Lecture 10 : Network Layer

Shankar Balachandran
Assistant Professor
Dept. of CSE, IIT Madras

10.1 Internetworking

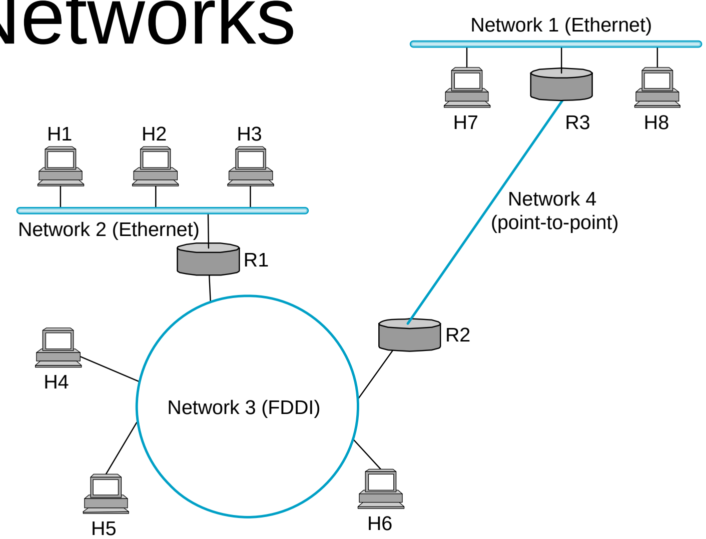
- Single network
  - Shared media
  - Point to point links
  - Switches
- In real world, people use different kinds of networks
- Problem : Heterogeneity and Scale
  - Not addressed so far
- Hosts of different networks may want to talk
  - Paths may be through other kinds of networks
  - Each step in between may have it's own
    - addressing mechanism
    - Media Access Control (MAC) protocol
    - …

Organization

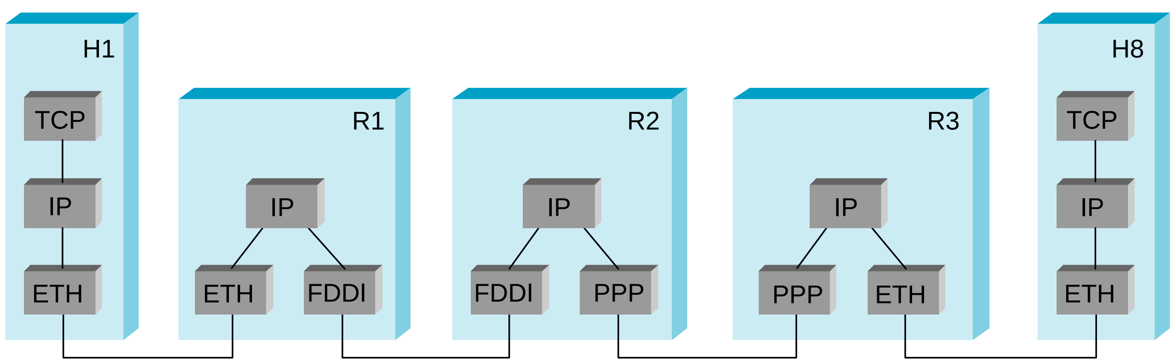
- Internet Protocol (IP)
  - Handling heterogeneity
  - Scale
    - Many other protocols available, but IP is the most interesting of all
- Finding paths from one network to another
  - aka routing
- Problems with internet
- Recent trends

IP Internet

- Concatenation of Networks



- Protocol Stack

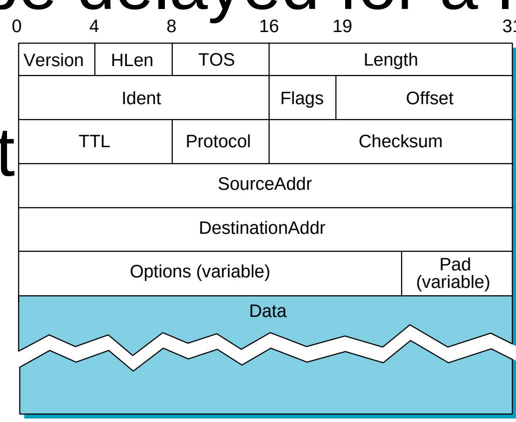


Service Model

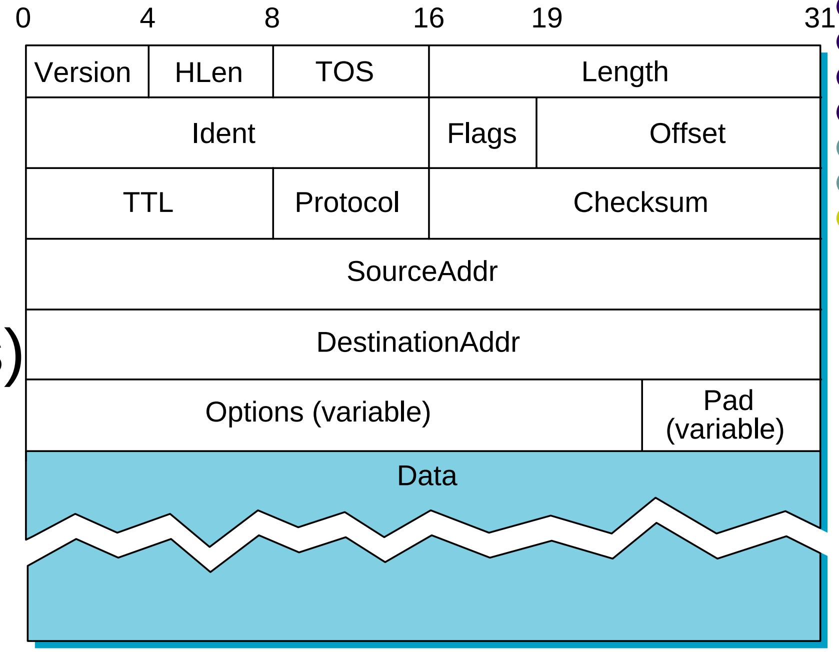
- Provide common services
  - Kept intentionally simple and less demanding
  - Any network technology can participate

