Georgia Tech Computer Science

CS2200
Systems and Networks
Spring 2022

Lecture 25: Networking

Alexandros (Alex) Daglis
School of Computer Science
Georgia Institute of Technology
adaglis@gatech.edu

Lecture slides adapted from Bill Leahy and Charles Lively of Georgia Tech

Networking

- Network stack overview – layer encapsulation
- Three transport protocol types
- Basic distinctions between TCP and UDP
- Network layer: subnets and basic routing

Today:

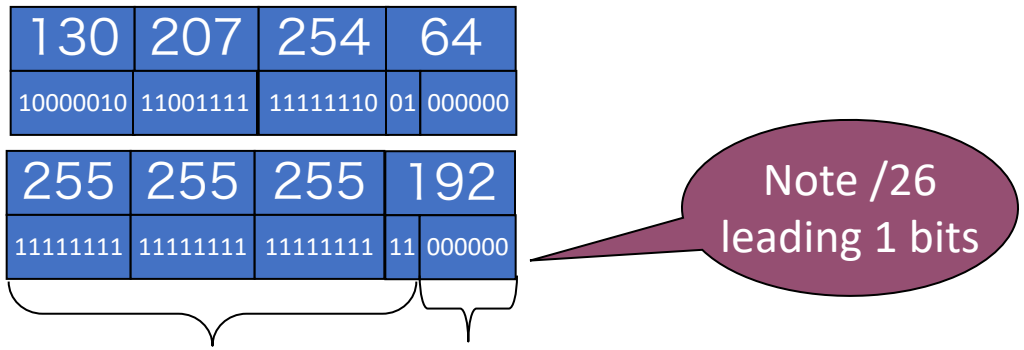
- Ports and connections
- Data Link layer (Ethernet)
- Performance metrics

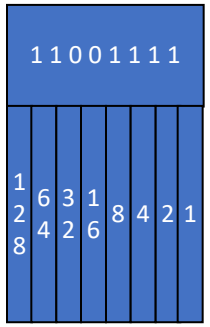
Thursday:

- Routing and IP table construction

Subnet and CIDR Notation

- Two different notations, same meaning:
  - 130.207.254.64 255.255.255.192
  - 130.207.254.64/26
- In this example, the network part is the first 26 bits, host is last 6
- If you're doing this every day, you learn to do it in your head; if not, use a subnet calculator, e.g. http://www.subnet-calculator.com/cidr.php
- Cisco equipment often uses the former notation; many other modern tools use the latter





- In the dark ages, subnet masks could be derived from the network address
- Now that's called "classful" addressing
- These can still show up as a default mask if you don't specify one

Class	Prefix	Range	Mask	Purpose
A	0…	0.0.0.0-127.255.255.255	8	
B	01…	128.0.0.0-191.255.255.255	16	
C	011…	192.0.0.0-223.255.255.255	24	
D	0111…	224.0.0.0-239.255.255.255	-	multicast
E	1111…	240.0.0.0-255.255.255.255	-	reserved

CIDR: Classless Inter-Domain Routing

- We ran out of IP addresses in 1995.
- CIDR introduces variable-length subnet masking for arbitrary-length network prefixes
- This was a very clever solution
- Turns out you only really need the network mask if you're connected to that network!
- Now we simply specify the mask for the local subnets on each host

IP Routing Table

Destination	Gateway	Flags	Interface
0.0.0.0/0	128.61.5.1	UG	
128.61.5.0/24	128.61.5.166	U	Eth0
128.61.5.21/32	128.61.5.166	U	Eth0
127.0.0.1/32	127.0.0.1	UH	

- If the destination is on our network, send the packet to the data link layer
- Look through the routing table and choose the match with the *longest prefix* (largest mask); send the packet to that gateway
- Question: Which line does 128.61.5.82 match?
  How about 130.207.5.9?
- The interesting part comes later: Creating the routing table

Questions: IP Routing rules

If the network address part of the interface and destination addresses are the **same**, then

A. I just want the participation credit
B. Pass the packet to layer 2 for delivery
C. Send the packet to the next-hop router based on the IP routing table
D. Return an Acknowledgement to the source address

Questions: IP Routing rules

If the network address part of the interface and destination addresses are **different**, then

20% A. I just want the participation credit

20% B. Pass the packet to layer 2 for delivery

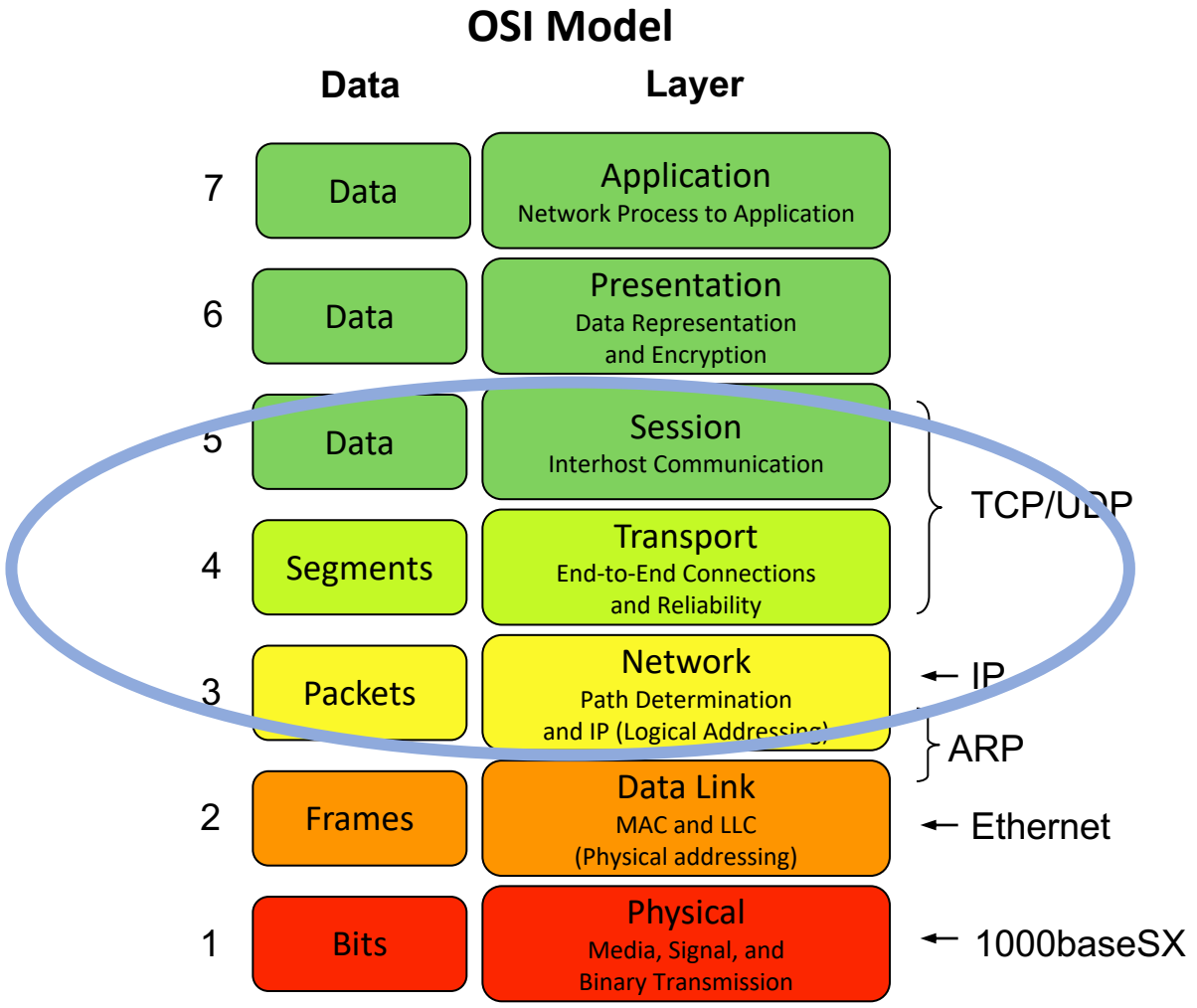
20% C. Send the packet to the next-hop router by choosing the longest match in the IP routing table

20% D. Send the packet to the next-hop router by choosing the shortest match in the IP routing table

20% E. Return an ICMP "No route to host" message

Now let's include Layers 3-5 in IP

- TCP and UDP are the two most common IP Transport protocols
- In IP, recall the division between L4 and L5 is not well defined



Ports and connection addressing

We know how to get packets from one host to another, but how do we identify conversations?

- How does one distinguish UDP and TCP traffic from different services that share the same network interface?
  - On each side of a conversation, port numbers (0-65535) are used to define unique connections
  - Often the destination port on the first packet is a Well-known or Registered port number
  - (Note that the fields are not actually ordered this way in the packet)
- With TCP and UDP protocols, always consider the addressing as a **quadruple**: *(source IP, source port, destination IP, destination port)*
- Let's look at where that ordered quad shows up in the segment...

TCP (and UDP) port addressing

IPv4 Header Format

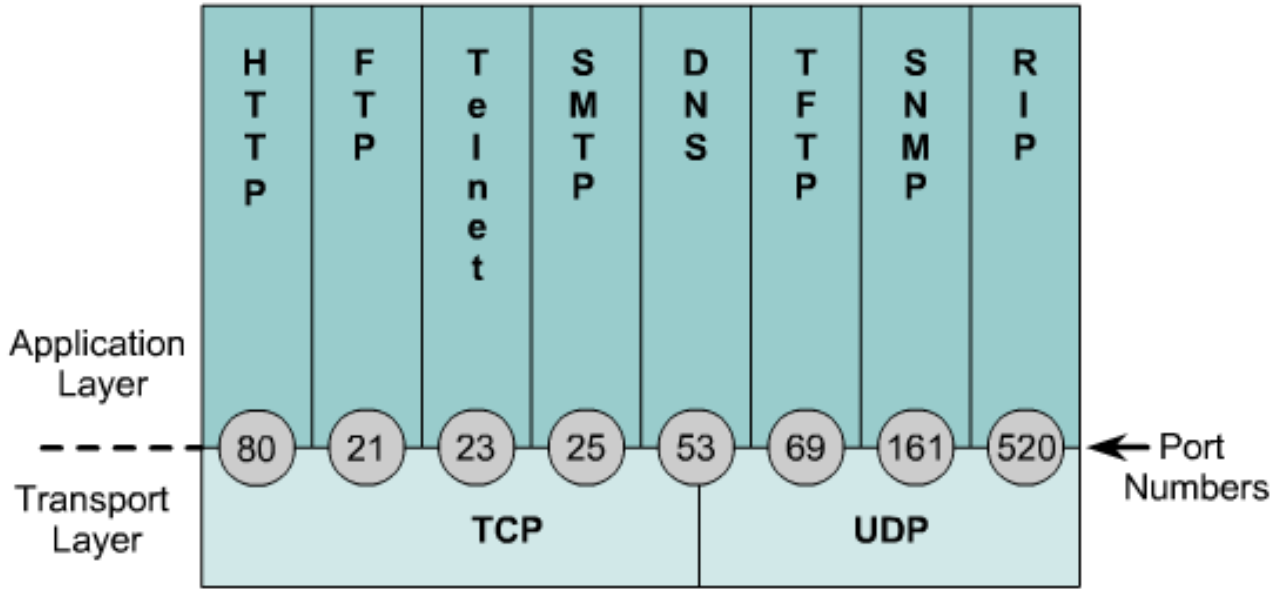
Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Version				IHL				DSCP						ECN		Total Length															
4	32	Identification																Flags			Fragment Offset											
8	64	Time To Live								Protocol								Header Checksum															
12	96	Source IP Address																															
16	128	Destination IP Address																															
20	160	Options (if IHL > 5)																															
24	192																																
28	224																																
32	256																																

TCP segment header

Offsets	Octet	0								1								2								3							
Octet	Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset				Reserved 0 0 0				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size														
16	128	Checksum																	Urgent pointer (if URG set)														
20 ...	160 ...	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.) ...																															

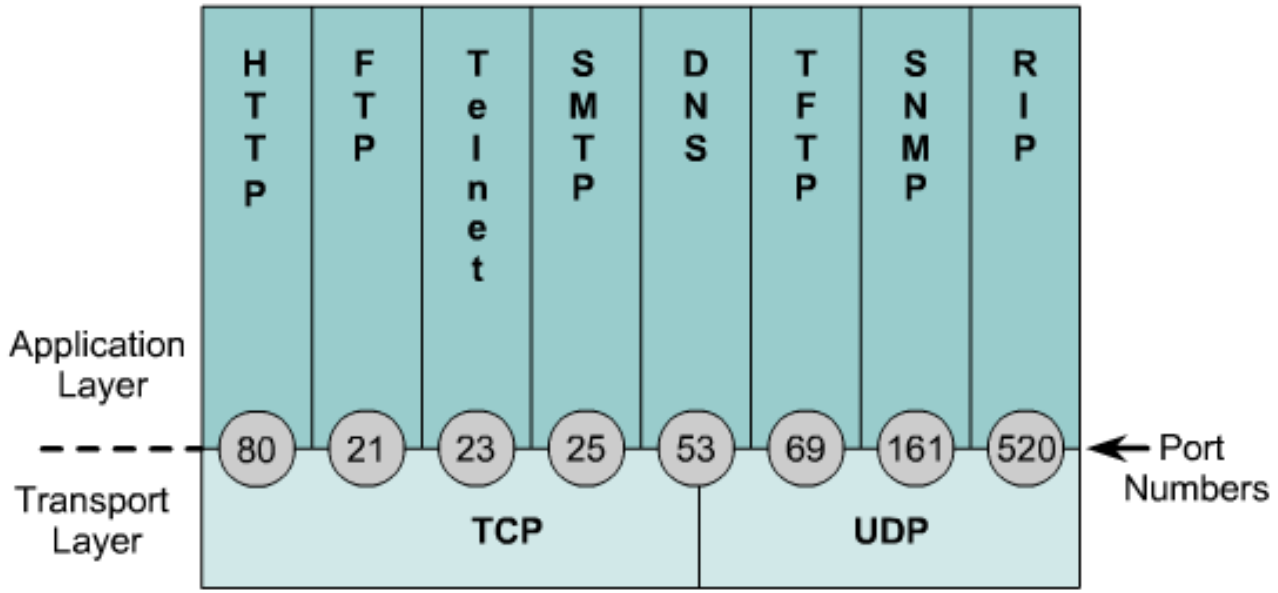
TCP and UDP port numbers

- Port numbers are used to keep track of different conversations.
- Well-known ports numbers are traditionally from 1-1023.
- Numbers 1024 and above are dynamic (ephemeral) port numbers.
- Registered port numbers for vendor-specific applications are now allowed above 1024.



TCP and UDP port numbers

- Destination ports are chosen based on the desired service
  - Often chosen based on the URL prefix, e.g. http: goes to 80, https: to 443
- Source ports are usually chosen by the IP stack to be unique among all open connections
- That makes the ordered quad unique among open connections
  - This is how we keep connections separate even when the IPs are the same



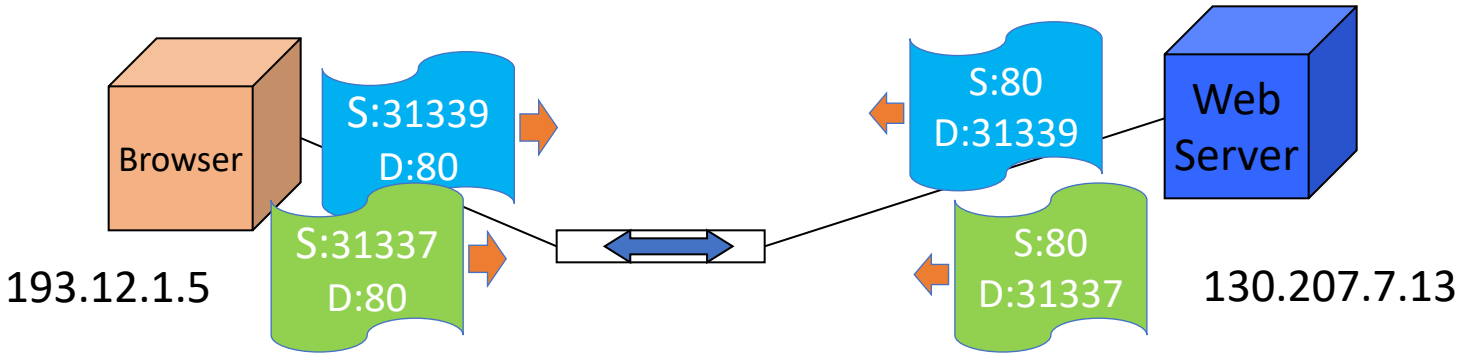
A browser opens two TCP connections to a web server

If the browser is running at 193.12.1.5, and the web server is offering service on port 80 at 130.207.7.13, how does TCP keep the conversations separate?

- 8% A. I just want the participation credit
- 17% B. IP does this automatically
- 42% C. The conversations use independent sequence numbers to disambiguate them
- 8% D. The conversations include a special session-identifier that disambiguates them
- 25% E. The browser lets the TCP stack choose a unique source port number for each connection

Conversations

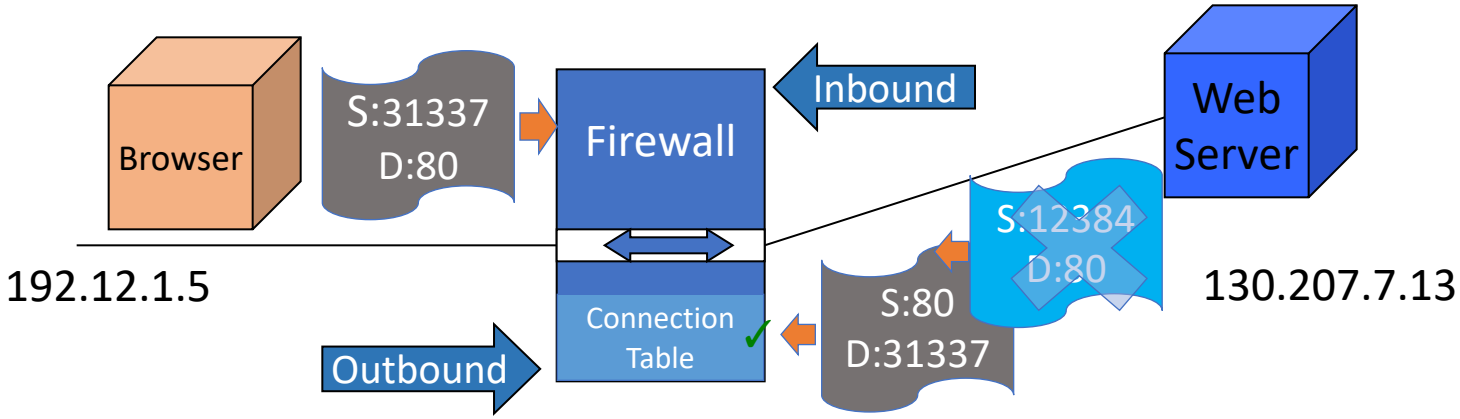
- When the web browser opens each connection, it doesn't specify a source port, allowing the TCP stack to pick a source port that isn't in use by any conversation on that host, say 31339 or 31337.
- The TCP stack uses the quads (193.12.1.5, 31337 or 31339, 130.207.7.13, 80) and its reverse to distinguish the two separate conversations
- Each connection gets its own sequence number, window size, etc. and is treated as a completely separate conversation by both the browser's and the web server's IP stacks



Stateful Firewalls

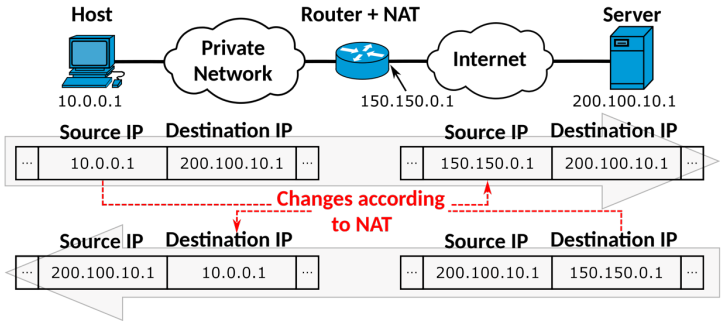
- Follow a predefined policy on which ordered quads are allowed from which direction
  - Example: Allow no inbound packets; Allow all outbound packets
- Records state on connections as it is opened
  - Requires a local state table (called a connection table)
- Compares packets to the connection table first to automatically allow return traffic
  - Example: If (192.12.1.5, 31337, 130.207.7.13, 80) is put in the connection table, then return traffic from (130.207.7.13, 80, 192.12.1.5, 31337) is automatically allowed
  - Any inbound traffic to other ports at 192.12.1.5 is dropped since no policy rule allows it
- Can do other protocol verification to drop malicious or badly-formed traffic

