CS 455: Introduction To Distributed Systems [Networking]

Shrideep Pallickara
Computer Science
Colorado State University

Frequently asked questions from the previous class survey

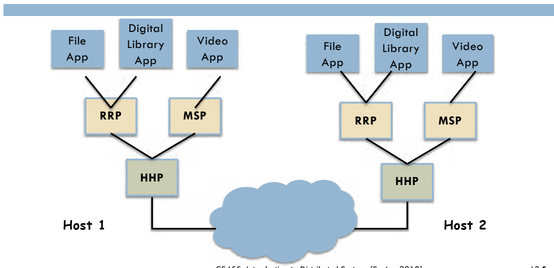
- 4B/5B: Does it solve problems of NRZI? Can it be sent over Manchester? Do they have error correction properties? Is just a chart or is there a way to calculate?
- What prevents packets (arriving concurrently) at a router from overwriting each other?
- What happens to data in flight if the connection is rejected?
- Who ends a socket connection? Server or Client
- Delay X Bandwidth product: Why do you have to fill the pipe before a bit arrives? Is there an upper limit on bandwidth?
- What is a flow? [Stream of bits]
- Switched networks: Is TCP a type of switched network?
- Layered protocols: What if I change the lower layers?

Topics covered in today's lecture

- Encapsulation
- OSI
- Internet Architecture
- IP routing

Encapsulation

Example of a protocol graph



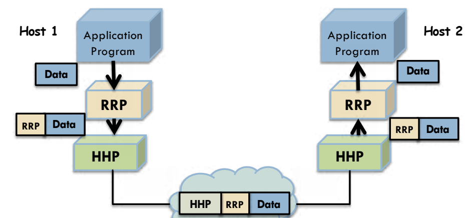
Encapsulation

- RRP receives a set of bytes to transmit from the application
  - E-mail, integers, images etc
- RRP is responsible for sending this data to its peer at the other end
  - Must communicate **control information** to its peer
  - Instruct how to **handle** the message

When asked to transmit info, lower level layers add information to the message

- Attach a **header** to the message
  - Small data structure
  - Few bytes to several dozen bytes
- Control info at the end of message: **trailer**
- Format is specific to the protocol
- Data being transmitted: body or **payload**
- Application data is said to be **encapsulated**

Encapsulating high-level messages inside low-level messages



Encapsulation: Some more info

- Low-level protocol does not **interpret** message given to it by high-level protocol
  - *Cannot extract* meaning
- Low-level protocol may apply simple **transformations** to the data it is given
  - Compress
  - Encrypt

Multiplexing is applicable up-and-down the protocol graph too

- RRP attaches header to every message that goes through it
  - Header include information to indentify the application
    - Called demultiplexing key or **demux key**
- At the destination host, RRP strips its header
  - Examines demux key
  - Demultiplexes message to correct application

Demux key is used at all levels of the protocol stack

- Some use an 8-bit field {**TCP** (6), **UDP** (17)}
  - Can support only 2^8 (256) high level protocols
  - Can also be 16/32-bits
- There could be a **single** demultiplexing field
  - Same demux key used at both ends
- There could be a **pair** of demultiplexing fields
  - Each side uses different key to identify high-level protocol

OSI Network Architecture

OSI network architecture

- Model is a product of the Open Systems Interconnection (OSI) project
  - At the International Organization for Standardization (ISO)
- Partitions network functionality into **7 layers**
- Physical Layer
  - Handles transmission of **raw bits**
  - Standardizes electrical, mechanical, and signaling interfaces
    - **0** bit should be received as **0** not **1**

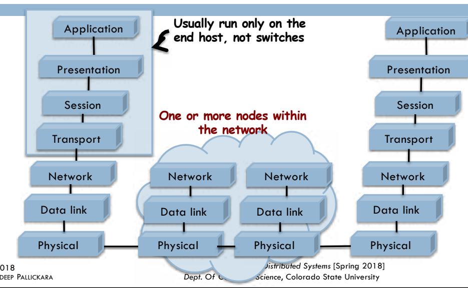
OSI network architecture: Data link Layer