- Connectionless (datagram-based)
- Best-effort delivery (unreliable service)
  - packets are lost
  - packets are delivered out of order
  - duplicate copies of a packet are delivered
  - packets can be delayed for a long time

- Datagram format



Datagram

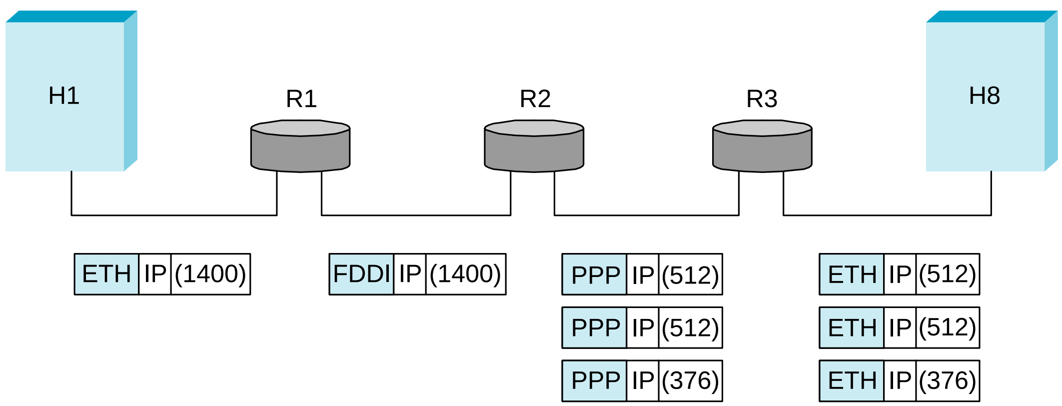


- Grouped as words(32 bits)
- Version : 4, 6 etc
  - Location
- HLen : 20 bytes max
- TOS : Type of service
- Length = < 64K
- Second Word : Fragmentation
- Third Word :
  - TTL in number of hops
  - TCP or UDP or …
- Source and destination address in every single packet

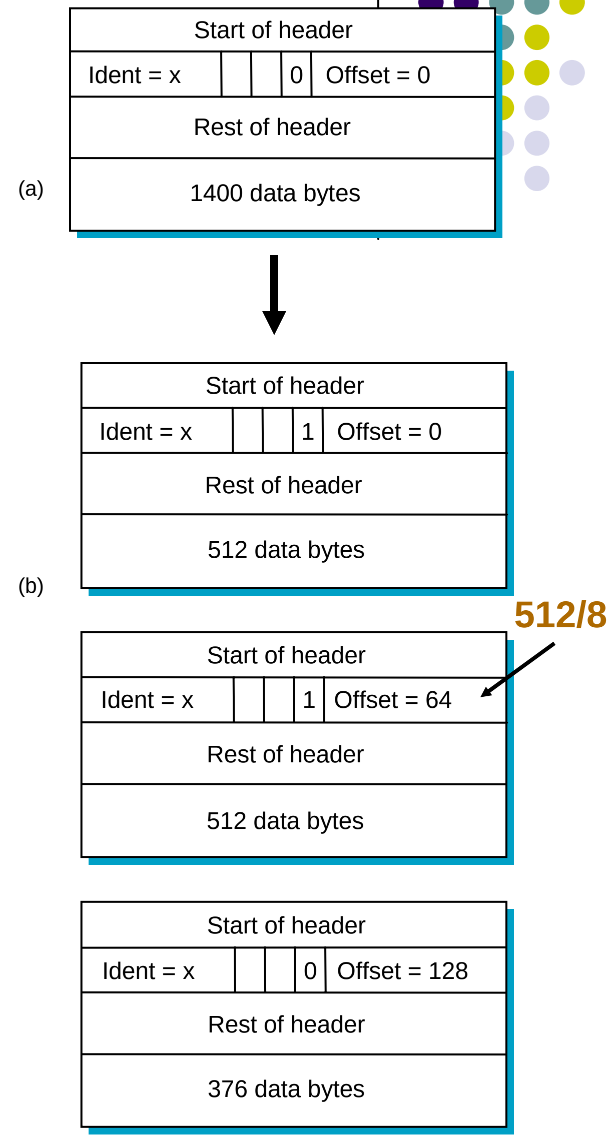
Fragmentation and Reassembly

- Each network has some MTU
- Design decisions
  - fragment when necessary (MTU < Datagram)
  - try to avoid fragmentation at source host
  - re-fragmentation is possible
  - fragments are self-contained datagrams
  - delay reassembly until destination host
  - do not recover from lost fragments

Example



H1
R1
R2
R3
H8
ETH IP (1400)
FDDI IP (1400)
PPP IP (512)
PPP IP (512)
PPP IP (376)
ETH IP (512)
ETH IP (512)
ETH IP (376)

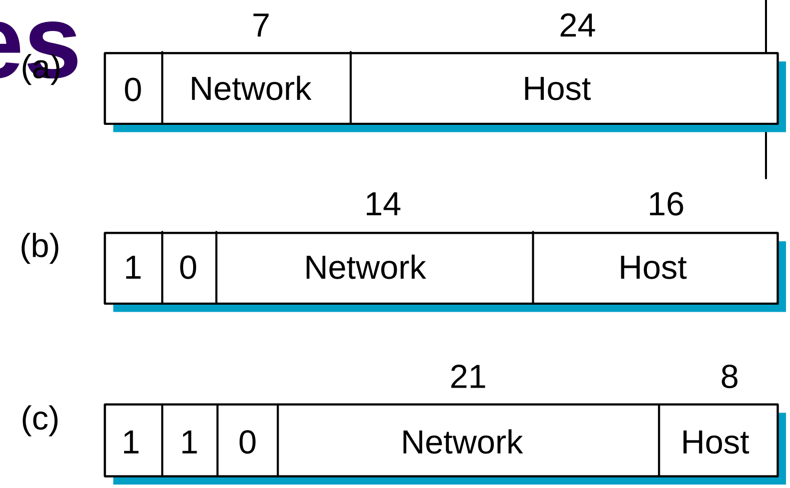


(a)
Start of header
Ident = x
0
Offset = 0
Rest of header
1400 data bytes
(b)
Start of header
Ident = x
1
Offset = 0
Rest of header
512 data bytes
512/8
Start of header
Ident = x
1
Offset = 64
Rest of header
512 data bytes
Start of header
Ident = x
0
Offset = 128
Rest of header
376 data bytes

