

IP Addresses

10000000 00001011 00000011 00011111

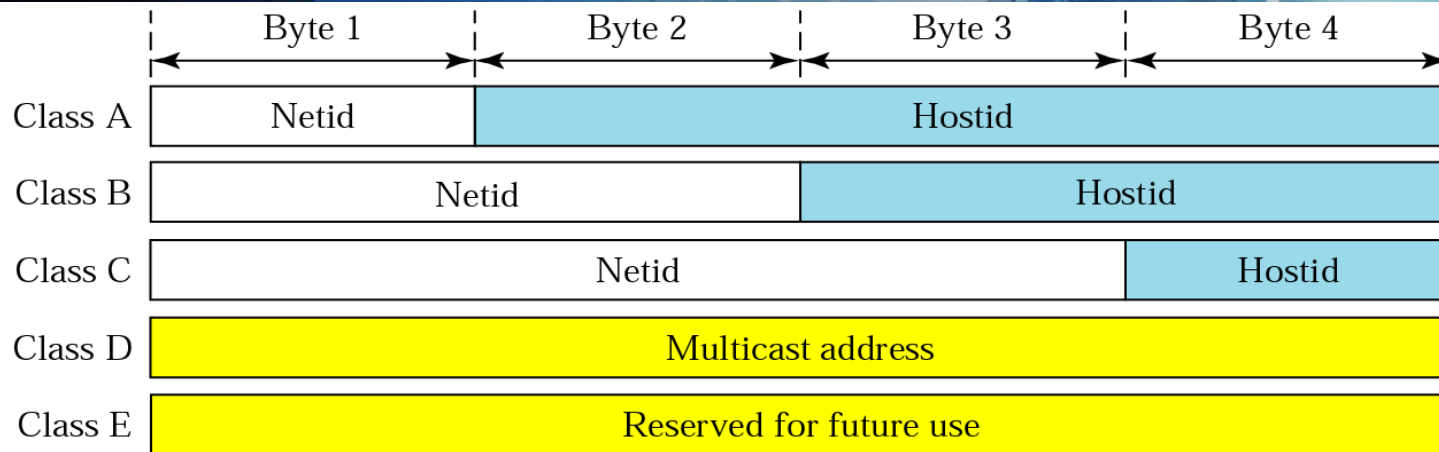
128.11.3.31

- 32-bit number
 - 4.294.967.296 addresses
- IP addresses are unique and universal
 - except private addresses

Range			Total
10.0.0.0	to	10.255.255.255	2^{24}
172.16.0.0	to	172.31.255.255	2^{20}
192.168.0.0	to	192.168.255.255	2^{16}

- Dotted decimal notation:
 - Bytes of binary notation represented as decimal separated by dot

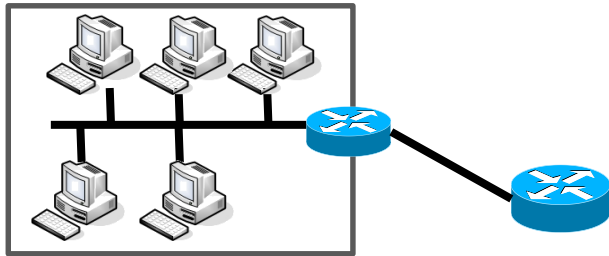
Classful Addresses



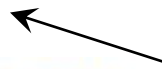
- Class A (international organisations)
 - 126 networks with 16,277,214 hosts each
- Class B (large companies)
 - 16,384 networks with 65,354 hosts each
- Class C (smaller companies)
 - 2,097,152 networks with 254 hosts each
- Inefficient use of hierarchical address space
 - Class C with 2 hosts ($2/254 = 0.78\%$ efficient)
 - Class B with 256 hosts ($256/65534 = 0.39\%$ efficient)