- Collects stream of bits into a **frame**
  - Puts special *bit pattern* at the start/end of each frame
  - Frames, not raw bits, are delivered to host
- Compute **checksum** for frame
  - Check for correctness and request retransmission
- Network adaptors & device drivers implement this

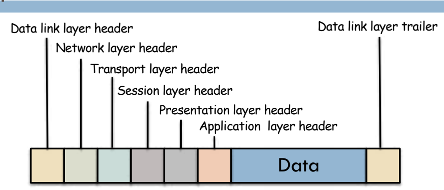
OSI network architecture

- Network layer
  - Handles routing among nodes in a **packet-switched** network
  - Unit of data exchanged is **packet** not frames
- Layers implemented on all network nodes?
  - Physical, data and network

OSI Architecture



How messages flowing through the OSI stack will appear on the network



OSI network architecture

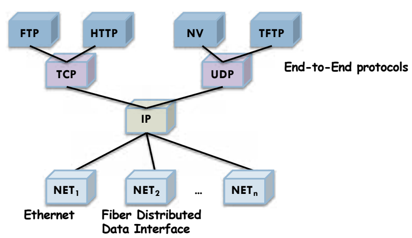
- Transport
  - Implements process-process **channel**
  - **Messages** {not packet or frame}
- Presentation
  - **Format** of data exchanged between peers
- Session
  - **Namespace** to tie different transport-streams that are part of the application

INTERNET ARCHITECTURE

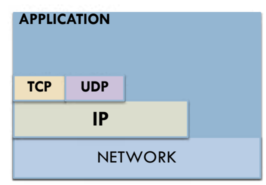
Internet architecture

- Evolved out of experiences with ARPANET
  - Funded by ARPA of the US DoD
- Around *before* the OSI architecture
- Unlike OSI, this is a **4-level** model

Internet protocol graph



Alternative view of the Internet Architecture



Internet architecture

- DOES NOT imply strict layering
  - Bypassing immediate lower layers is possible
- Layer has an **hour-glass** shape
  - Wide at top and bottom
  - Narrow in the middle
  - **IP** is the **focal point** of the architecture

Protocol implementation issues
Where are the processes?

- Process-per-protocol
- Process-per-message

Protocol implementation issues
Process-per-protocol model

- Each protocol implemented in *separate* process
- Process/protocol passes message to another process/protocol
- **Context-switch** required at each level of the protocol graph
  - Expensive!

Process-per-message model: Associate processes per message

- Treat each protocol as a **static** piece of code
- Protocol graph **traversed in sequence** of procedure calls
- When message arrives:
  - Dispatch process to move message up the protocol graph
  - At each level procedure implementing protocol is **invoked**
- Sending message?
  - Application process invokes appropriate procedures

Comparison

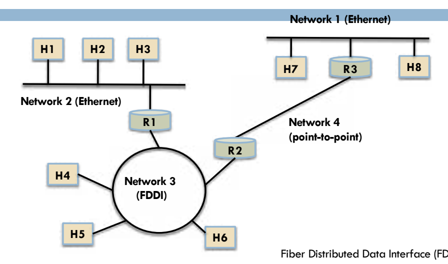
- Process-per-protocol
  - Context switch per level
- Process-per-message
  - Procedure call per level

INTERNETWORKING

Internetwork

- Arbitrary collection of **interconnected** networks
  - To provide some sort of host-host packet delivery service
- Network of networks
  - Made up of lots of smaller networks

A simple internetwork

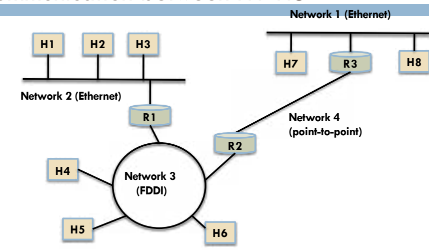


Fiber Distributed Data Interface (FDDI)

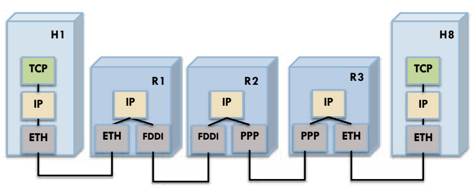
Internet Protocol (IP)