Global Addresses

- Thought process
  - Why don’t we use Ethernet addresses directly?
    - After all, they are unique
    - Problem : Does not reveal structure of the network.
    - Problem : Not all hosts have ethernet interfaces
  - Need : A mechanism to provide
    - Unique addresses
    - hierarchical: network + host

Global Addresses



- Dot Notation
  - 10.3.2.4
  - 128.96.33.81
  - 192.12.69.77
- Class A
  - 126 n/ws possible
  - 2^24 – 2 hosts in every n/w
- Class B
  - 14 bits for n/w; 64K hosts per network
- Class C
  - 2^21 Class C networks possible
- Addressing is flexible
  - Can arrange networks in their natural way and take care of naming with one of the classes

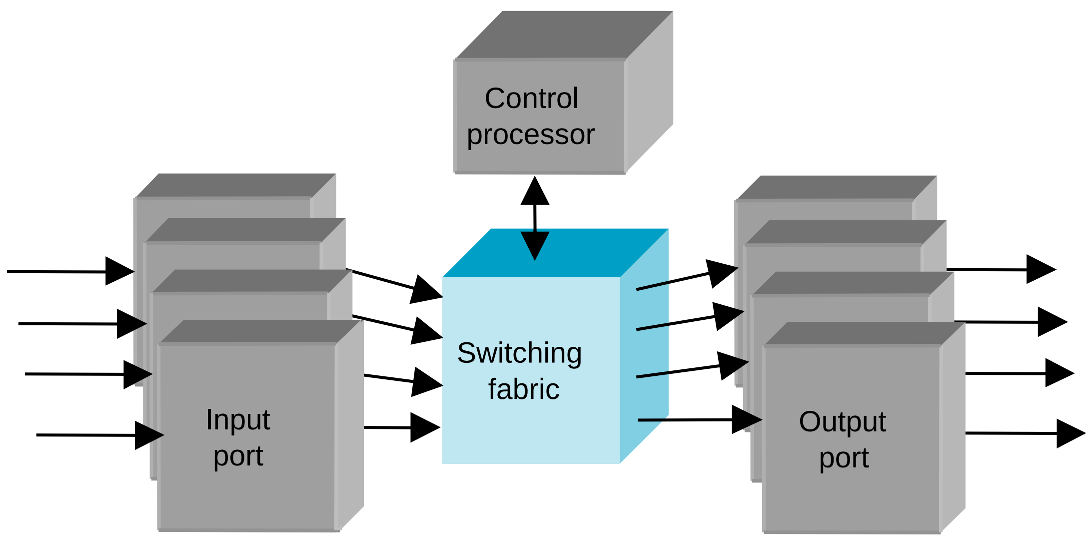
Datagram Forwarding

- Strategy
  - every datagram contains destination's address
  - if connected to destination network, then forward to host
  - if not directly connected, then forward to some router
  - forwarding table maps network number into next hop
  - each host has a default router
  - each router maintains a forwarding table

- Example (R2)

Network Number	Next Hop
1	R3
2	R1
3	interface 1
4	interface 0

Router Implementation



Address Translation

- Map IP addresses into physical addresses
  - destination host
  - next hop router
- Techniques
  - encode physical address in host part of IP address
  - table-based
- ARP
  - table of IP to physical address bindings
  - broadcast request if IP address not in table
  - target machine responds with its physical address
  - table entries are discarded if not refreshed

ARP Details

- Request Format
  - HardwareType: type of physical network (e.g., Ethernet)
  - ProtocolType: type of higher layer protocol (e.g., IP)
  - HLEN & PLEN: length of physical and protocol addresses
  - Operation: request or response
  - Source/Target-Physical/Protocol addresses
- Notes
  - table entries timeout in about 10 minutes
  - update table with source when you are the target
  - update table if already have an entry
  - do not refresh table entries upon reference

ARP Packet Format

<table>
<tr><th>0</th><th>8</th><th>16</th><th>31</th></tr>
<tr><td colspan="2">Hardware type = 1</td><td colspan="2">ProtocolType = 0x0800</td></tr>
<tr><td>HLen = 48</td><td>PLen = 32</td><td colspan="2">Operation</td></tr>
<tr><td colspan="4">SourceHardwareAddr (bytes 0–3)</td></tr>
<tr><td colspan="2">SourceHardwareAddr (bytes 4–5)</td><td colspan="2">SourceProtocolAddr (bytes 0–1)</td></tr>
<tr><td colspan="2">SourceProtocolAddr (bytes 2–3)</td><td colspan="2">TargetHardwareAddr (bytes 0–1)</td></tr>
<tr><td colspan="4">TargetHardwareAddr (bytes 2–5)</td></tr>
<tr><td colspan="4">TargetProtocolAddr (bytes 0–3)</td></tr>
</table>