Makes sense if we don't offer any services



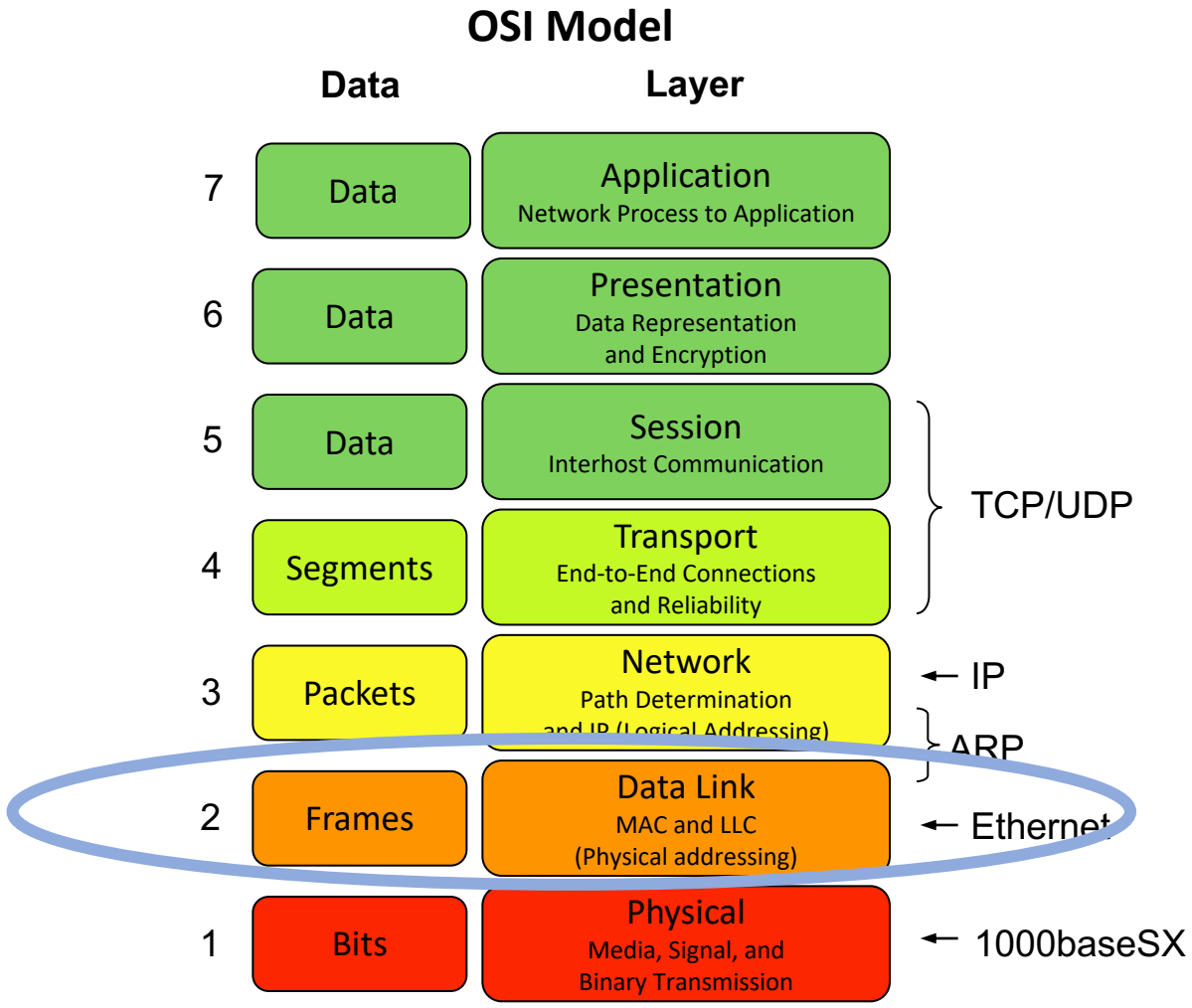
Network Address Translation

- Note there is confusing terminology
- Types
  - Static one-to-one NAT
    - A mapping is configured so that source address or destination address is changed from/to the mapped address on transit; checksums are recalculated
  - Dynamic one-to-one NAT
    - Same as Static, but a public address from a pool isn't mapped until a private address attempts to cross the NAT device; the public address is returned to the pool when the conversation ends
  - PAT or NPAT (network and port address translation)
    - Building on Dynamic NAT, both the source address and source port are mapped to an address and port from the public pool (often of size 1); return traffic is mapped by destination address and port.
  - And a lot of other variants
- NPAT is typically the way we share a single public IP address among multiple hosts -- another way we've managed the scarcity of public IP addresses



Now let's look at the data link layer

- Ethernet is a very common data link network



Let's talk about Ethernet

- Ethernet is a particular type of data link network
- It was designed for relatively small areas, like a building, hence the term Local Area Network (LAN)
- It has its very own standard (IEEE 802.3)
- It's evolved a great deal since the 1980s

Terminology

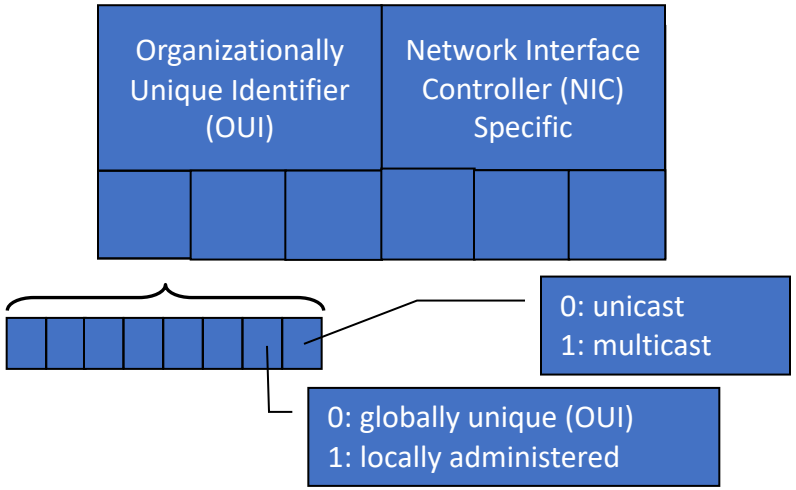
- Collision domain - All the NICs that share a physical ethernet; only one can transmit at a time
- Broadcast domain - All the NICs that can hear an ethernet broadcast frame

---

- Repeater/hub - layer 1 replication/amplification of bits (both sides stay in one collision domain)
- Bridge/switch - layer 2 re-transmission of ethernet frames (both sides stay in one broadcast domain)
- Router - layer 3 forwarding of packets based on a layer-3 (IP) routing table
- Host – a computer with one or more NICs; if it forwards traffic using multiple NICs, we're going to call it a router, not a host.

Ethernet Addressing

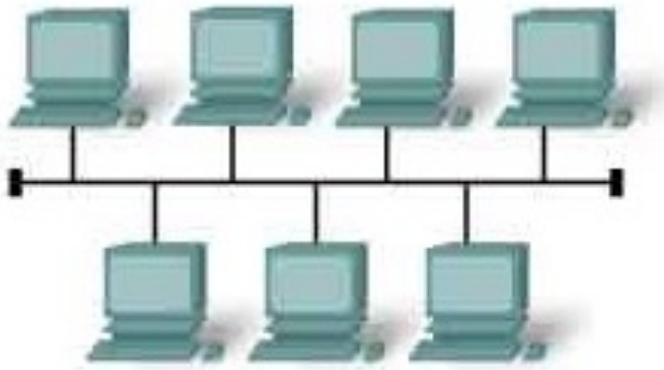
Every ethernet device gets a unique burned-in 48-bit MAC address



- 6 bytes/48 bits
  - 3 bytes OUI
  - 3 bytes NIC specific
- Examples
  - Apple: 00:25:00:f4:6d:c2
  - Oracle: 00-03-ba-a1-02-df
- To identify the OUI
  - https://www.wireshark.org/tools/oui-lookup.html

Ethernet Frames

Preamble	Destination MAC	Source MAC	802.1Q tag	Ether type/ Length	Payload	CRC	Interframe Gap
8	6	6	0-4	2	46-1500	4	12

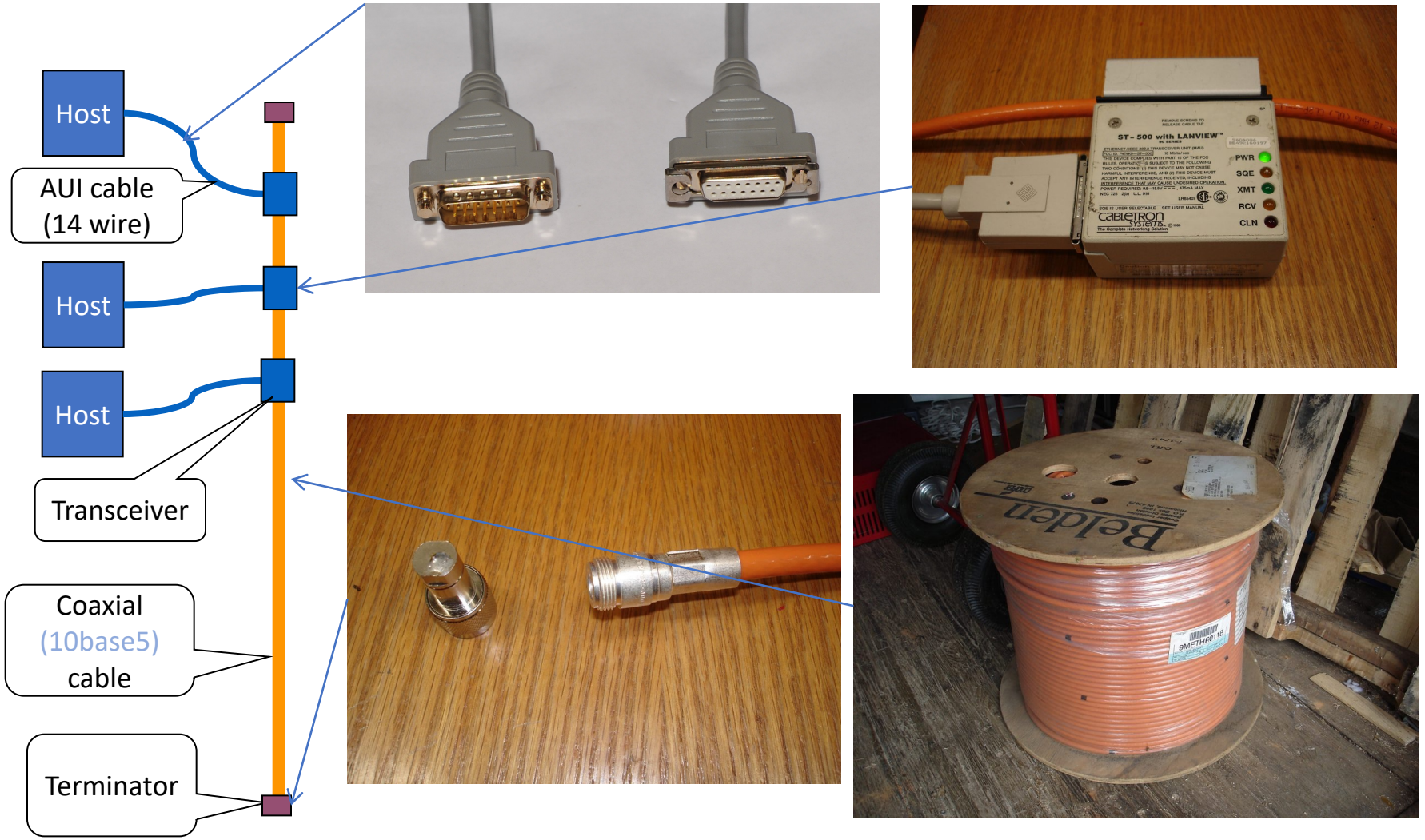


- To send a frame
  - Wait until no one using the cable (carrier sense)
  - Assert carrier signal on the cable
  - Broadcast the frame to all stations on the cable
  - If you detect a collision while transmitting, stop and requeue the frame for retransmission
- If you hear a frame on the cable
  - Accept it if the destination MAC belongs to us (or if it has a broadcast or multicast destination MAC)
  - Drop it if you see a collision during its receipt or its CRC doesn't match

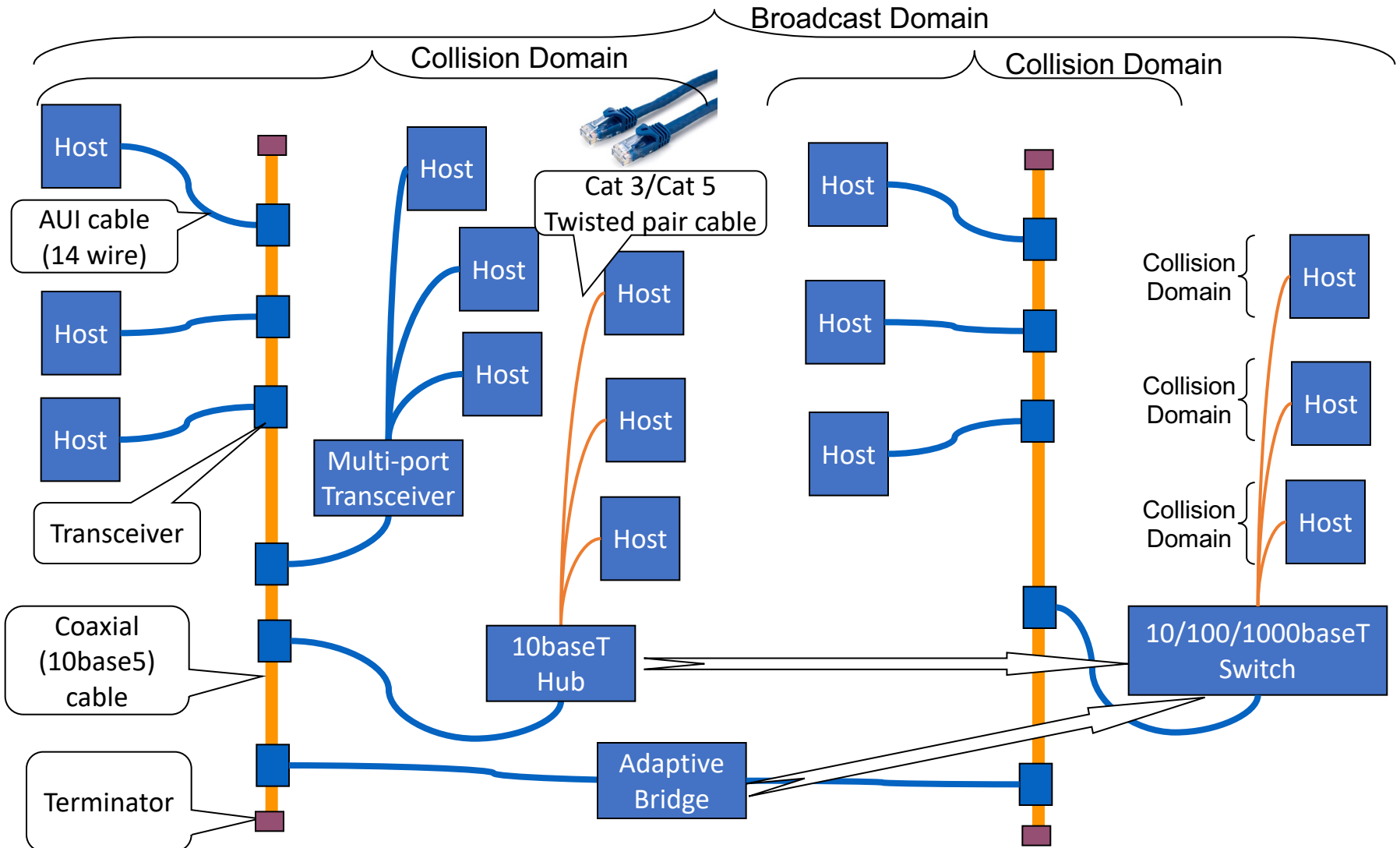
Something is Missing…

- Only one station can send at a time?
- Where are the IP addresses?
- We'll get back to this…

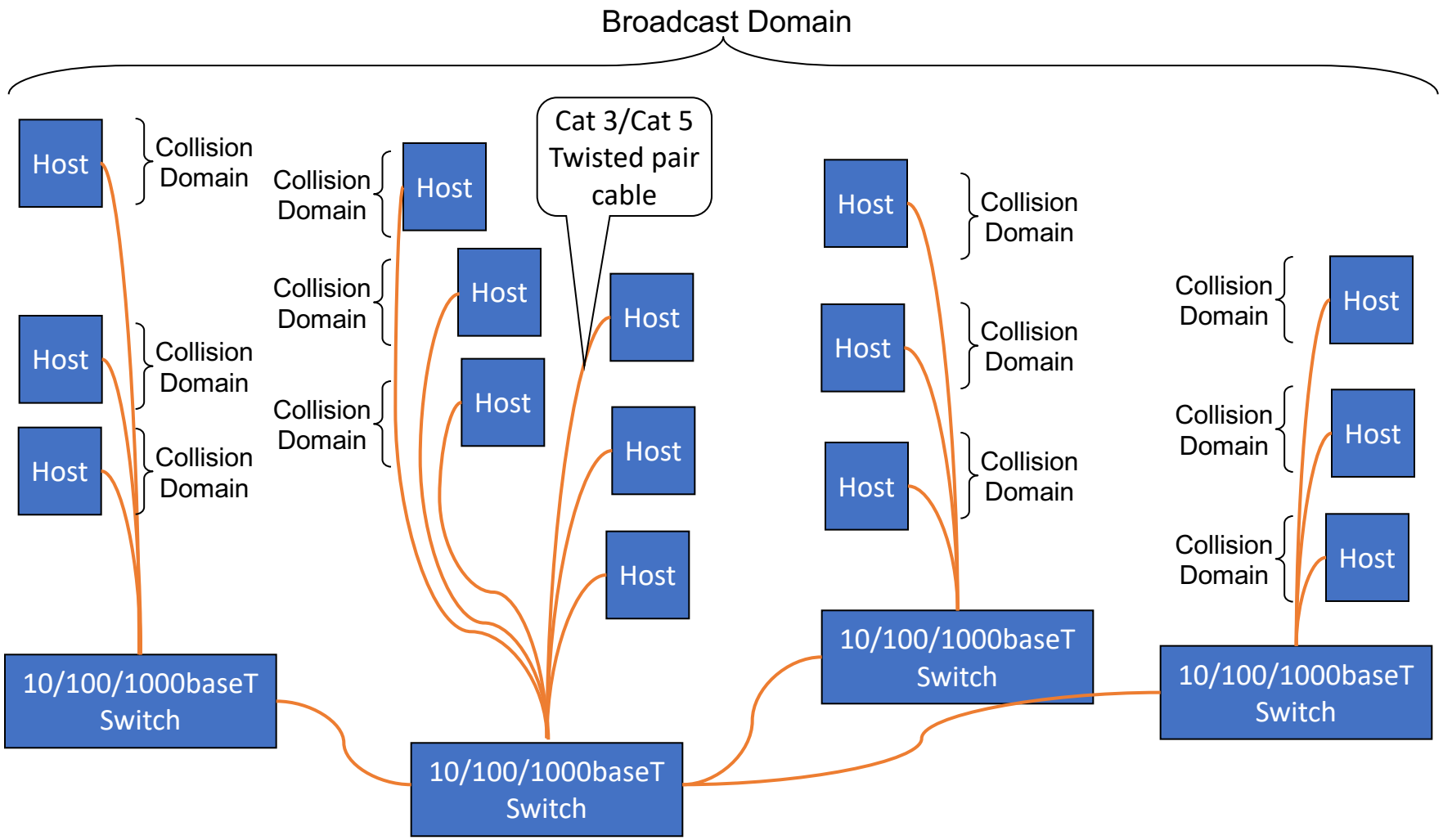
Ethernet Topology (some history)



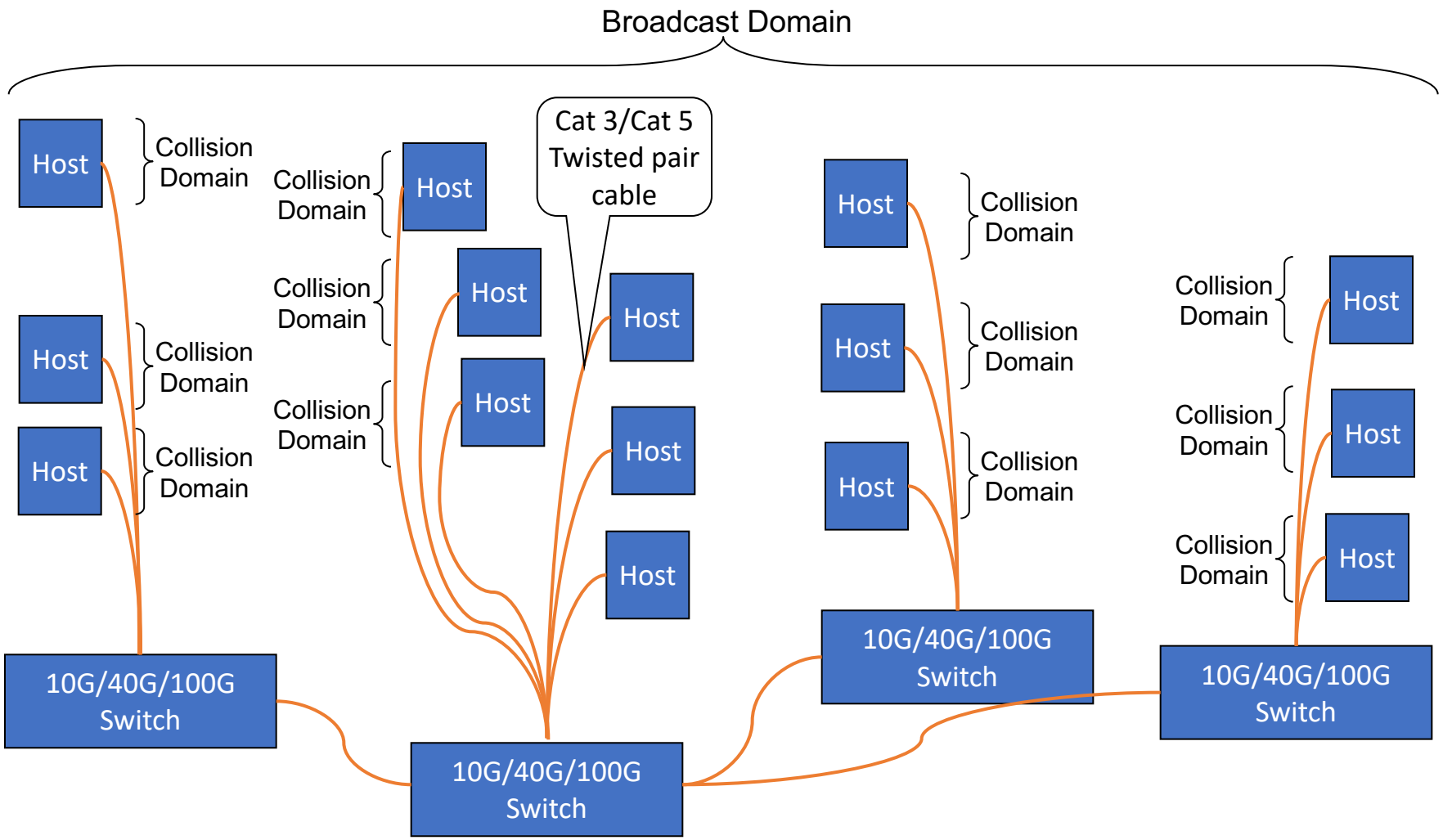
Ethernet Topology (some history)



Ethernet Topology (some history)



Ethernet Topology (some history)



Ethernet Bridging Algorithm (simplified)

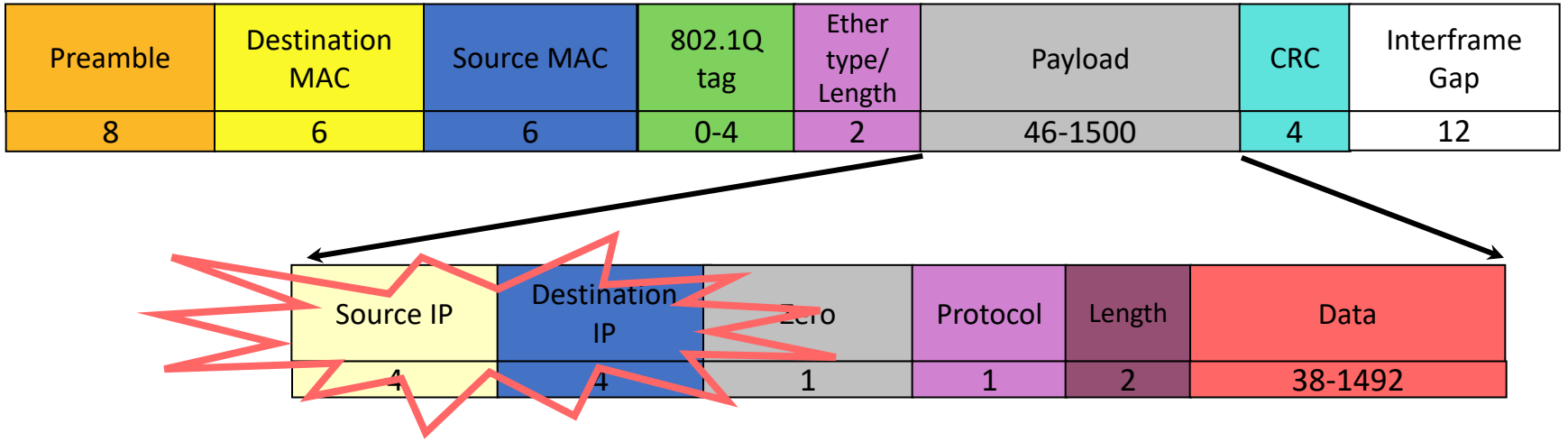
Bridge Table		
MAC Addr	Last Seen on Port	Last Used
002080 00324a	*FastEthernet 1/4*	*12 sec ago*
0003ba 51733a	*FastEthernet 2/9*	*72 sec ago*
. . .	. . .	. . .