Network Layer

Trinity College



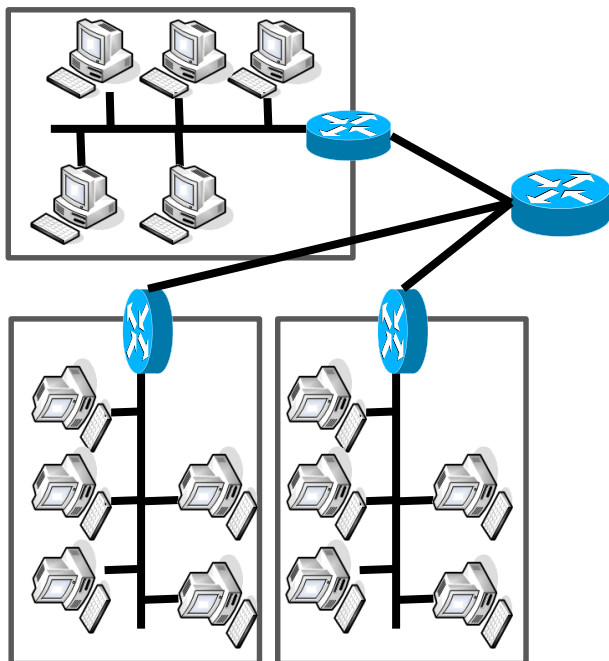
134.226.0.0



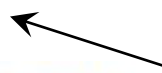
Class B

Network Layer

Trinity College



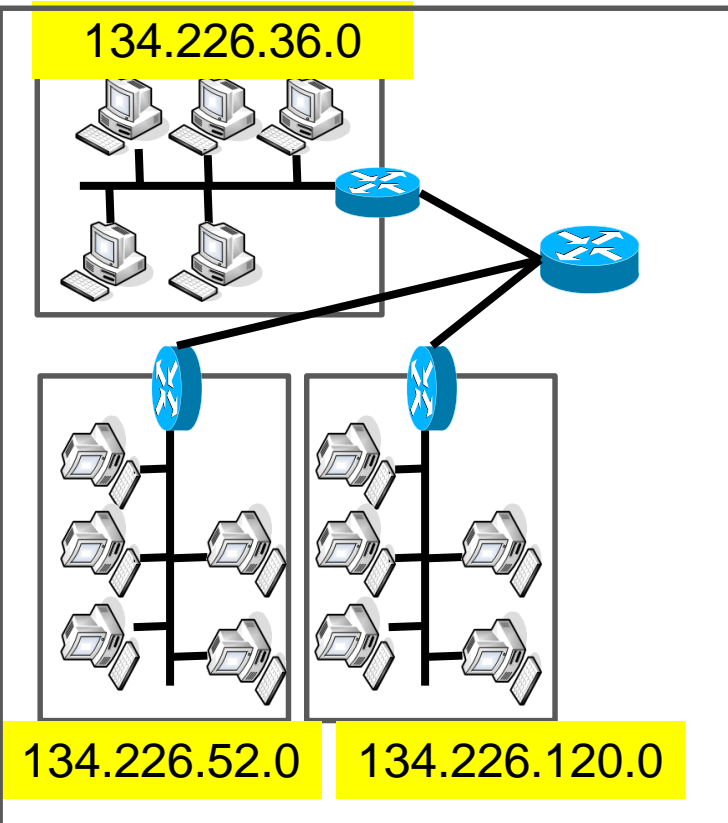
134.226.0.0



Class B

Network Layer

Trinity College

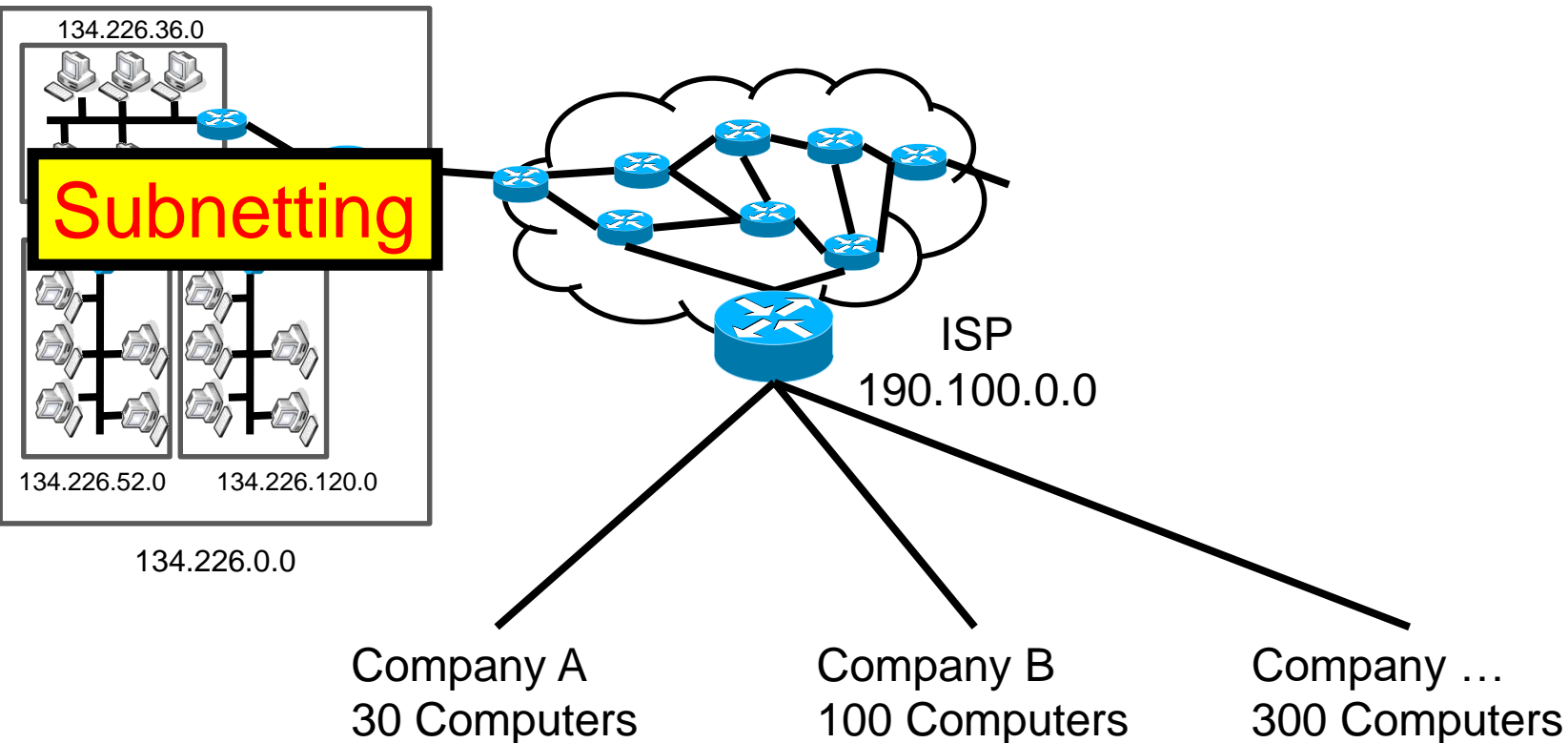


134.226.0.0

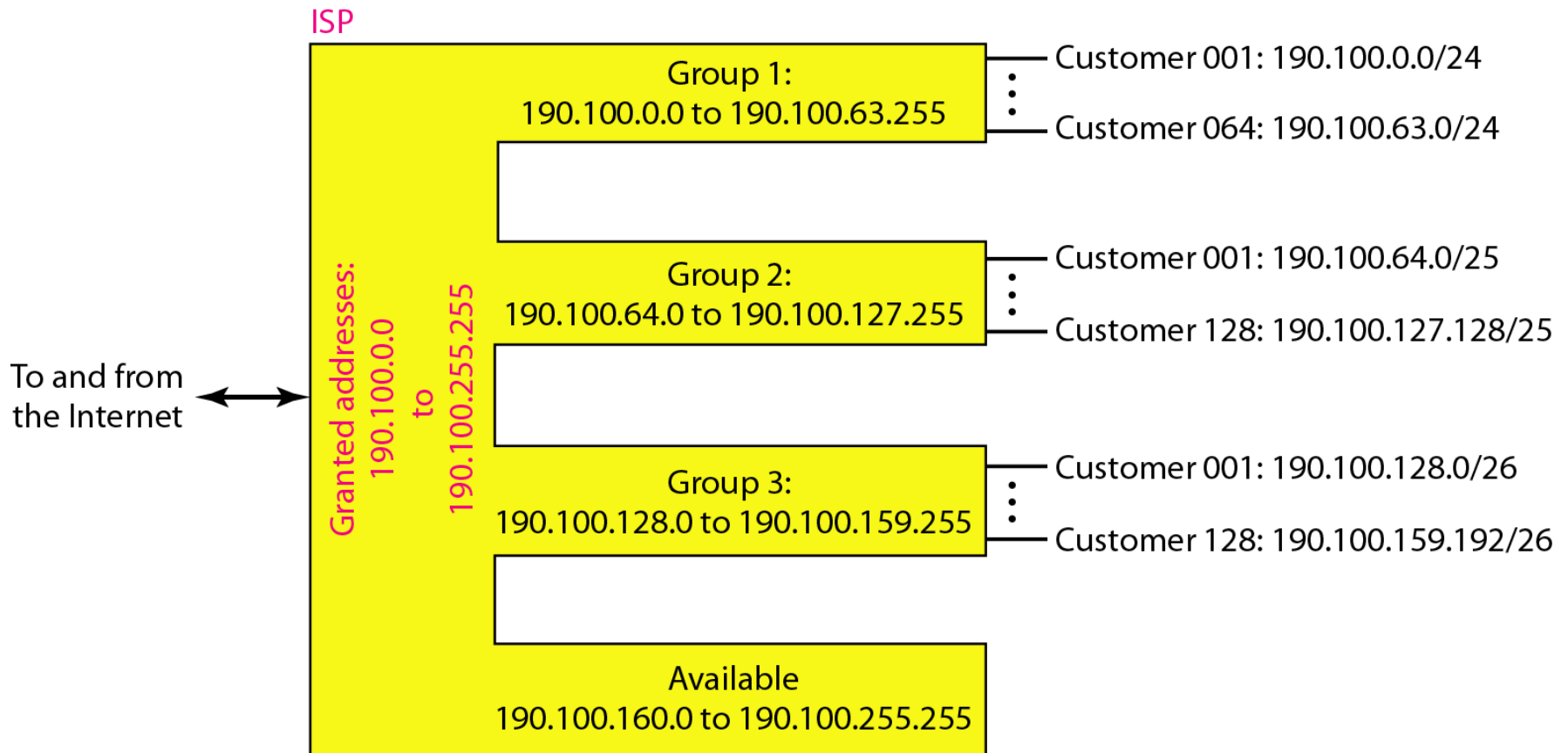
Class B ← Classful Addresses

Network Layer

Trinity College



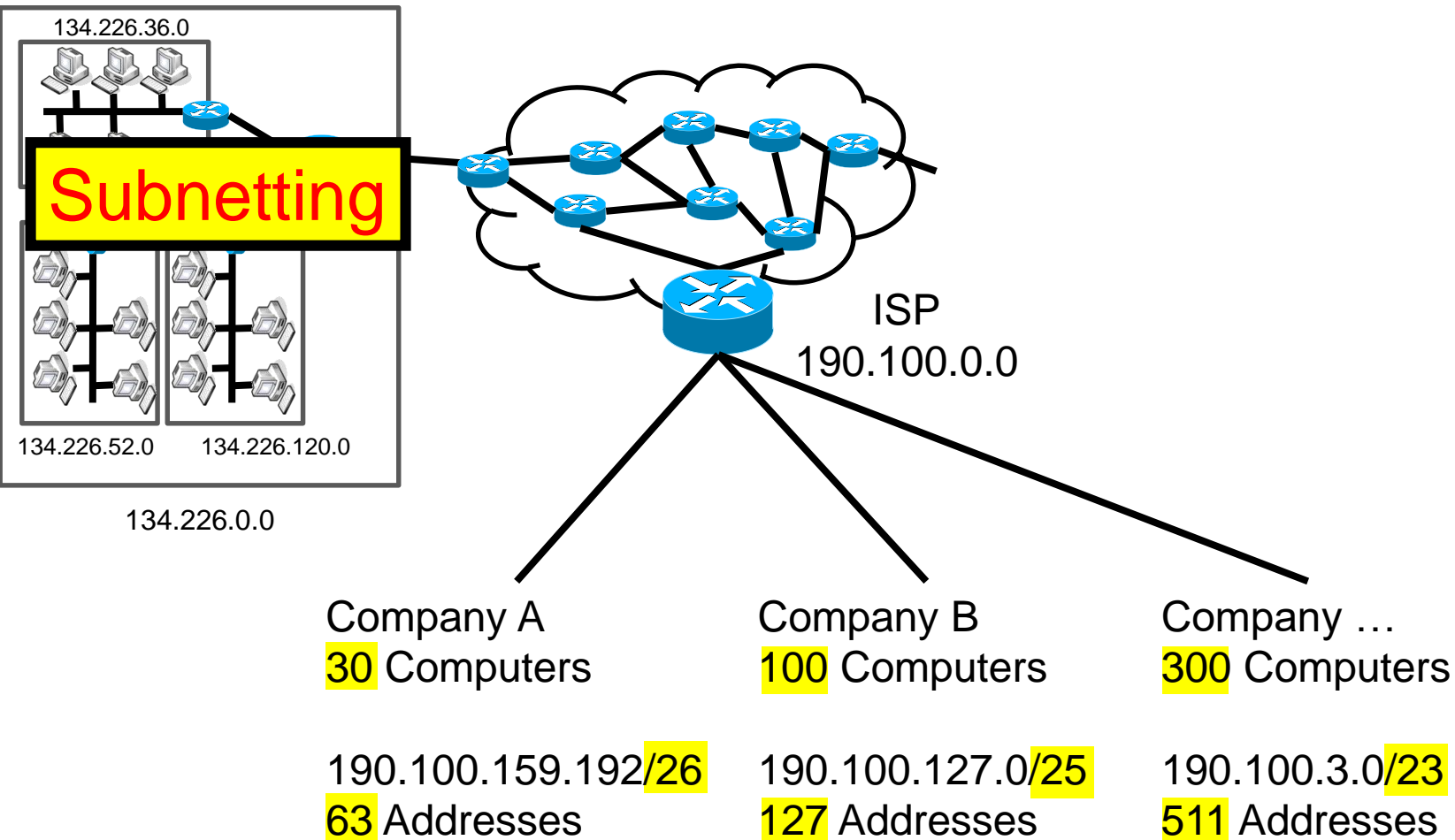
ISPs & Classless Addresses



* Figure is courtesy of B. Forouzan

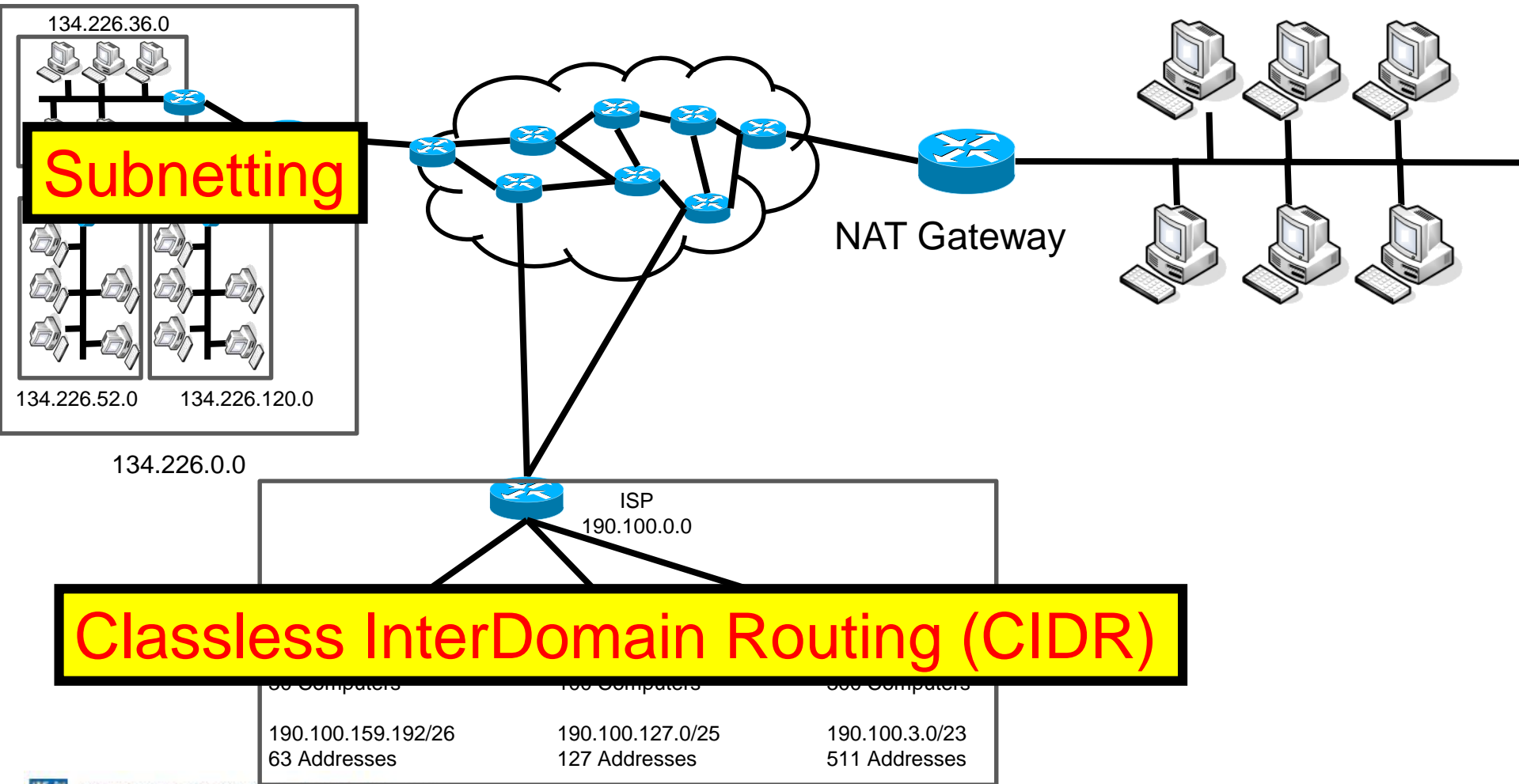
Network Layer

Trinity College

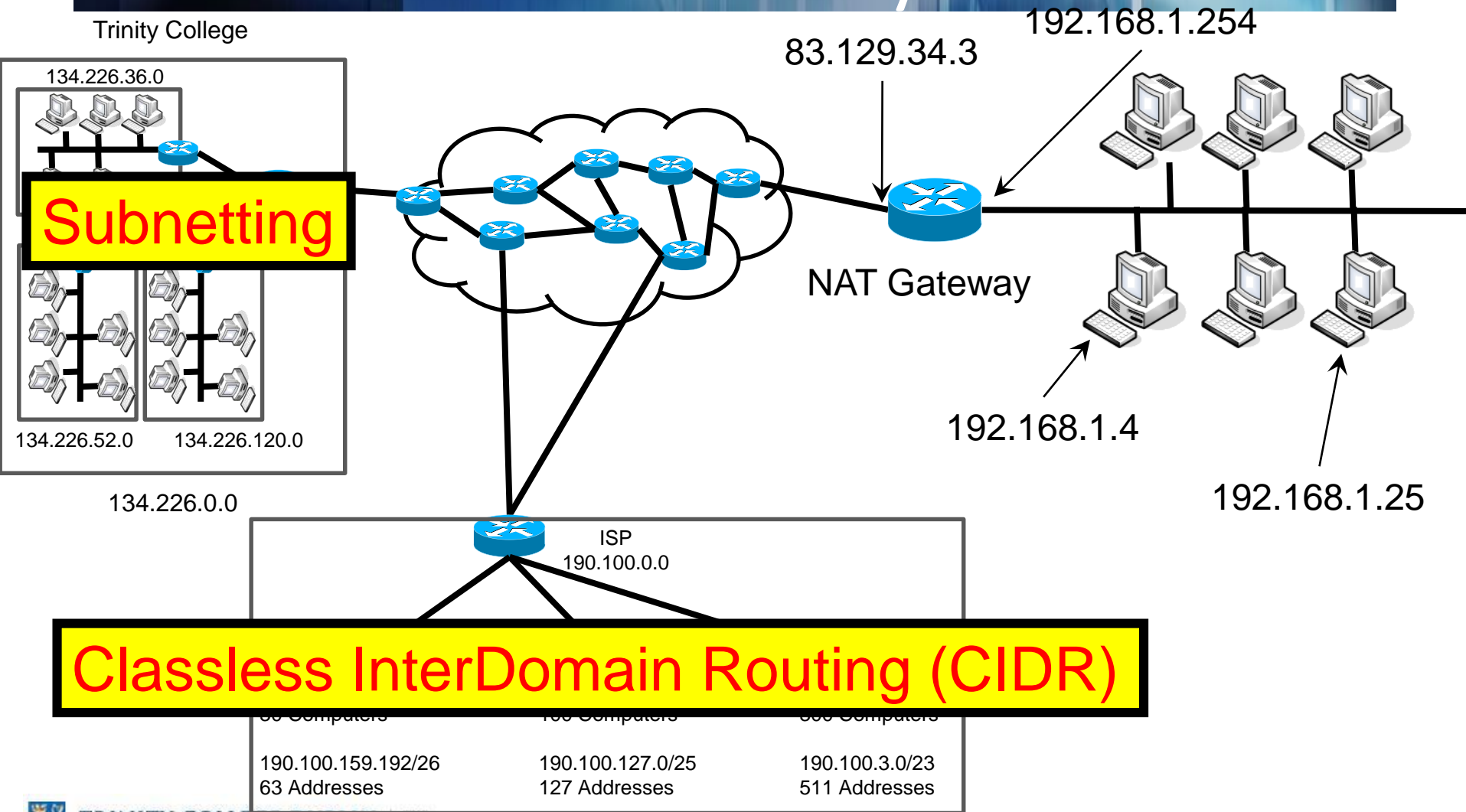


Network Layer

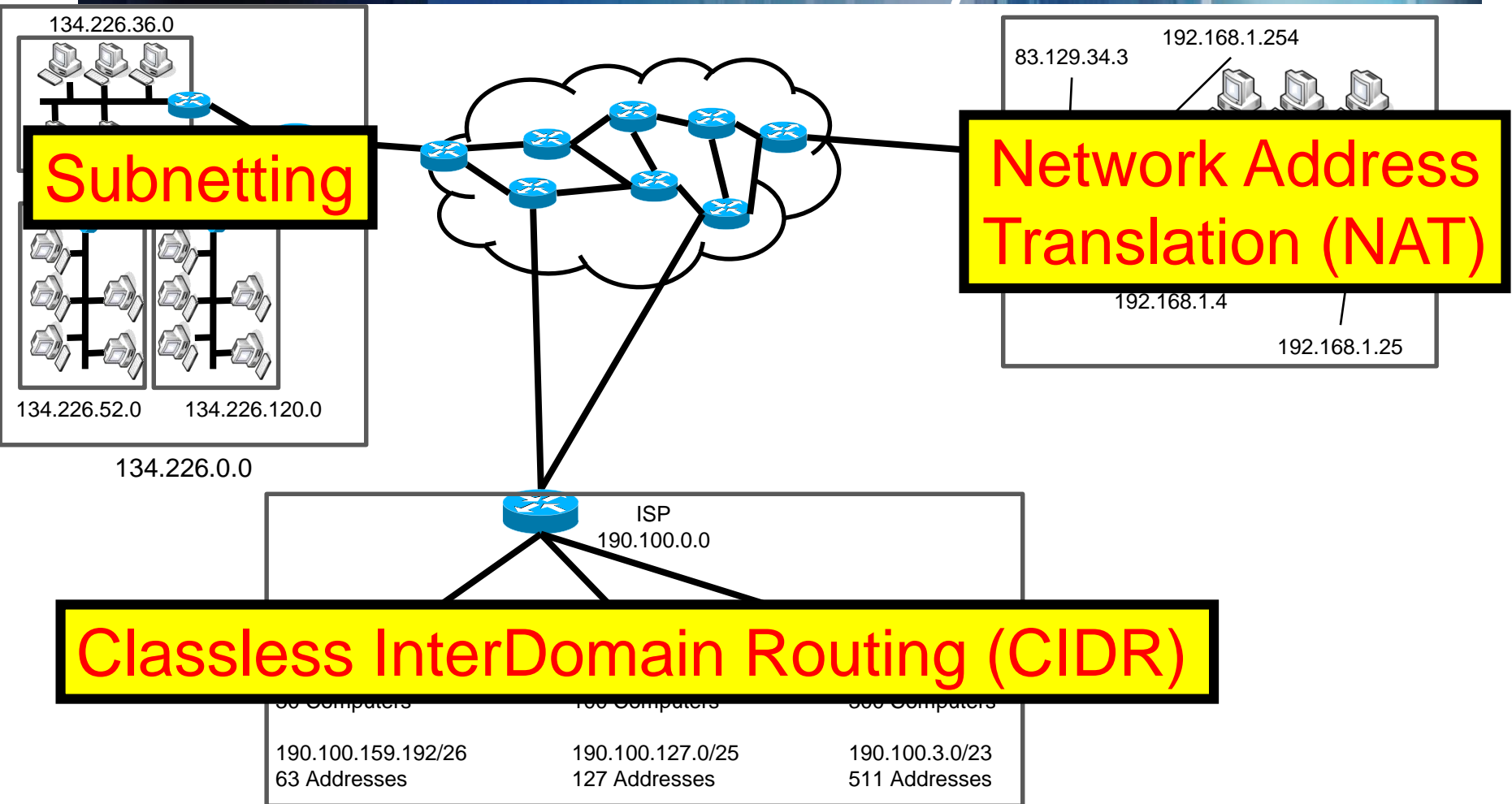
Trinity College



Network Layer



Network Layer



Summary: Network Layer&Addresses

- Doted-decimal notation
- Classful addresses
 - Classes A, B, and C for networks, D for multicast
- Subnetting
- Classless Inter-Domain Routing (CIDR)
 - / notation = significant bits in subnet mask
- Network address translation (NAT)

Internet Protocol (IP)

Fragmentation

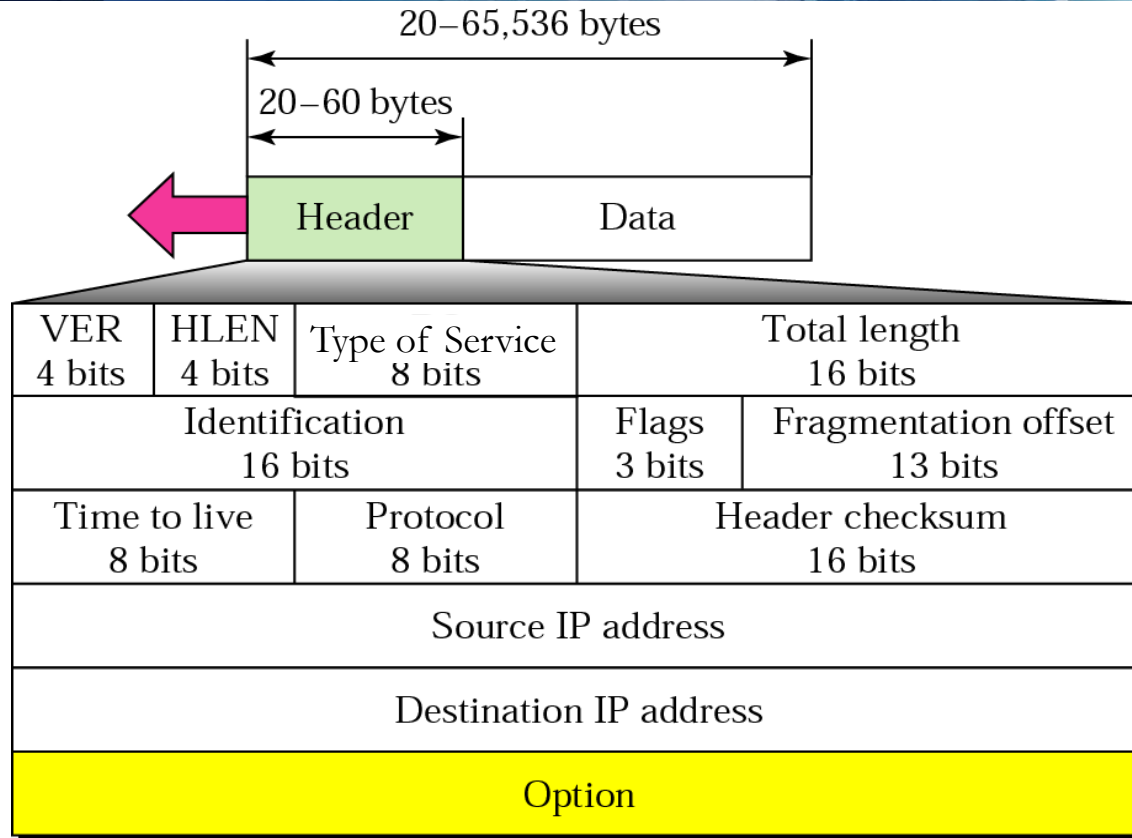
IP Datagram Format

- The unit of IP data delivery is called a datagram
- Datagrams can have different sizes
- Includes header area and data area



- Header area generally 20 bytes
 - but can be more depending on options

IP Datagram

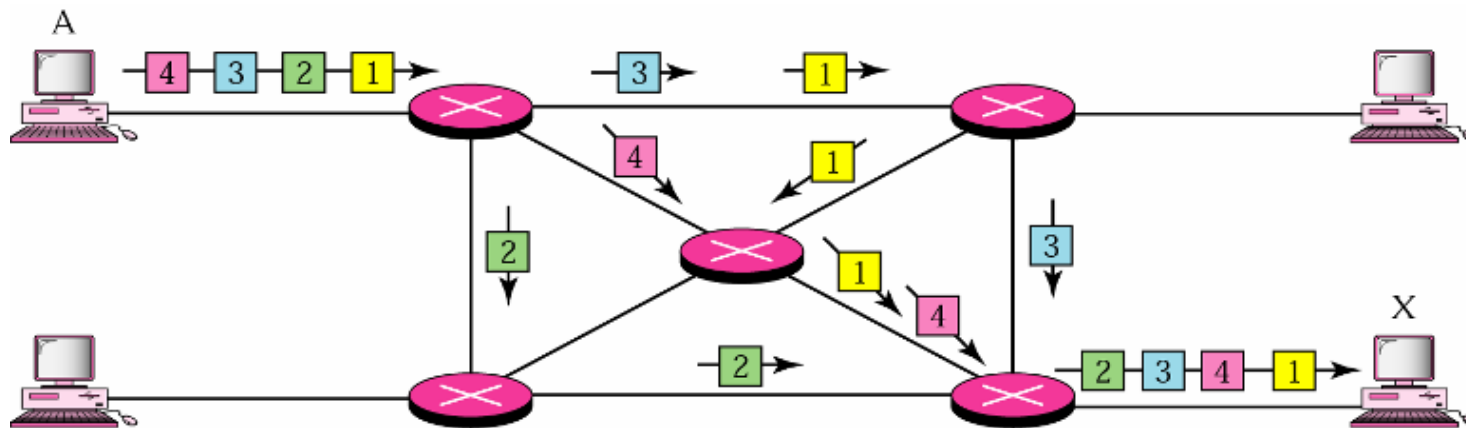


- The total length field defines the total length of the datagram including the header.

IP Datagram Header Fields

Length in bit	Name	Description
4	Version	Version of IP (4, 6)
4	HLEN	Length of the header
8	Service Type	Sender's preference for services e.g. low latency, etc
16	Total Length	Total length of datagram in bytes/octets
16	Identification	Unique identification for each datagram
4	Flags	Control fragmentation of datagrams
12	Fragment Offset	Used in fragmentation of datagrams
8	TTL	Time to live; decremented at each hop it passes
8	Protocol	Higher-layer protocol such as TCP, UDP, etc
16	Header Checksum	Checksum to verify correctness of header information
32	Source address	IP address of the sender
32	Dest. address	IP address of the destination

IP Service Model



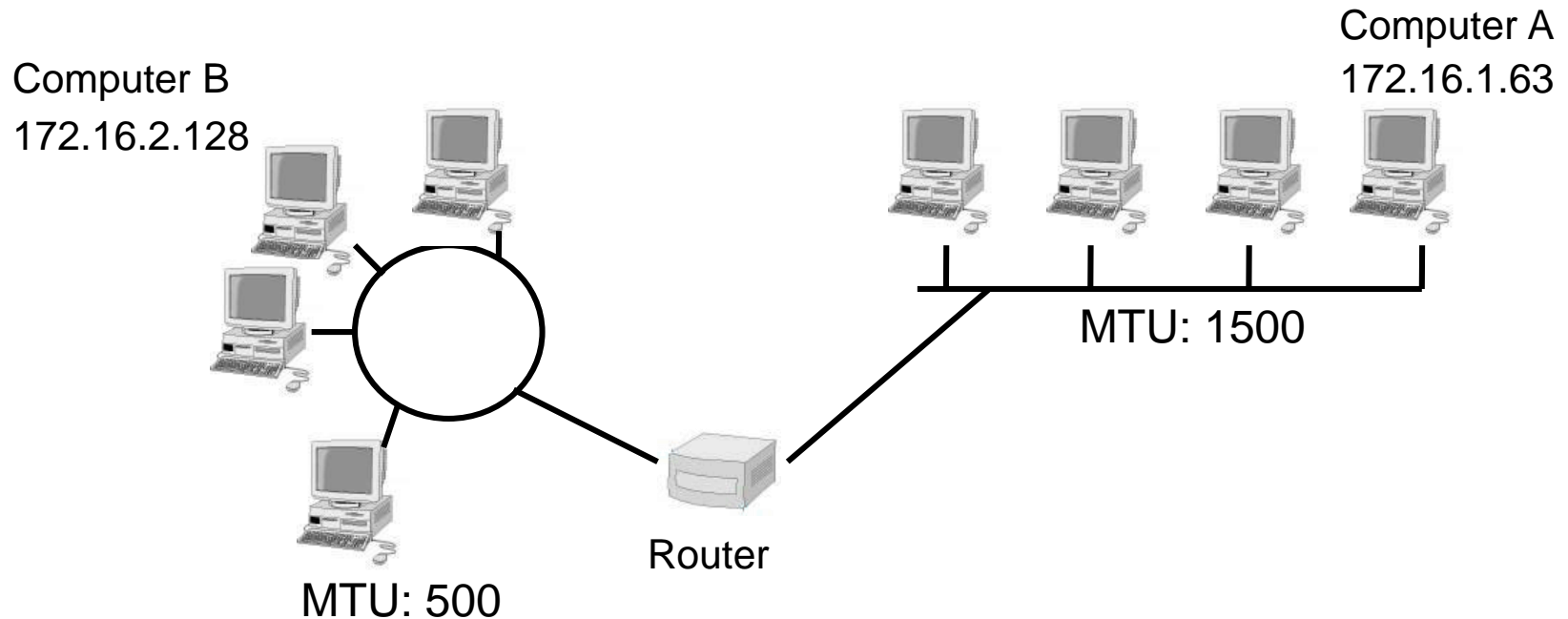
- Connection-less Communication
 - No state is kept about individual packets
- Order not guaranteed
- Best-effort delivery (unreliable service)
 - Packets may be lost
 - Packets may be delivered out of order
 - Duplicate copies of a packet may be delivered
 - Packets can be delayed for a long time