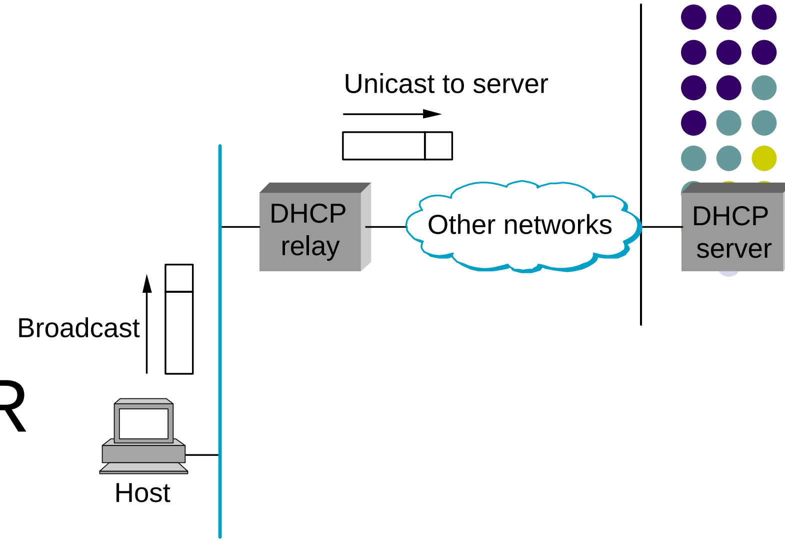
Host Configuration (Assigning Host ID's)

- Thought Process :
  - Use ethernet : Will fix hosts to networks
- Need : Something that's reconfigurable
  - Letting a host participate in any network that it wants to
  - IP addresses may just help
- What else is needed?
  - Default router : Router that you are going to connect to if sender and receiver not on same shared network
- Manually configure IPs
  - Error-prone
  - Centralized
  - Cumbersome

Dynamic Host Configuration Protocol (DHCP)

- DHCP Server
  - At least one per administrative domain
  - Configuration info is stored in the server
  - Clients retrieve on Power Up
  - Admin still controls who gets what
- Variation of configuration
  - List of unassigned addresses maintained

Mechanism

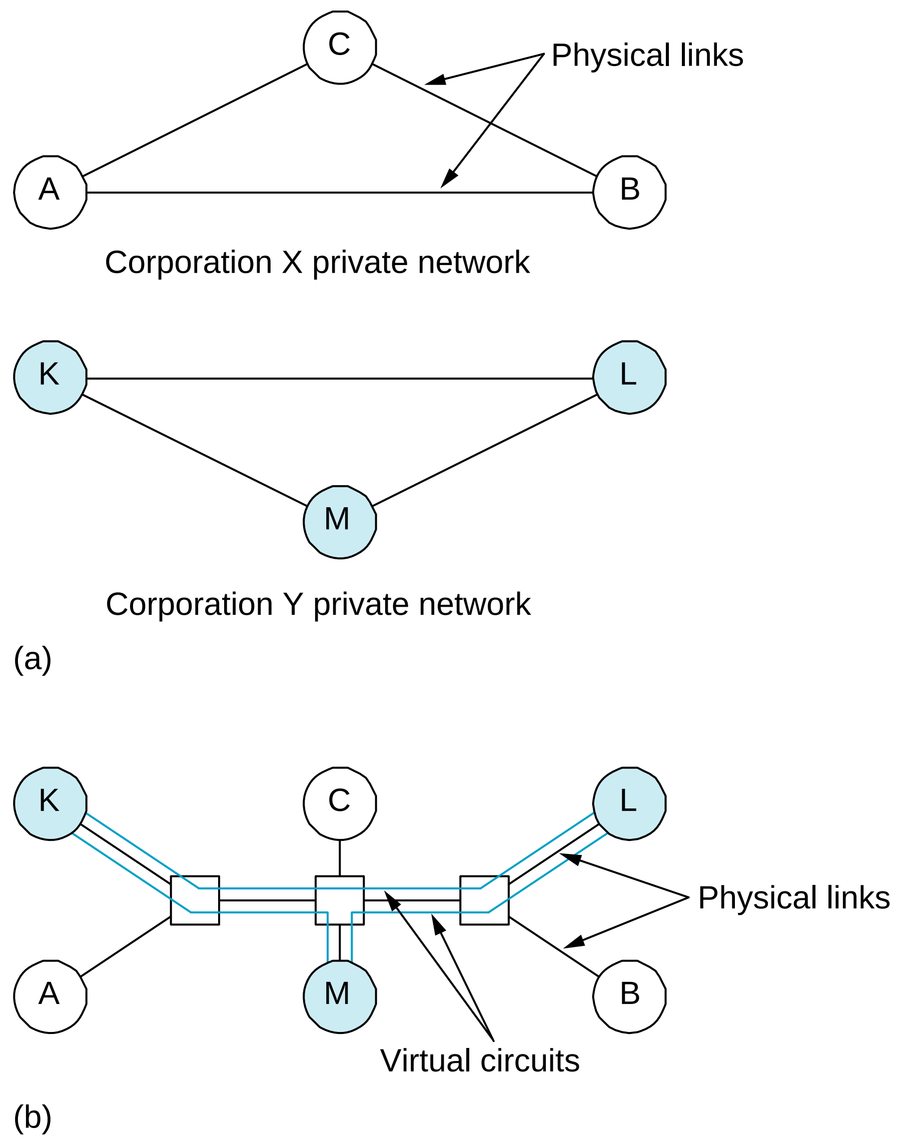


- A host powers up
- Sends a DHCPDISCOVER to 255.255.255.255
- Retrieved by all hosts and routers on that n/w
  - Routers do not forward
- Cases :
  - One node is the DHCP server. It responds.
    - So many DHCP servers all around !!
  - *Relay Agent :* One relay agent per network
    - Knows IP Address of the DHCP server
    - A DHCPDISCOVER observed is unicast to the DHCP server
    - Reply from DHCP server is passed on to the host