- When a frame enters the bridge (switch):
  - If the **destination** MAC address is in the bridge table
    - Send the frame out the port listed in the bridge table entry
  - Otherwise
    - Flood the frame out all ports except the ingress port
  - Put the **source** MAC address in the bridge table, if necessary, along with the frame ingress port and mark that entry “recently used”
  - Remove any “stale” entries in the bridge table

Something Is Still Missing

- Now more than one message can be in transit at a time
- We've only talked about MAC addresses.
- Where are the IP addresses?

Layer 3 (IP) packets



- IP addresses appear in the Ethernet frame payload, not the header
- But Ethernet hardware only examines the MAC addresses in the Ethernet frame(!)
- How do we connect these two layers?

ARP

- Address Resolution Protocol
- Used with Ethernet to discover IP addresses
- How?
  - IP stack needs to send to IP address *a*, but doesn't know what destination MAC address to use.
  - Broadcast an ARP-request message: "Who has IP address *a*?"
  - Wait on unicast ARP-response: "I have IP address *a*."
  - Store the response in a local *ARP Table*, discarding entries when they become stale.
  - Use the corresponding entry in the ARP table to set the destination MAC address of the frame.

ARP Example

```
bash-3.00# arp -an
Net to Media Table: IPv4
Device   IP Address               Mask      Flags      Phys Addr
------ -------------------- --------------- -------- ---------------
bge0   130.207.165.229      255.255.255.255 o        00:50:56:91:4a:2b
bge0   130.207.165.248      255.255.255.255 o        00:14:4f:f2:8c:ce
bge0   130.207.165.240      255.255.255.255 o        00:50:56:91:62:be
bge0   130.207.165.242      255.255.255.255 o        00:14:4f:21:14:f6
bge0   130.207.165.245      255.255.255.255 o        00:14:4f:94:d7:0c
bge0   130.207.165.194      255.255.255.255 o        00:50:56:91:59:25
bge0   130.207.165.197      255.255.255.255 o        00:50:56:91:46:a3
bge0   130.207.165.221      255.255.255.255 o        00:50:56:91:4c:af
bge0   130.207.165.163      255.255.255.255 o        00:14:4f:7d:7f:58
bge0   130.207.165.137      255.255.255.255          00:03:ba:1a:28:a7
bge0   130.207.165.138      255.255.255.255          00:03:ba:1a:2a:ff
bge0   130.207.165.158      255.255.255.255 o        00:14:4f:1e:26:ec
bge0   130.207.165.96       255.255.255.255 o        00:1b:24:1d:eb:7c
bge0   130.207.165.103      255.255.255.255 o        00:50:56:a4:7e:df
bge0   130.207.165.122      255.255.255.255 o        00:50:56:91:43:f8
```

VLANs

- Virtual LAN, a.k.a. VLAN
  - A group of hosts that can communicate as if they were attached to the same **broadcast domain**, regardless of their physical location.
  - A VLAN has the same attributes as a physical LAN, but it allows for hosts to be grouped together even if they are not connected to the same network switch.
  - Network reconfiguration can be done through software instead of physically relocating devices and wires.

Geographic Addressing

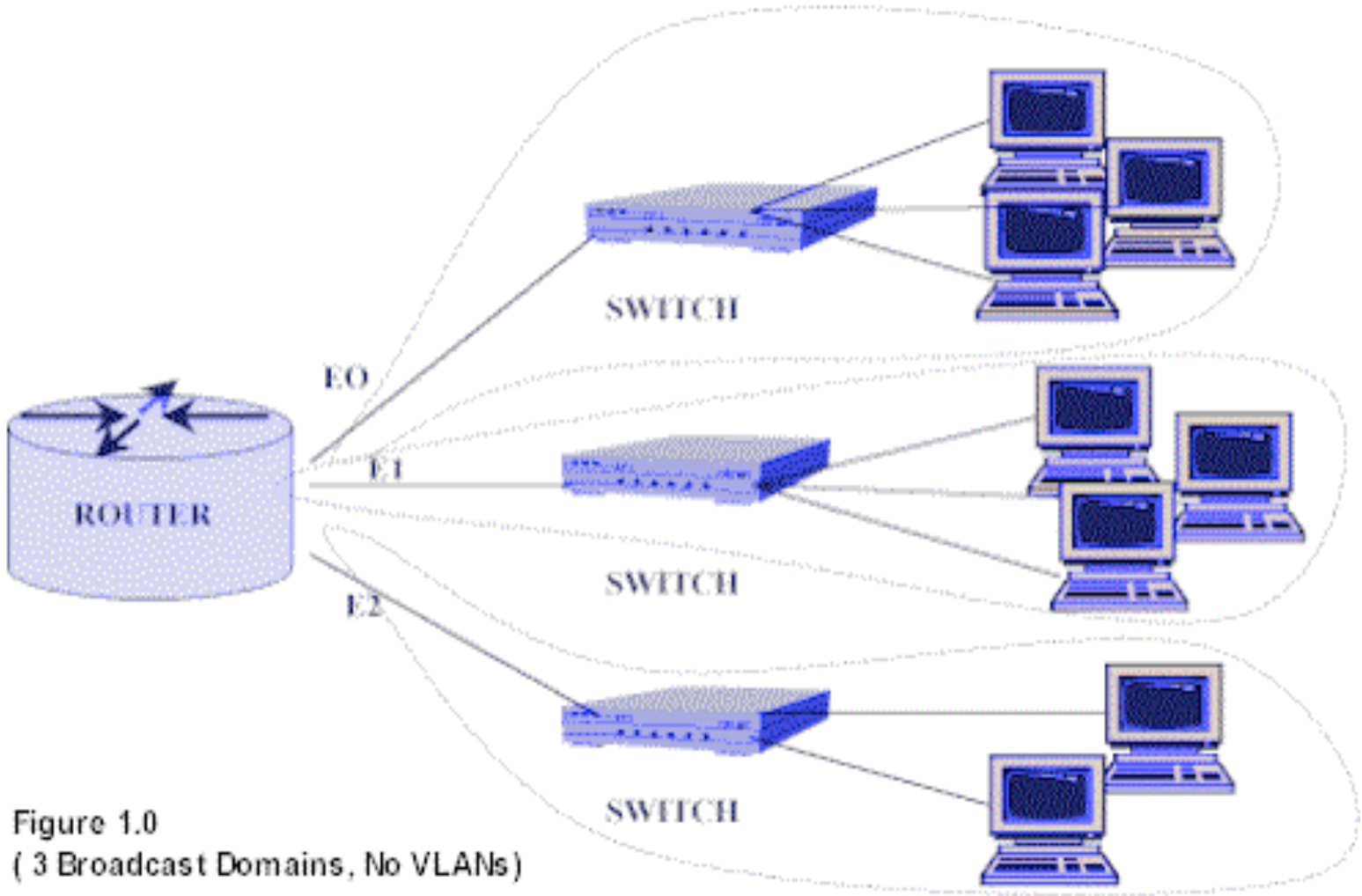
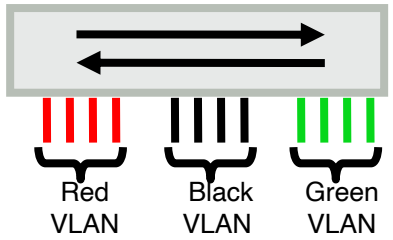


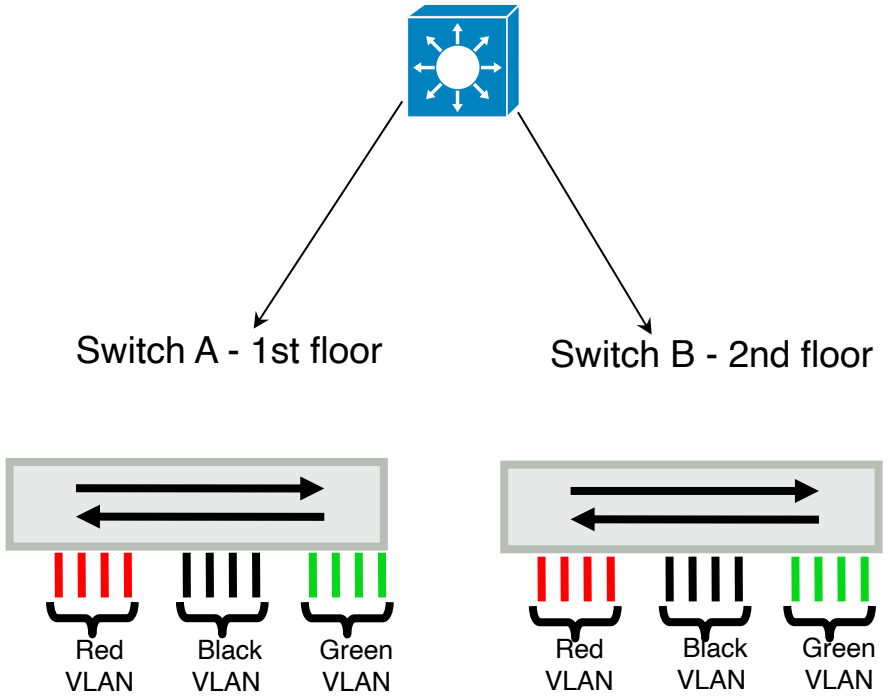
Figure 1.0
(3 Broadcast Domains, No VLANs)

After VLANs

Switch A



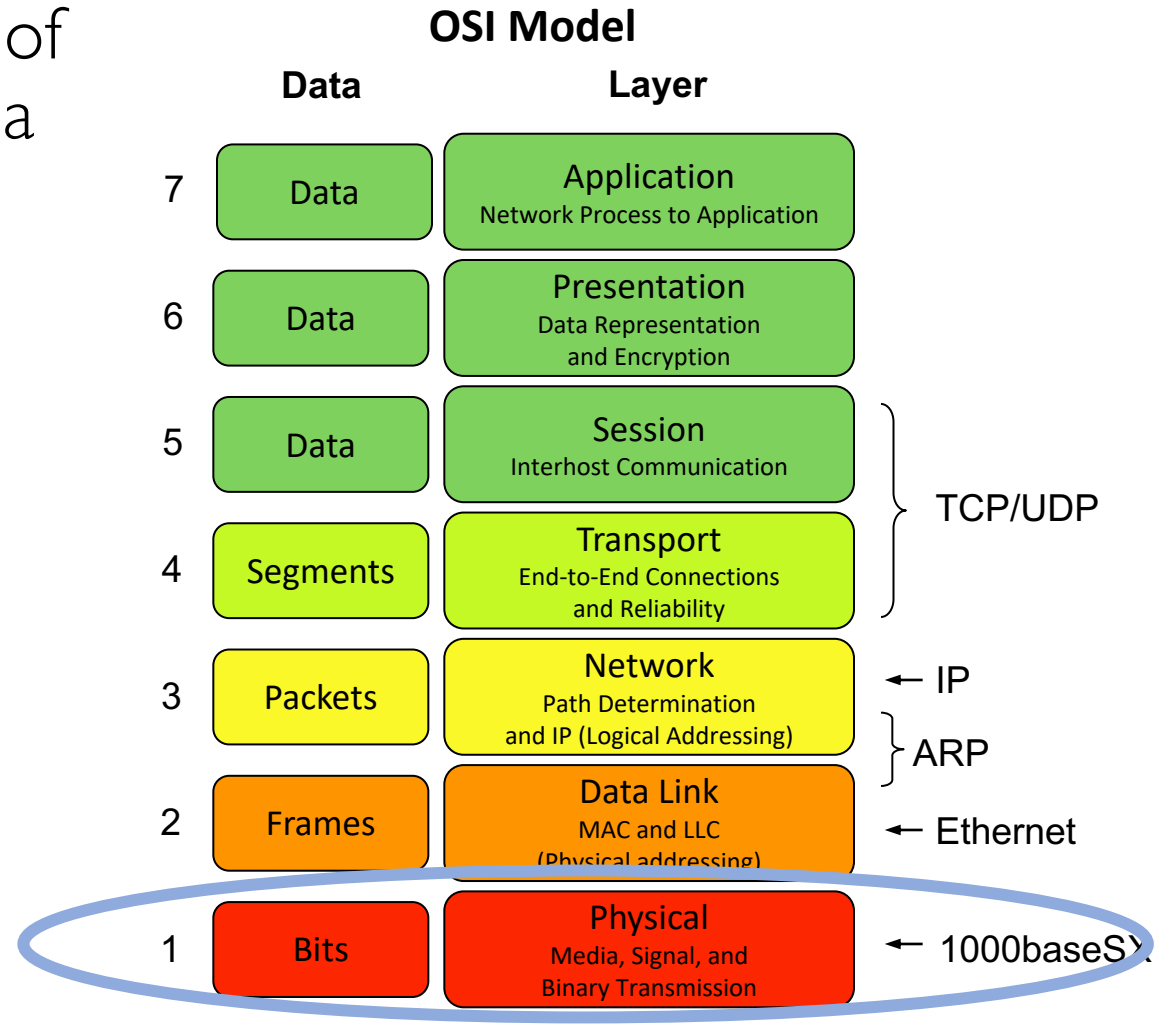
- Each logical VLAN is like a separate physical bridge
  - College of Engineering (red)
  - College of Computing (black)
  - Classroom network (green)



- Each logical VLAN is like a separate physical bridge
- VLANs can span across multiple switches

Now let's look at the last layer

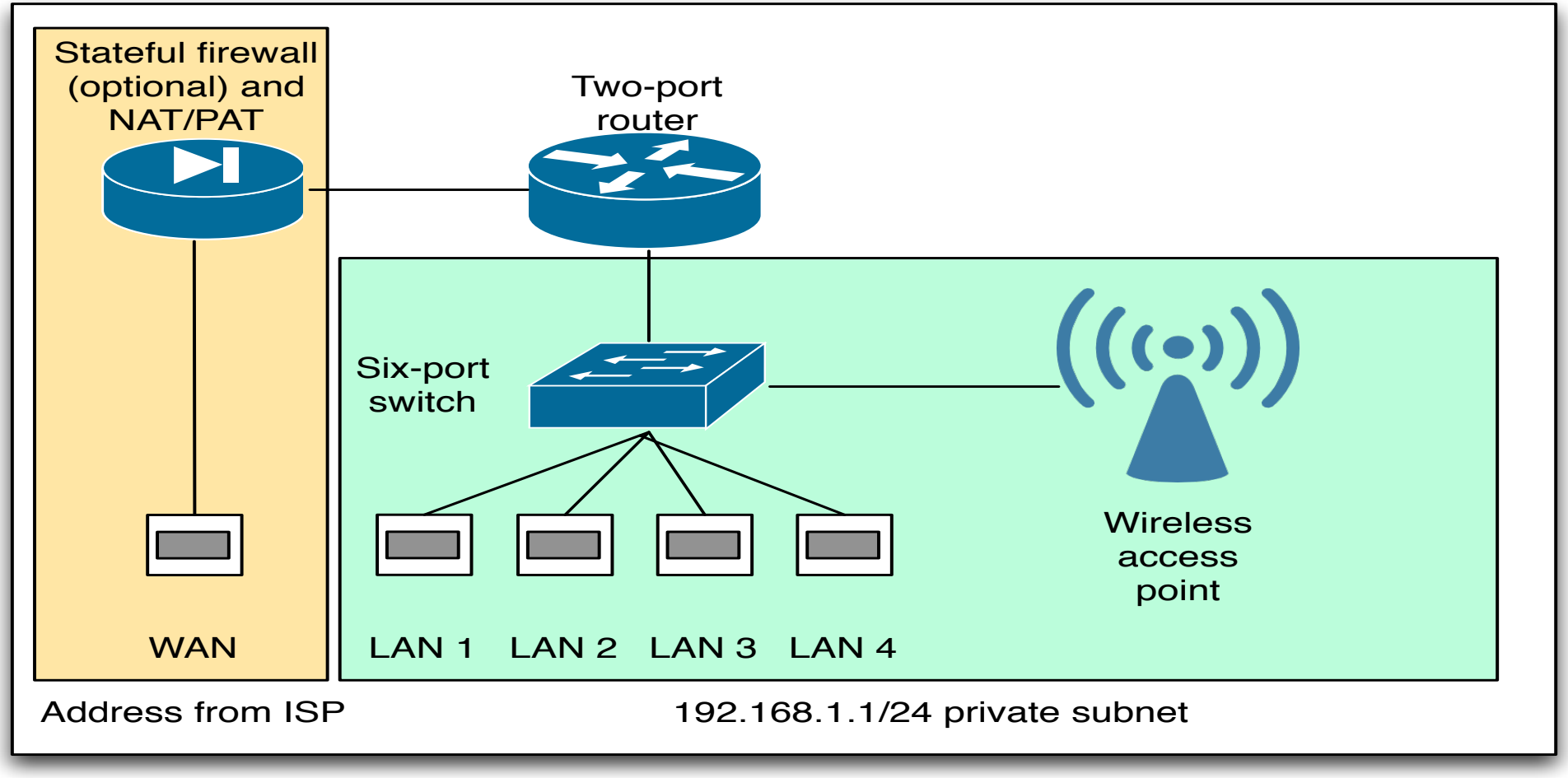
- Ethernet standards allow use of several types of physical media



Ethernet physical media standards

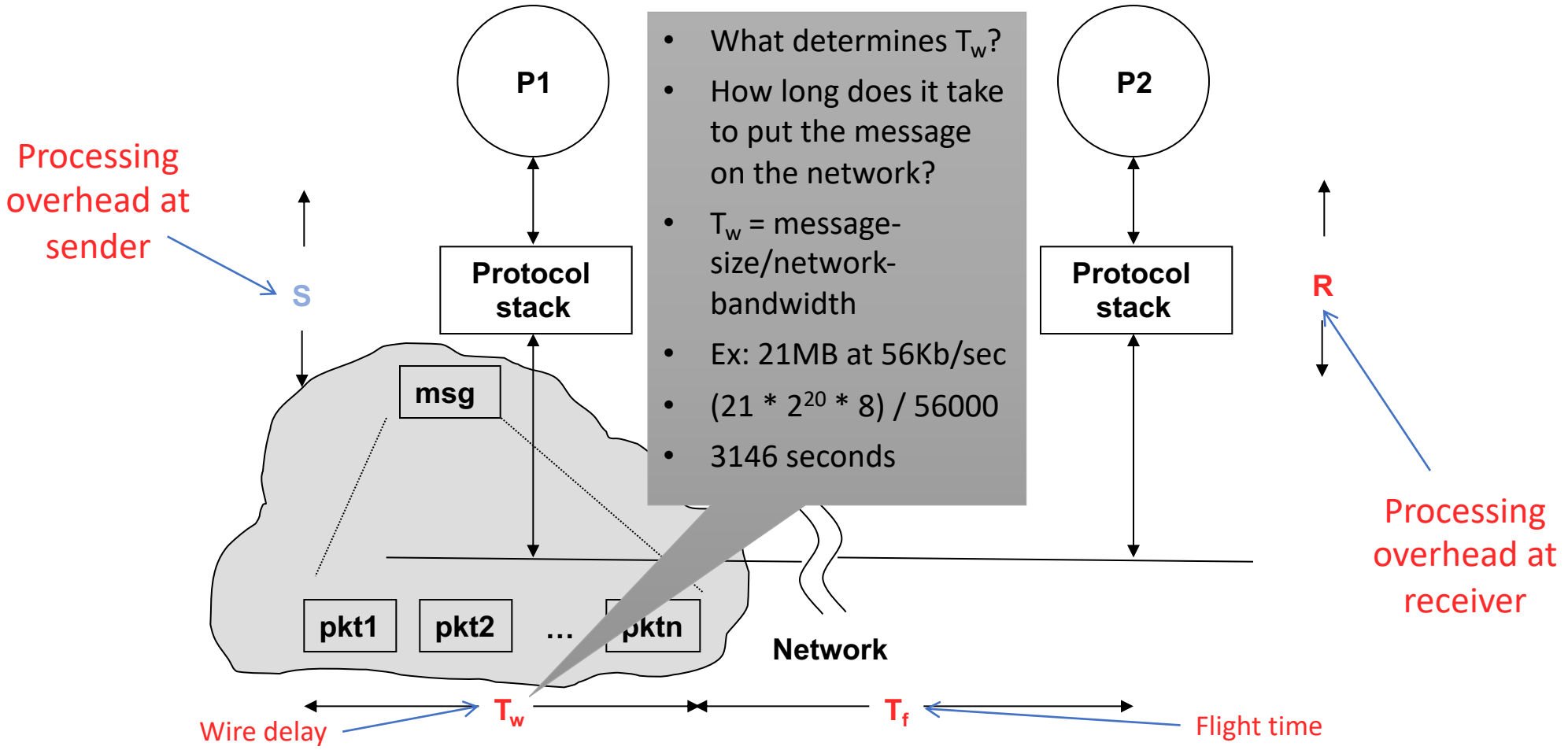
- Twisted pair copper (1m – 100m reach)
  - 10base-T
    100base-T
    1000base-TX
    10Gbase-T
- Fiber optic (100m – 40km reach)
  - 1000base-SX, 1000base-LX
    10Gbase-SR, 10Gbase-LR
    40Gbase-SR4, 40Gbase-LR4
    100Gbase-SR4, 100Gbase-SR10, 100Gbase-LR4
    and many more

What's in a Home Router?