Maximum Transmission Unit (MTU)

- Maximum size of a data unit depends on underlying hardware architecture
- Maximum frame size determines *Maximum Transmission Unit (MTU)*

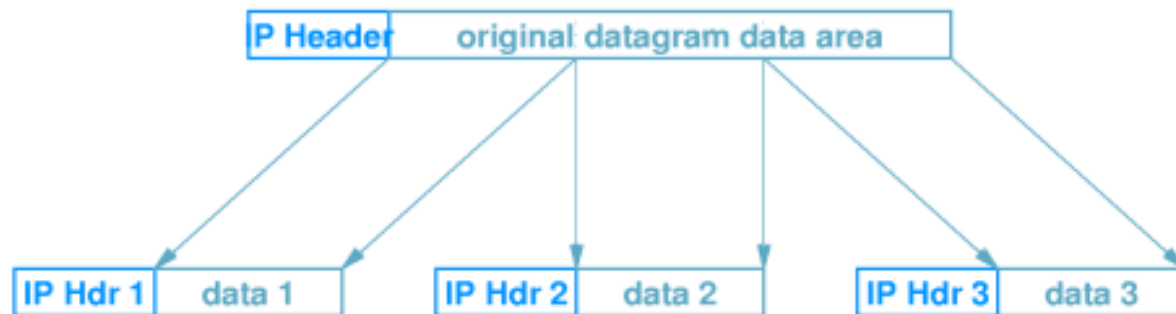
<i>Protocol</i>	<i>MTU</i>
Hyperchannel	65,535
Token Ring (16 Mbps)	17,914
Token Ring (4 Mbps)	4,464
FDDI	4,352
Ethernet	1,500
X.25	576
PPP	296

IP & Network Architectures



- IP datagrams may be larger than most MTUs of underlying network architectures
- Maximum frame size limits maximum size of an IP datagram for a network

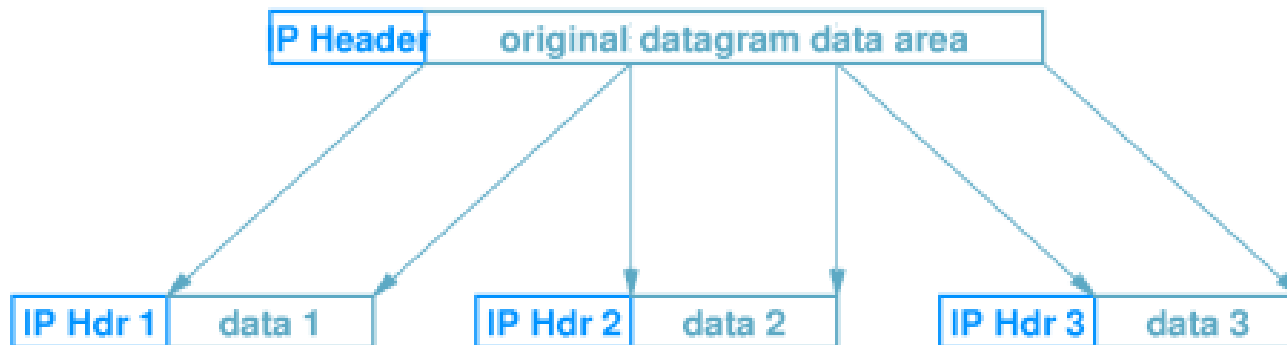
Fragmentation



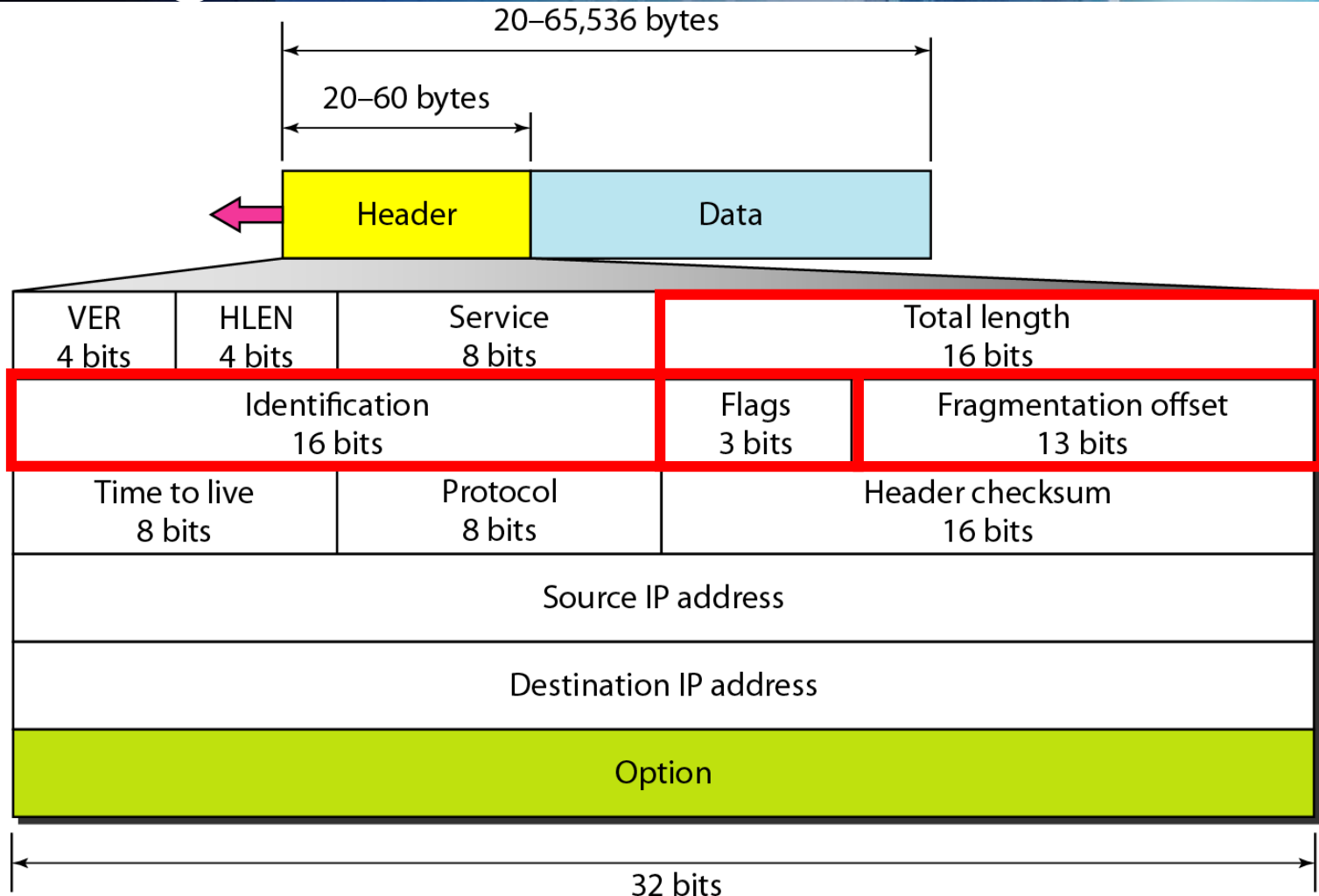
- Possible techniques:
 - Limit datagram size to smallest MTU of any network
 - Adjust datagram size as packet progresses through networks
 - Router detects datagram larger than network MTU and splits into pieces
- IP Strategy
 - Fragment when necessary (Datagram > MTU)
 - Try to avoid fragmentation at source host
 - Re-fragmentation is possible
 - **Delay reassembly until destination host**
 - Do not recover from lost fragments
 - If one fragment is lost all fragments are discarded

Fragmentation (details)

- Each fragment is an independent datagram
 - Includes all header fields
 - Bit in header indicates datagram is a fragment
 - Other fields have information for reconstructing original datagram
 - FRAGMENT OFFSET gives original location of fragment



Fragmentation & IPv4 Header

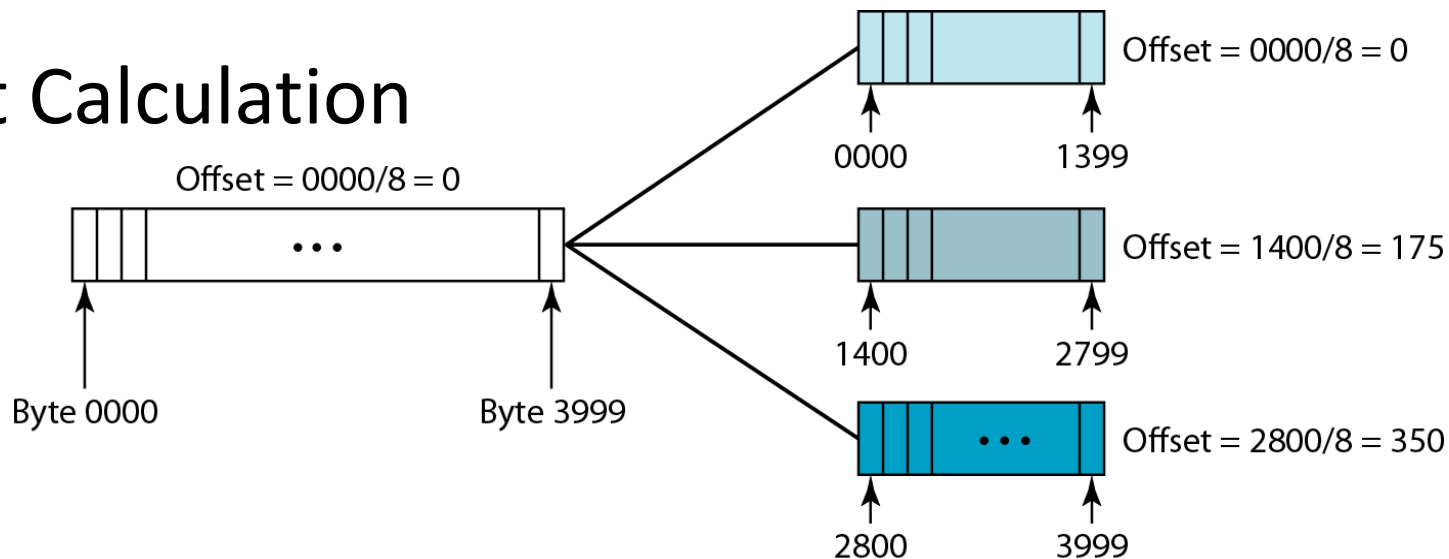


Header Fields

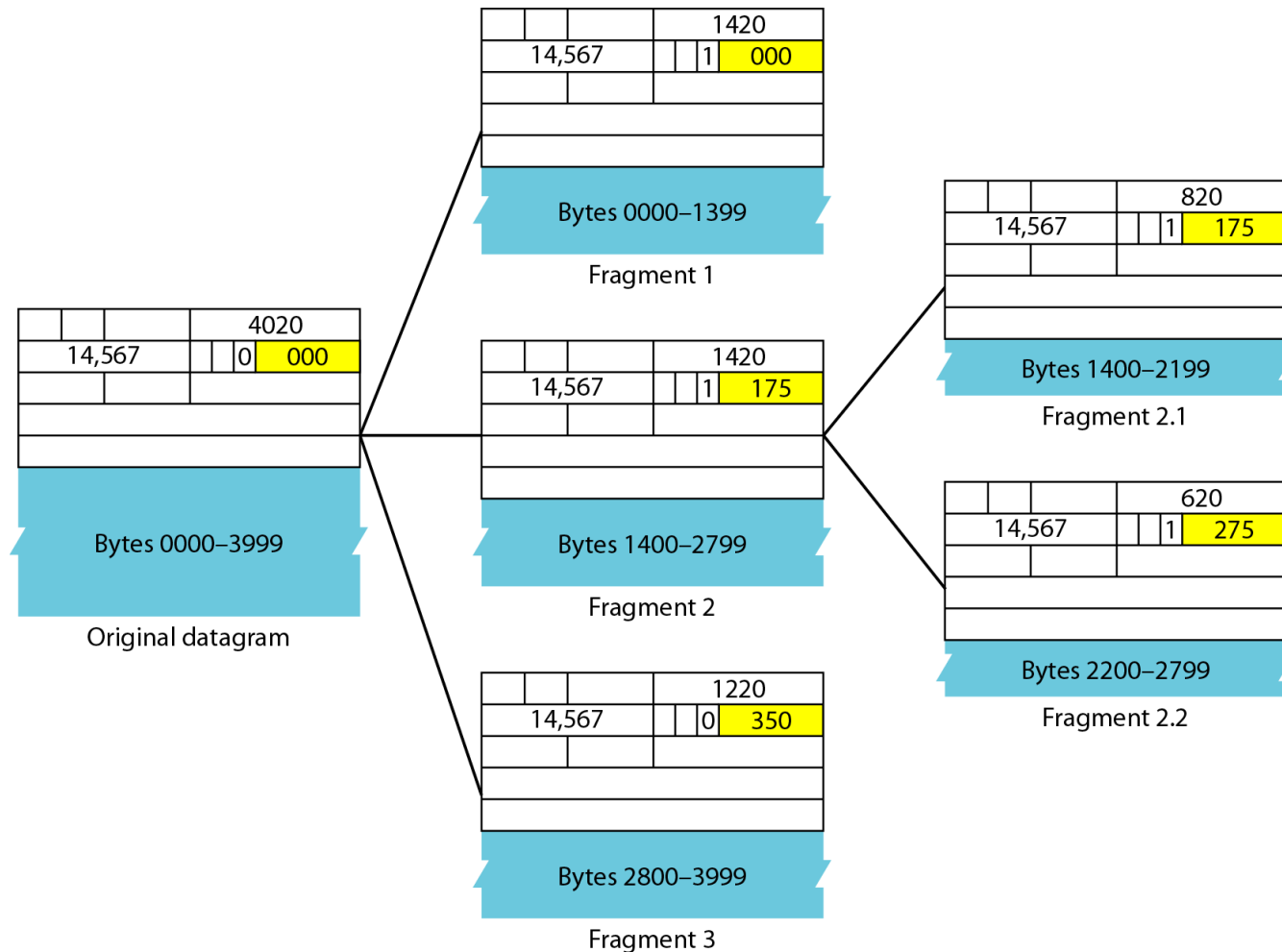
- “Do not fragment”-Request
- More Fragments



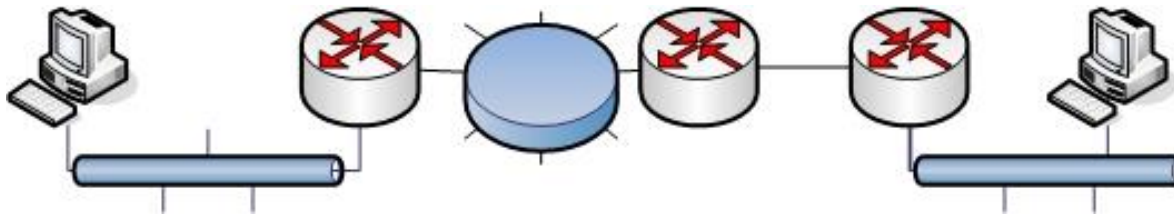
- Offset Calculation



Fragmentation Example I



Fragmentation Example II



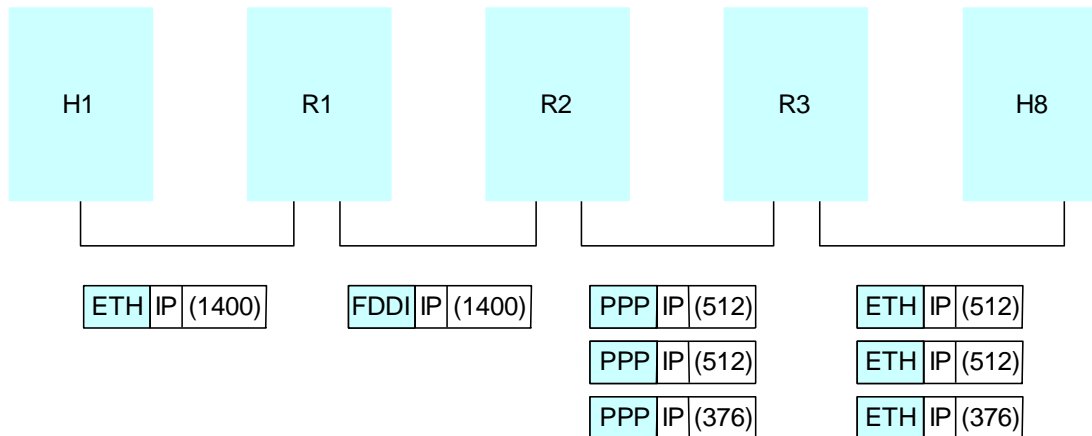
Start of header			
Ident= x		0	Offset= 0
Rest of header			
1400 data bytes			



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

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			



ETH IP (1400)

FDDI IP (1400)

PPP IP (512)

PPP IP (512)

PPP IP (376)

ETH IP (512)

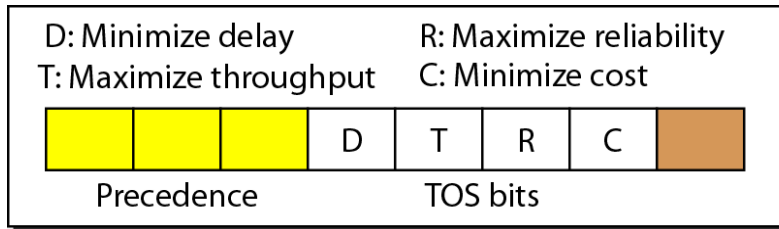
ETH IP (512)

ETH IP (376)

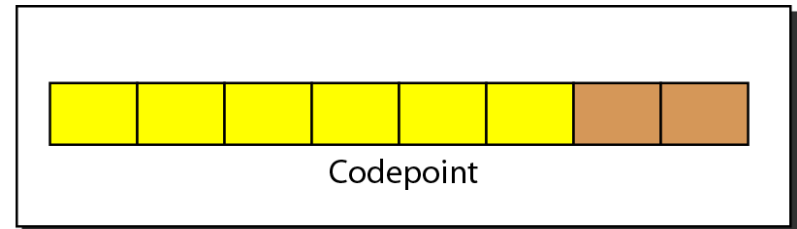
Fragment loss

- A fragment may be lost/dropped in transfer
- What happens to original datagram?
 - Destination drops entire original datagram
- How does destination identify lost fragment?
 - Sets timer with each fragment
 - If timer expires before all fragments arrive, fragment assumed lost
 - Datagram dropped
- Source is assumed to retransmit