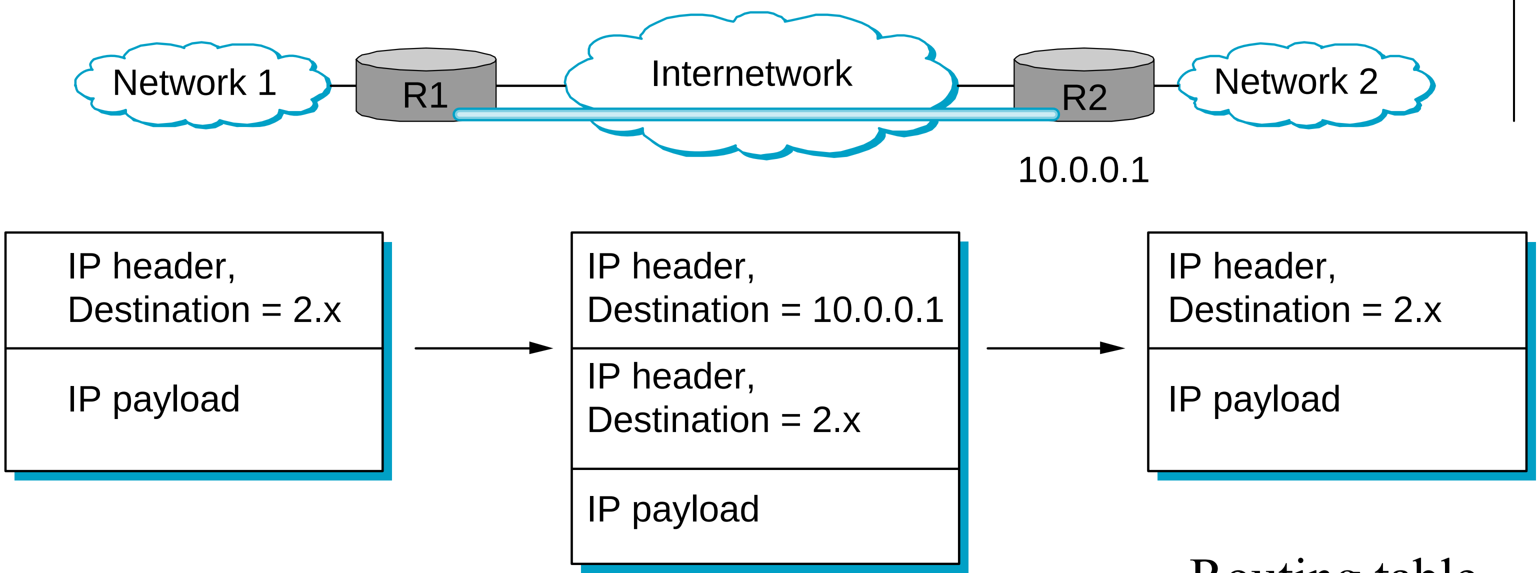
Internet Control Message Protocol (ICMP)

- Echo (ping)
- Redirect (from router to source host)
- Destination unreachable (protocol, port, or host)
- TTL exceeded (so datagrams don't cycle forever)
- Checksum failed
- Reassembly failed
- Cannot fragment

Virtual Networks



Implementing a VPN



Encapsulate traffic from R1 to R2 inside IP packets addressed to R2. Together with encryption, this tunneling of packets is an effective way to implement a VPN.

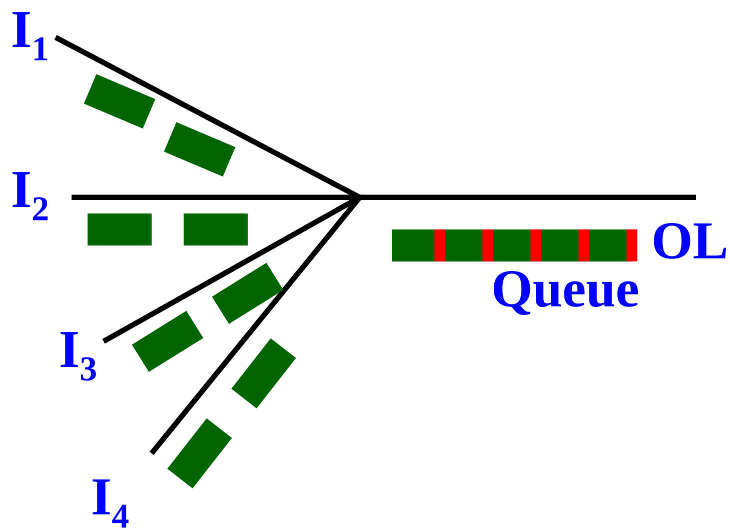
Routing table

NetworkNum	NextHop
1	Interface 0
2	Virtual Interface 0
Default	Interface 1

10.2 Congestion vs. Flow Control

- Flow control:
  - End-to-end
- Congestion control
  - Router to Router

Congestion control vs. Flow control:



- Congestion -
  - buffer length
    - Drop packets
  - Slow processor at the router even though line capacity is high
  - Mismatch between different parts of the system

Congestion vs. Flow Control

- **Router discards packets when it cannot serve**
  - **Sender retransmits until acknowledged**
  - **Congestion builds up**
- **Flow Control**
  - Pt – Pt links between a given sender and a given receiver
  - Fast sender does not overwhelm receiver
  - Receiver can tell sender directly to slow down