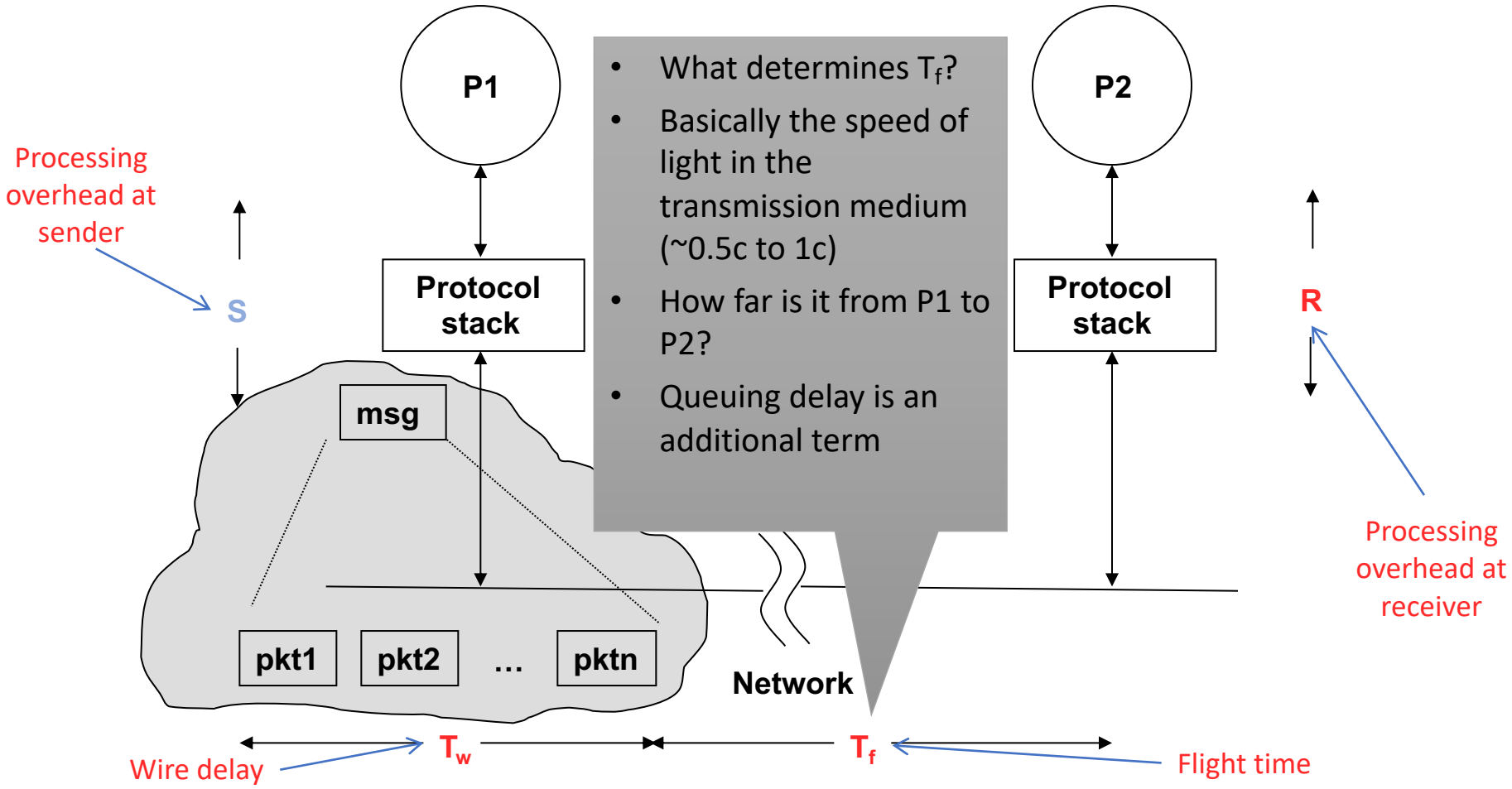
Performance metrics

Transmission time or end-to-end latency = $S+T_W+T_f+R$
Message throughput = message-size/end-to-end-latency



Performance metrics

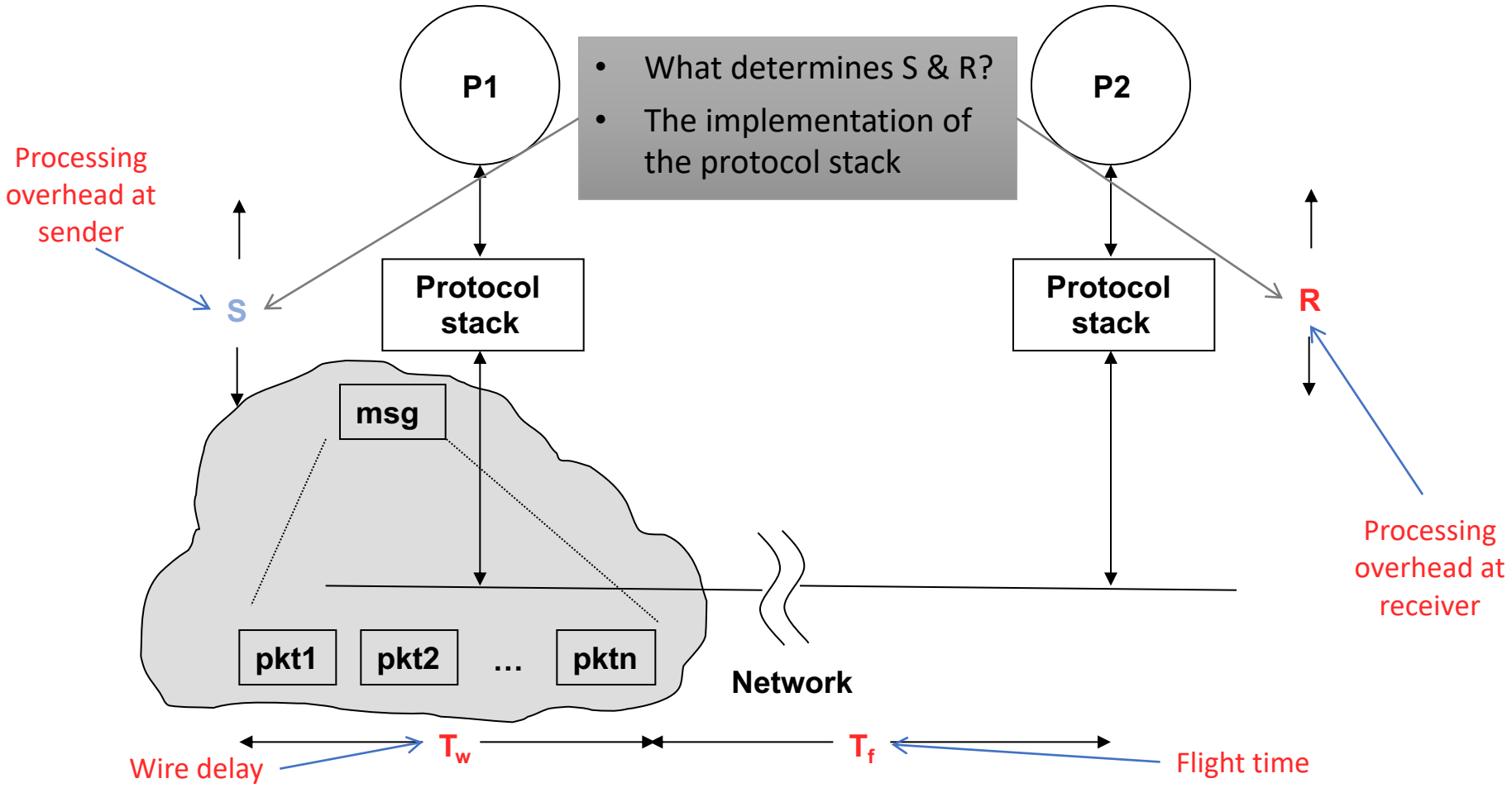
Transmission time or end-to-end latency = $S+T_W+T_f+R$
Message throughput = message-size/end-to-end-latency



- What determines T_f?
- Basically the speed of light in the transmission medium (~0.5c to 1c)
- How far is it from P1 to P2?
- Queuing delay is an additional term

Performance metrics

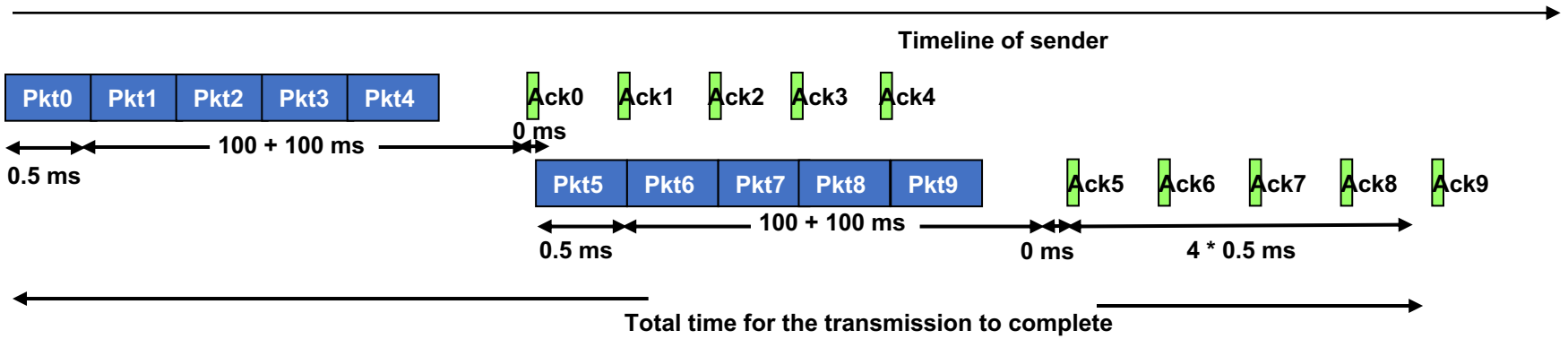
Transmission time or end-to-end latency = $S+T_W+T_f+R$
Message throughput = message-size/end-to-end-latency



Timeline example

100ms latency
10 packets
Window size = 5

Wire delay (data) = 0.5ms
Wire delay (ack) = ~0
Receiver & Sender overhead = ~0



End-to-end latency for Pkt0
= $S + T_w + T_f + R$
0 + 0.5 + 100 + 0
100.5 ms

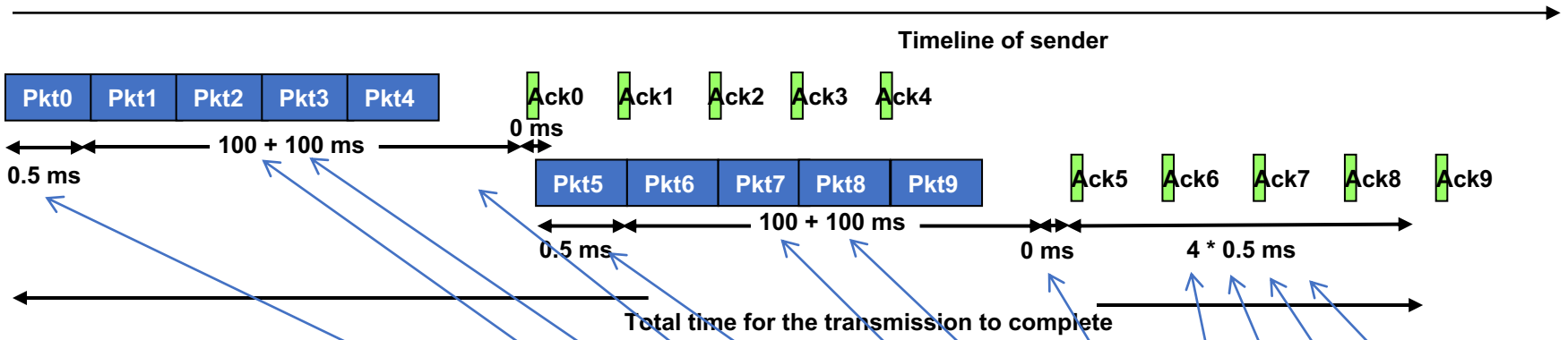
End-to-end latency for Ack0
= $S + T_w + T_f + R$
0 + 0.0 + 100 + 0
100.0 ms

Total transmission time
= 0.5 + 100 + 100 + 0 + 0.5 + 100 + 100 + 0.5 + 0.5 + 0.5 + 0.5
0.5 + 200 + 0.5 + 200 + 4 * 0.5
403 ms

Timeline example

100ms latency
10 packets
Window size = 5

Wire delay (data) = 0.5ms
Wire delay (ack) = ~0
Receiver & Sender overhead = ~0



End-to-end latency for Pkt0
= $S + T_w + T_f + R$
0 + 0.5 + 100 + 0
100.5 ms

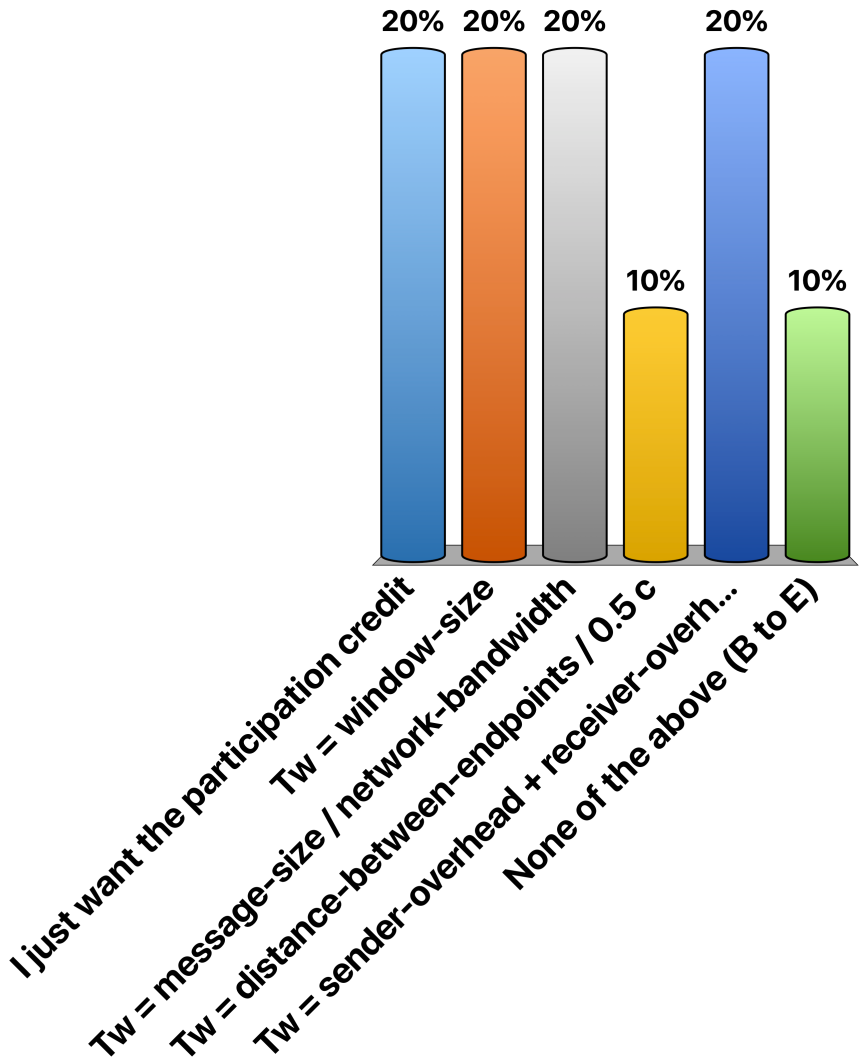
End-to-end latency for Ack0
= $S + T_w + T_f + R$
0 + 0.0 + 100 + 0
100.0 ms

Total transmission time
= 0.5 + 100 + 100 + 0 + 0.5 + 100 + 100 + 0 + 0.5 + 0.5 + 0.5 + 0.5
0.5 + 200 + 0.5 + 200 + 4 * 0.5
403 ms

Session ID: dag2200

In the expression $S + T_w + T_f + R$

A. I just want the participation credit
B. T_w = window-size
C. T_w = message-size / network-bandwidth
D. T_w = distance-between-endpoints / 0.5 *c*
E. T_w = sender-overhead + receiver-overhead
F. None of the above (B to E)

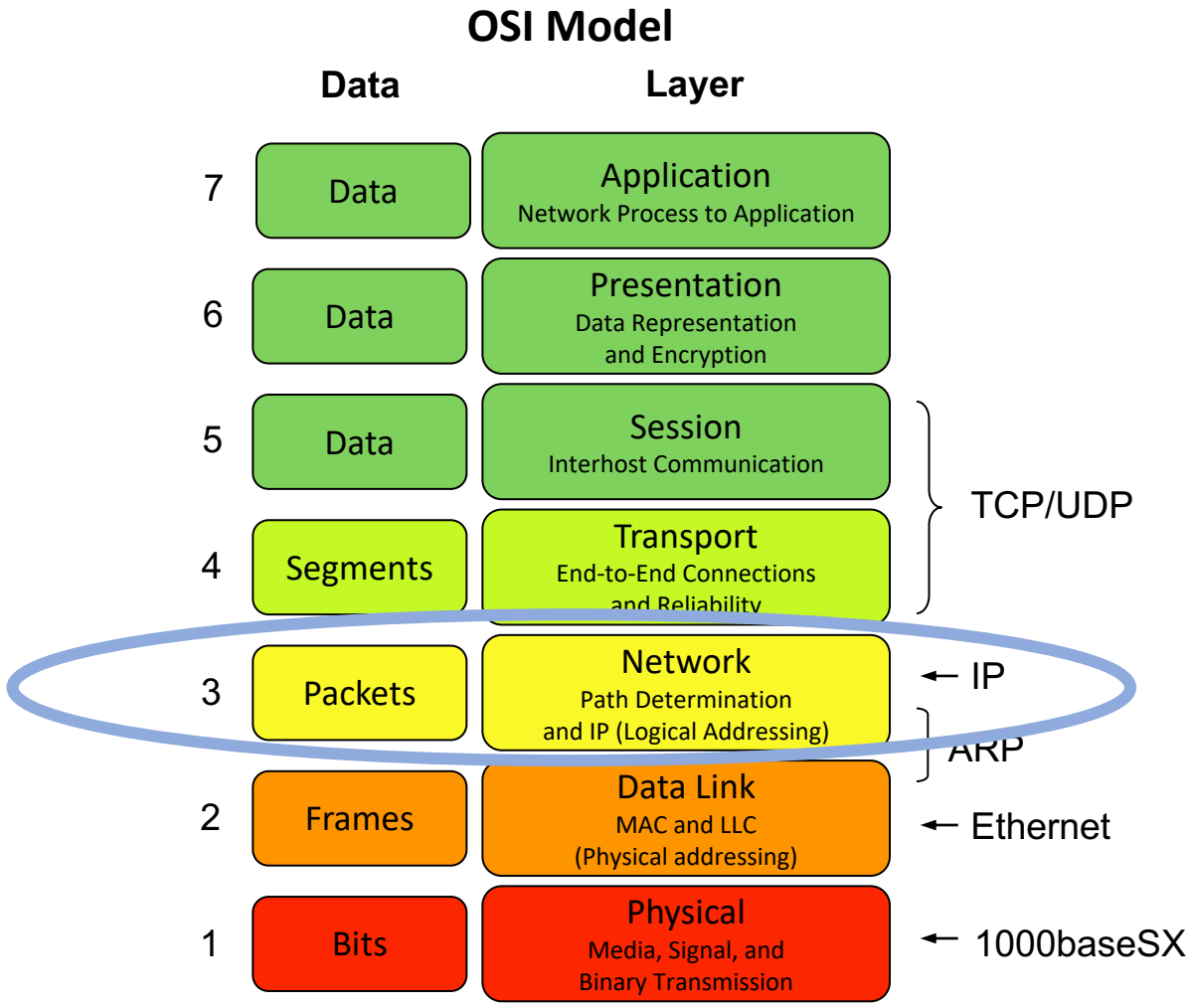


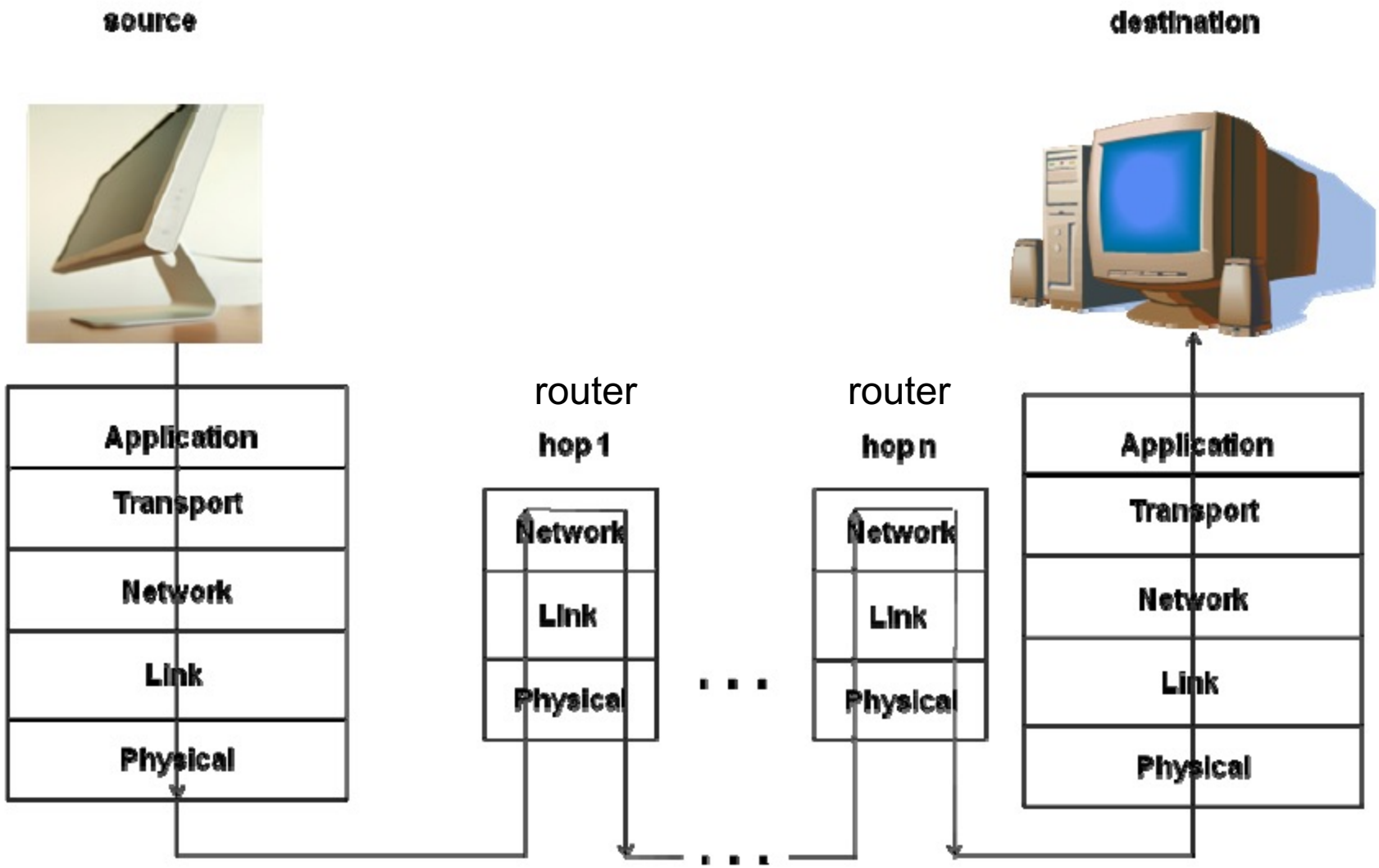
Acknowledgement example

- A transport protocol uses a window of 10 packets.
- For a message that consists of 101 packets, how many acknowledgement packets does the destination generate?
- Assume no packets are lost.
- 101

Back to the network layer

- Why are we back here?
- Recall, IP routing happens here
- We didn't answer a question.
- How does a router know where to route (i.e. who sets up the routing table)?