Service types



Service type



Differentiated services

<i>TOS Bits</i>	<i>Description</i>
0000	Normal (default)
0001	Minimize cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimize delay

Protocols & TOS Bits

<i>Protocol</i>	<i>TOS Bits</i>	<i>Description</i>
ICMP	0000	Normal
BOOTP	0000	Normal
NNTP	0001	Minimize cost
IGP	0010	Maximize reliability
SNMP	0010	Maximize reliability
TELNET	1000	Minimize delay
FTP (data)	0100	Maximize throughput
FTP (control)	1000	Minimize delay
TFTP	1000	Minimize delay
SMTP (command)	1000	Minimize delay
SMTP (data)	0100	Maximize throughput
DNS (UDP query)	1000	Minimize delay
DNS (TCP query)	0000	Normal
DNS (zone)	0100	Maximize throughput

Checksum in IPv4 Header

- Calculation omits:
 - Service field
 - Fragment fields and offset
 - Checksum field

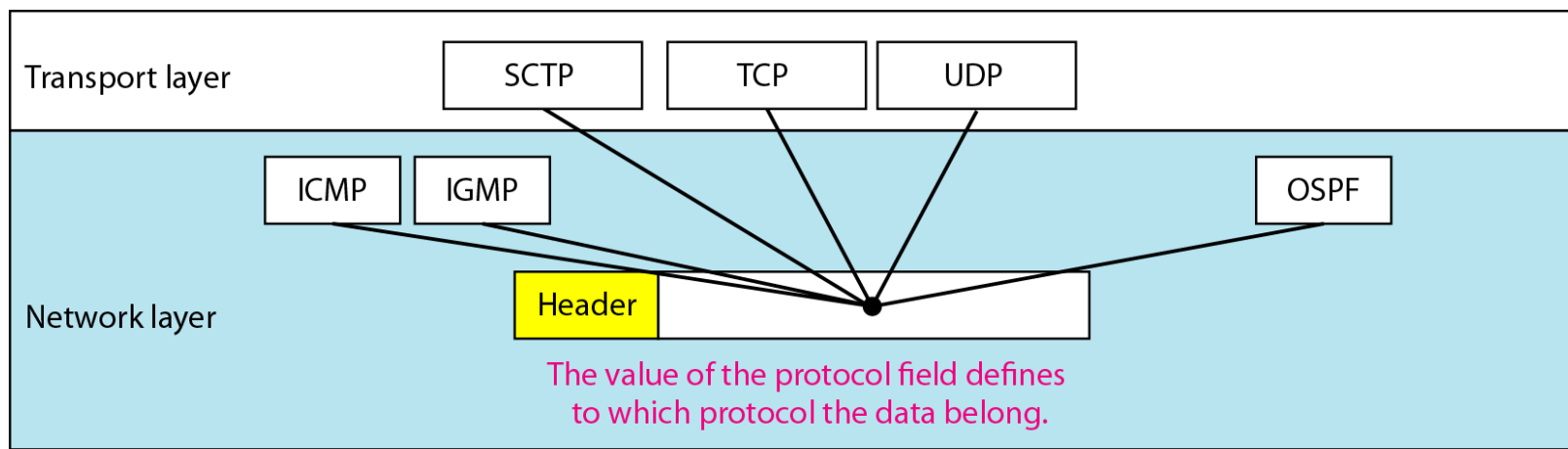
4	5	0	28	
1		0	0	
4	17	0		
10.12.14.5				
12.6.7.9				

4, 5, and 0	→	4	5	0	0
28	→	0	0	1	C
1	→	0	0	0	1
0 and 0	→	0	0	0	0
4 and 17	→	0	4	1	1
0	→	0	0	0	0
10.12	→	0	A	0	C
14.5	→	0	E	0	5
12.6	→	0	C	0	6
7.9	→	0	7	0	9
Sum	→	7	4	4	E
Checksum	→	8	B	B	1

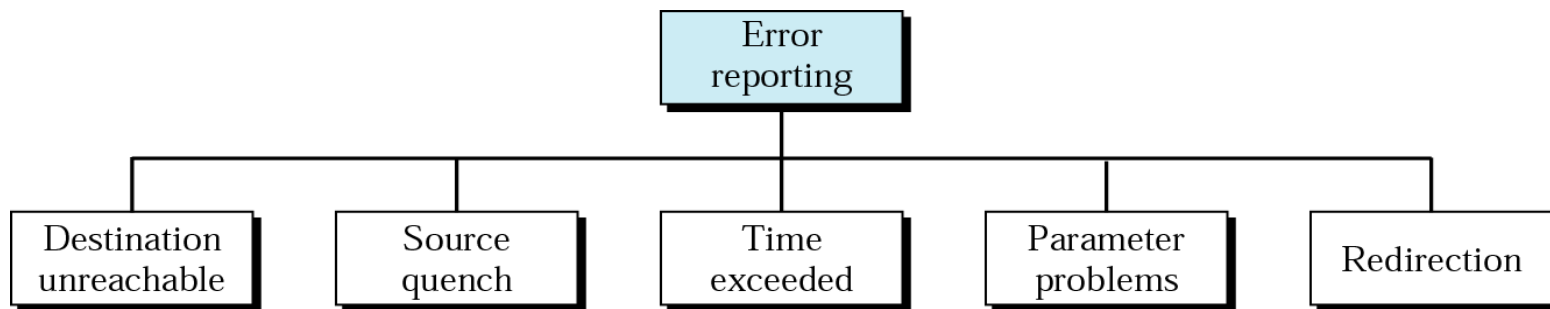
Protocol Field

- Specifies next protocol in stack

<i>Value</i>	<i>Protocol</i>
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF

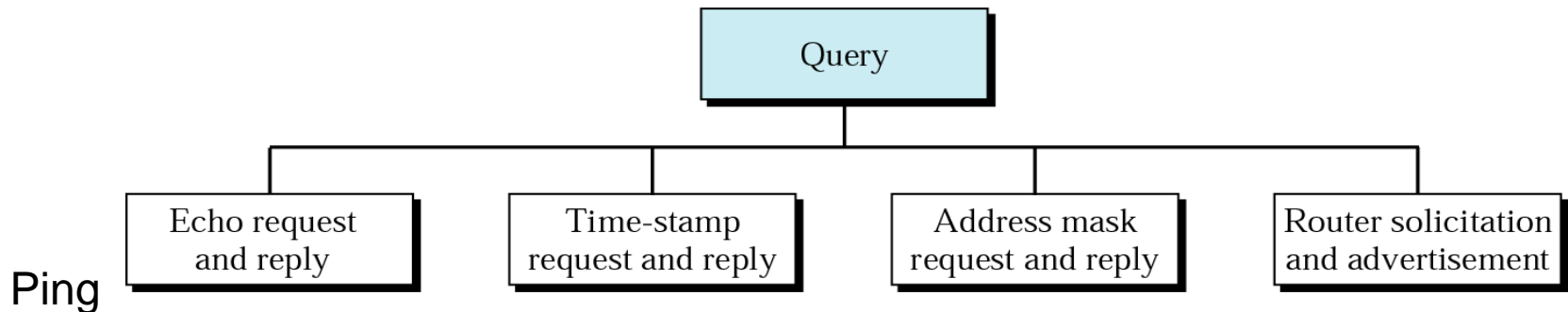


Internet Control Message Protocol



- Destination unreachable
- Source quench
 - Mechanism for destination and intermediate nodes to limit traffic from source
- Time exceeded
 - Send when datagram discarded due TTL value of 0
- Parameter problems
 - Send when datagram discarded due to parameter ambiguity
- Redirection
 - Send from router to update routing table of source

ICMP Queries



uses the ICMP protocol's mandatory ECHO_REQUEST datagram to elicit an ICMP ECHO_RESPONSE from a host or gateway. ECHO_REQUEST datagrams have an IP and ICMP header, followed by a struct timeval and then an arbitrary number of bytes used to fill out the packet.

```
$ ping www.tcd.ie
```

```
PING dux6.tcd.ie (134.226.1.61): 56 data bytes
```

```
64 bytes from 134.226.1.61: icmp_seq=0 ttl=64 time=0.781 ms
```

```
64 bytes from 134.226.1.61: icmp_seq=1 ttl=64 time=0.466 ms
```

```
64 bytes from 134.226.1.61: icmp_seq=2 ttl=64 time=0.490 ms
```

```
4 packets transmitted, 4 packets received, 0% packet loss
```

```
round-trip min/avg/max/stddev = 0.457/0.548/0.781/0.135 ms
```

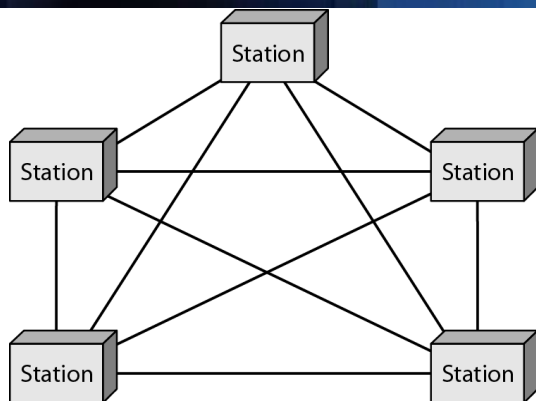

Summary: Internet Protocol

- IP Service Model
 - Connection-less, no order guaranteed
- IP Header
 - 20 bytes + options
- Fragmentation
 - Datagrams split into fragments to fit MTUs
 - Only re-assembled at destination
- Internet Control Message Protocol (ICMP)
 - Error Reporting e.g. source quench
 - Querying e.g. Ping

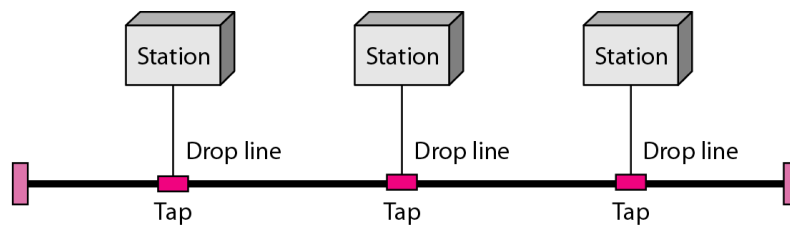


That's all
folks

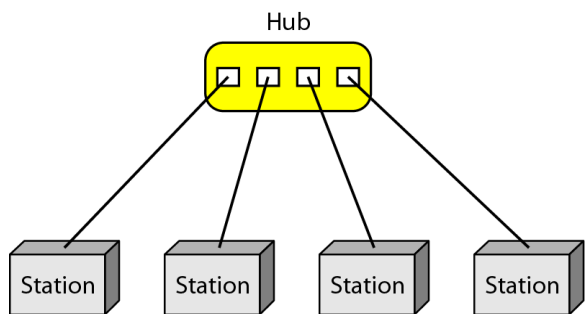
Topologies



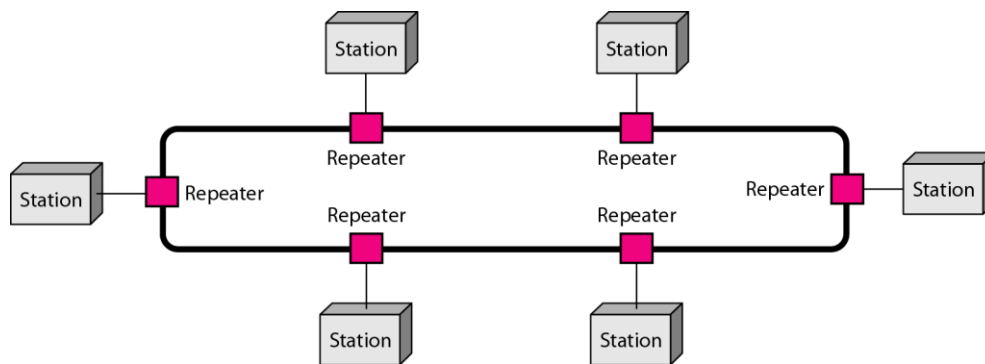
Mesh



Bus

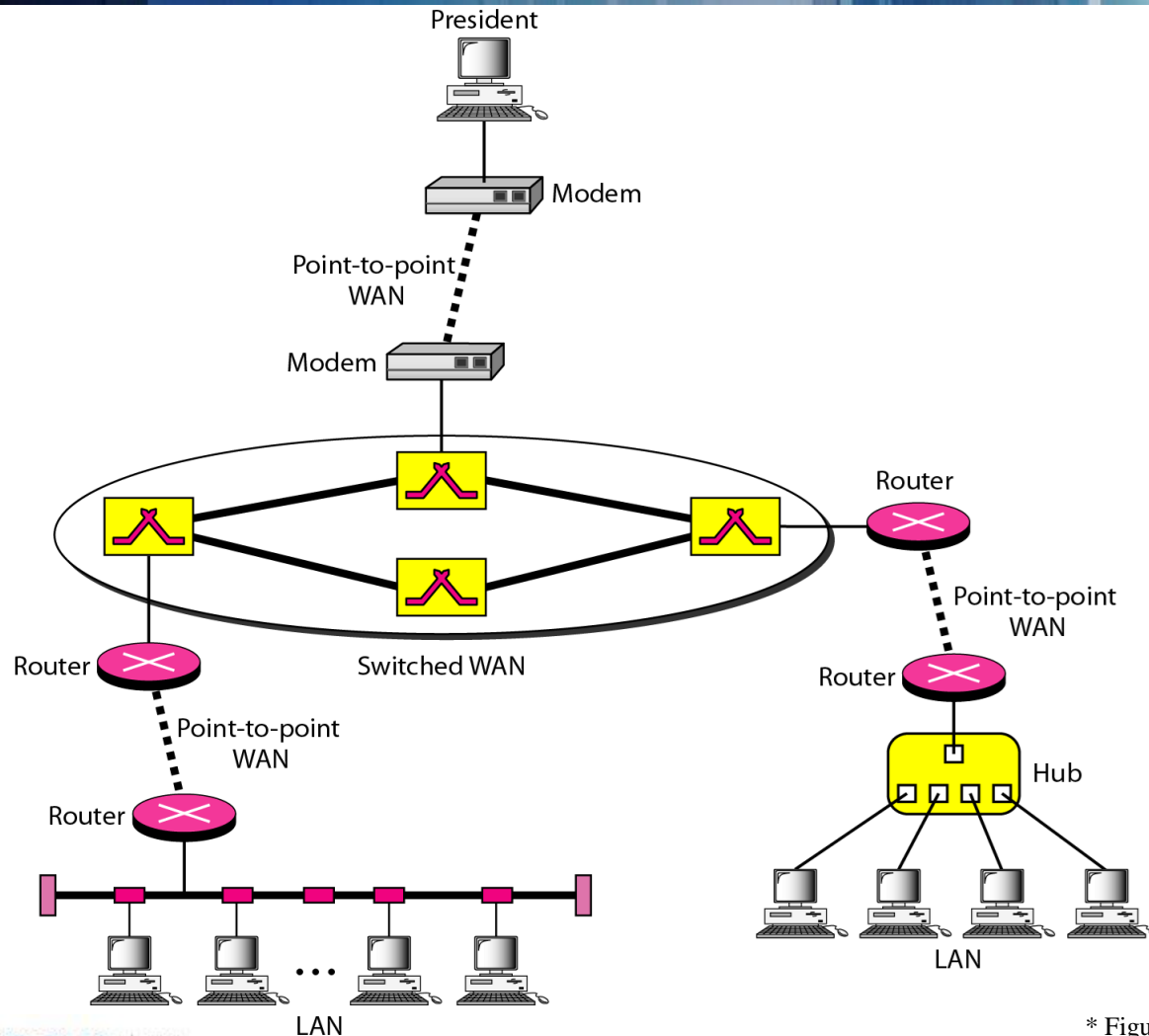


Star



Ring

Combined Networks



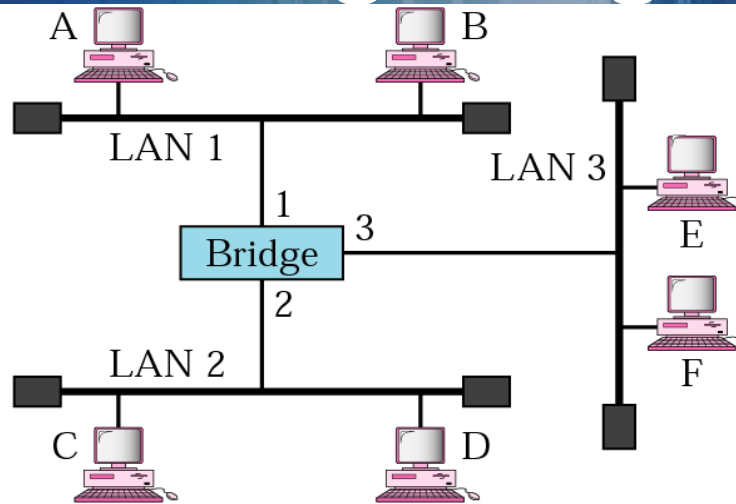
CS2031

Telecommunications II

Learning Bridges



Learning Bridges



Address	Port

a. Original

Address	Port
A	1

b. After A sends a frame to D

Address	Port
A	1
E	3

c. After E sends a frame to A

Address	Port
A	1
E	3
B	1

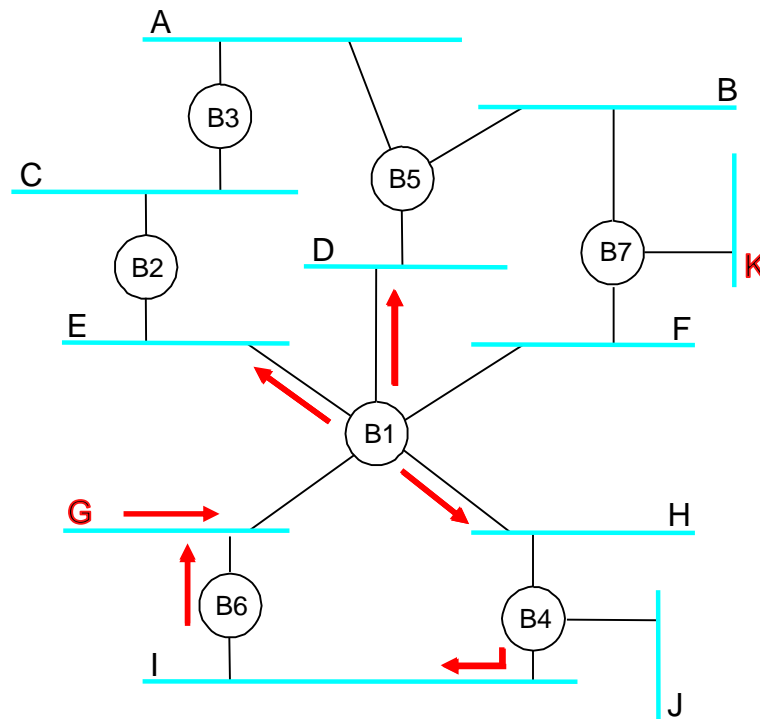
d. After B sends a frame to C

- Initially bridge forwards frames on all segments except incoming port
- Learns addresses from frames that pass through

