



# CS519: Computer Networks

Lecture 2: Feb 2, 2004

*IP (Internet Protocol)*



# A hypothetical service

---



CS519

- You want a mail delivery service
- You have two choices:
  - Acme Guaranteed Mail Delivery Service
    - “We never fail”
  - Rocko’s Mail Delivery and Hubcap
    - “We’ll get it there if we feel like it”
- Which do you choose???



Some selection criteria???

---



**CS519**



# Some selection criteria

---

CS519

- What is the price?
  - Rocko: \$1, Acme: \$2
- How often do they fail?
  - Rocko: 5%, Acme: Never
- How important is the mail?
  - Important: Worth \$100,000!!!
- Which do you choose???



# A selection analysis

---

CS519

- Rocko's service costs \$5001/mail on average
  - Because 1/20 mails are lost at a cost of \$100000!!!
- Acme's service costs \$2/mail
- Acme appears to be the best choice...



# But....more selection criteria

---

CS519

- What is being delivered (copy or original)?
  - Copy
- How long does it take?
  - Acme and Rocko, both one day max
- How soon do we need delivery?
  - Within a week
- Now which do we choose???



# More selection analysis



CS519

- Assume that the recipient calls to report delivery. If no call, make another copy and send mail again
  - Rocko gets 7 tries at .05 failure per try means  $.05^7 = .00000000078$  failure prob
- Now Rocko costs  $\sim \$1/\text{mail}$ , Acme costs  $\$2/\text{mail}???$ 
  - Ah, but assume phone call costs \$2
- Rocko costs  $\sim \$3/\text{mail}$ , Acme costs  $\$2/\text{mail}$
- Still appears that Acme is the best choice...



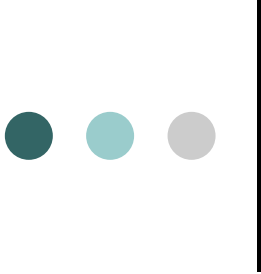
# A couple of network layer service models

---

CS519

- Datagram (IP)
  - “Best Effort” (packets may not arrive, they may be out of order, they may be duplicated)
  - Send packet anytime
- Virtual Circuit (X.25)
  - Guaranteed (no loss, in order, no duplicate)
  - Send packet only if VC established
    - Can try to establish a VC anytime
  - Send packet only if network is ready to receive

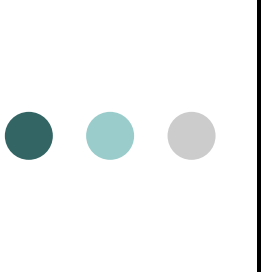




# This was the choice about 20 years ago

**CS519**

- Many people did an analysis and concluded that virtual circuit (VC) services made more sense
  - In fact, the whole idea of an unreliable network service seemed absurd!
- In part: VC services implied simple end devices, complex switches
  - But far more end devices than switches, and switches easier to access (for management and repair)

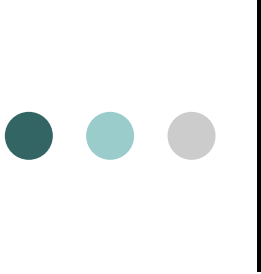


This was the choice about 20  
years ago

---

**CS519**

- But datagram service won in the marketplace
- Why????



# One reason (of many): The end-to-end argument

---

**CS519**

- This may be the single most important concept in network design....



# Mail selection criteria again: Something fishy???

---

CS519

- What is the price?
  - Rocko: \$1, Acme: \$2
- How often do they fail?
  - Rocko: 1%, Acme: Never
- How important is the mail?
  - Important: Worth \$100,000!!!
- Which do you choose???