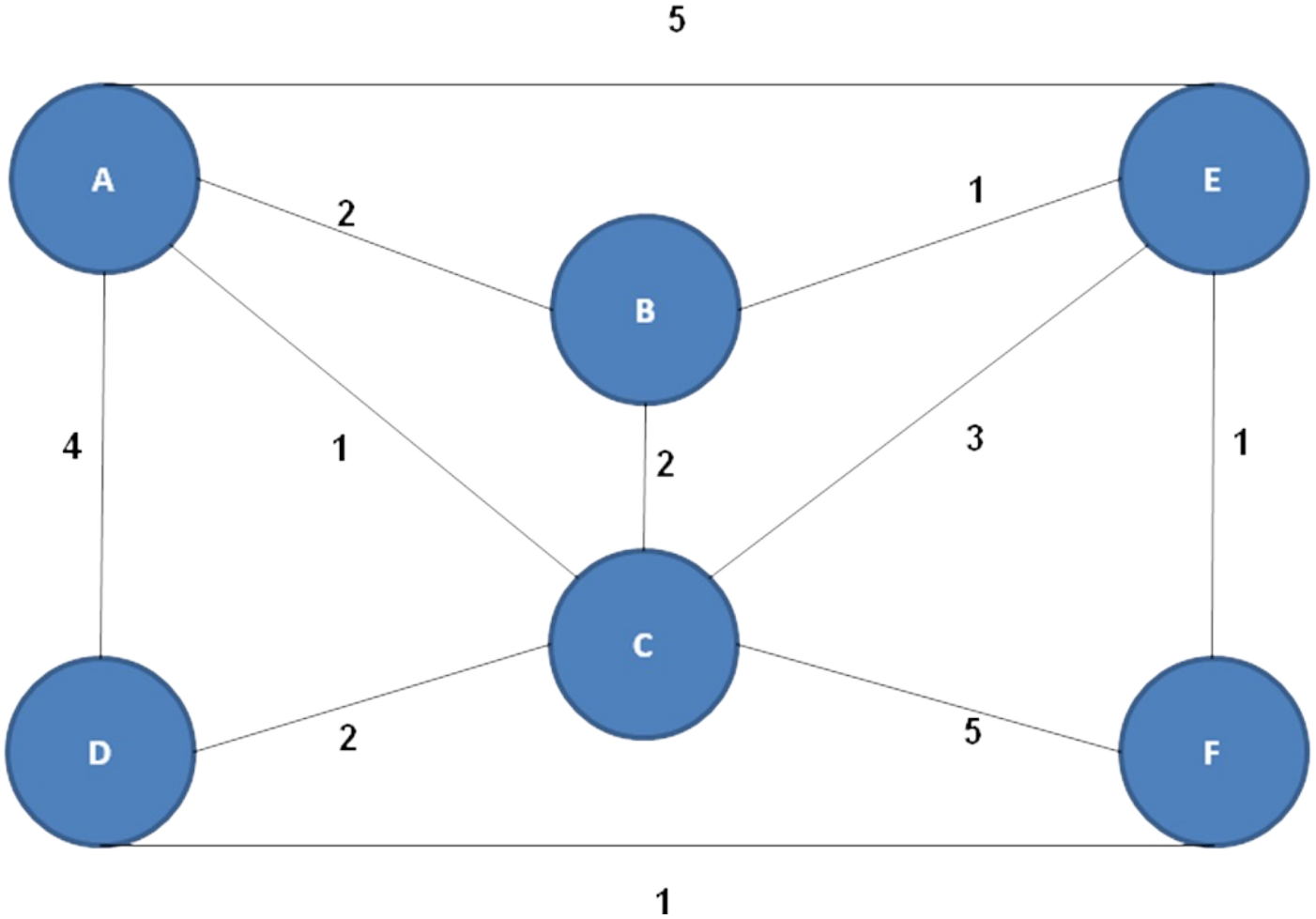

source
destination
router
hop 1
router
hop n
Application
Transport
Network
Link
Physical
Network
Link
Physical
Network
Link
Physical
Application
Transport
Network
Link
Physical

Network layer

- Routing information protocols
  - Link state protocols
  - Distance vector protocols
  - Hierarchical routing
- Addressing

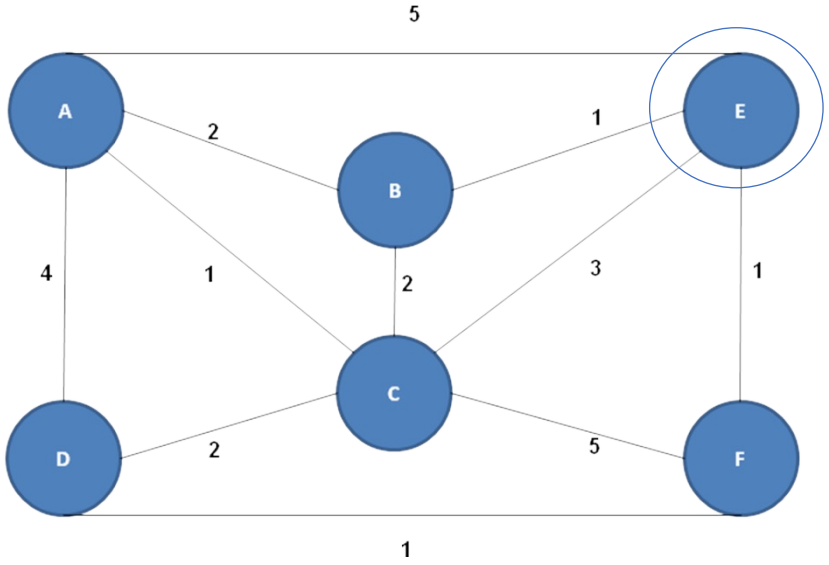
Intuition behind routing algorithms (link state, distance vector)

What's the latency to get to your neighbor?



Distance vector in action

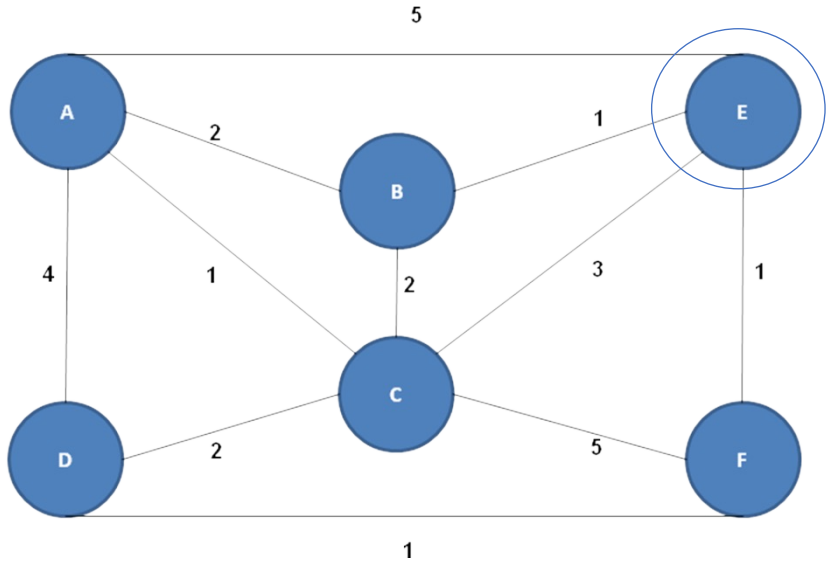
- Uses neighbor info, local algorithm (i.e. every node only sees its neighbors)
- Each node knows the cost to each of its neighbors
- Each node sends its idea of routing to each of its neighbors
- Known colloquially as "routing by rumor"
- For every other node, each node determines the lowest cost next-hop
- Algorithm runs simultaneously on every node
- How does it work?
- Node E knows its costs to its neighbors:



Routes from E	A	B	C	D	F
Cost/next hop	5/A	1/B	3/C	∞/	1/F

Distance vector in action

Routes from E	A	B	C	D	F
Cost/next hop	5/A	1/B	3/C	∞/	1/F

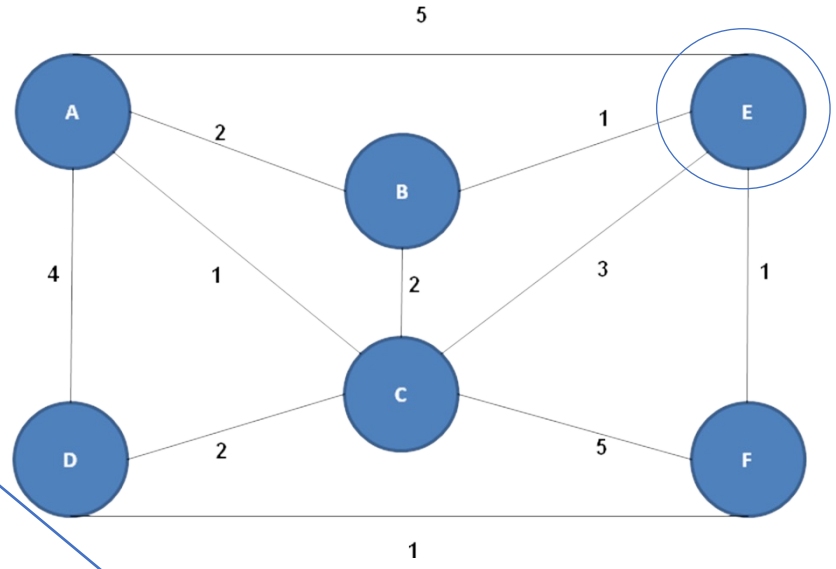


- Node E listens to messages from its neighbors:
  - A says its routes are 0/A, 5/E, 2/B, 1/C, 4/D, 4/F
  - B says its routes are 0/B, 1/E, 2/C, 2/A, 5/F, 4/D
  - C says its routes are 0/C, 1/A, 2/D, 5/F, 3/E, 2/B
  - F says its routes are 0/F, 5/C, 1/D, 1/E, 2/B, 4/A
  - D isn't heard because it isn't connected to E
- Node E looks at each route from each neighbor N; it adds its cost to get to N to N's cost to get to each other node.
- For each node, if the sum is lower cost than what E has in its route table, E changes its own route to run through the neighbor

- Reviewing node A's message
  - 5+2/B, 5+1/C, 5+4/D, 5+4/F
  - Only 9/D is less
  - We'll set the next hop to A

Distance vector in action

Routes from E	A	B	C	D	F
Cost/next hop	5/A	1/B	3/C	9/A	1/F

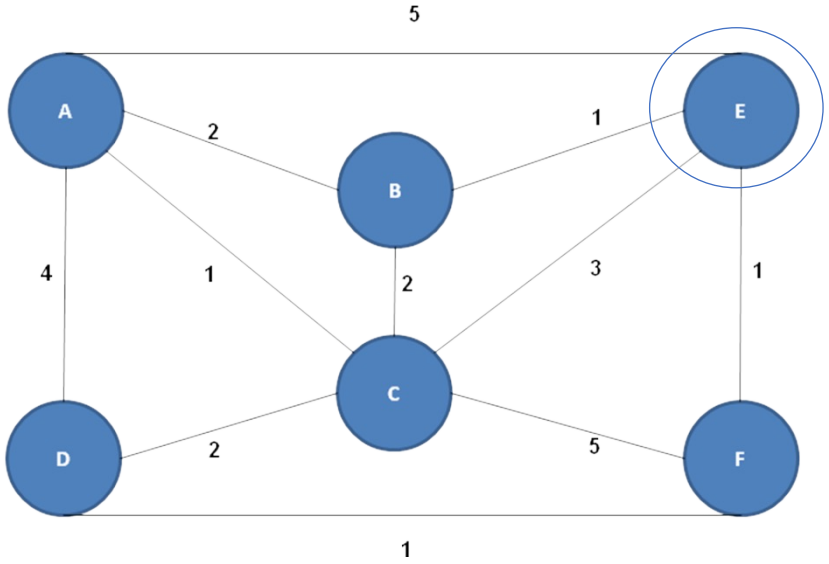


- Node E listens to messages from its neighbors:
  - A says its routes are 0/A, 5/E, 2/B, 1/C, 4/D, 4/F
  - B says its routes are 0/B, 1/E, 2/C, 2/A, 5/F, 4/D
  - C says its routes are 0/C, 1/A, 2/D, 5/F, 3/E, 2/B
  - F says its routes are 0/F, 5/C, 1/D, 1/E, 2/B, 4/A
  - D isn't heard because it isn't connected to E
- Node E looks at each route from each neighbor N; it adds its cost to get to N to N's cost to get to each other node.
- For each node, if the sum is lower cost than what E has in its route table, E changes its own route to run through the neighbor

- Reviewing Node B's message
  - 1+2/C, 1+2/A, 1+5/F, 1+4/D
  - 3/A and 5/D are lower cost
  - We set their next hop to B

Distance vector in action

Routes from E	A	B	C	D	F
Cost/next hop	3/B	1/B	3/C	5/B	1/F

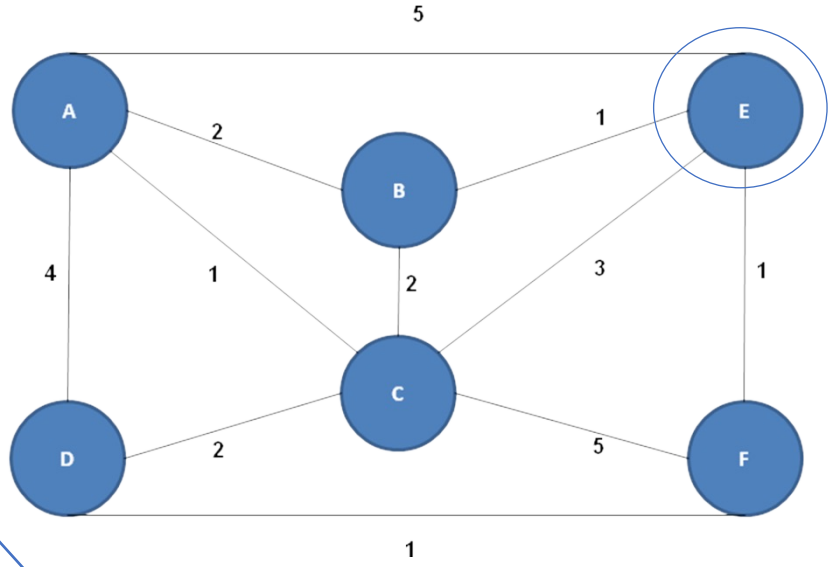


- Node E listens to messages from its neighbors:
  - A says its routes are 0/A, 5/E, 2/B, 1/C, 4/D, 4/F
  - B says its routes are 0/B, 1/E, 2/C, 2/A, 5/F, 4/D
  - C says its routes are 0/C, 1/A, 2/D, 5/F, 3/E, 2/B
  - F says its routes are 0/F, 5/C, 1/D, 1/E, 2/B, 4/A
  - D isn't heard because it isn't connected to E
- Node E looks at each route from each neighbor N; it adds its cost to get to N to N's cost to get to each other node.
- For each node, if the sum is lower cost than what E has in its route table, E changes its own route to run through the neighbor

- Reviewing Node C's message
  - 3+1/A, 3+2/D, 3+5/F, 3+2/B
  - None are lower cost

Distance vector in action

Routes from E	A	B	C	D	F
Cost/next hop	3/B	1/B	3/C	5/B	1/F

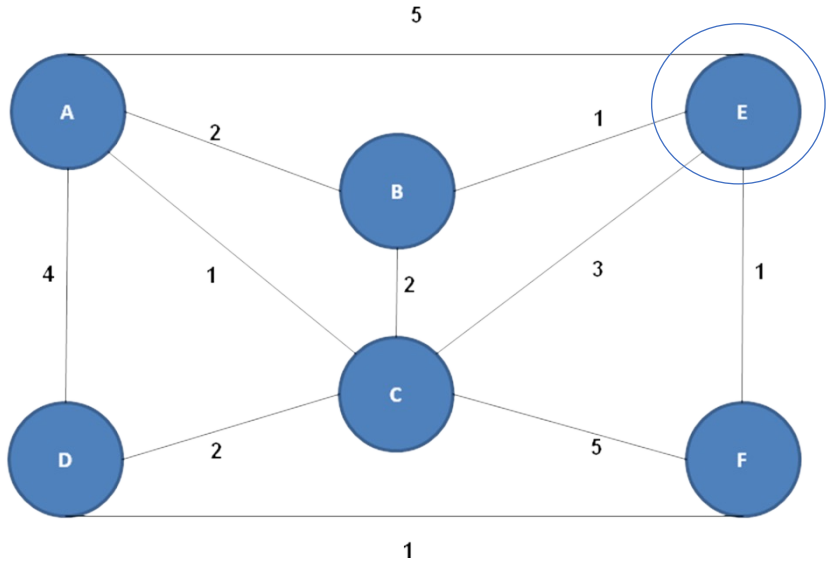


- Node E listens to messages from its neighbors:
  - A says its routes are 0/A, 5/E, 2/B, 1/C, 4/D, 4/F
  - B says its routes are 0/B, 1/E, 2/C, 2/A, 5/F, 4/D
  - C says its routes are 0/C, 1/A, 2/D, 5/F, 3/E, 2/B
  - F says its routes are 0/F, 5/C, 1/D, 1/E, 2/B, 4/A
  - D isn't heard because it isn't connected to E
- Node E looks at each route from each neighbor N; it adds its cost to get to N to N's cost to get to each other node.
- For each node, if the sum is lower cost than what E has in its route table, E changes its own route to run through the neighbor

- Reviewing Node F's message
  - 1+5C, 1+1/D, 1+2/B, 1+4/A
  - 2/D is lower cost
  - We'll set the next hop to F

Distance vector in action

Routes from E	A	B	C	D	F
Cost/next hop	3/B	1/B	3/C	2/F	1/F



- Node E listens to messages from its neighbors:
  - A says its routes are 0/A, 5/E, 2/B, 1/C, 4/D, 4/F
  - B says its routes are 0/B, 1/E, 2/C, 2/A, 5/F, 4/D
  - C says its routes are 0/C, 1/A, 2/D, 5/F, 3/E, 2/B
  - F says its routes are 0/F, 5/C, 1/D, 1/E, 2/B, 4/A
  - D isn't heard because it isn't connected to E
- Node E looks at each route from each neighbor N; it adds its cost to get to N to N's cost to get to each other node.
- For each node, if the sum is lower cost than what E has in its route table, E changes its own route to run through the neighbor

- Do you agree that E now knows the next hop for the lowest cost routes to the other nodes?
- This of course presumed the other nodes knew their lowest cost routes
- When this runs on all nodes simultaneously, they will ***converge*** on this solution in a relatively short time

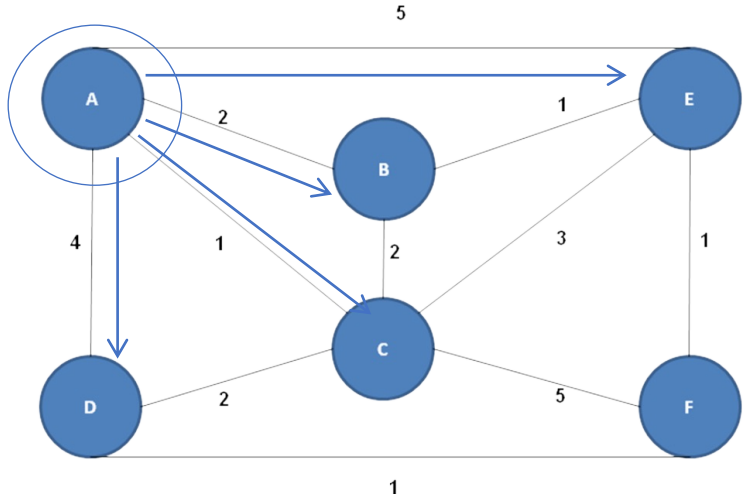
Distance Vector vs Link State Routing

- Distance Vector
  - Each node knows the cost to reach each of its neighbors
  - Each node sends its routing table to only its neighbors
  - Each node revises its own routing table every time it receives a routing table from a neighbor
  - Slower convergence but lower traffic