Problem with Learning Bridges

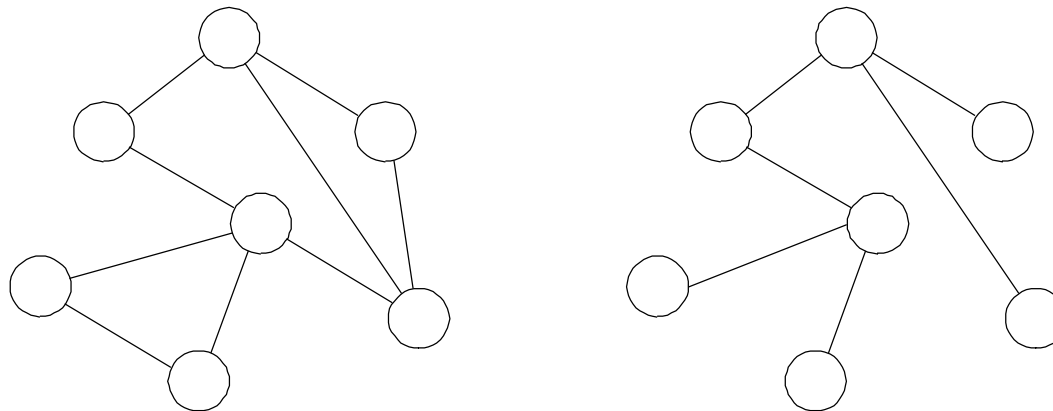
- Problem with learning bridges is loops

$G \rightarrow K$



Spanning Tree Algorithm

- Bridges run a distributed spanning tree algorithm
 - select which bridges actively forward
 - developed by Radia Perlman
 - now IEEE 802.1 specification
- Spanning tree includes all nodes
- Spanning tree only includes edges on the shortest path from the root to any other given node

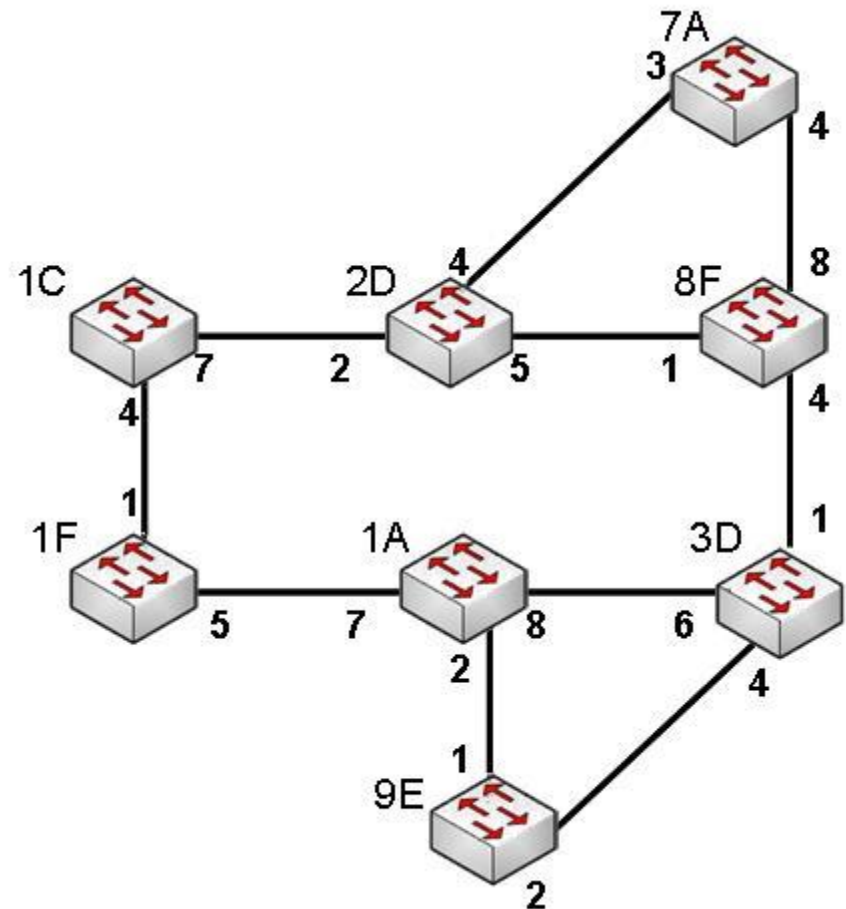


Spanning Tree Algorithm

1. Bridge with smallest ID is selected as root bridge
2. Mark port on each bridge with least-cost to root bridge as root port
3. Select designated bridge for each LAN that has root port with least-cost to root bridge – if two bridges with same cost select bridge with lowest ID
4. Mark root ports and designated ports as forwarding ports; other ports as blocking ports

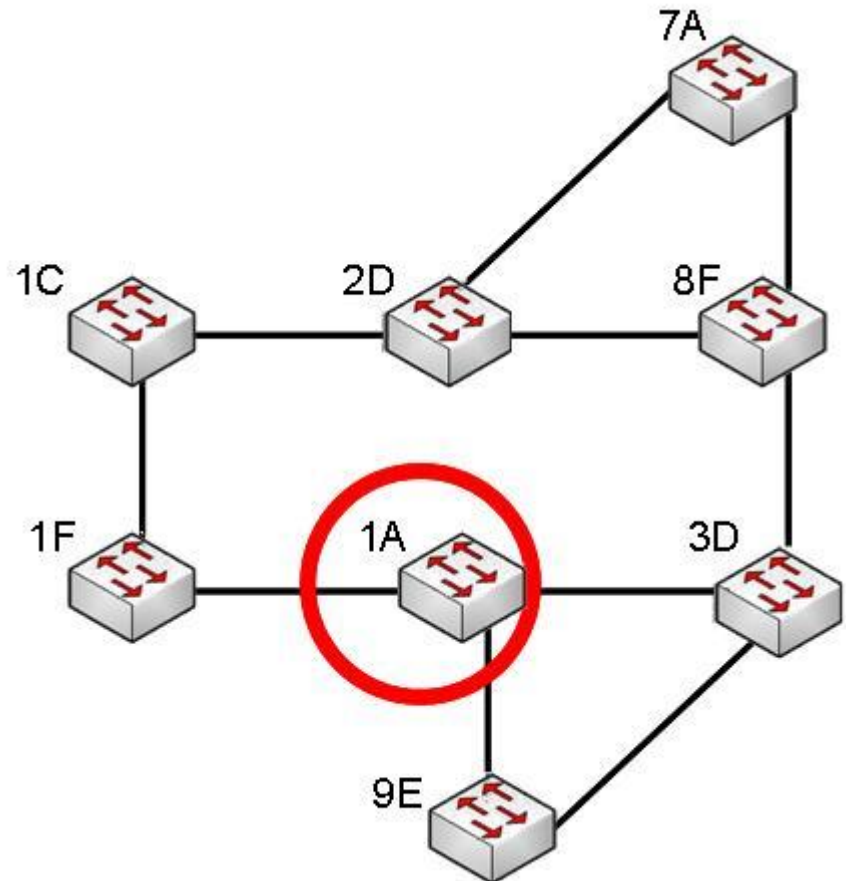
Spanning Tree Algorithm

1. Bridge with smallest ID is selected as root bridge
2. Mark port on each bridge with least-cost to root bridge as root port
3. Select designated bridge for each LAN that has root port with least-cost to root bridge – if two bridges with same cost select bridge with lowest ID
4. Mark root ports and designated ports as forwarding ports; other ports as blocking ports



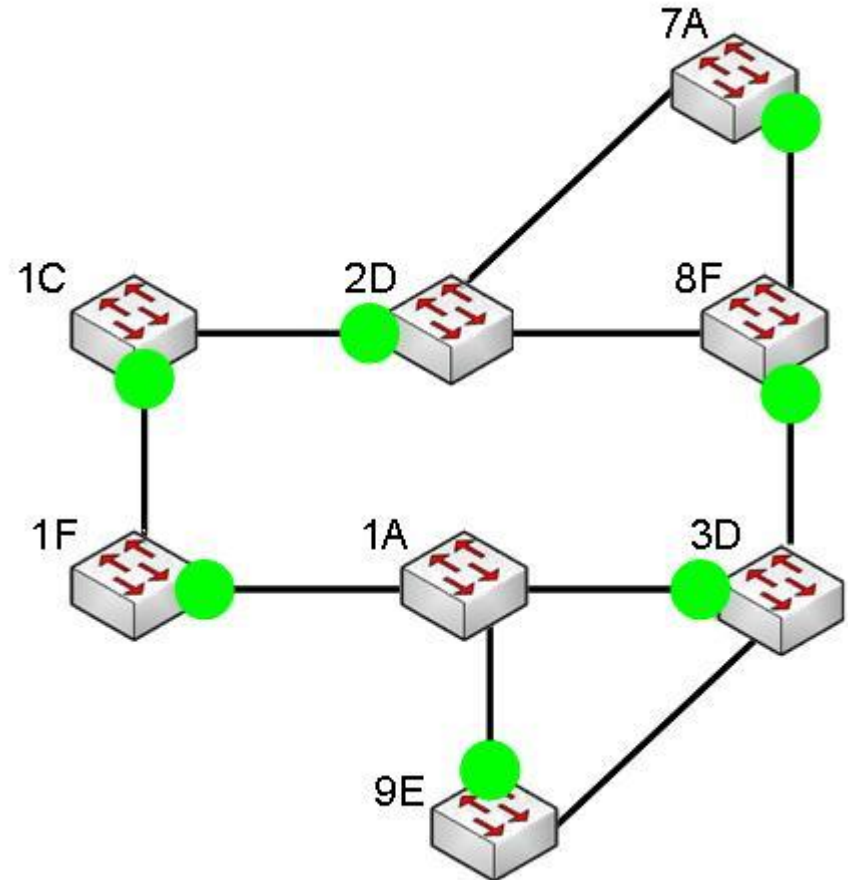
Spanning Tree Algorithm

1. Bridge with smallest ID is selected as root bridge



Spanning Tree Algorithm

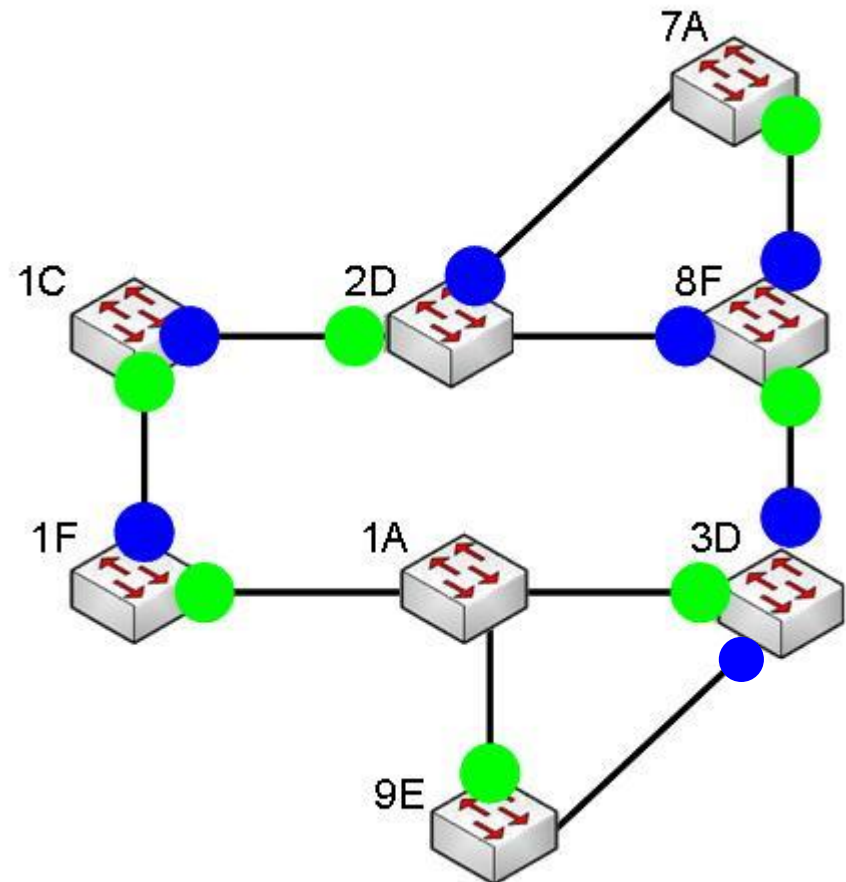
1. Bridge with smallest ID is selected as root bridge
2. Mark port on each bridge with least-cost to root bridge as root port



● Root Port

Spanning Tree Algorithm

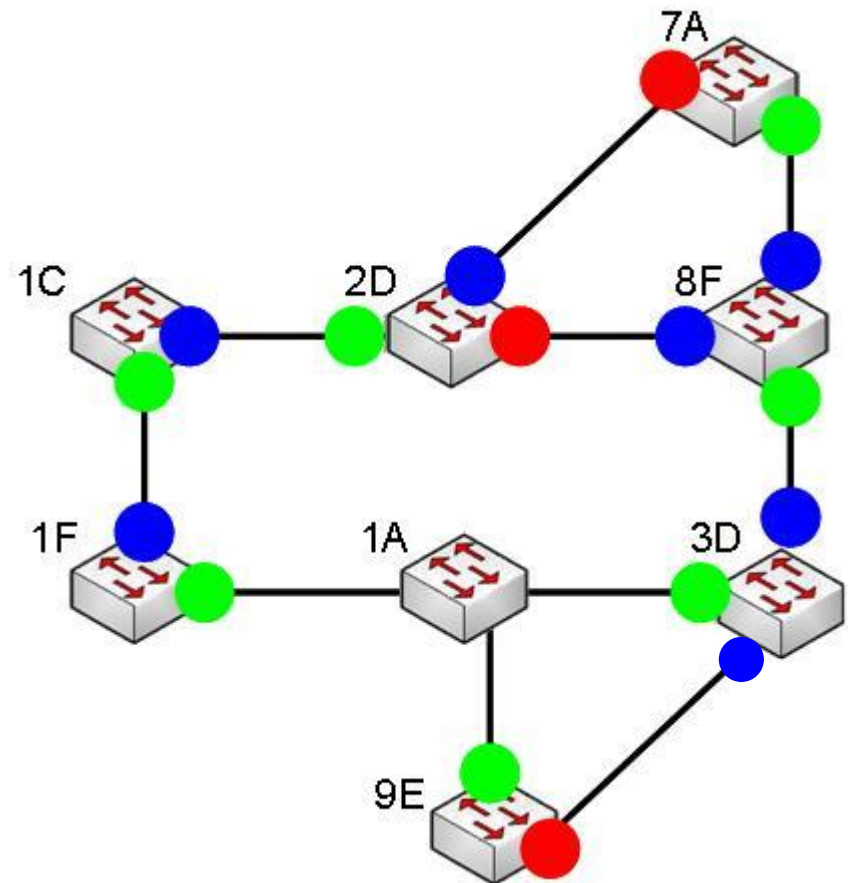
1. Bridge with smallest ID is selected as root bridge
2. Mark port on each bridge with least-cost to root bridge as root port
3. Select designated bridge for each LAN that has root port with least-cost to root bridge – if two bridges with same cost select bridge with lowest ID



● Root Port ● Port to LAN

Spanning Tree Algorithm

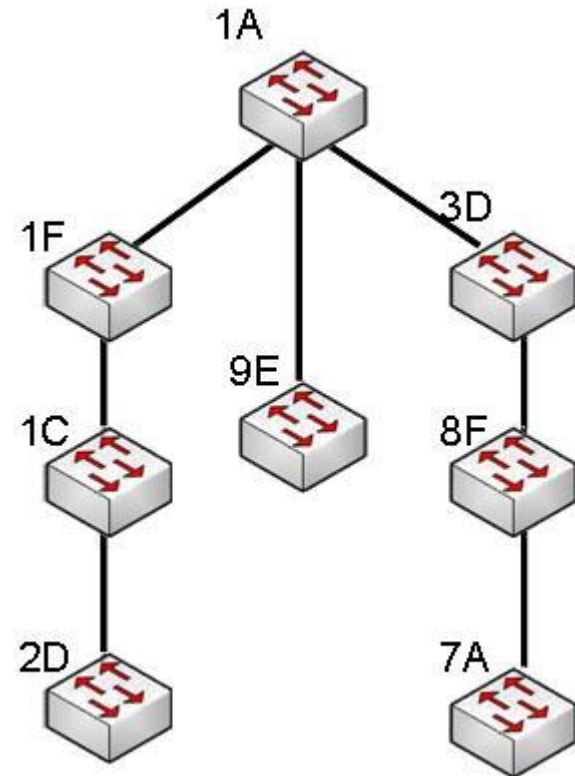
1. Bridge with smallest ID is selected as root bridge
2. Mark port on each bridge with least-cost to root bridge as root port
3. Select designated bridge for each LAN that has root port with least-cost to root bridge – if two bridges with same cost select bridge with lowest ID
4. Mark root ports and designated ports as forwarding ports; other ports as blocking ports



● Root Port ● Port to LAN ● Blocked Port

Spanning Tree Algorithm

1. Bridge with smallest ID is selected as root bridge
2. Mark port on each bridge with least-cost to root bridge as root port
3. Select designated bridge for each LAN that has root port with least-cost to root bridge – if two bridges with same cost select bridge with lowest ID
4. Mark root ports and designated ports as forwarding ports; other ports as blocking ports