- Key tool to build scalable, **heterogeneous** networks
- Runs on all nodes (hosts and routers)
- Allows nodes and networks to *function as a single logical network*
- Possible to build an internetwork without IP
  - But IP is the only one that has faced scale issues

A simple internetwork: Communication between H1-H8



Example depicting how hosts (H1-H8) are logically connected



The IP service model

- Datagram model of **delivery**
  - Connectionless
  - Best effort
- **Addressing** scheme
  - Identifies all hosts in the internetwork

Datagram delivery

- Datagram is a type of packet
  - Sent in a **connectionless** fashion
- No need for any *advance* setup mechanisms
  - That tell network what do when packet arrives
- Every datagram contains enough information
  - To forward packet to correct destination

The network makes a best effort to send datagrams across

- Things that could go **wrong** with the packets
  - Lost
  - Corrupted
  - Misdelivered
  - Out of order and duplicates
- When things go wrong, the network does **nothing**
  - <u>No attempt to recover</u> from the failure

Keeping routers simple was one of the original design goals of IP

- Important to **run over anything**
- Putting extra functionality into routers to make up for network deficiencies?
  - Not a good idea
- Higher-level protocols/apps that run above IP need to be aware of failure modes

The IP Packet format consists of a header followed by bytes of data

- Represented as a succession of **32-bit** words
- Packet formats designed to align on 32-bit boundaries
  - Simplifies task of processing in software
- Transmission order
  - ***Top word*** transmitted first
  - ***Leftmost byte*** of each word transmitted first

The IPv4 packet header

0	4	8	16	19	31
Version	HLen	TOS	Length		
Ident			Flags	Offset	
TTL		Protocol	Checksum		
SourceAddr					
DestinationAddr					
Options (variable)				Pad (variable)	
Data					

IP Packet format [1/5]

- **Version**
  - Makes it easy to **redefine** packet format later on
- **HLen**
  - Specifies length of header **in 32-bit words**
  - When there are no options (most of the time)
    - Header is **5 words** or 20 bytes
- **TOS** (type of service)
  - Allow packets to be treated differently
    - Based on application needs

IP Packet format [2/5]

- **Length**
  - Length of the datagram **in bytes**
  - Maximum size of IP datagram is 2^{16} bytes
- Second word of IP packet
  - {Ident, Flags, Offset}
  - Information about **fragmentation**

IP Packet format [3/5]

- **TTL** (time to live)
  - Hop-count not timer (as originally intended)
- **Protocol** field
  - **Demultiplexing** key
    - Identifies higher-level protocol
    - TCP (6), UDP (17)
- **Checksum**
  - Consider IP header as a sequence of 16-bit words

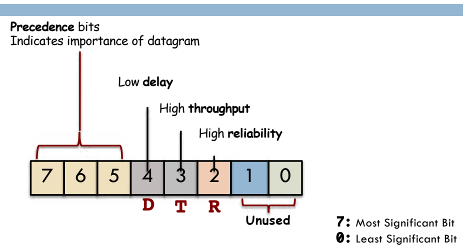
IP Packet format [4/5]

- **SourceAddr**
  - Decide if packet should be accepted
  - Also used for replies
- **DestinationAddr**
  - Full address of destination
  - Forwarding decisions are made at each router
- Presence or absence of **options**
  - Can be checked **based on size of HLen**

IP Packet format [5/5]
TOS field {Type of Service}

- Meant to specify ***how*** the datagram should be ***handled*** as it traversed the internet
  - Preference for low delay
  - Preference for high reliability
- In practice TOS was not widely implemented

The 8 bits allocated to TOS can be divided into 5 parts



Providing host-to-host service model over heterogeneous collection of networks

- Each network technology has its own idea of how large a packet can be
  - Ethernet v2: 1500 bytes
  - FDDI: 4500 bytes