# Mail selection criteria again: Something fishy???

CS519

- You would definitely make a \$2 phone call to make sure a \$100,000 document was received!!!
  - End-to-end verification
- Turns out that a “reliable” network fails enough that you want end-to-end verification
  - End systems no simpler after all



# End-to-end argument

---

CS519

- Ultimately the end system must be responsible for insuring reliability
  - The network can't fully be trusted
- If the end system has to insure reliability, no reason to do it in the middle!
- Keep the middle simple!
- This is the design principle behind IP



# IP Problem Statement

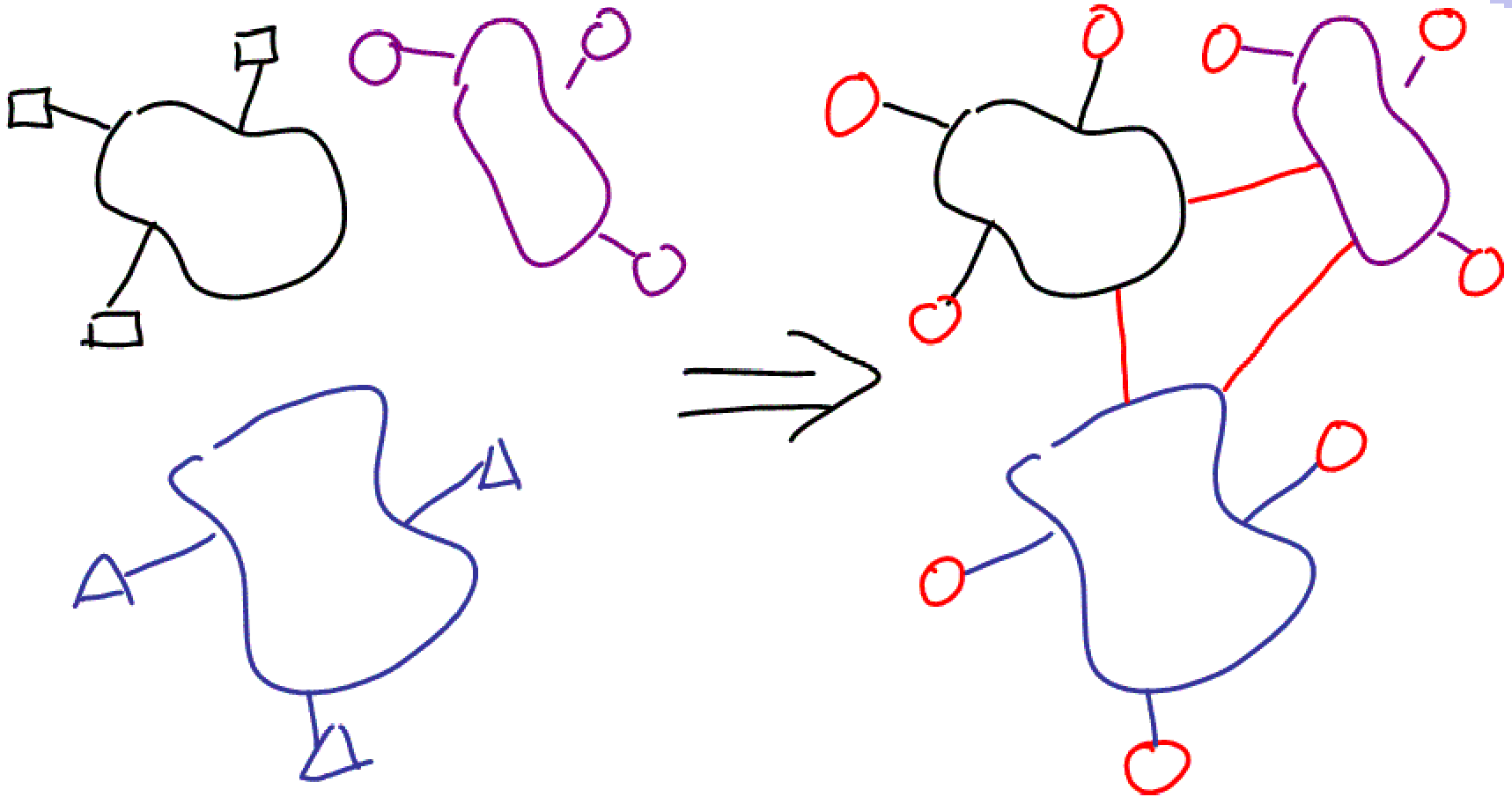
---

**CS519**

- Design a network protocol that can operate over and bridge multiple different kinds of packet networks
- Why this problem statement?
  - Because at the time, DARPA had multiple networks and wanted to make them interwork

