damages, duplicates, and/or reorders packets

Reliable
- Doesn't do any of the above.

Different layers *within a single network archtecture* may provide different service characteristics

HDLC	(link layer)	- Reliable connection oriented packet stream
IP	(network layer)	- Unreliable connectionless
TCP	(transport layer)	- Reliable connnection oriented byte stream
SCTP	(transport layer)	- Reliable connnection oriented packet stream
UDP	(transport layer)	- Uneliable connnection oriented packet stream
ATM	(cell transport)	- Unreliable connection oriented message (cell) stream

Protocols:

Sets of rules that defines the *syntax*, *semantics*, and *sequencing* of messages transmitted in a network.

Syntax - the locations and lengths and format of specific fields within a message or message header. (e.g.: The address is specified as a two byte binary number at offset 8 within the header).

Semantics - A value of 0x83 in the packet type field specifies that this is a standalone acknowledgment.

Sequencing - You will not send more than 7 packets before receiving an acknowledgment. You will not send packet class "A" while in state 19.

These rules govern the interchange of data and state information between *peer layers* in *different* nodes of the net.

Significant layer characteristics and functionality

Physical:

Modulation and Coding Scheme (MCS): How to physically encode 0's and 1's, how long each signaling state and error correct code.
Dedicated point-to-point or shared medium channel
Duplex or simplex: Whether or not data can flow both ways simultaneously
Framing: Signal start and end of a packet

Medium Access Control (MAC) :

Determining which station can (attempt to) transmit on a share channel
CSMA-CD Original shared medium ethernet
CSMA-CA WiFI
Reservation based WiMax, DOCSIS (cable TV systems)
Polling

Logical Link Control (LLC) :

Addressing: 48 bit unique address
Address resolution (ARP) mapping network (IP) address of dest to LLC address
Error detection: 32 bit cyclic redundancy check

Network:

Addressing: 32 bit IPV4 or 128 bit IPV6
Routing: Determine outgoing interface and next hop IP address.
Congestion control

Transport:

Breaking logical messages into packets of constrained size
Error control if required by the application
Flow control (preventing a fast producer of data from sending faster than a slow consumer can consume it.)
Routing of packets to proper application (incoming messages for different apps carry the *same network address* but *different transport addresses)*.

Application:

E-Mail (*smtp*)
Web (*http*)
File transfer (*ftp, scp*)
Remote session (*ssh, rlogin*)

An ideal operational perspective

View each layer as consisting of two threads

The *up* thread is in charge of moving packets up the stack
The *down* thread moves them down

Two message queues exist *between* each layer

The *up* queue holds packets moving up the stack
The *down* queue holds packets moving down

Both threads act as consumer/producer processes

The *up* thread consumes from the *up* queue below
It performs necessary processing on the packet
Header removal
Error checking
It then produces onto the *up* queue above

The down thread works analogously

The layered structure of the implementation may refine that of the protocol

The linux implementation breaks the *transport* and *link* layers into identifiable subcomponents

socket	A thin layer with a single implementation instance
protocol_family	A functional layer with one instance per protocol family (e..g PF_INET, PF_ATMPVC, PF_X25, etc.) It contains the generic components of the transport layer
transport_protocol	For each protocol family there is one instance of the this layer for each transport protocol. It contains the esoteric components of the particular transport (e.g., UDP, TCP, etc.). Your project will be to build one of these.
network_layer	One instance of this for each protocol family. For PF_INET this component contains the implementation of IP.
generic device(dev)	There is a single instance of the generic device layer that service all protocol families. It contains the NIC independent aspects of the MAC layer protocol, output packet scheduling, and demultiplexing of input packets with deliver to the proper network layer.
device class	There is one instance of this layer for each device class (e.g., ethernet, ATM, token ring, etc.) It contains components that are common to the MAC protocol but independent of the requirements of any specific NIC.
device driver	There is one instance of this layer for each specific NIC or family of NICs (e.g. 3C59x, e100, e1000, sk98lin)

Key:

transport layer
network layer
link layer