- Link State
  - Each node advertises its neighbor links and costs to every other node in the network
  - Each node collects the entire topology of the network from these advertisements
  - Using a "shortest path" algorithm, each node calculates its own routing table
  - Faster convergence but more traffic

Link state in action

- Dijkstra's shortest path algorithm
- Uses global info, local algorithm (i.e. every node has this graph)
- n-1 iterations
- One least-cost route to a destination
- Runs simultaneously on each node



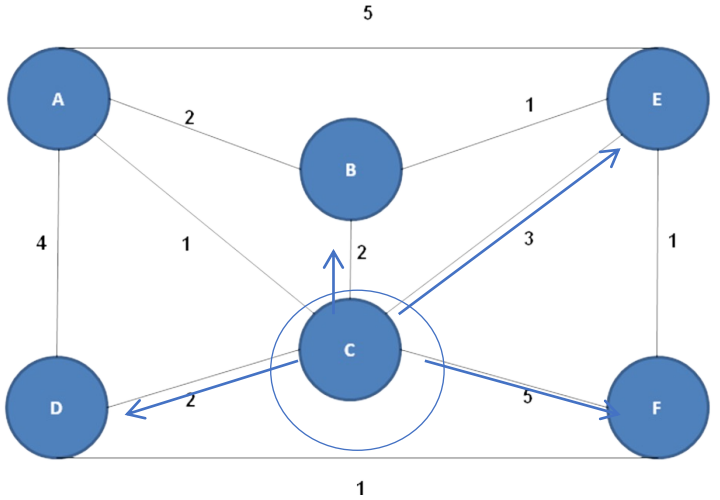
Calculating on Node A

(each node does its own calculation)

Iteration Count	Nodes to which least-cost routes known	B Cost/ route	C Cost/ route	D cost/ route	E Cost/ route	F Cost/ route
Init	A	2/AB	1/AC	4/AD	5/AE	∞
1						
2						
3						
4						
5						

Link state in action

- Dijkstra's shortest path algorithm
- Uses global info, local algorithm (i.e. every node has this graph)
- n-1 iterations
- One least-cost route to a destination
- Runs simultaneously on each node



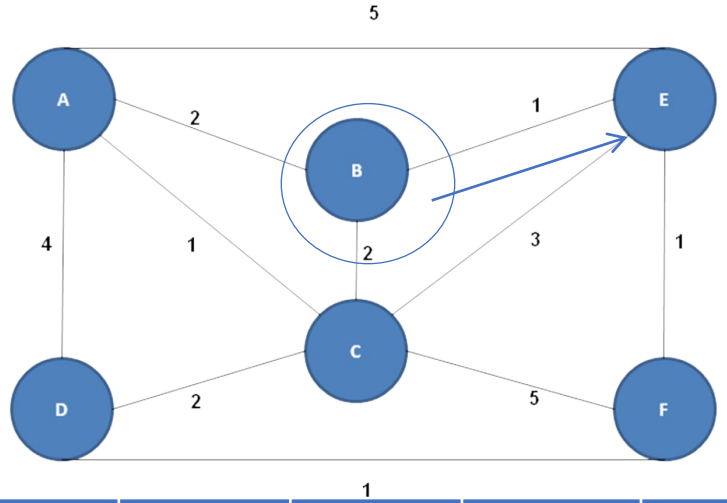
Calculating on Node A

(each node does its own calculation)

Iteration Count	Nodes to which least-cost routes known	B Cost/ route	C Cost/ route	D cost/ route	E Cost/ route	F Cost/ route
Init	A	2/AB	1/AC	4/AD	5/AE	∞
1	A,C	2/AB	1/AC	3/ACD	4/ACE	6/ACF
2						
3						
4						
5						

Link state in action

- Dijkstra's shortest path algorithm
- Uses global info, local algorithm (i.e. every node has this graph)
- n-1 iterations
- One least-cost route to a destination
- Runs simultaneously on each node



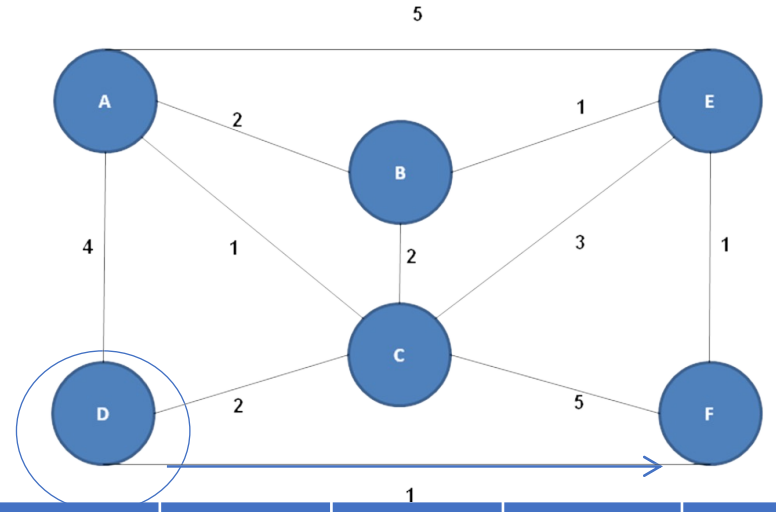
Calculating on Node A

(each node does its own calculation)

Iteration Count	Nodes to which least-cost routes known	B Cost/ route	C Cost/ route	D cost/ route	E Cost/ route	F Cost/ route
Init	A	2/AB	1/AC	4/AD	5/AE	∞
1	A,C	2/AB	1/AC	3/ACD	4/ACE	6/ACF
2	A,C,B	2/AB	-	3/ACD	3/ABE	6/ACF
3						
4						
5						

Link state in action

- Dijkstra's shortest path algorithm
- Uses global info, local algorithm (i.e. every node has this graph)
- n-1 iterations
- One least-cost route to a destination
- Runs simultaneously on each node



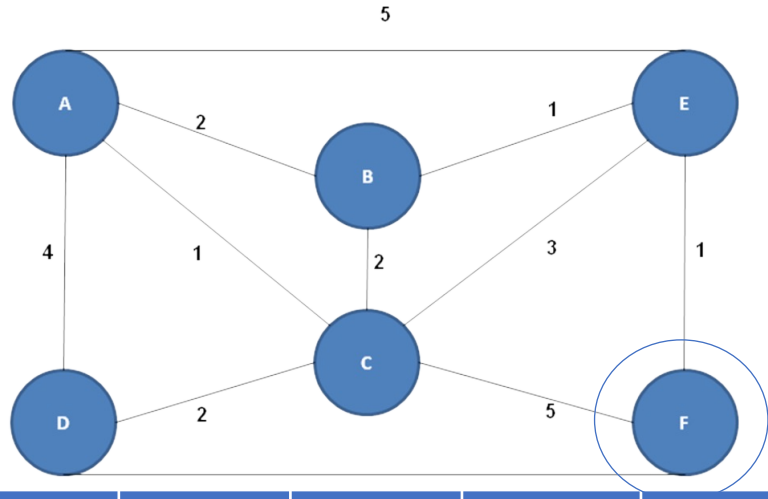
Calculating on Node A

(each node does its own calculation)

Iteration Count	Nodes to which least-cost routes known	B Cost/ route	C Cost/ route	D cost/ route	E Cost/ route	F Cost/ route
Init	A	2/AB	1/AC	4/AD	5/AE	∞
1	A,C	2/AB	1/AC	3/ACD	4/ACE	6/ACF
2	A,C,B	2/AB	-	3/ACD	3/ABE	6/ACF
3	A,C,B,D	-	-	3/ACD	3/ABE	4/ACDF
4						
5						

Link state in action

- Dijkstra's shortest path algorithm
- Uses global info, local algorithm (i.e. every node has this graph)
- n-1 iterations
- One least-cost route to a destination
- Runs simultaneously on each node



Calculating on Node A

(each node does its own calculation)

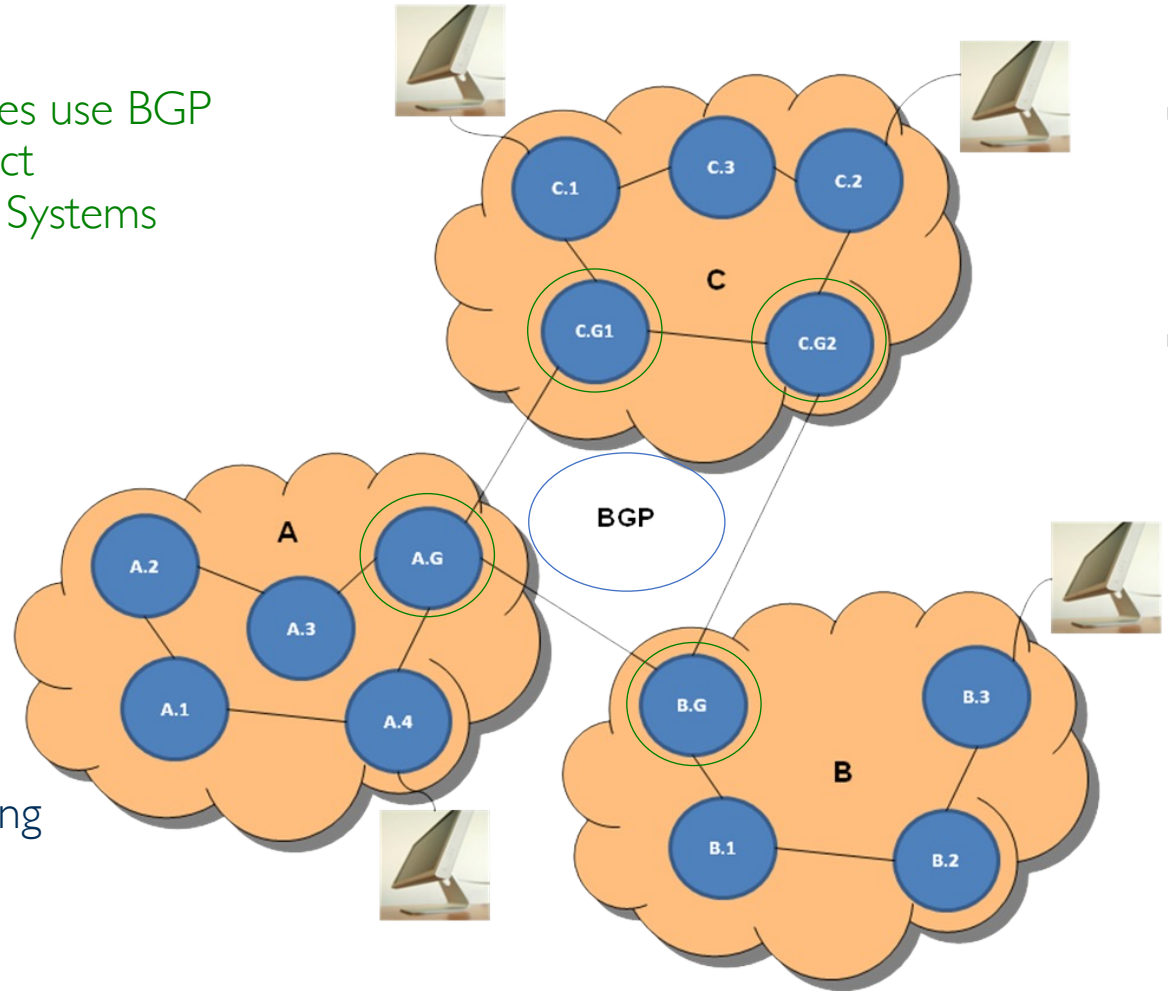
Iteration Count	Nodes to which least-cost routes known	B Cost/ route	C Cost/ route	D cost/ route	E Cost/ route	F Cost/ route
Init	A	2/AB	1/AC	4/AD	5/AE	∞
1	A,C	2/AB	1/AC	3/ACD	4/ACE	6/ACF
2	A,C,B	2/AB	-	3/ACD	3/ABE	6/ACF
3	A,C,B,D	-	-	3/ACD	3/ABE	4/ACDF
4	A,C,B,D,E	-	-	-	3/ABE	4/ACDF
5	A,C,B,D,E,F	-	-	-	-	4/ACDF

Hierarchical routing algorithms

- Network of networks
- Very large scale, frequent state changes
- Compartmentalize so routing information protocol issues don't spread and cause widespread outages
- What better example than the Internet itself!
- Autonomous Systems (AS)
  - It is a collection of IP routing prefixes under the control of common administrative policy that presents a common, clearly defined routing policy to the internet.
  - Gateway nodes connect to other AS gateway nodes for inter-AS routing

Hierarchical routing algorithms

- Gateway nodes use BGP to interconnect Autonomous Systems
- We use **BGP** as the internet inter-AS routing protocol



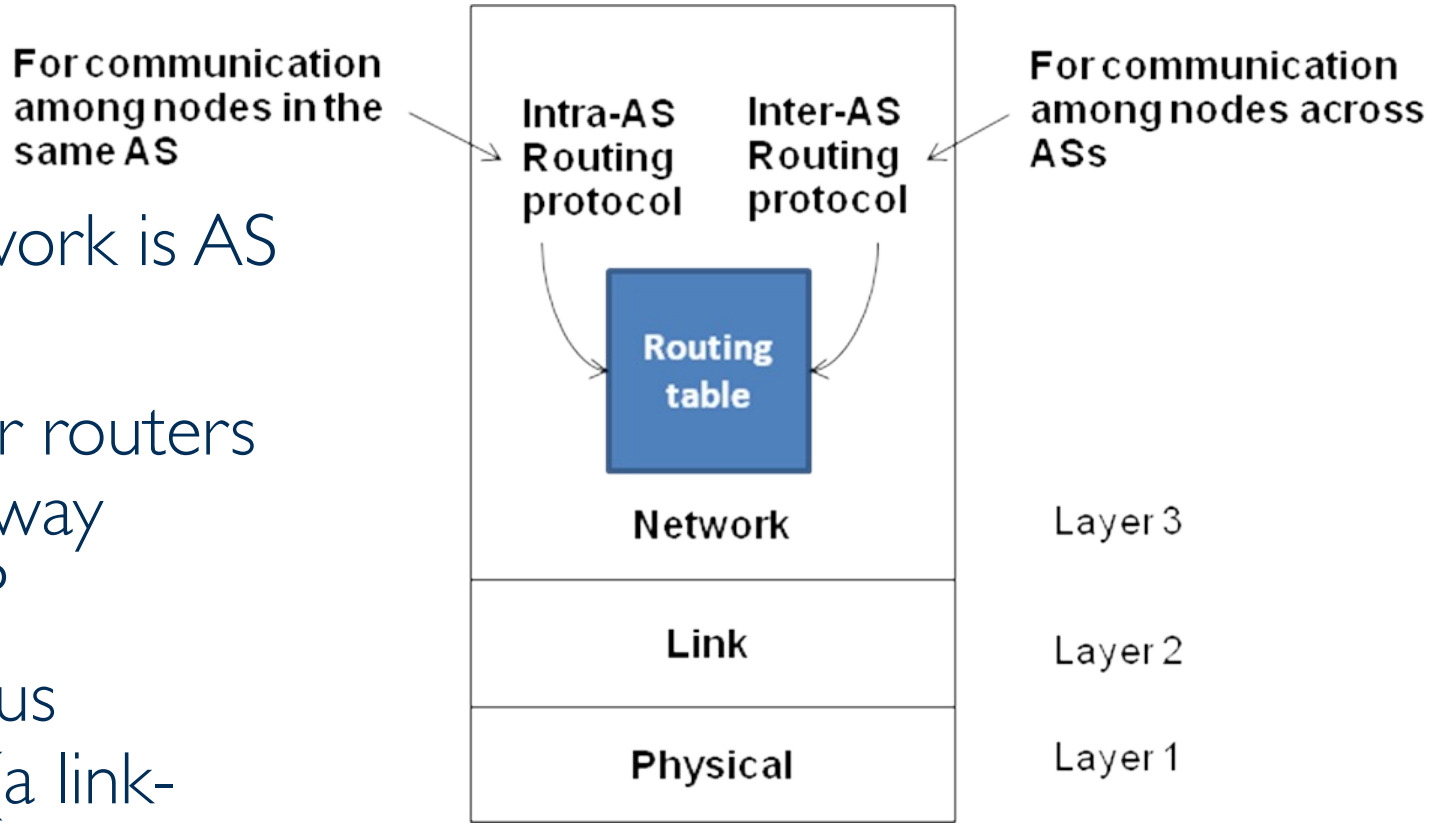
- ASs are assigned numbers by IANA/ARIN
- Nodes within an AS use Link State or Distance Vector (or other) routing protocol

Border Gateway Protocol (BGP)

- BGP is a path-vector protocol. (We're not going into much detail about that.)
- The gateway routers send path-vector messages to advertise the reachability of networks.
- Each router that receives a path vector message must verify the advertised path according to its policy.
- If the message is compliant, the router modifies its routing table and the message before sending the message to the next neighbor.
  - It modifies the routing table to maintain the autonomous systems that are traversed in order to reach the destination system.
  - It modifies the message to add its AS number and to replace the next router entry with its identification.

Routing at GT

- The GT campus network is AS 2367
- There are two Juniper routers that function as Gateway Routers and run BGP
- The rest of the campus network runs OSPF (a link-state routing protocol)



Session ID: dag2200

A host using link state routing protocol

0% A. I just want the participation credit

25% B. Knows only about the routes of its neighbors

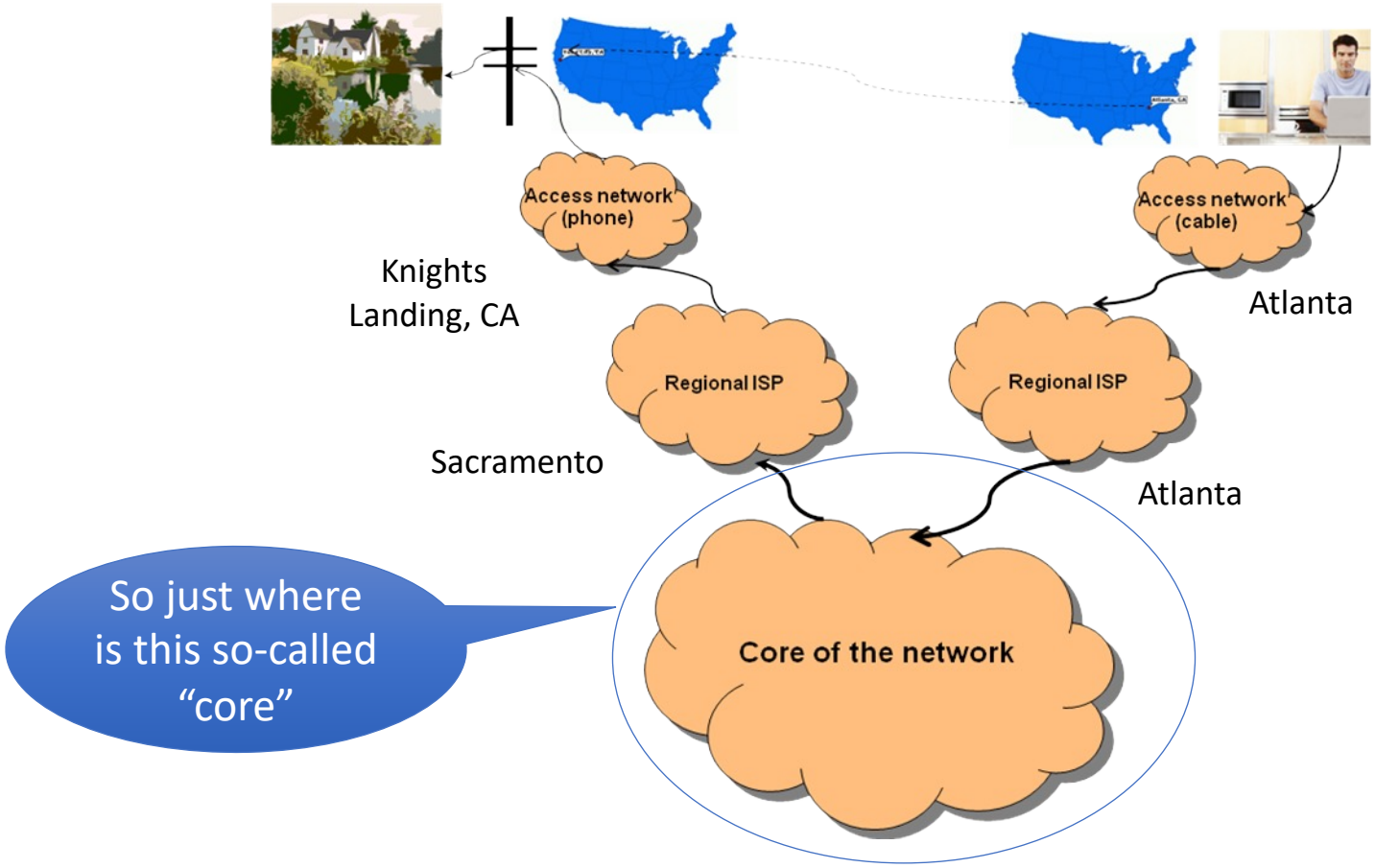
13% C. Knows the routing table of every other host in the local network

38% D. Uses a breadth-first search to calculate distances to other hosts

25% E. Uses less CPU time than a distance vector protocol host would

Remember this slide?

- Now consider an email from Joe to his grandmother




This Jen, is “The Internet”.