# IP Problem Statement

CS519

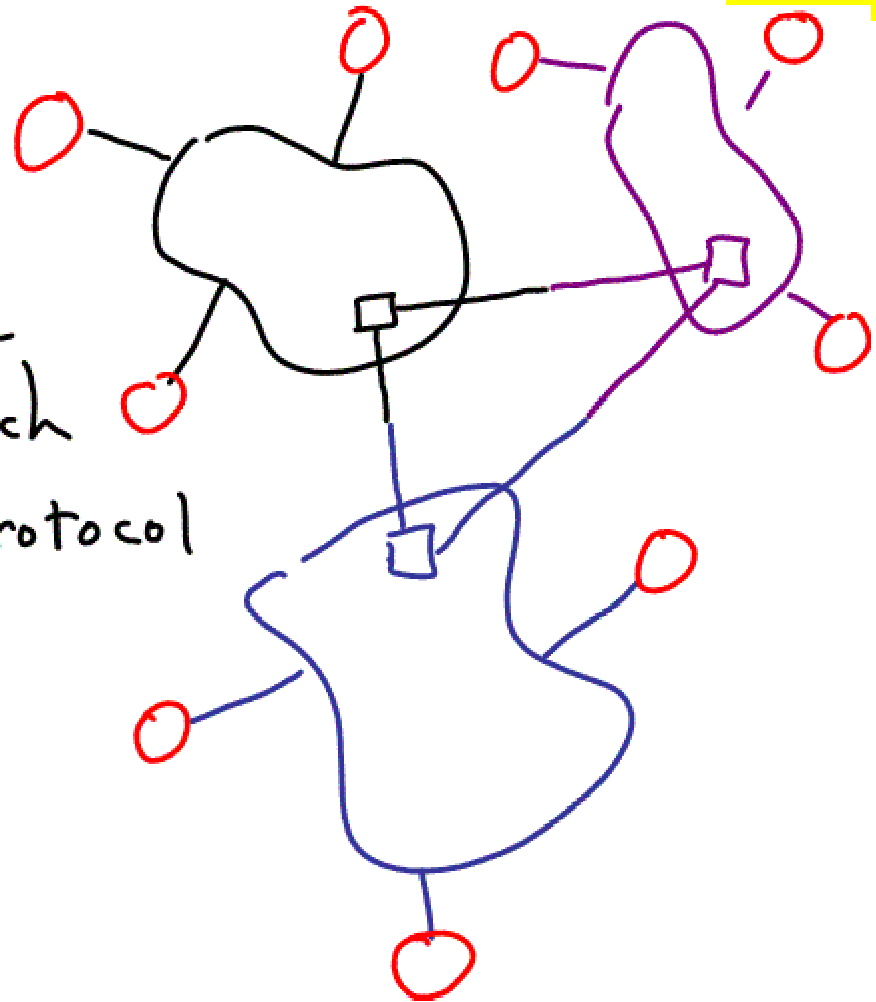




# IP Problem Statement

CS519

But, can't  
directly connect a  
switch running one  
protocol to a switch  
running another protocol