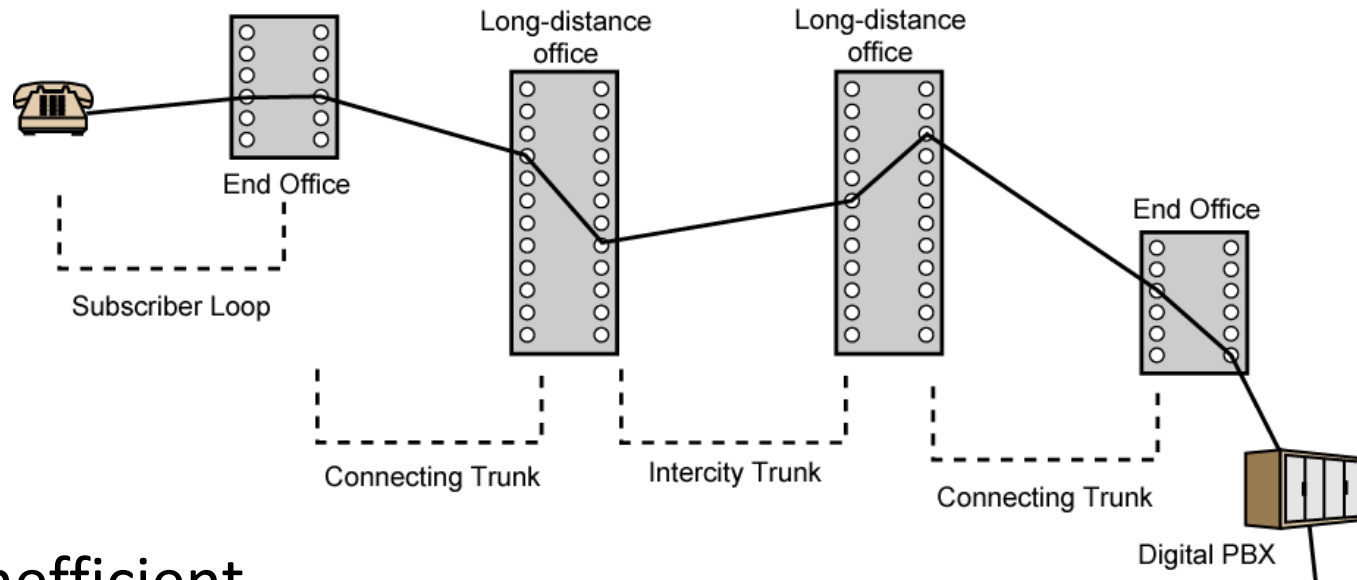


CS2031

Telecommunications II

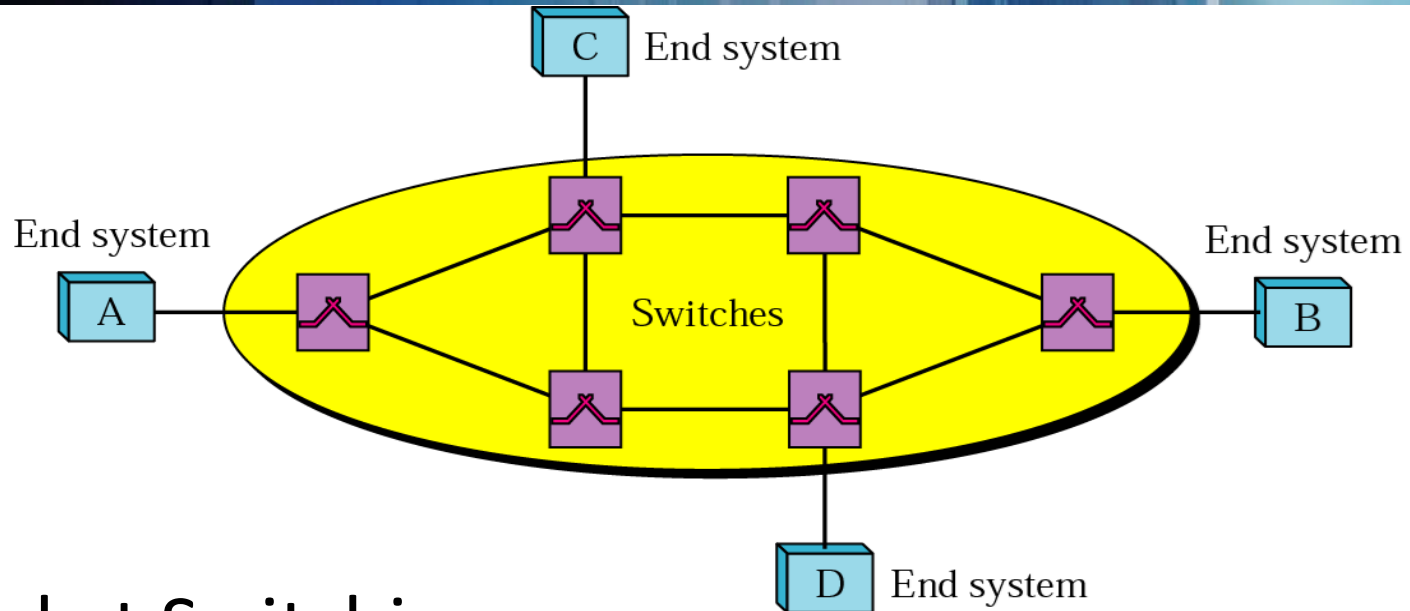
Circuit Switching & Packet Switching

Public Circuit Switched Network



- Inefficient
 - Channel capacity dedicated for duration of connection
 - If no data, capacity wasted
- Set up of connection takes time
- Once connected, transfer is transparent

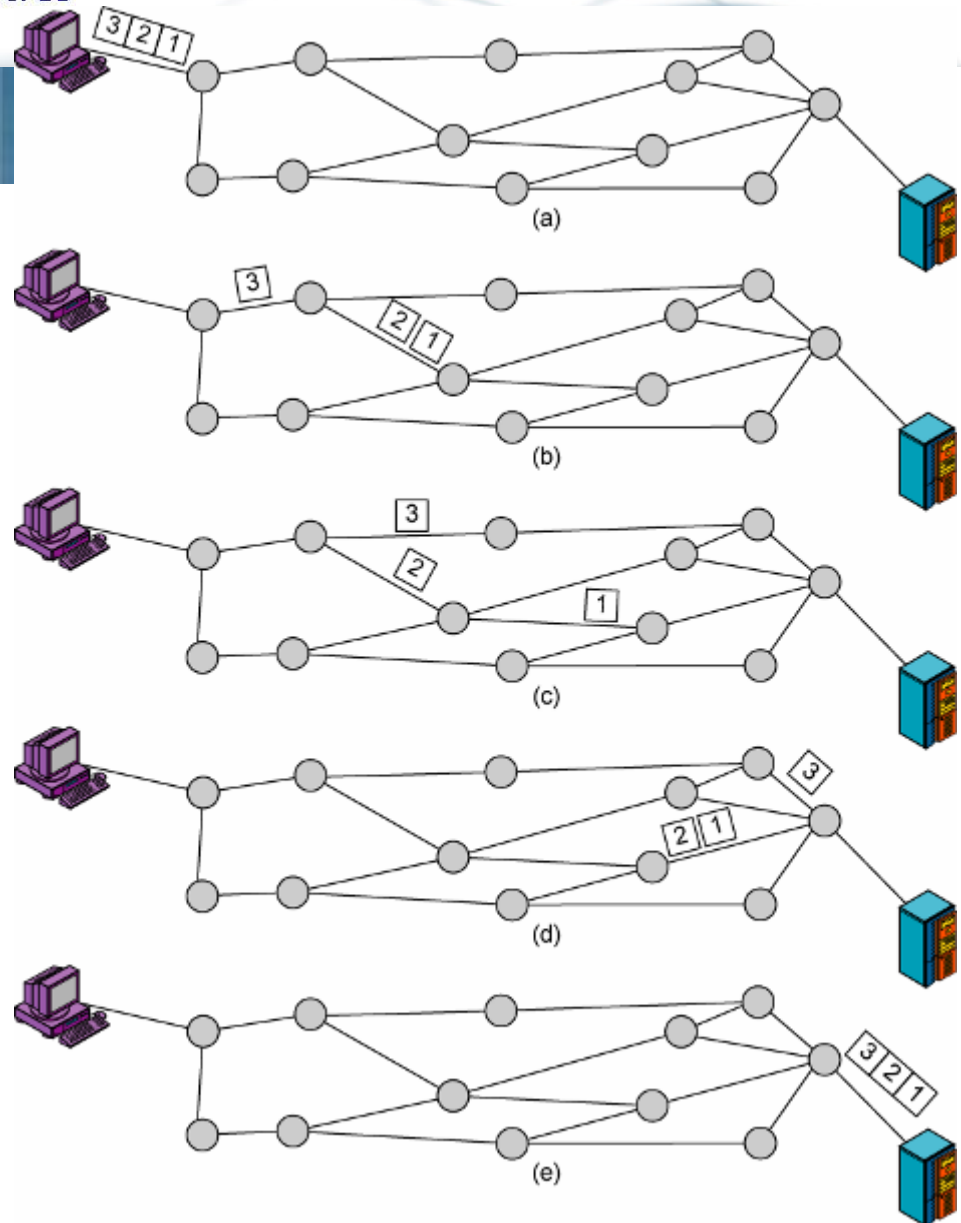
Switched Networks



- Packet Switching:
 - Switching decisions are made on individual packets
- Virtual Circuit Switching:
 - A circuit is setup explicitly for individual connections

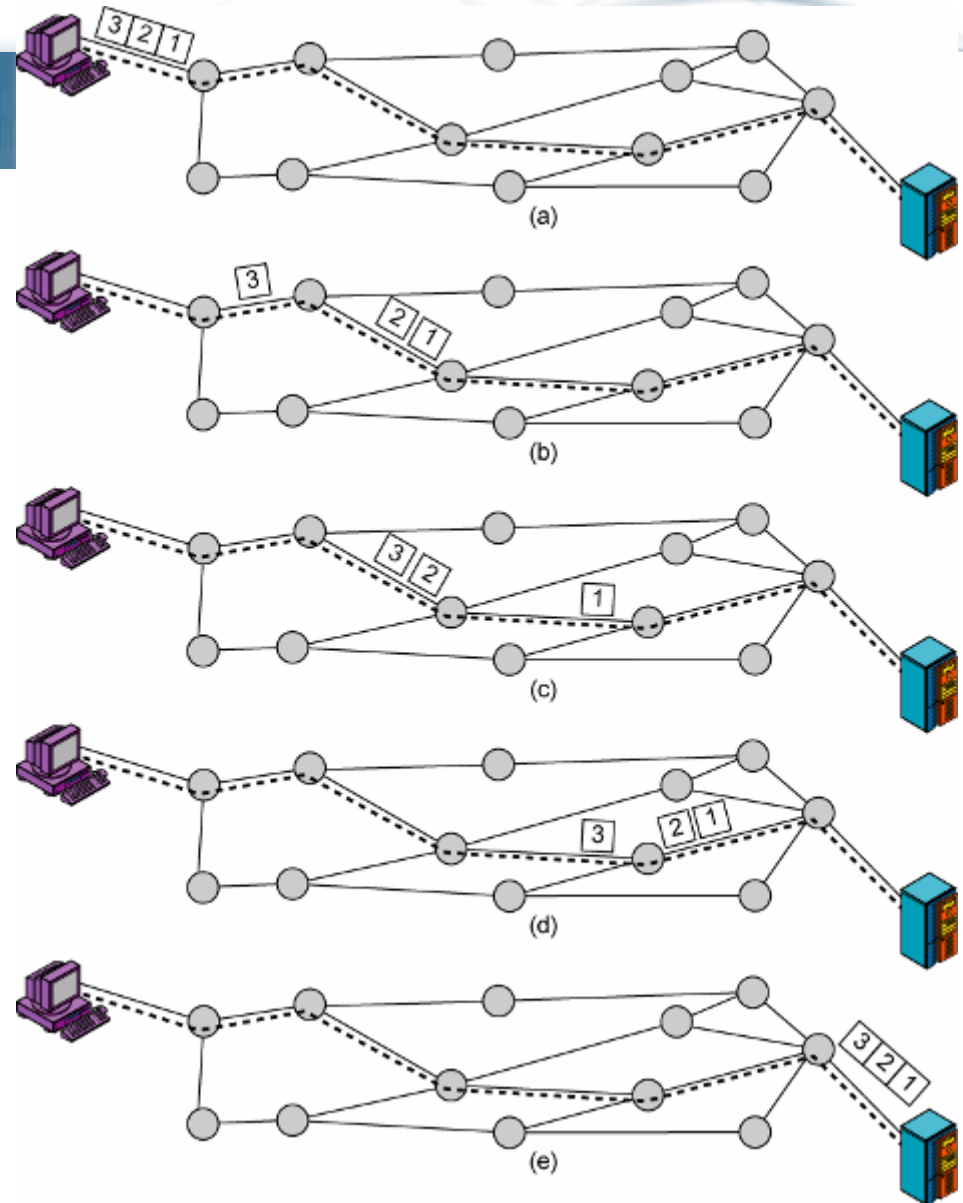
Packet switching

- Frames can be transferred over different paths in the network
- Reliability is generally delegated to higher layers
- Order is not necessarily maintained



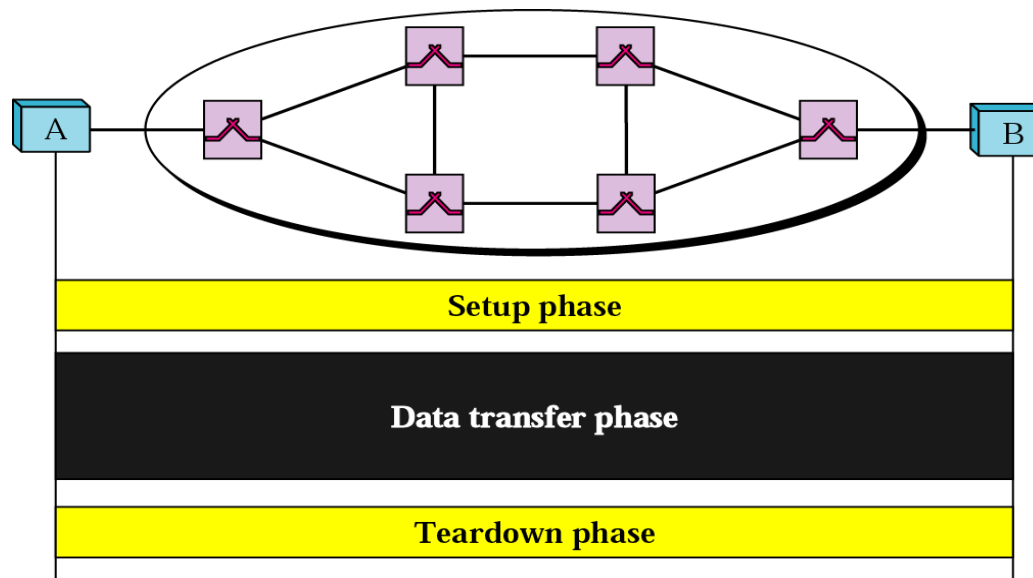
Virtual Circuits

- Connection-oriented communication
- Connection is established before communication
- The network maintains order

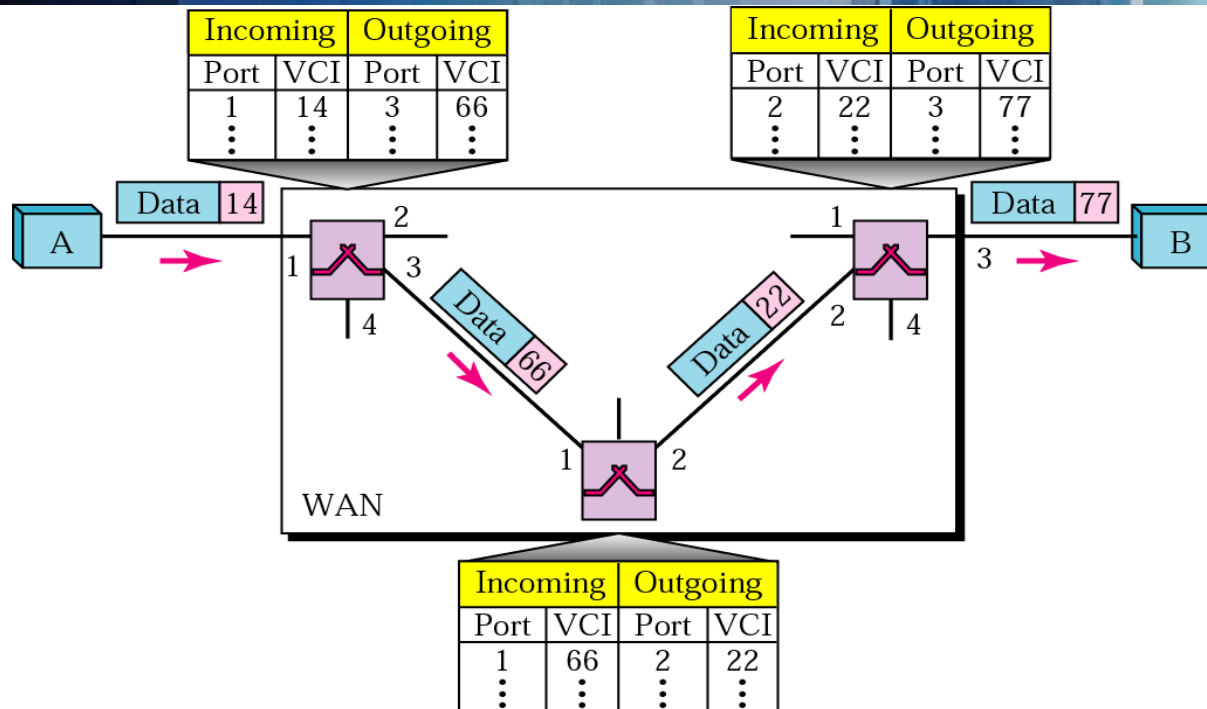


Phases for Virtual Circuit Communication

- Three phases
 - Connection setup
 - Data Transfer
 - Connection termination



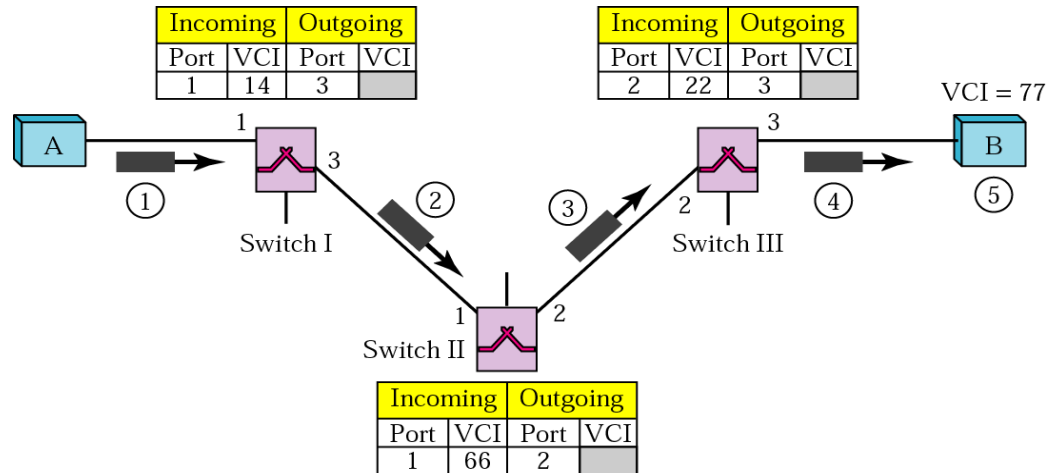
Virtual Circuit Switching



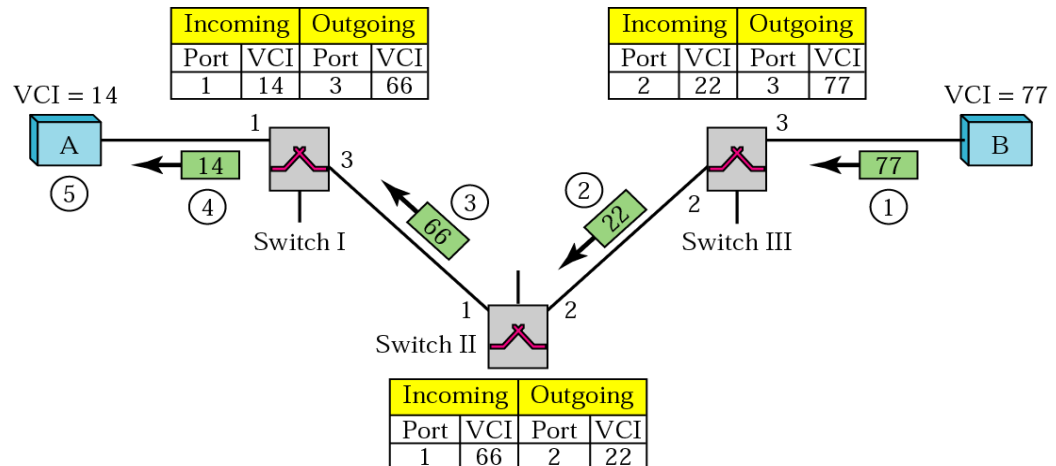
- Every switch maintains a table
 - For duration of communication one entry for incoming and outgoing line
 - Incoming and outgoing line are identified by port number and virtual circuit identifier

Setup Phase

- Setup request

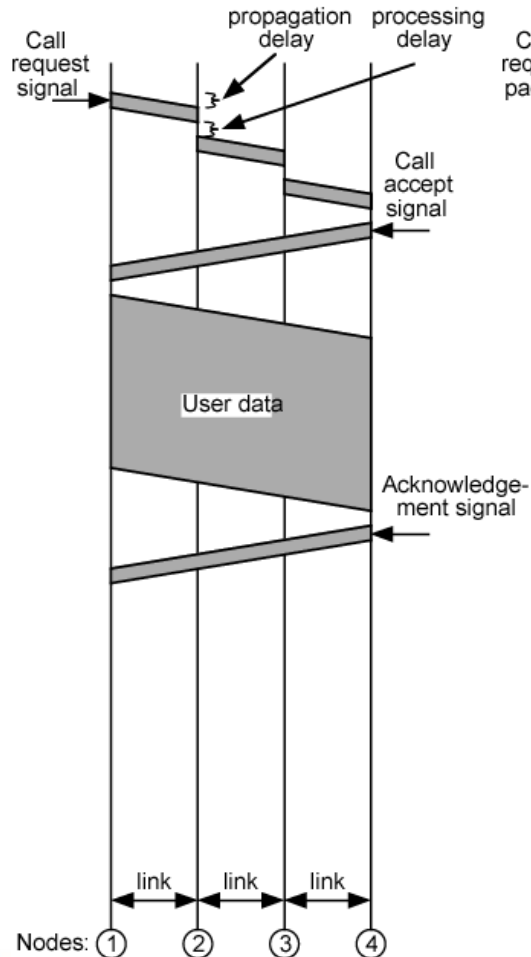


- Setup acknowl.