There really is no internet core …

- Instead there are dozens of Internet Exchange Points (IXPs) where groups of Internet Service Providers (ISPs) interconnect their networks

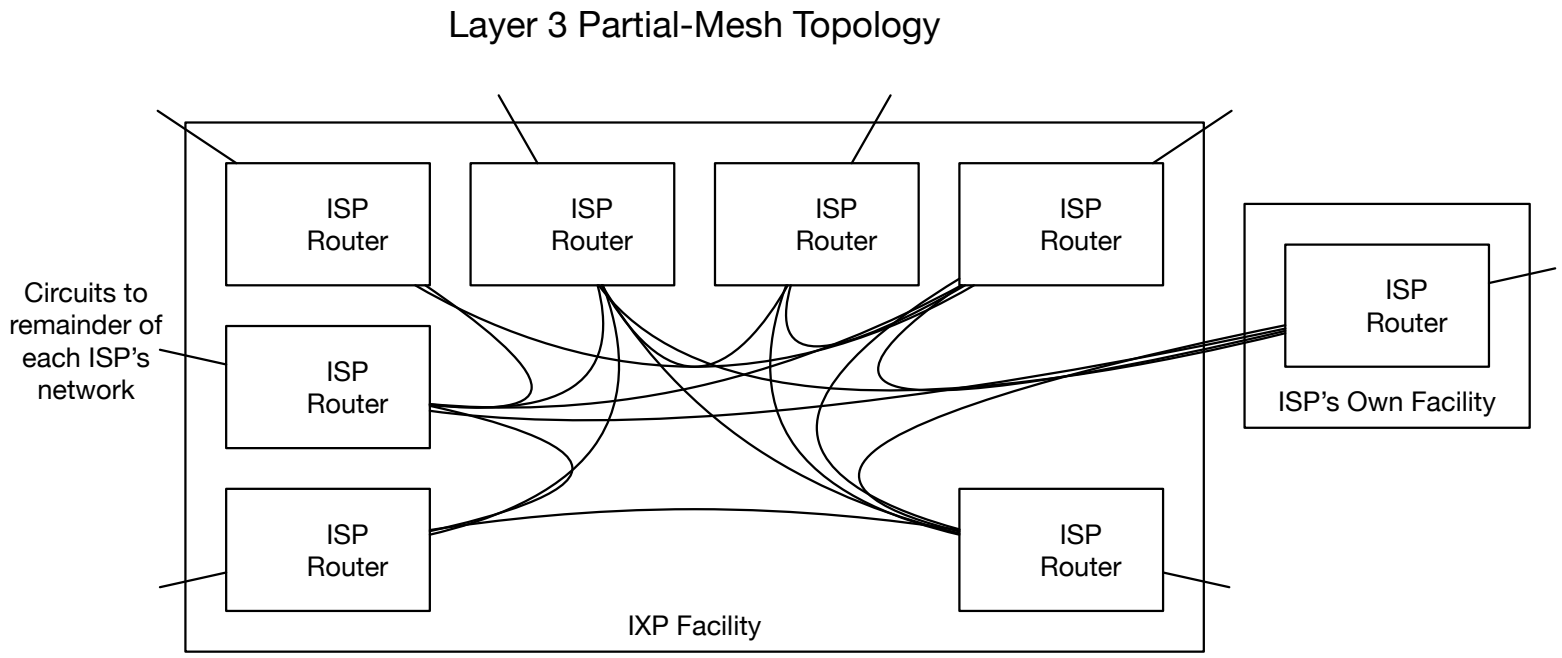
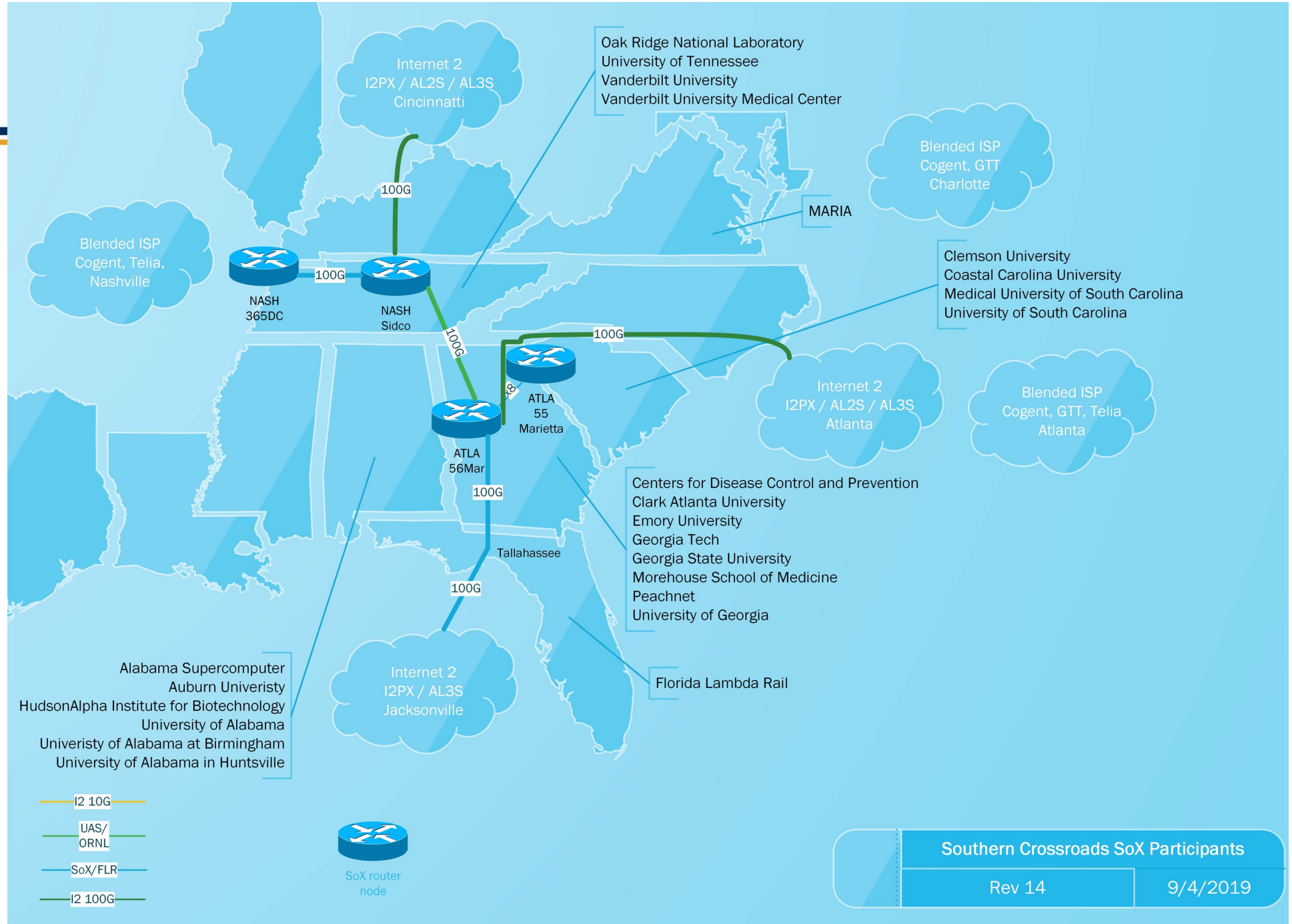


Diagram by Bill Woodcock, Packet Clearing House, rights released for use on Wikipedia

There really is no internet core …

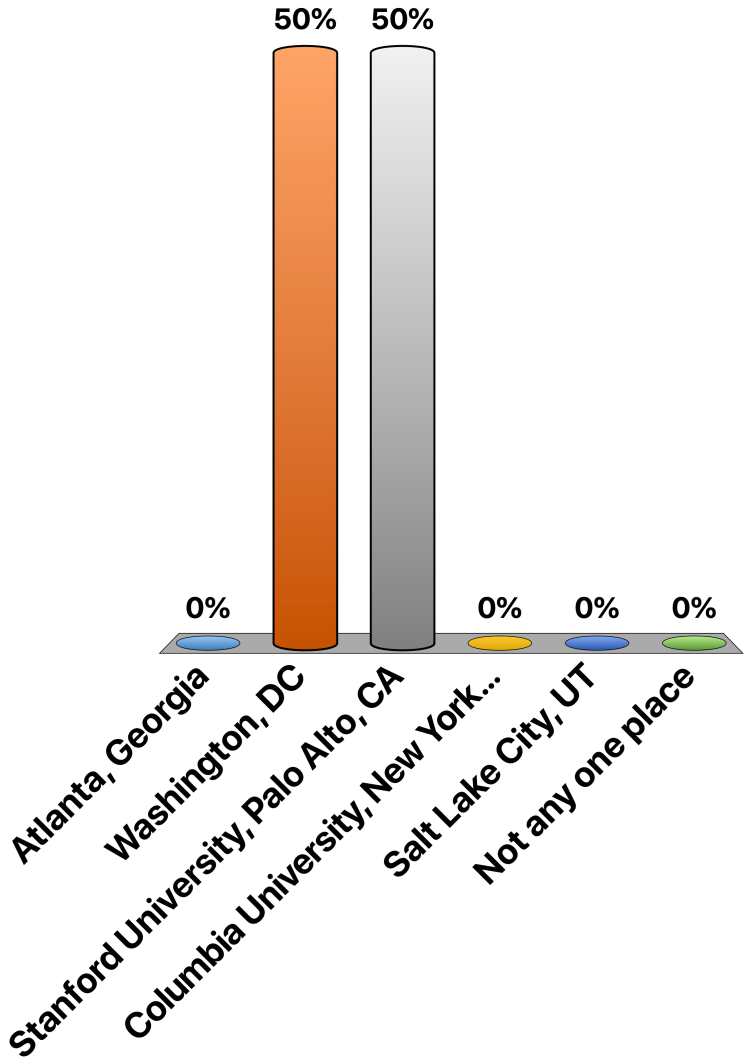
- The ISPs peer using BGP such that all the gateway routers know multiple routes to every regional ISP
- You can begin to see that the Internet has a highly distributed "core" created by multiple IXPs
- There are at least four IXPs in Atlanta; GT participates in and operates an ISP, SoX, which provides network access for about 3 dozen southeastern universities and research centers
- The diagram on the next page shows just the IXPs in which SoX participates. Note that traffic is allowed to traverse SoX networks to get from one ISP to another.
- The diagram doesn't show the SoX peering with other entities such as Google, AT&T, NASA, Dept of Energy, and Comcast


Internet 2
I2PX / AL2S / AL3S
Cincinnati
Oak Ridge National Laboratory
University of Tennessee
Vanderbilt University
Vanderbilt University Medical Center
Blended ISP
Cogent, GTT
Charlotte
MARIA
Blended ISP
Cogent, Telia,
Nashville
100G
NASH
365DC
NASH
Sidco
Clemson University
Coastal Carolina University
Medical University of South Carolina
University of South Carolina
100G
100G
100G
ATLA
55
Marietta
Internet 2
I2PX / AL2S / AL3S
Atlanta
Blended ISP
Cogent, GTT, Telia
Atlanta
ATLA
56Mar
100G
Centers for Disease Control and Prevention
Clark Atlanta University
Emory University
Georgia Tech
Georgia State University
Morehouse School of Medicine
Peachnet
University of Georgia
Tallahassee
100G
Internet 2
I2PX / AL3S
Jacksonville
Florida Lambda Rail
Alabama Supercomputer
Auburn Univeristy
HudsonAlpha Institute for Biotechnology
University of Alabama
Univeristy of Alabama at Birmingham
University of Alabama in Huntsville
I2 10G
UAS/
ORNL
SoX/FLR
I2 100G
SoX router
node
Southern Crossroads SoX Participants
Rev 14
9/4/2019

Session ID: dag2200

Where is the core of the internet located?

A. Atlanta, Georgia
B. Washington, DC
C. Stanford University, Palo Alto, CA
D. Columbia University, New York City
E. Salt Lake City, UT
F. Not any one place



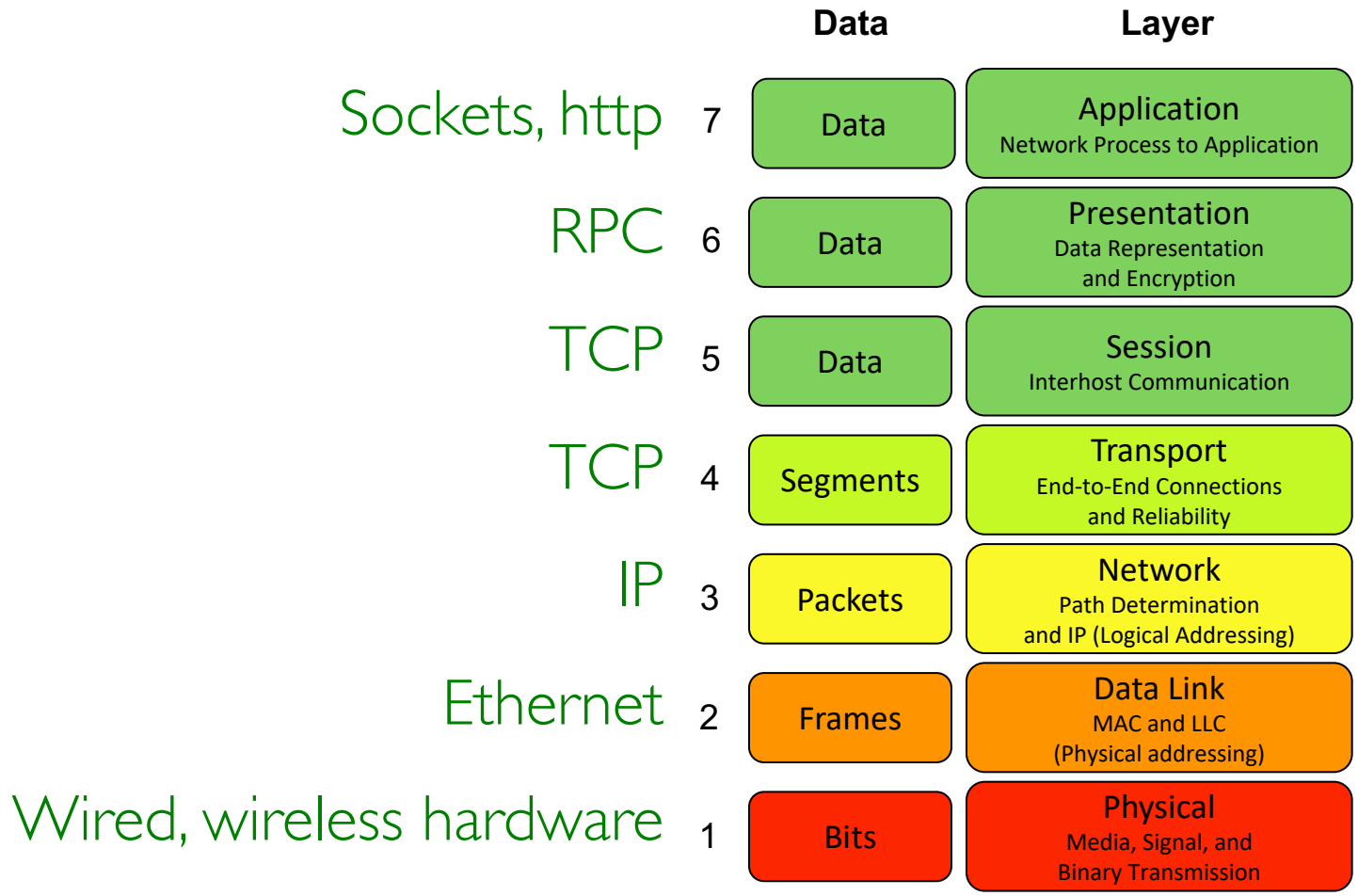
Network Layer Summary

Network Terminology	Definition/Use
Circuit switching	A network layer technology used in telephony. Reserves the network resources (link bandwidth in all the links from source to destination) for the duration of the call; no queuing or store-and-forward delays
Packet switching	A network layer technology used in wide area Internet. It supports best effort delivery of packets from source to destination without reserving any network resources en route.
Message switching	Similar to packet switching but at the granularity of the whole message (at the transport level) instead of packets.
Switch/Router	A device that supports the network layer functionality. It may simply be a computer with a number of network interfaces and adequate memory to serve as input and output buffers.
Input buffers	These are buffers associated with each input link to a switch for assembling incoming packets.
Output buffers	These are buffers associated with each outgoing link from a switch if in case the link is busy.
Routing table	This is table that gives the next hop to be used by this switch for an incoming packet based on the destination address. The initial contents of the table as well as periodic updates are a result of routing algorithms in use by the network layer.

Network Layer Summary

Network Terminology	Definition/Use
Delays	The delays experienced by packets in a packet-switched network
Store and forward	This delay is due to the waiting time for the packet to be fully formed in the input buffer before the switch can act on it.
Queuing	This delay accounts for the waiting time experienced by a packet on either the input or the output buffer before it is finally sent out on an outgoing link.
Packet loss	This is due to the switch having to drop a packet due to either the input or the output buffer being full and is indicative of traffic congestion on specific routes of the network.
Service Model	This is the contract between the network layer and the upper layers of the protocol stack. Both the datagram and virtual circuit models used in packet-switched networks provide **best effort delivery** of packets.
Virtual Circuit (VC)	This model sets up a virtual circuit between the source and destination so that individual packets may simply use this number instead of the destination address. This also helps to simplify the routing decision a switch has to make on an incoming packet. (ATM and X.25)
Datagram	This model does not need any call setup or tear down. Each packet is independent of the others and the switch provides a best effort service model to deliver it to the ultimate destination using information in its routing table. (IP)

Summary

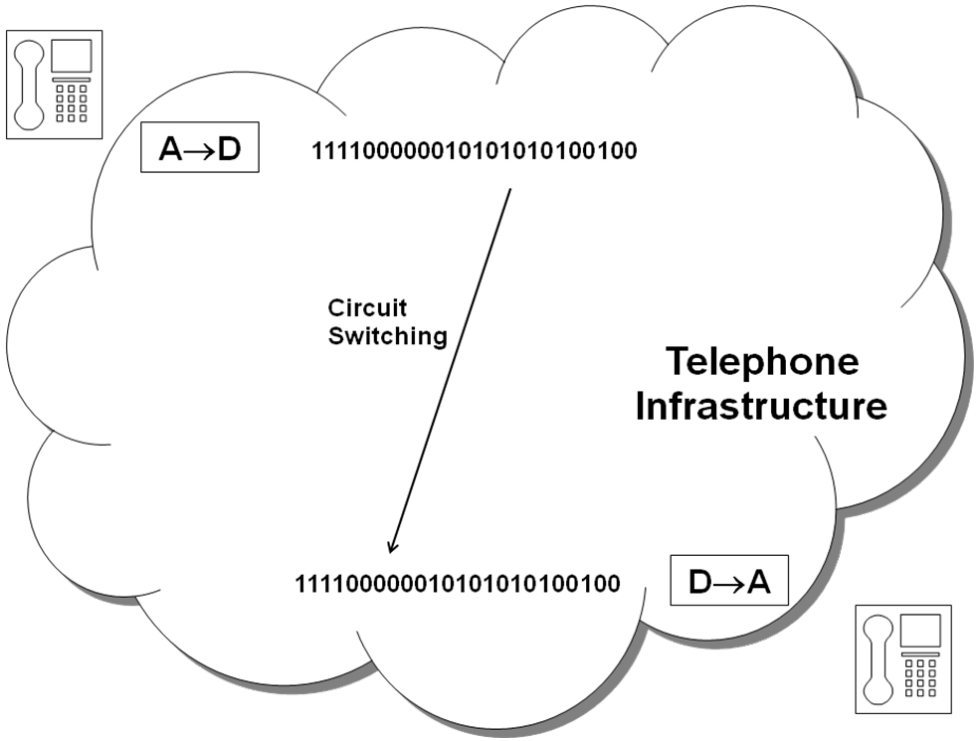

Data
Layer
Sockets, http
7
Data
Application
Network Process to Application
RPC
6
Data
Presentation
Data Representation
and Encryption
TCP
5
Data
Session
Interhost Communication
TCP
4
Segments
Transport
End-to-End Connections
and Reliability
IP
3
Packets
Network
Path Determination
and IP (Logical Addressing)
Ethernet
2
Frames
Data Link
MAC and LLC
(Physical addressing)
Wired, wireless hardware
1
Bits
Physical
Media, Signal, and
Binary Transmission

A brief history of electronic communication

- Credit for the following slides goes to Prof. Kishore Ramachandran, which provide interesting historical perspective
- Wikipedia is a great place to look for more detail on these events

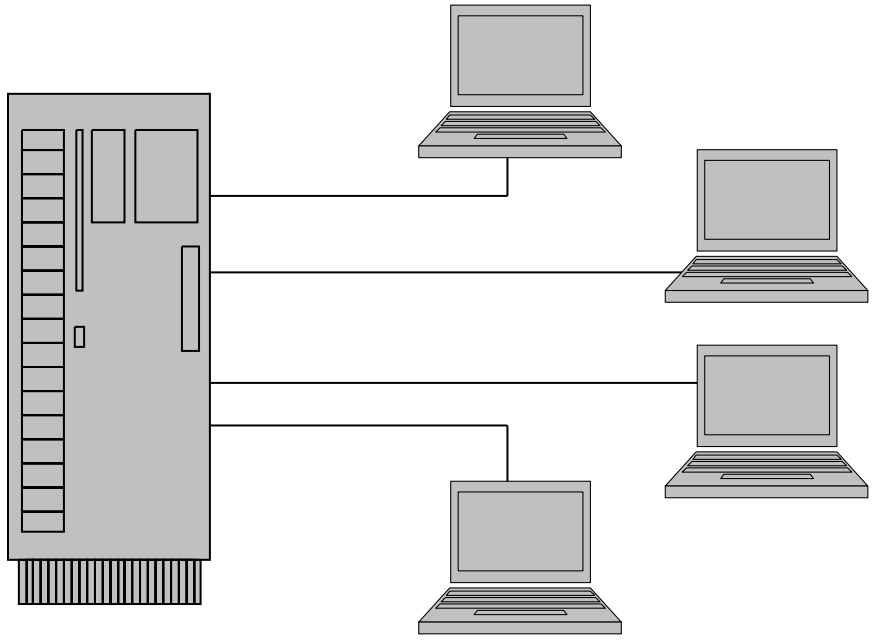
From Telephony to Computer Networking

- 1875 Telephone invented…analog system
- 1960 Telephone infrastructure goes digital