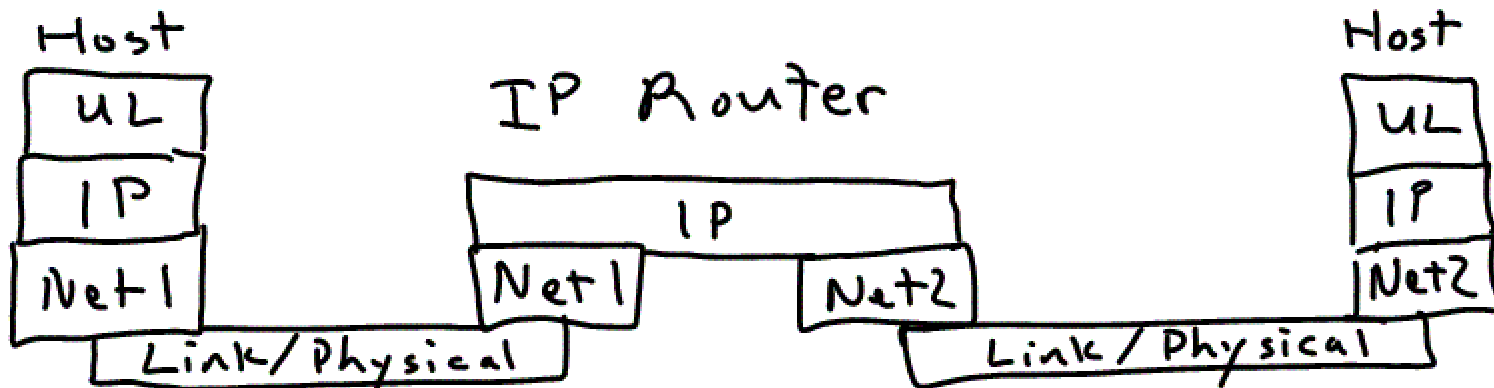
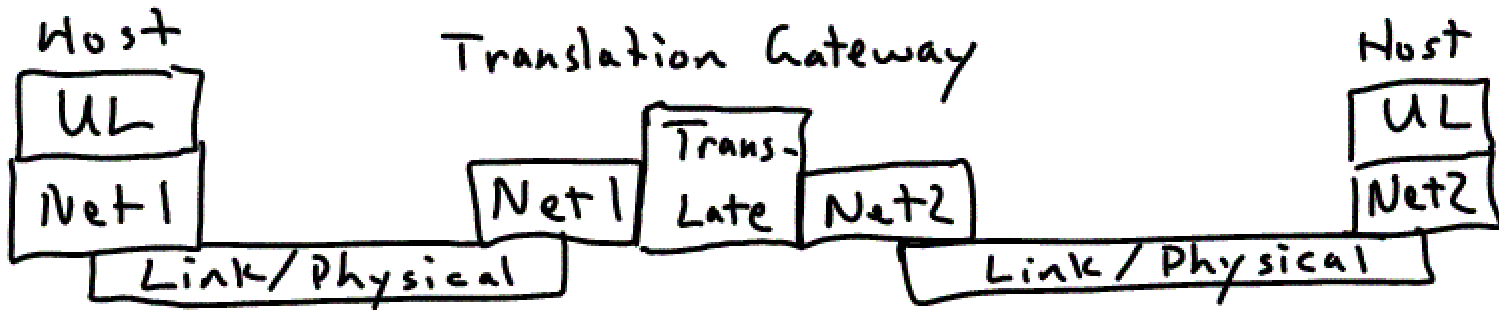
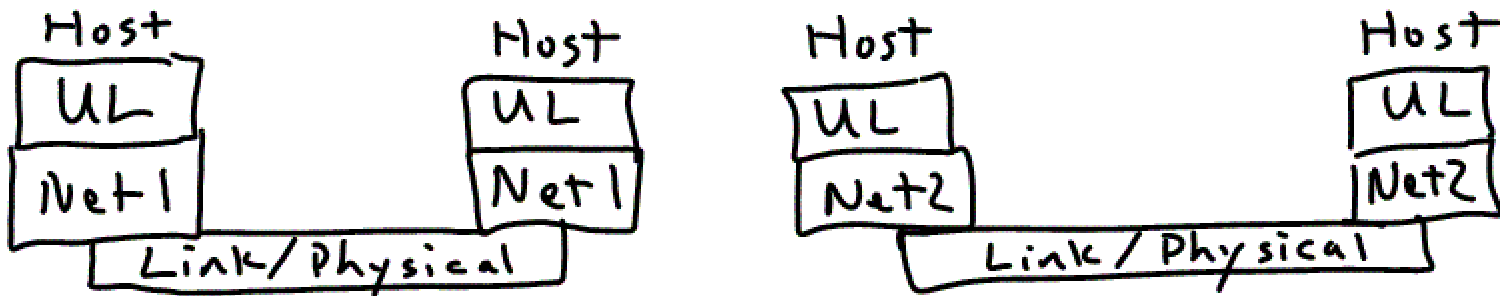
# Two basic approaches were considered