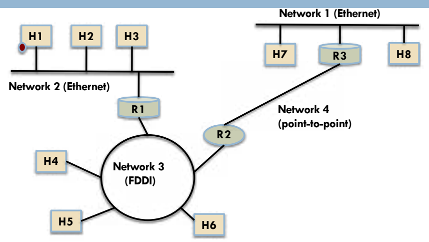
Every network type has a Maximum Transmission Unit (MTU)

- Largest IP datagram that it can carry in its frame
- Smaller than the largest packet-size of network
  - IP datagram needs to fit in payload of **link-layer frame**

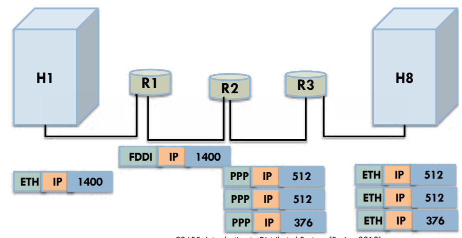
Fragmentation necessary when datagram path includes network with smaller MTU

- All fragments carry same identifier in **Ident** field
  - To enable fragment reassembly
  - Chosen by the source host
- If all fragments do not arrive at receiving host?
  1. Receiver ***gives up*** reassembly [reassembly timeout: 15 seconds RFC0791]
  2. ***Discards*** fragments that did arrive
- IP **does not attempt** to recover from missing fragments

A simple internetwork: Sending IP datagrams from H1 to H8



IP datagrams traversing a sequence of physical networks



IPv4 Packet header

0	4	8	16	19	31
Version	HLen	TOS	Length		
Ident			Flags	Offset	
TTL		Protocol	Checksum		
SourceAddr					
DestinationAddr					
Options (variable)				Pad (variable)	
Data					

Header fields used in IP fragmentation:
Fragmentation occurs at 8-byte boundaries

Start of header				
Ident = x	0	DF	0	Offset = 0
Rest of header				
1400 data bytes				

Unfragmented packet

Start of header				
Ident = x			1	Offset = 0
Rest of header				
512 data bytes				

Fragmented packet

Header fields used in IP fragmentation:
Fragmentation occurs at 8-byte boundaries

Start of header				
Ident = x			1	Offset = 64
Rest of header				
512 data bytes				

Fragmented packet

Start of header				
Ident = x			0	Offset = 128
Rest of header				
376 data bytes				

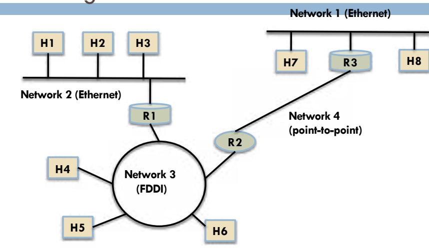
Fragmented packet

DATAGRAM FORWARDING

Datagram forwarding in IP: Datagrams contains IP address of destination

- Network part uniquely identifies a single physical network
- Hosts/routers that share the network part
  - Connected to *same* physical network
- Every physical network has a router
  - Connected to at least *one other* physical network

A simple internetwork: Forwarding table for router R2



Example forwarding table: For Router R2

Network Num	Next Hop
1	R3
2	R1

Error Reporting in IP communications

- IP drops datagrams when the going gets tough
  - But does not fail silently
- IP always configured with a **companion** protocol
  - Internet Control Message Protocol (ICMP)

ICMP defines a collection of error messages

- When router/host is unable to process datagram successfully
  - ICMP error message **sent back to source**
- Examples
  - Destination host is unreachable
  - Reassembly process failed
  - TTL reached 0
  - IP header checksum failed

ICMP also defines some control messages

- Router sends **control messages** back to host
- Example: `ICMP-Redirect` tells that there is a better route to destination
  - Network has two routers `R1` and `R2` and host uses `R1` as default
  - When `R1` receives a datagram and it knows `R2` is a better choice?
    - Send `ICMP-Redirect` to host
    - Host then uses `R2` for *future* datagrams to that host

The contents of this slide-set are based on the following references

- *Computer Networks: A Systems Approach. Larry Peterson and Bruce Davie. 4th edition. Morgan Kaufmann. ISBN: 978-0-12-370548-8. Chapter [4]*