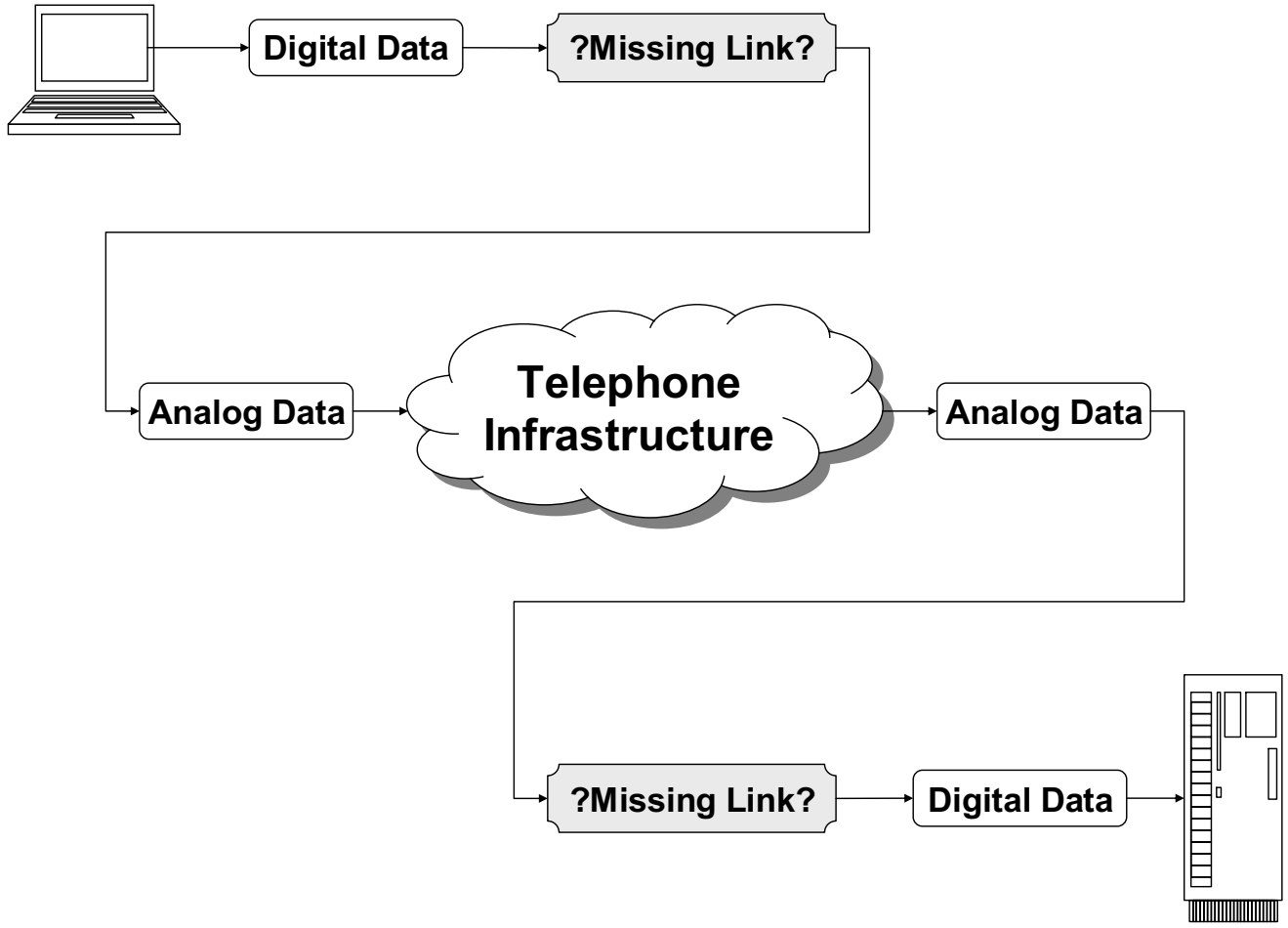


From Telephony to Computer Networking

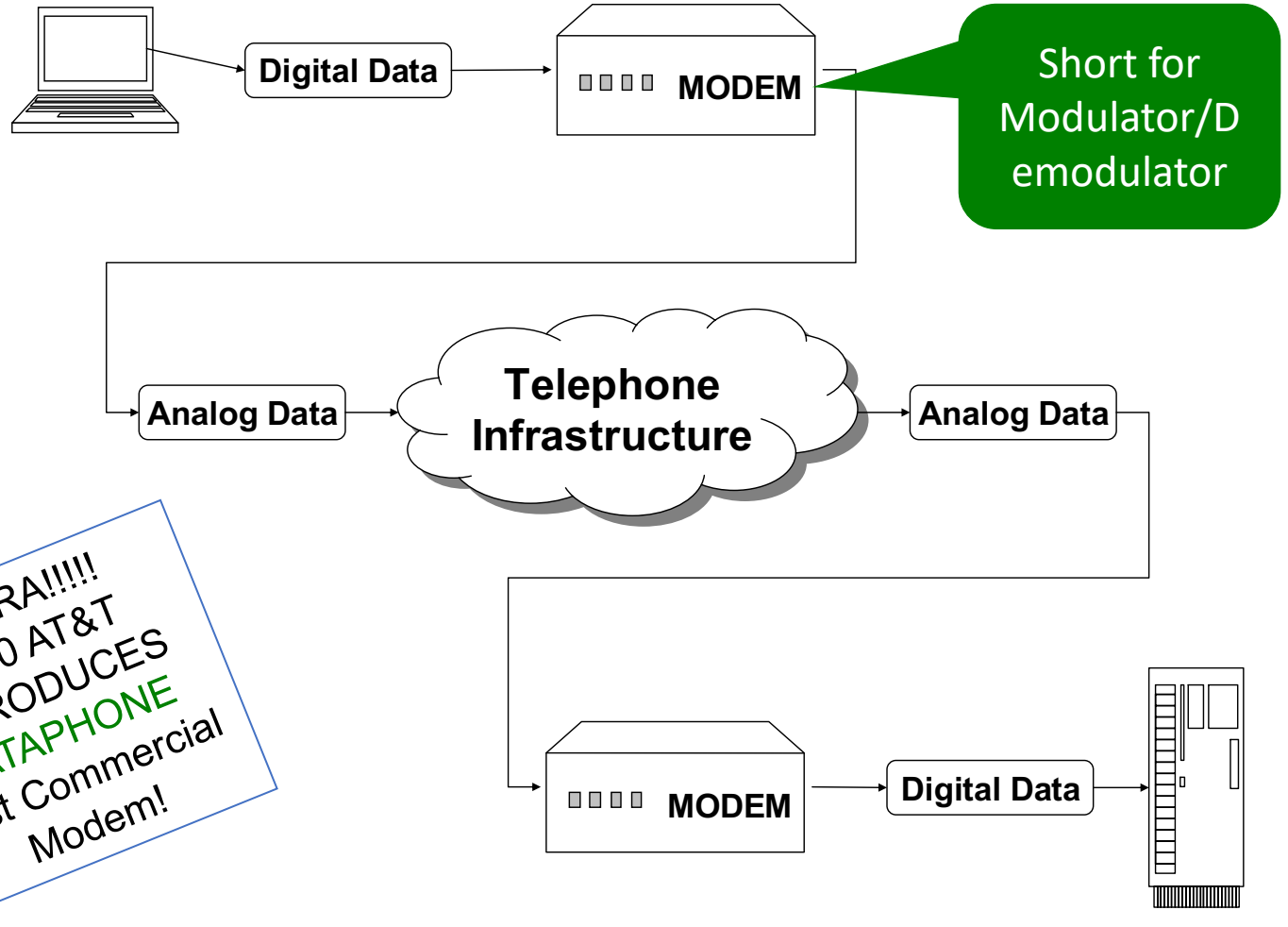
- 1940's Mainframe computers developed
- 1960's Transition
  - Batch-oriented card-input/output
  - CRT I/O and timesharing



From Telephony to Computer Networking



From Telephony to Computer Networking



EXTRA!!!!!
1960 AT&T INTRODUCES DATAPHONE
First Commercial Modem!

From Telephony to Computer Networking

- 1968/9 Carterphone decision allowed devices which were beneficial and not harmful to the network to be connected to the Public Switched Telephone Network (PSTN).
- Allowed non-Bell system equipment to be connected and paved the way for computers to communicate using the telephone switching infrastructure.

Evolution of the Internet

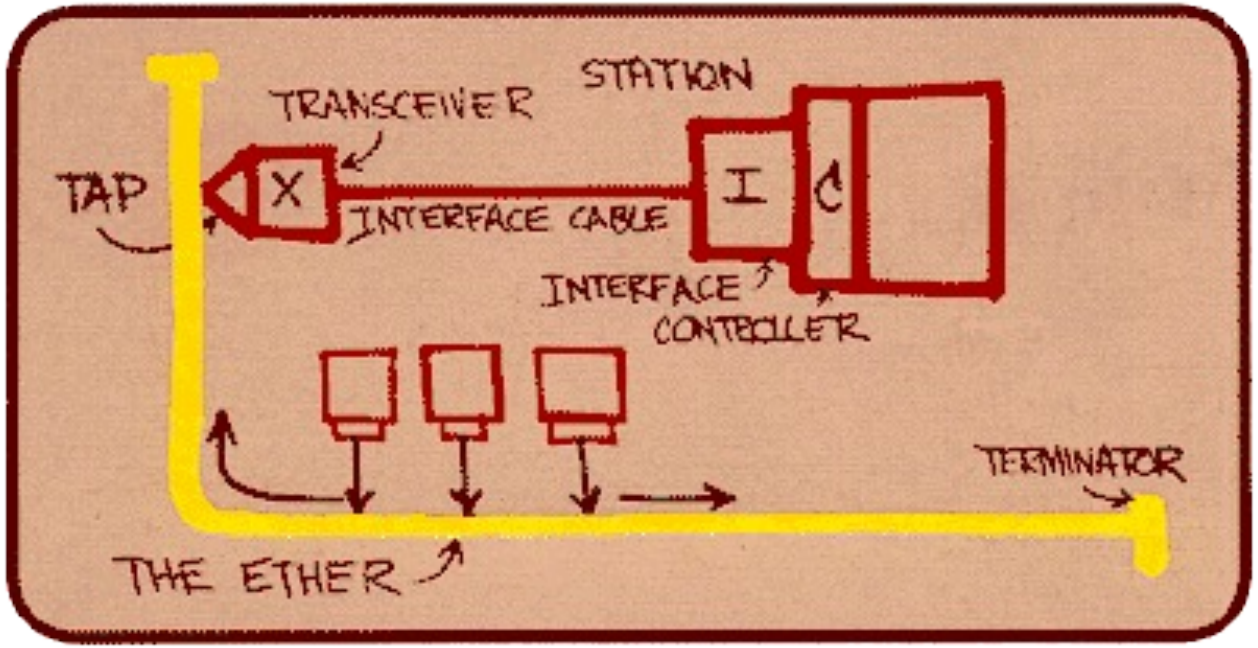
- 1965 DoD DARPA plans first computer network
- 1969 ARPANET connects 4 computers using packet switched network
  - Stanford Research Institute, UCLA, UC Santa Barbara, and the University of Utah
  - Networking luminary Leonard Kleinrock, is credited with successfully sending the first network "message" from UCLA to Stanford.

Evolution of the Internet

- "Router" in the network was called *Interface Message Processor (IMP)*, built by a company called BBN (which stands for Bolt, Beranak, and Newman Inc.).
  - IMP system architecture required a careful balance of the hardware and software that would allow it to be used as a store-and-forward packet switch among these computers.
  - IMP's used modems and leased telephone lines to connect to one another.
- 1971 The ARPANET grows to 23 hosts connecting universities and government research centers around the country.

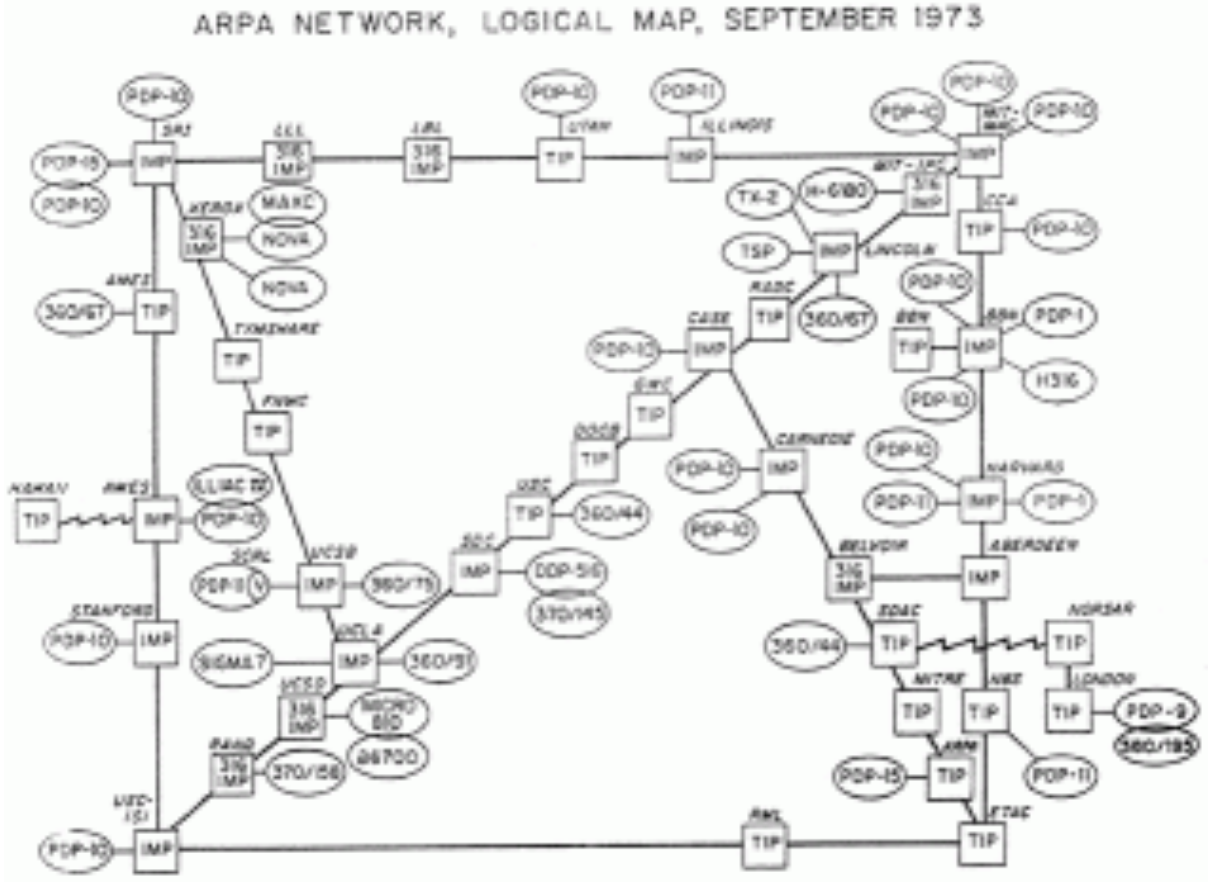
Evolution of the Internet

1973 Robert Metcalfe and David Boggs invent the Ethernet networking system at the Xerox Palo Alto Research Center.



Evolution of the Internet

- 1973 The ARPANET goes international



Evolution of the Internet

- 1975 Internet operations transferred to the Defense Communications Agency
- 1978 Hayes Microcomputer Products releases the first mass-market modem, transmitting at 300 bps (0.3K).
- 1980 John Shoch at Xerox creates the first "worm" program, with the capacity to travel through networks.
- 1981 Ungermann-Bass ships the first commercial Ethernet network interface card.

Evolution of the Internet

- 1981 ARPANET has 213 hosts. A new host is added approximately once every 20 days.
- 1982 The term 'Internet' is used for the first time.
- 1983 TCP/IP becomes the universal language of the Internet. Developed by Vinton Cerf and Robert Kahn
- 1984 CISCO founded
- Early 80's Unix and IBM OS included TCP/IP

Evolution of the Internet

- Late 90's Internet becomes household term
  - Needed PC
  - Needed "Killer app" i.e. WWW & browsers

PC and the arrival of LAN

- 1971 Intel introduces the first microprocessor - the Intel 4004.
- 1971 The Kenbak-1, the first microcomputer, is introduced in Scientific American, selling a total of 40 units in 2 years.

  Used 130 IC's with a 256 byte memory and 8-bit words, processed 1000 instructions per second, and cost $750.

PC and the arrival of LAN

- 1972 Intel launches the 8-bit 8008 - the first microprocessor which could handle both upper and lowercase characters.
- 1972 Xerox develops the Xerox Alto - the first computer to use a Graphic User Interface.

The **Alto** consists of four major parts: the graphics **display**, the **keyboard**, the graphics **mouse**, and the disk s**torage/processor** box. Each **Alto** is housed in a beautifully formed, textured beige metal cabinet that hints at its $32,000 price tag (1979US money). With the exception of the disk storage/processor box, everything is designed to sit on a desk or tabletop

PC and the arrival of LAN

- 1974 Intel introduces the 8080 microprocessor
  - 5 times faster than the 8008.
  - And the heart of the future Altair 8800.
- 1975 MITS markets the Altair 8800 - the first mass-market microcomputer, launching the Personal Computer Revolution.
- 1975 Bill Gates and Paul Allen form the Microsoft company to create software for the new Altair 8800.

PC and the arrival of LAN

- 1976 Apple Computer is formed by Steve Jobs, Steve Wozniak, and Ron Wayne, and launches the Apple Computer.
- 1977 Tandy Radio Shack ships its first personal computer - the TRS-80. It sells over 10,000 units, tripling expectations.
- 1977 Apple Computer launches the Apple II, which sets new standards for sophisticated personal computer systems.

PC and the arrival of LAN

- 1978 The C programming language is completed at AT&T Bell Laboratories, offering a new level of programming.
- 1978 Apple and Tandy ship PCs with 5.25" floppy disks, replacing cassette tape as the standard storage medium for PCs.
- 1978 Hayes Microcomputer Products releases the first mass-market modem, transmitting at 300 bps (0.3K).

PC and the arrival of LAN

- 1978 Intel ships the Intel 8086 microprocessor, with 29,000 transistors, and running at 4.77 megahertz.
- 1979 Personal Software creates VisiCalc for the Apple II, the first electronic spreadsheet program, selling over 100,000 copies.
- 1979 Intel develops the 8088 microprocessor, which would later become the heart of the IBM PC.

PC and the arrival of LAN

- 1979 Motorola develops the Motorola 68000 microprocessor, offering a new level of processing power.
- 1979 Robert Metcalf founded 3COM
- 1980 Seagate Technology introduces the first microcomputer hard disk, capable of holding 5 megabytes of data.
- 1980 Philips introduces the first optical laser disk, with many times the storage capacity of floppy or hard disks.

PC and the arrival of LAN

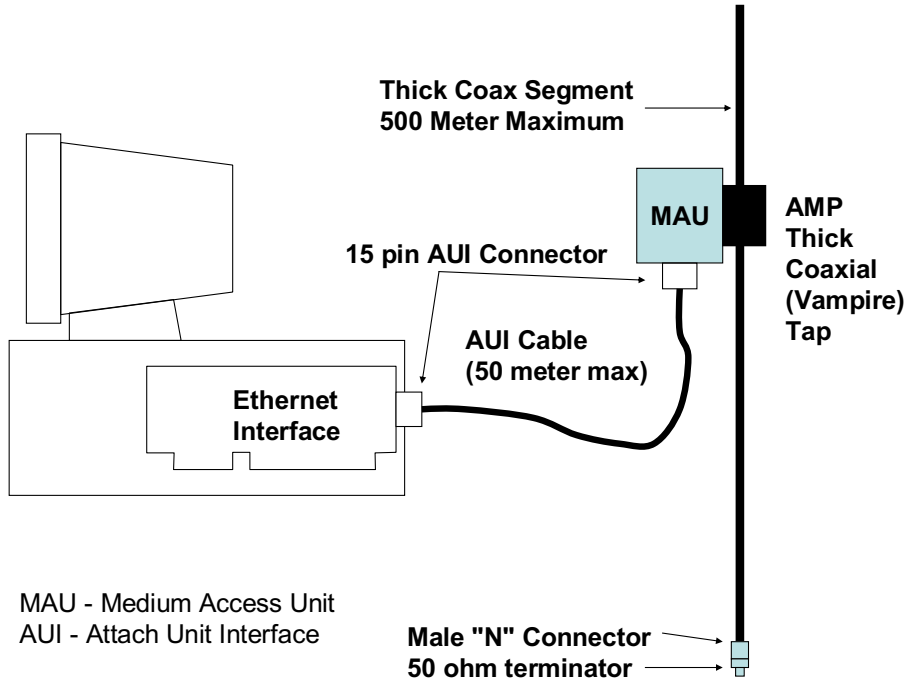
- 1980 Xerox creates Smalltalk - the first object-oriented programming language.
- 1981 Ungermann-Bass ships the first commercial Ethernet network interface card.
- 1981 Xerox introduces the Xerox Star 8010, the first commercial Graphic User Interface computer, for $16,000-$17,000.

PC and the arrival of LAN

- 1981 Microsoft supplies IBM with PC-DOS (which it would also sell as MS-DOS), the OS that would power the IBM PC.
- 1981 IBM brings to market the IBM PC, immediately establishing a new standard for the world of personal computers.

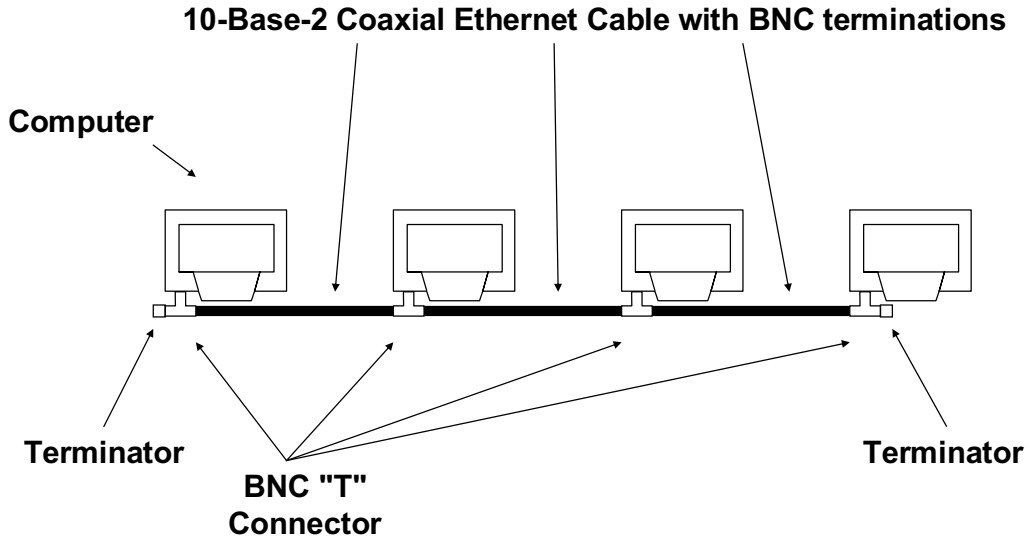
Evolution of LAN

- Thicknet
  - Coaxial cable/Vampire taps
  - 10base5 (10 Mbits/sec, baseband, 500 meters)
  - 1979-1985



Evolution of LAN

- Thinnet
  - Coaxial cable/BNC connectors
  - 10base2 (10 Mbits/sec, baseband, 200 meters)
  - 1985-1993



Evolution of LAN

- Fast Ethernet
  - Move "ethernet" into the box
  - 100baseT (T for twisted pair)
  - RJ45 Connectors