CS519

- Build a translation *gateway* for every pair of network protocols
  - $N^2$  types of gateways
  - Every host has an address on every network
  - Gateways know how to map from an address in one network address space to an address in another network space
- Create a new protocol layer that runs above the existing network protocols
  - This of course is IP

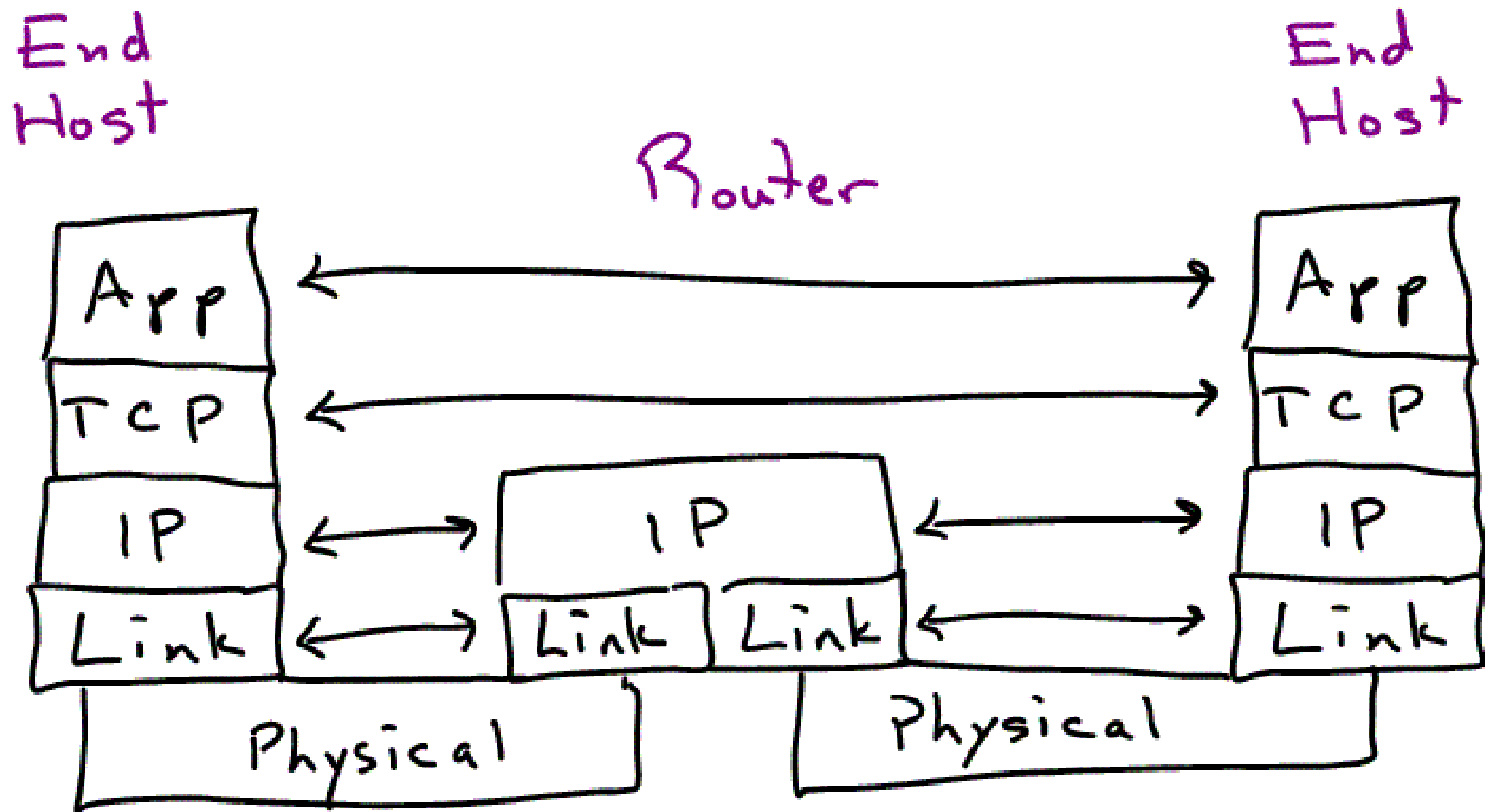
# Translation versus new layer

CS519



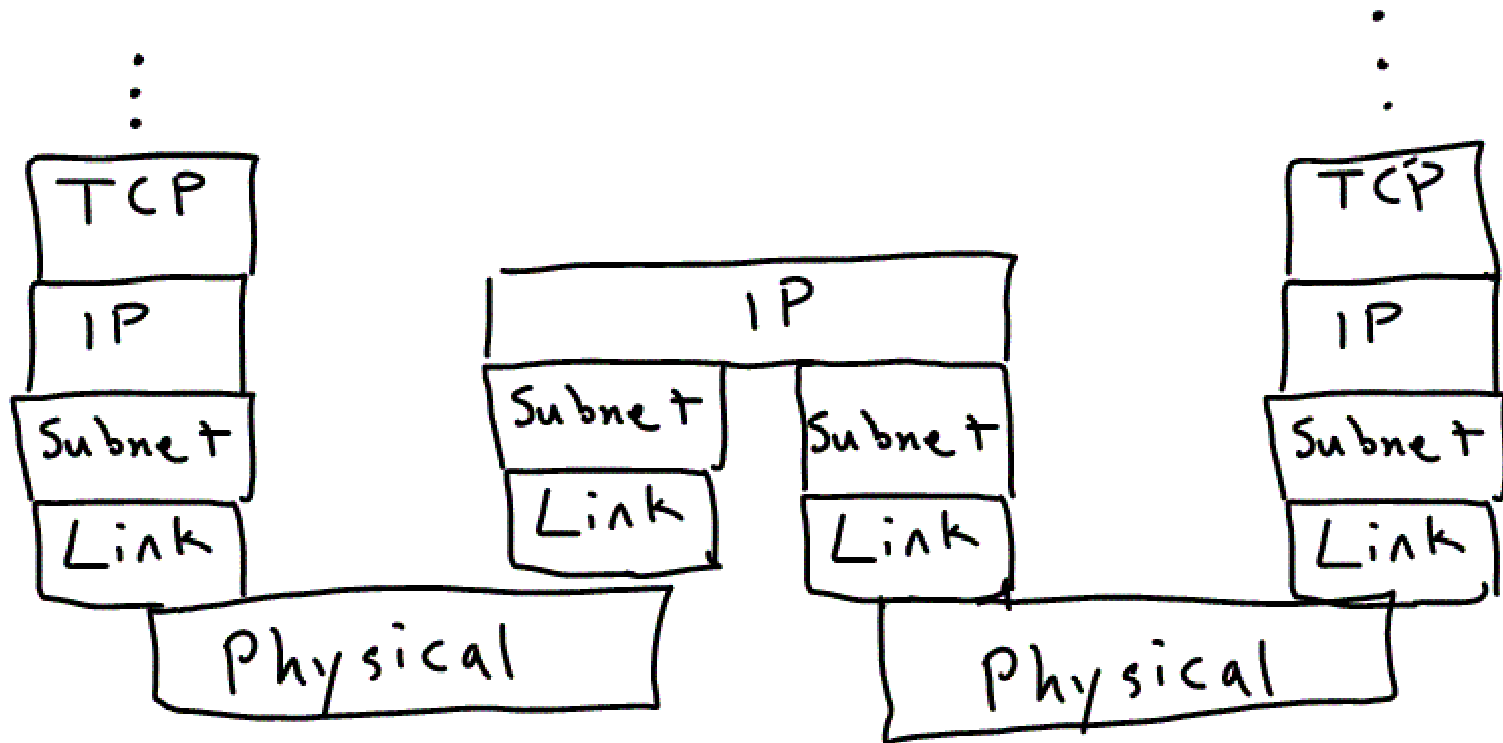
# Contrast with previous protocol stack picture

CS519



# This is a more accurate stack picture

If there is a "network" below IP, we often call it a "subnet".