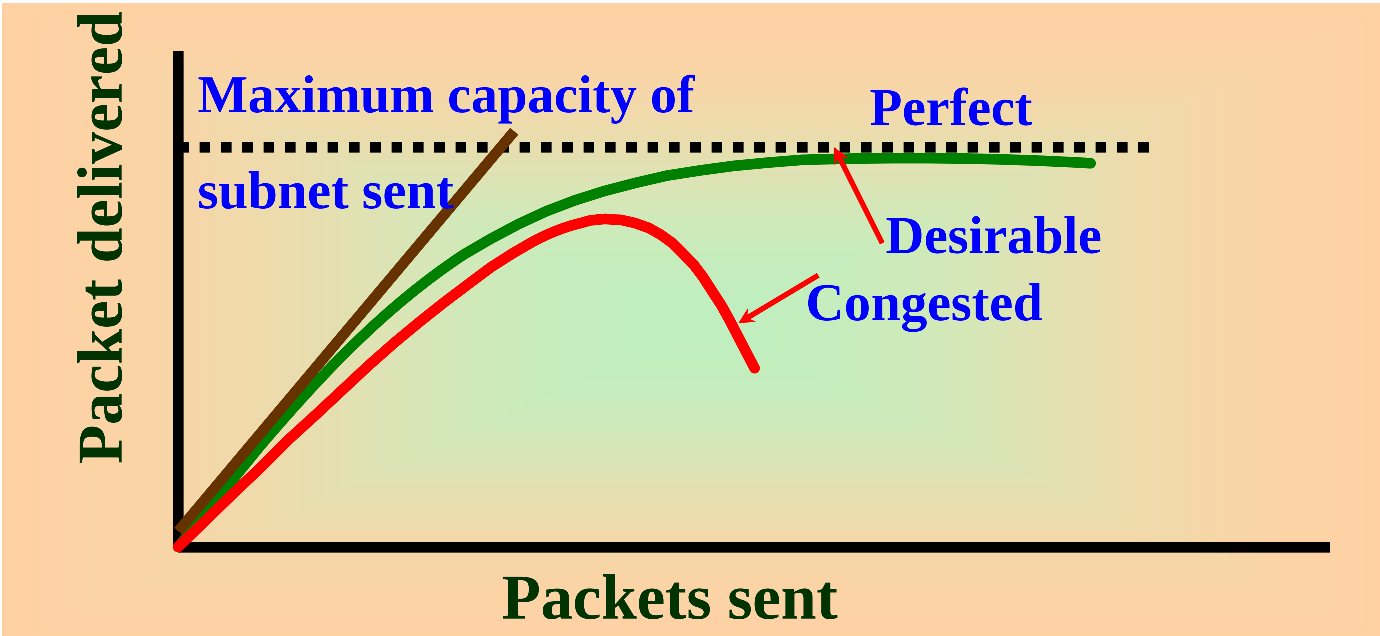
Congestion Control

- General principle of congestion:
  - Monitor system to detect when and where congestion occurs
  - Pass this information to places where action can be taken
  - Adjust system operation to correct the problem

Congestion vs. Flow Control

- Policing traffic at routers
  - Token bucket / leaky bucket
  - non trivial
- Alternative flow specifications:
  - Agreed between sender and receiver
  - pattern of injected traffic
  - QoS desired by Application

Congestion Control Algorithm:



- Routers loose packets
- Buffering?
    - No use
    - Packet reaches front of Queue, duplicate generated

Traffic Shaping

- Traffic monitoring:
  - Monitoring a traffic flow
  - VC no problem
    - Can be done for each VC separately since connection oriented
- DG - Transport layer

Congestion: Reasons

Congestion causing policies:

- Transport Layer
  - Retransmission
  - Out of order caching policy
  - Ack policy
  - Flow control policy
  - Time out
- Network Layer:
  - VC versus datagram inside subnet
  - Packet queuing and service policy
  - Packet discard policy
  - Routing algorithm policy
  - Packet lifetime management policy

Congestion Control (contd.)

- Solution:
  - Traffic prediction?
  - Router informs neighbour of possible congestion
  - Traffic shaping
  - Regulate the packet rate
  - VC - traffic characteristics
    - Not too important for file transfer but important for audio and video

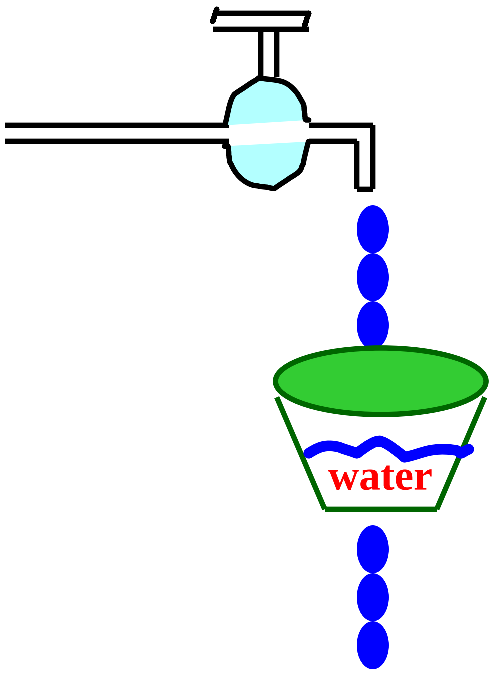
Congestion Control (contd.)

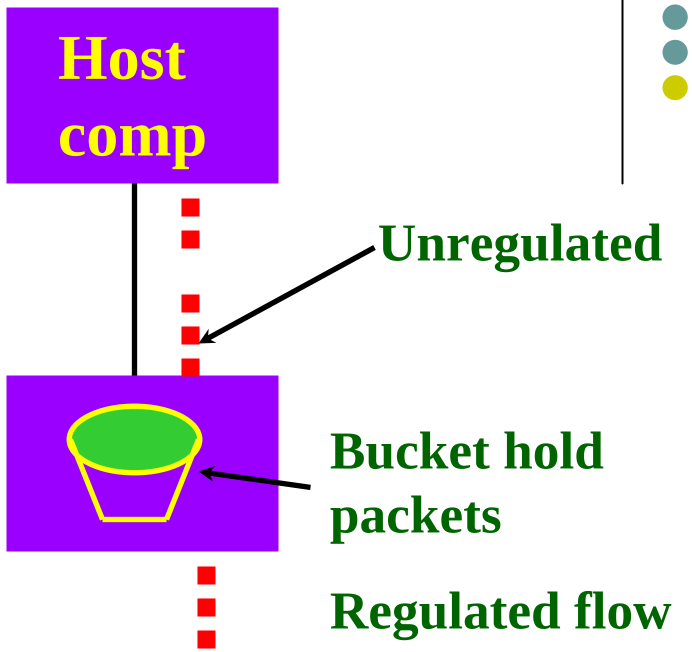
- Send probe packets periodically ask about congestion
  - Road congestion – use helicopters flying over cities
  - Bang bang operation of router – how does one prevent it
  - Feed back and control required