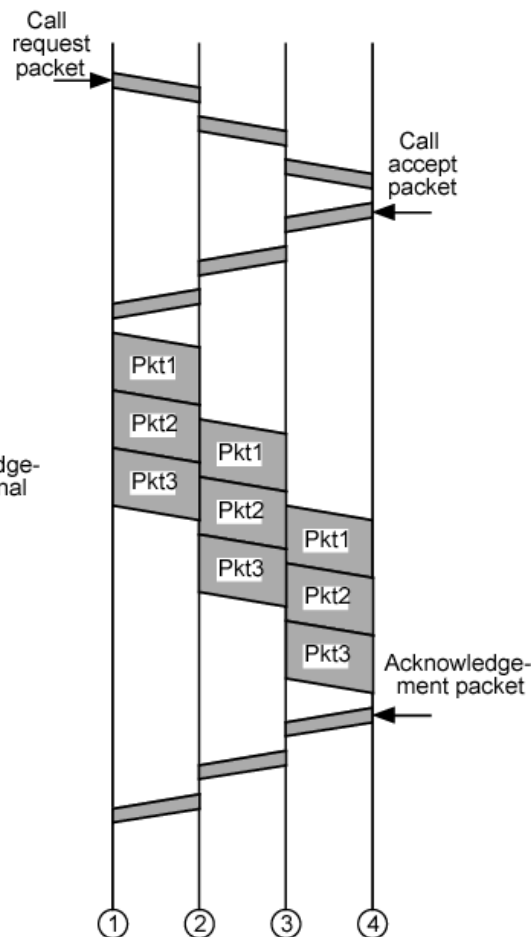


Event Timing

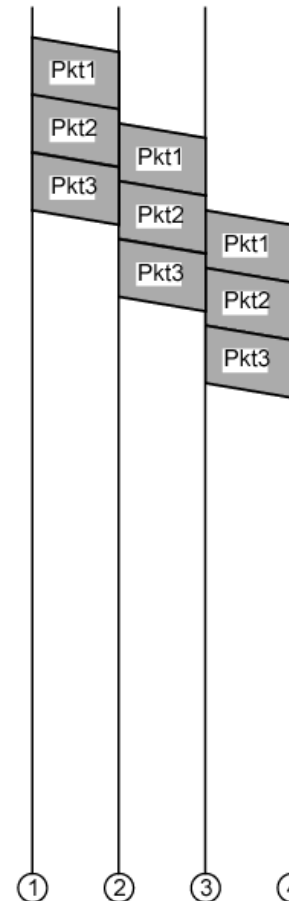
(a) Circuit switching



(b) Virtual circuit packet switching



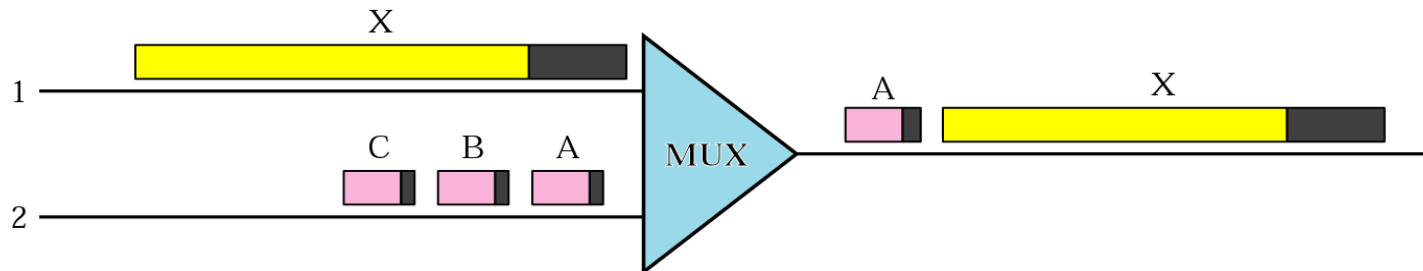
(c) Datagram packet switching



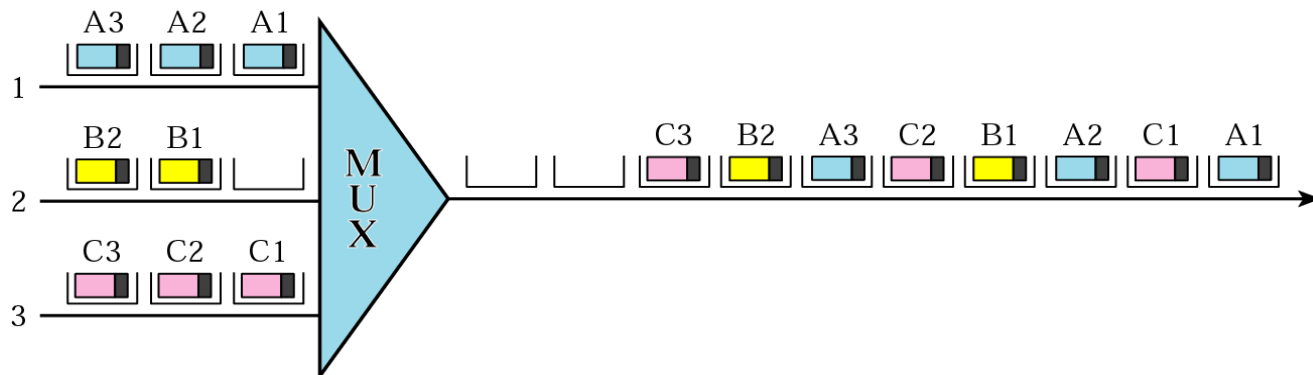
Asynchronous Transfer Mode (ATM)

- Example of virtual circuit switching
 - Cell-Switching
- Similarities between ATM and packet switching
 - Transfer of data in discrete chunks
 - Multiple logical connections over single physical interface
- In ATM flow on each logical connection is in fixed sized packets called cells
- Minimal error and flow control
 - Reduced overhead

Motivation for ATM

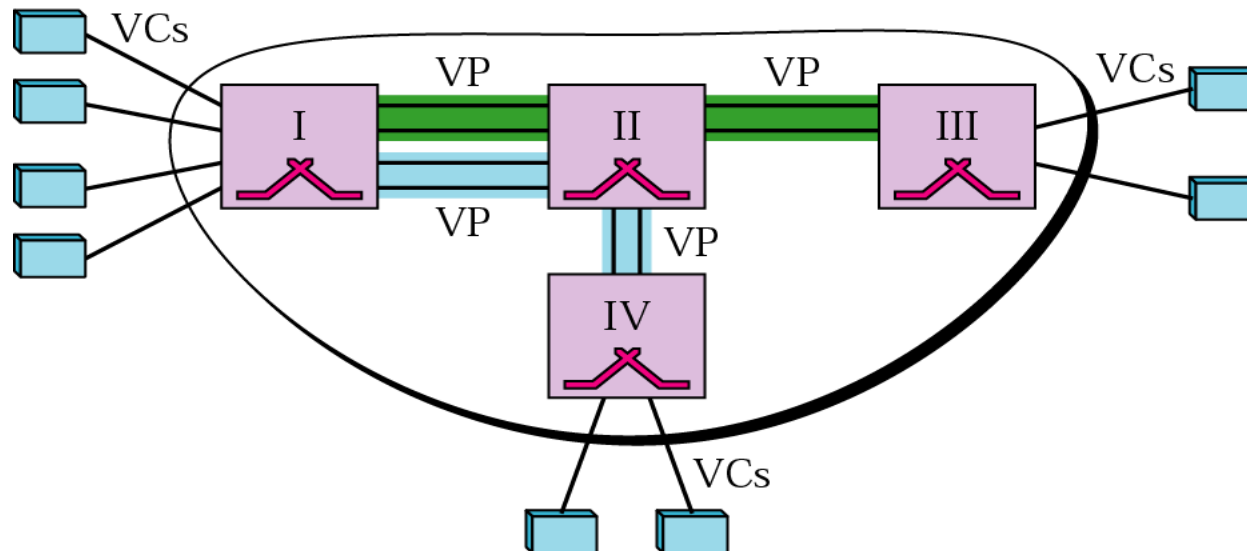


- Frames at a switch may be handled in any order and occupy switch for underspecified time



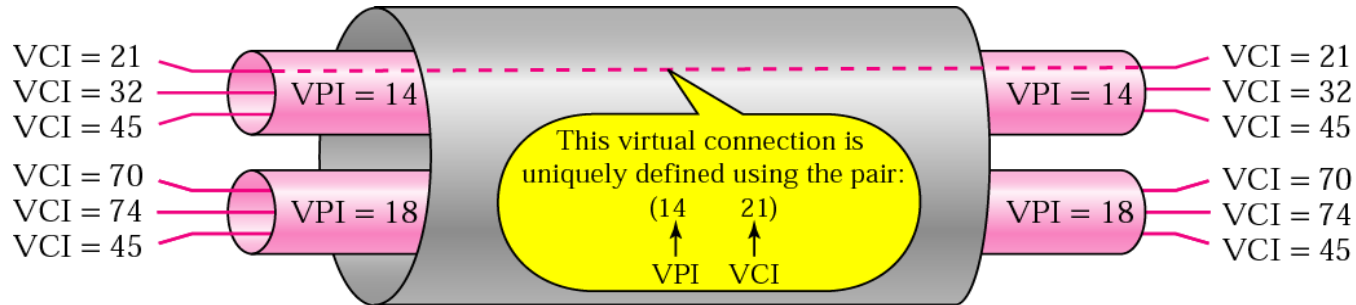
- Small, fixed-size frames allow simple, fast switches

Virtual Circuits / Virtual Paths

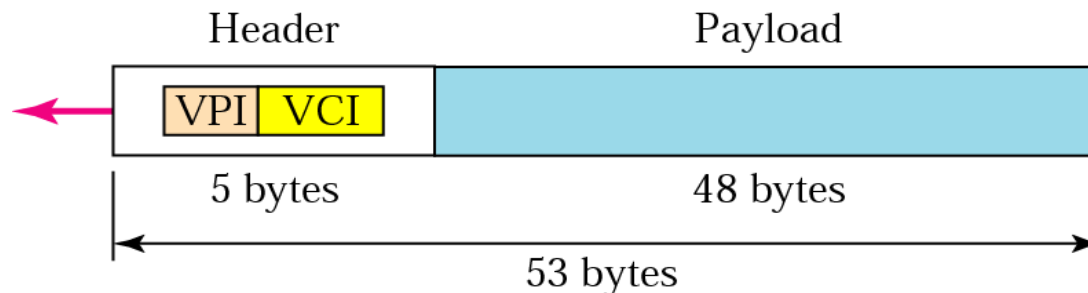


- Virtual circuits are collected into virtual paths

ATM Packet

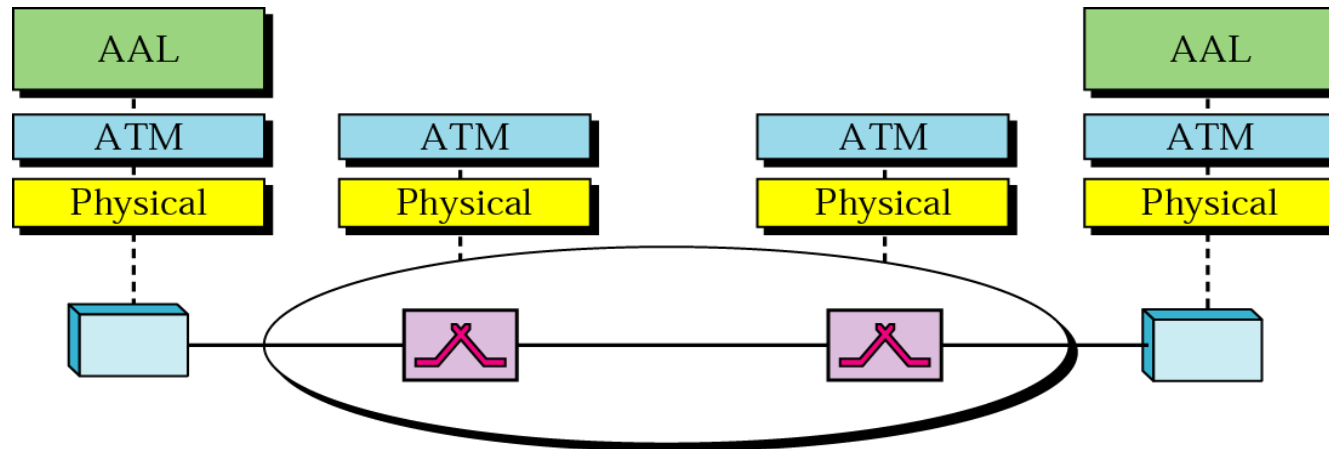


- Connection is specified by combination of Virtual Path ID and Virtual Circuit ID



- Every frame is exactly 53 bytes

Application Adaptation Layer (AAL)



- ATM defined a number of AALs for various purposes (each has its own header format):
 - AAL1: Constant bit rate e.g. multimedia
 - AAL2: Variable-data-rate
 - AAL3/4: Connection-oriented data services
 - Sequencing and Error Control
 - AAL5: Simple and efficient adaptation layer (SEAL)

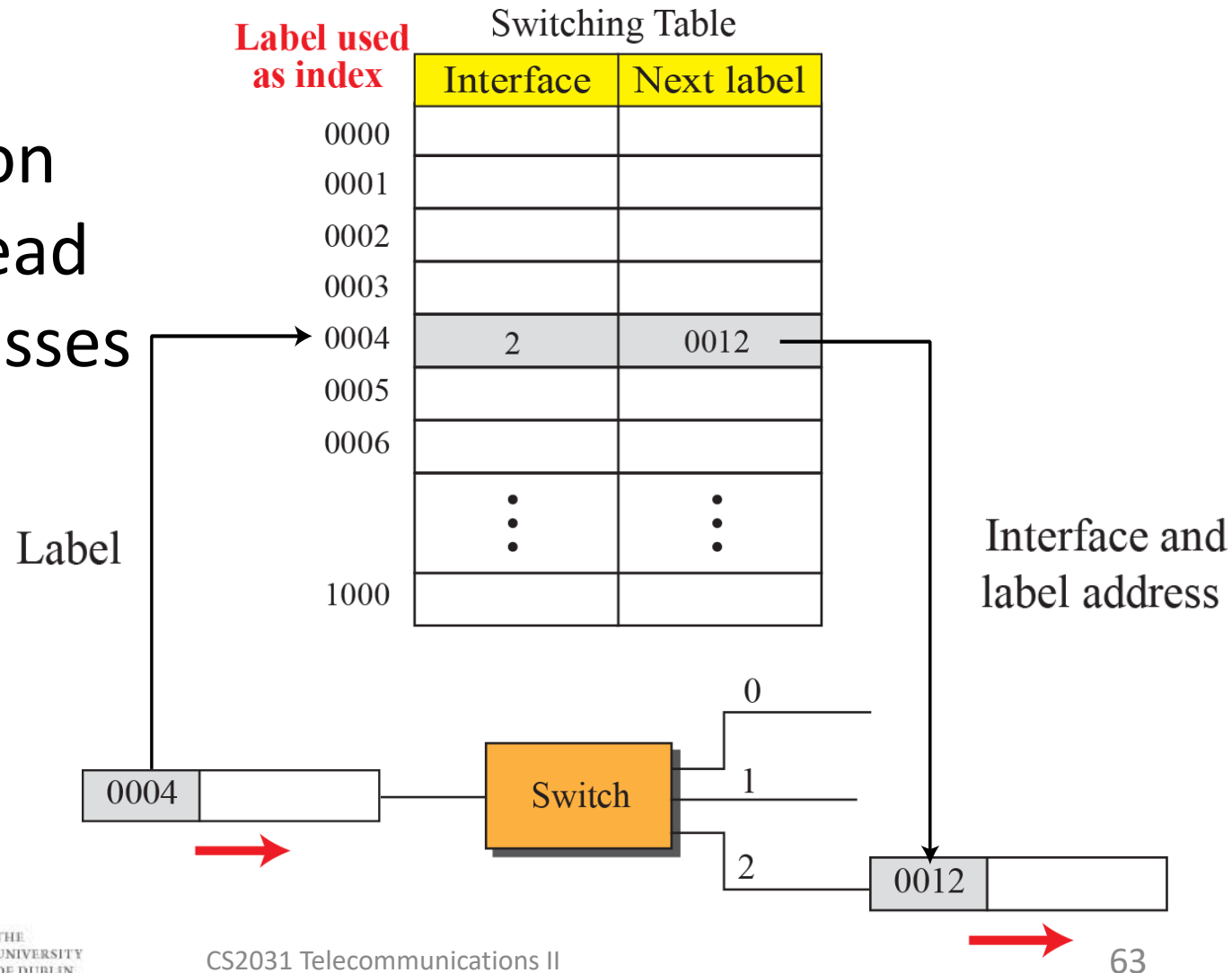
ATM – It Didn't Happen

- From Tanenbaum:

“ATM was going to solve all the world’s networking and telecommunications problems by merging voice, data, cable television, telex, telegraph, carrier pigeons, ...”
- It didn't happen:
 - Bad Timing
 - Technology
 - Implementation
 - Politics

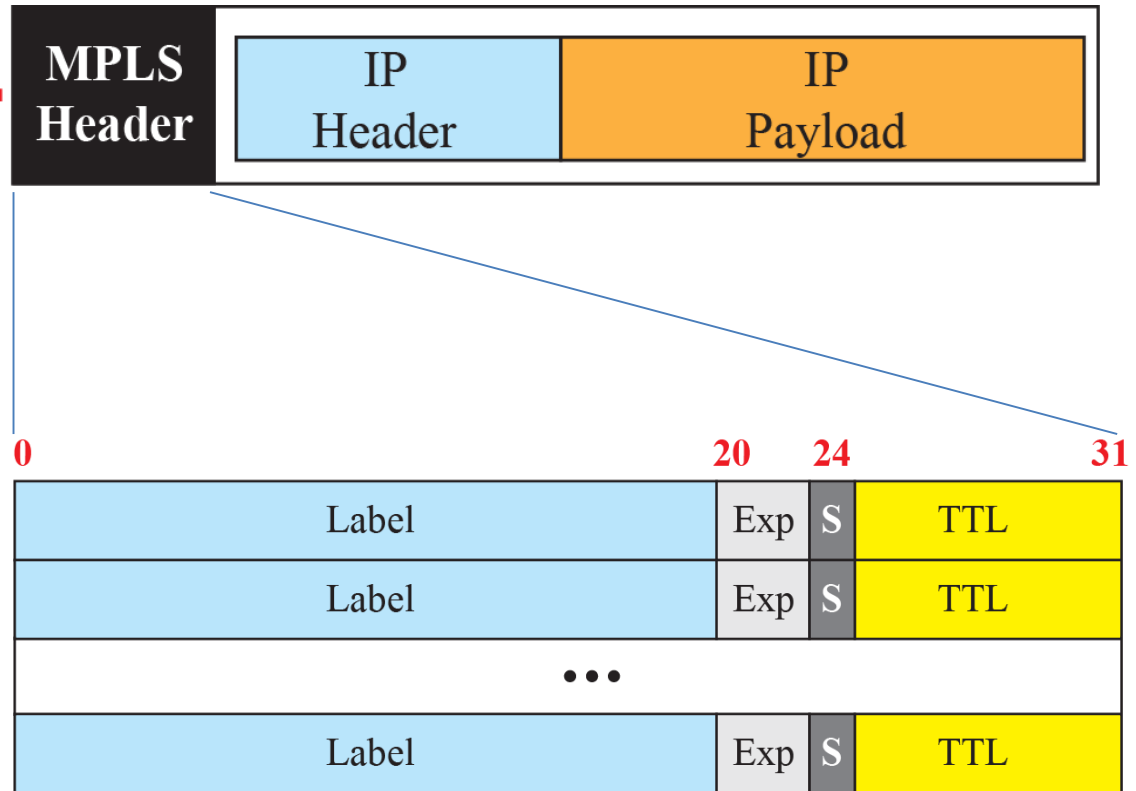
Multiprotocol Label Switching (MPLS)

- Enables switching on labels instead of IP addresses

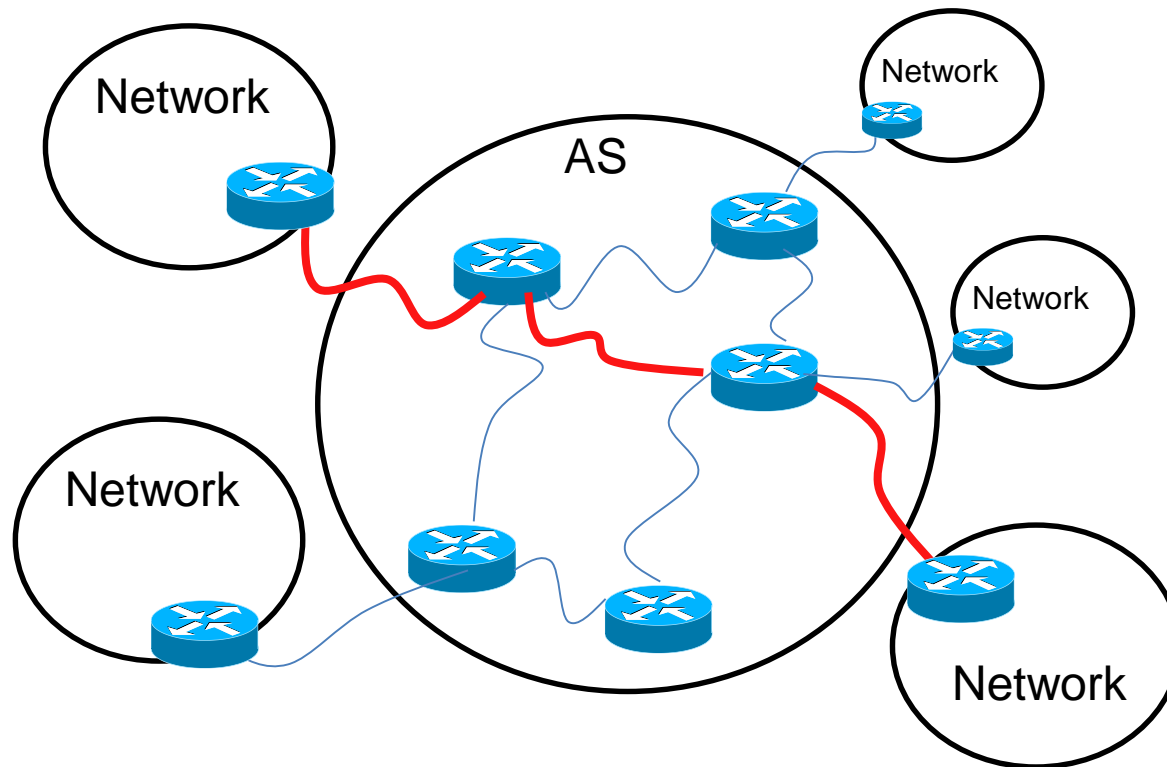


MPLS Header

- MPLS header as stack of labels



MPLS Use Case



- Creating a virtual network

CS2031 Telecommunications II



Summary: Virtual Circuit Switching – ATM

- Virtual Circuit Switching
 - Preplanned route established before any frames sent
 - Call request and call accept frames establish connection (handshake)
 - Each frame contains a virtual circuit identifier instead of destination address
 - No routing decisions required for each frame
 - Clear request to drop circuit
 - Not a dedicated path
- Asynchronous Transfer Mode (ATM)
 - Example for virtual circuit switching
 - Cells consist of 5-byte header and 48-byte payload
 - Circuits identified by virtual circuit ID and virtual path ID
 - Application adaptation layer (AAL) for specific application areas

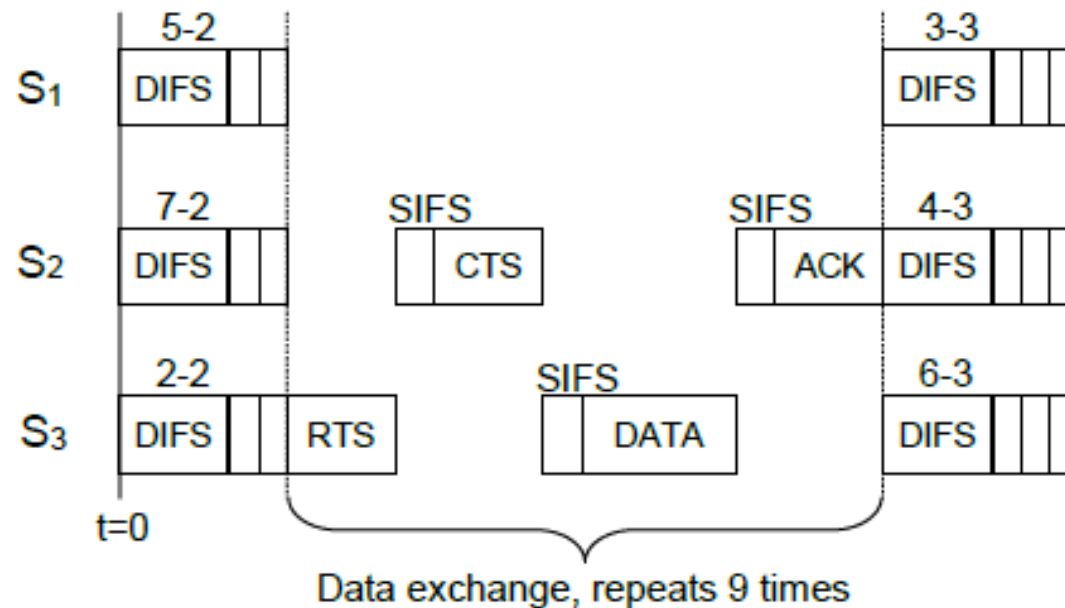
Tutorial - Question 3

3 stations intent to transmit each 3 data frames

Transmission times for

- data frames and the Beacon: 190us
- RTS and Poll: 180us
- CTS, ACK and CF-End: 132us

DCF - Individual Transmissions



$$\begin{aligned}
 & \text{DIFS} + \text{RTS} + \text{CTS} + \text{data} + \text{ack} + 3 * \text{SIFS} \\
 &= 34\mu\text{s} + 180\mu\text{s} + 132\mu\text{s} + 190\mu\text{s} + 132\mu\text{s} + 3 * 16\mu\text{s} \\
 &= 716\mu\text{s}
 \end{aligned}$$

DCF – Backoff Procedure

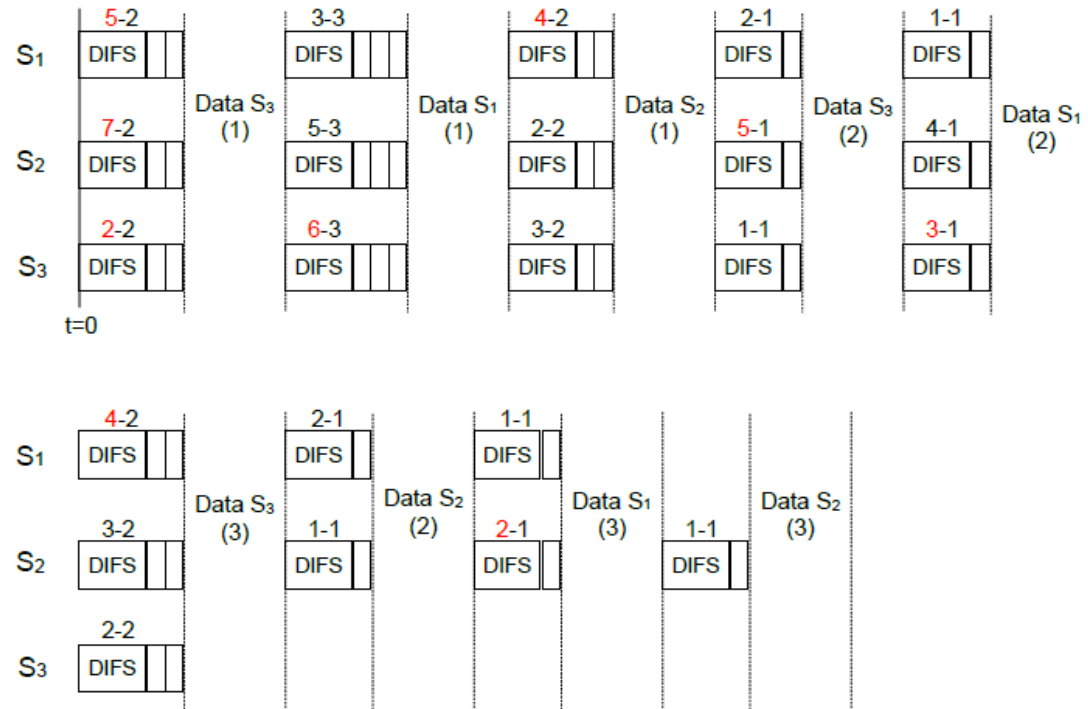
Step 1:

S₁: 5, 4

S₂: 7, 5, 2

S₃: 2, 6,

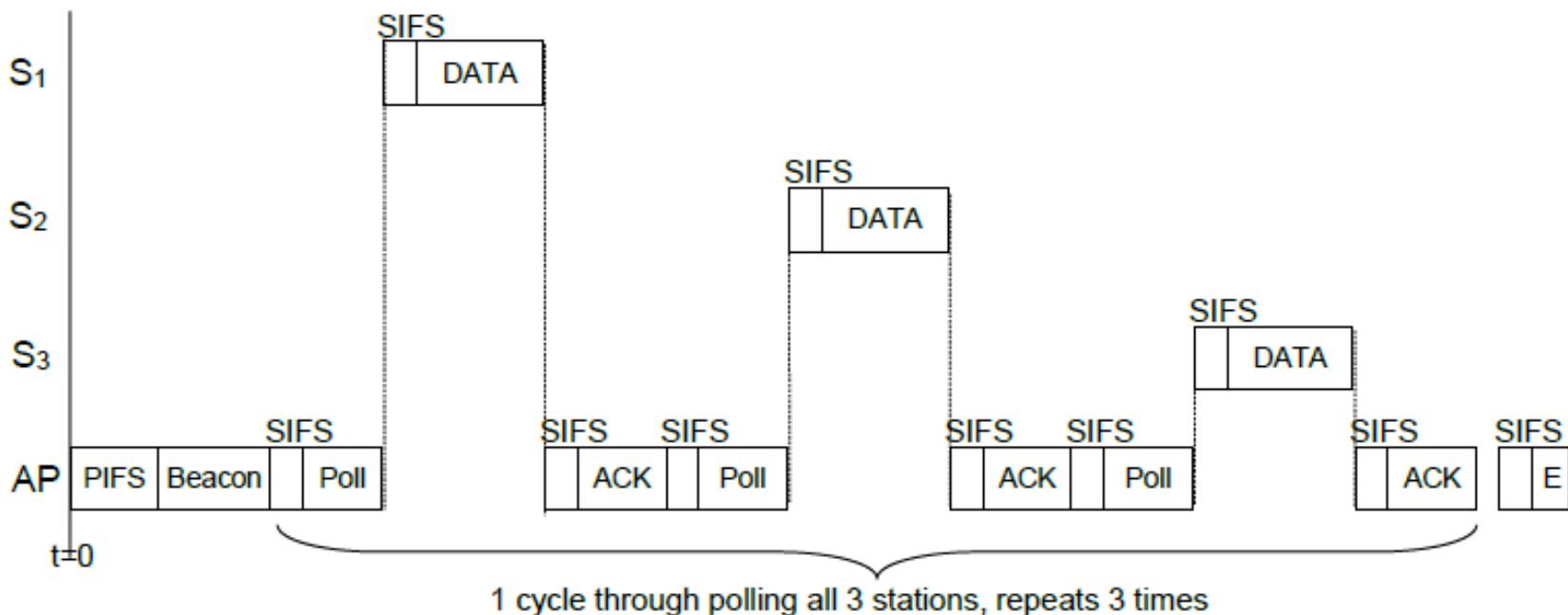
Step 2:



$$9 \cdot T_x + 14 \cdot \text{slots} = 9 \cdot 716 \mu\text{s} + 14 \cdot 9 \mu\text{s}$$

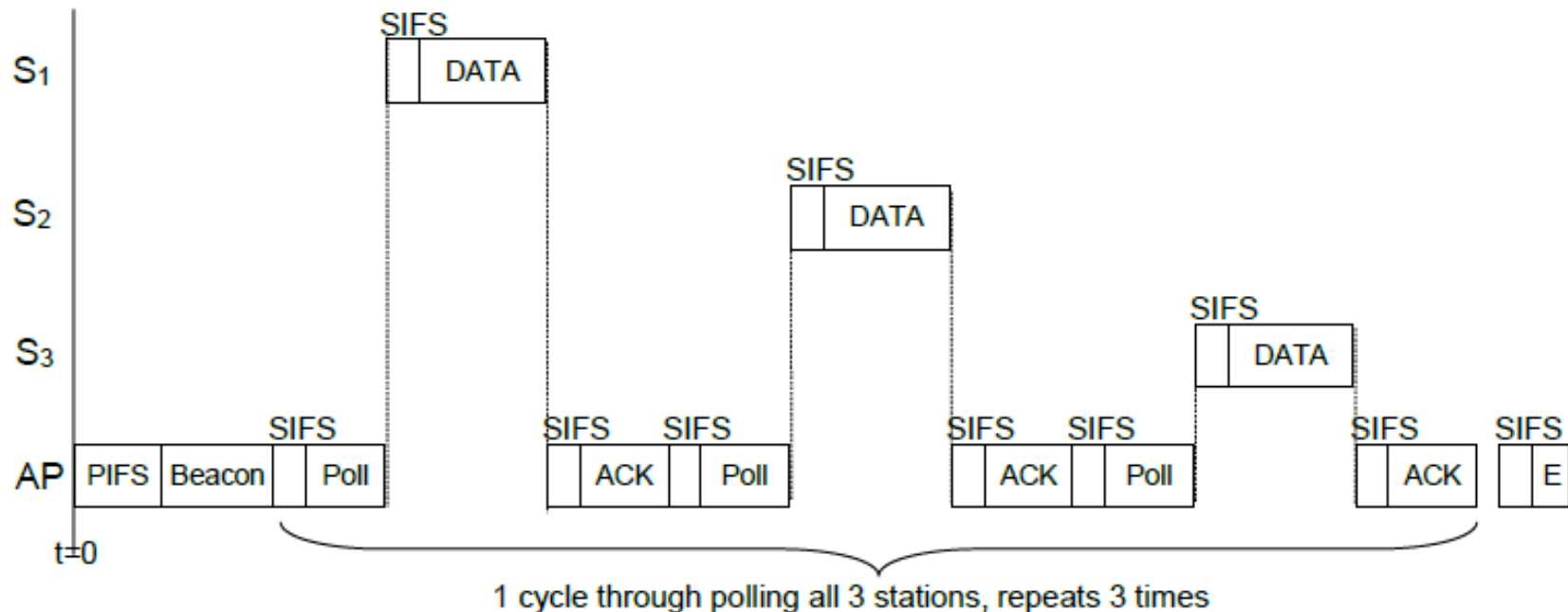
$$= 6,570 \mu\text{s} = \sim 0.006 \text{ms}$$

PCF – Round of Polls



$$\begin{aligned} & 9 * \text{SIFS} + 3 * \text{poll} + 3 * \text{data} + 3 * \text{ack} \\ &= 9 * 16 \mu\text{s} + 3 * 180 \mu\text{s} + 3 * 190 \mu\text{s} + 3 * 132 \mu\text{s} \\ &= 1,650 \mu\text{s} \end{aligned}$$

PCF – Round of Polls



$$\begin{aligned} & \text{PIFS} + \text{beacon} + 3 \cdot D_x + \text{SIFS} + \text{CF-End} \\ &= 25\mu\text{s} + 190\mu\text{s} + 3 \cdot 1650\mu\text{s} + 16\mu\text{s} + 132\mu\text{s} \end{aligned}$$

$$= 5,313\mu\text{s} = \sim 0.005\text{ms}$$