Congestion Control Algorithms

- Leaky Bucket Algorithm
  - Regulate output flow
    - Packets lost if buffer is full
- Token Bucket Algorithm
  - Buffer filled with tokens
    - transmit ONLY if tokens available

Leaky bucket algorithm:



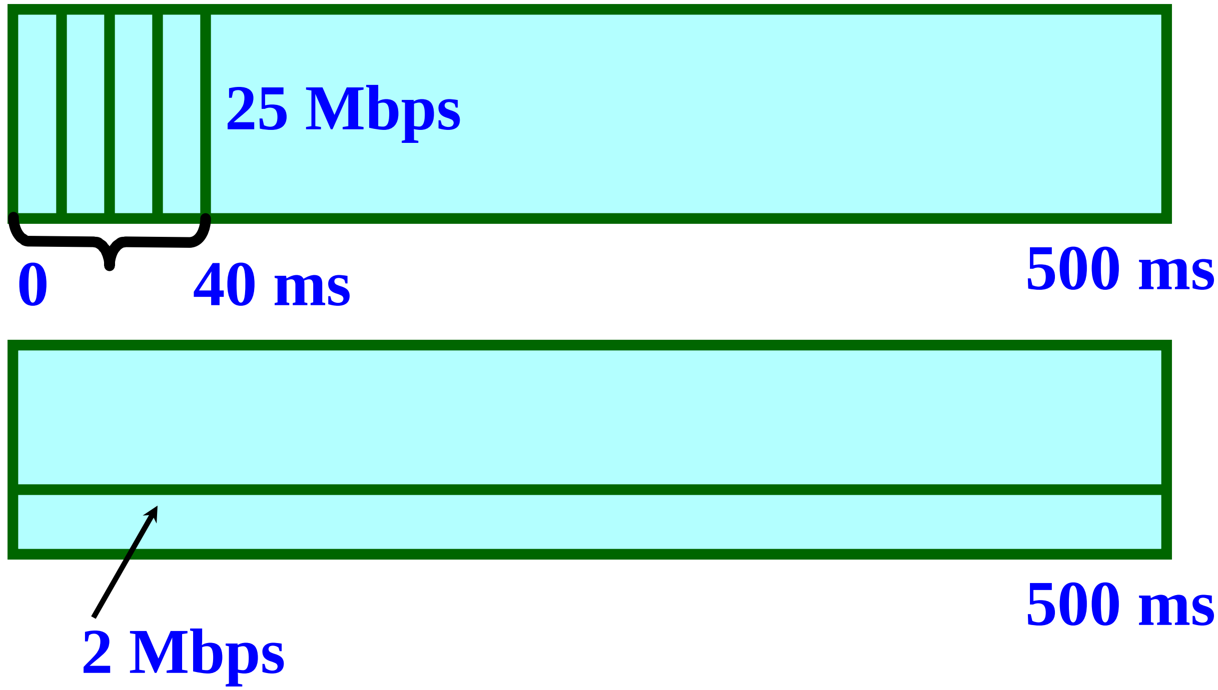


Bucket full – lost packets

- Output flow constant
  - when water in bucket – zero when no water
- Converts uneven flow to even flow
  - Packets Queued
  - Packets output at regular intervals only

Leaky Bucket Algorithm

- Queue full, packet discarded.
  - What if packets are different size and fixed bytes/ unit time.
- Leaky bucket example
  - Input burst 25 Mb/s every 40 ms
    - Network speed 25 Mbps – every second
    - Capacity of bucket C – 1 Mb
    - Reduce average rate – 2 Mbps
    - bucket can hold upto 1 Mb without data loss,
    - burst spread over 500 ms irrespective of how fast they come

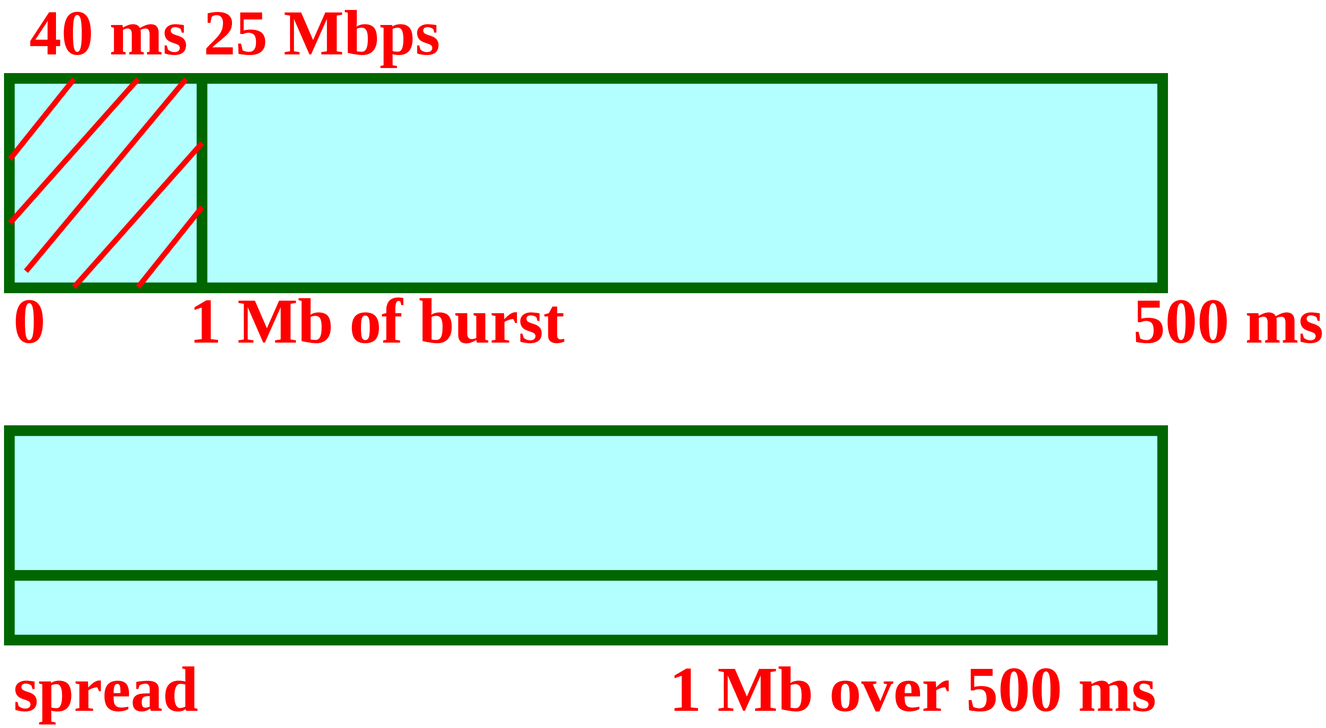


$$\frac{25 \times 40}{1000} = 1 \text{ Mb}$$

$25 \quad - \quad 1000 \qquad \dfrac{40 \times 25}{1000}$

? - 40 = 1 Mb every sec

- spread it over 500 ms

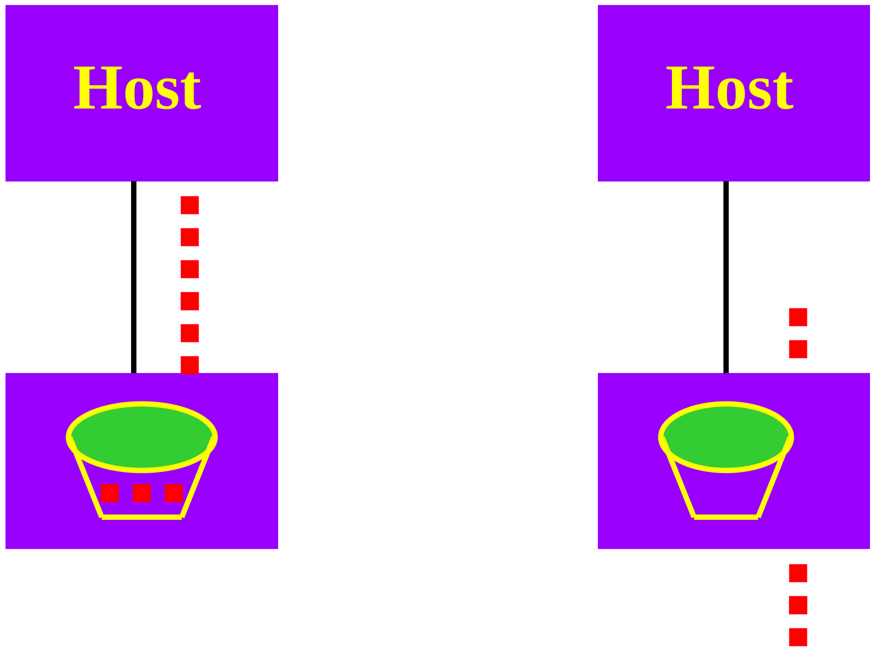


output rate 2 Mbps

Leaky bucket issues:

* **Drops packets**
* **Does not allow host to save permission to transmit large burst later**

Token bucket Algorithm



- Host save packets upto maximum size of bucket, n
- n packets send at once – some burstiness
- Host captures token
- Never loose data
- Tokens not available packets queue up! – not discarded

Token Bucket Algorithm

- Packet gets tokens and only then t transmitted
  - A variant – packets sent only if enough token available - token - fixed byte size
  - Token bucket holds up n tokens
    - Host captures tokens
    - Each token can hold some bytes
    - Token generated every T seconds
    - Allows bursts of packets to be sent - max n
    - Responds fast to sudden bursts
    - If bucket full – thrown token packets not lost

Token Bucket Algorithm (example)

Calculation of length of maximum rate burst:

- Tokens arrive while burst output

Example

S – burst length in S

M – Maximum output rate

MS – Maximum length in bytes

ρ – Token arrival rate

C – Capacity of token bucket in byte

Token Bucket Algorithm (Example)

Maximum output burst = $C + \rho S = MS$

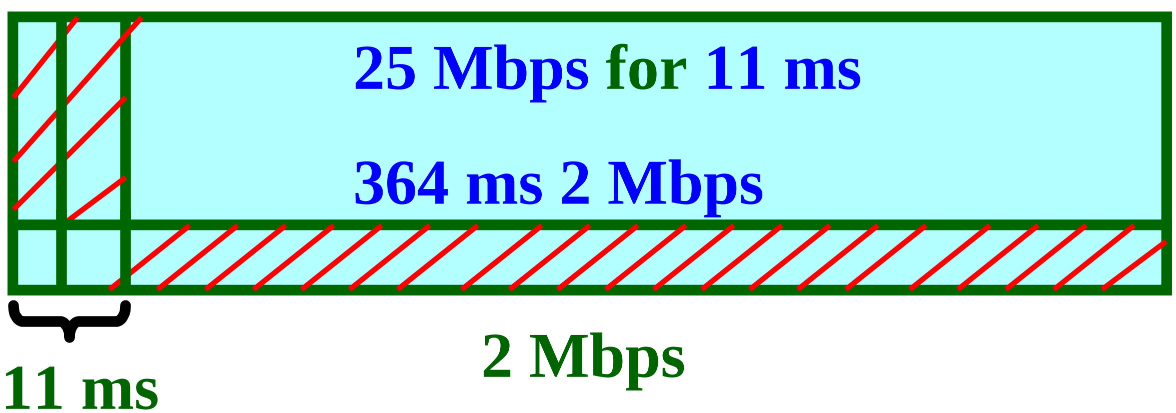
$$S = \left(\frac{C}{M-\rho}\right)$$

C = 250 Kb

M = 25 Mbps

ρ = 2 Mbps

S = 11 ml



Flow Control

- Flow Control is specified end to end
  - Sliding window protocol
  - Fast sender vs. slow receiver
    - Sender does not overwhelm receiver
  - Advertisement of window size
    - receiver tells sender DIRECTLY
  - Process to process
- See More about flow control in TCP