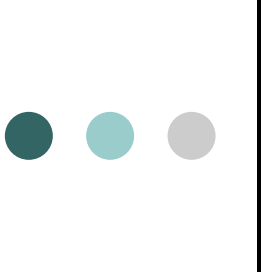


# Advantage of translation gateway approach

---

**CS519**

- No changes to existing hosts required
  - Each host thinks the remote host is on the same network
- This was a nice advantage (and is an approach often used today), but . . .



# Main problems with translation gateway approach

---

CS519

- Service mismatch
  - Networks may offer different services
    - Reliability or not, resource reservation or not, congestion control or not
  - A host on one network “thinks” the remote host is on the same network, and so has the same services
- Lack of address space
  - Most networks built with only enough address space for themselves
  - Can’t accommodate hosts on other networks



# New Protocol Layer

---

CS519

- Can make the address space as big as needed
  - This solves the “lack of address space” problem
- But what service should the new protocol layer provide?





# New Protocol Layer Services



**CS519**

- Reliability (sequencing):
  - This can be provided by the end hosts
  - Don't need it from the networks
- Resource reservation:
  - This is hard to provide unless every network in the middle participates
  - Can't get it from the networks
- Congestion control:
  - This is useful even if not all networks provide it
  - Provide this service (even though different networks may signal this differently and require different responses)!



# New functions required by the new protocol layer

---

CS519

- Address resolution
  - How to determine the subnet address of the next hop (router or host)
  - A hard problem in the general case
- Fragmentation and reassembly
  - How to accommodate different MTUs (Maximum Transmission Unit) in different subnets



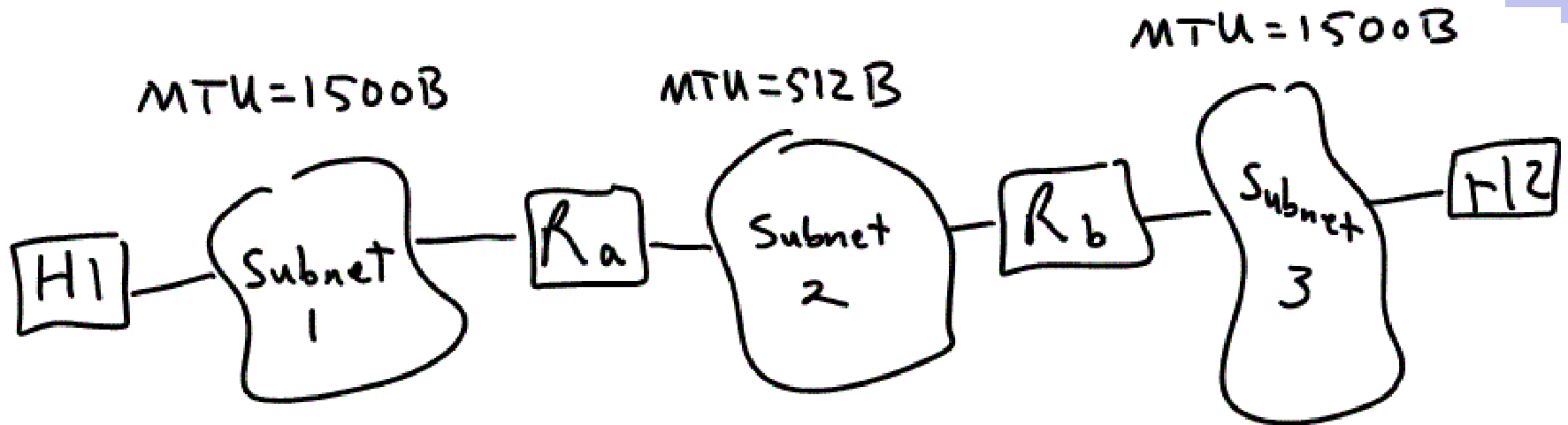
# Maximum Transmission Unit (MTU)

CS519

- Every subnet/link has a maximum packet size it can handle
  - This fixes design of incoming buffers, etc., in hardware
- This is called the MTU
- With multiple subnets, an IP packet may be larger than some MTUs in the path
- The smallest MTU in the path is known as the Minimum MTU

# Minimum MTU Example

CS519





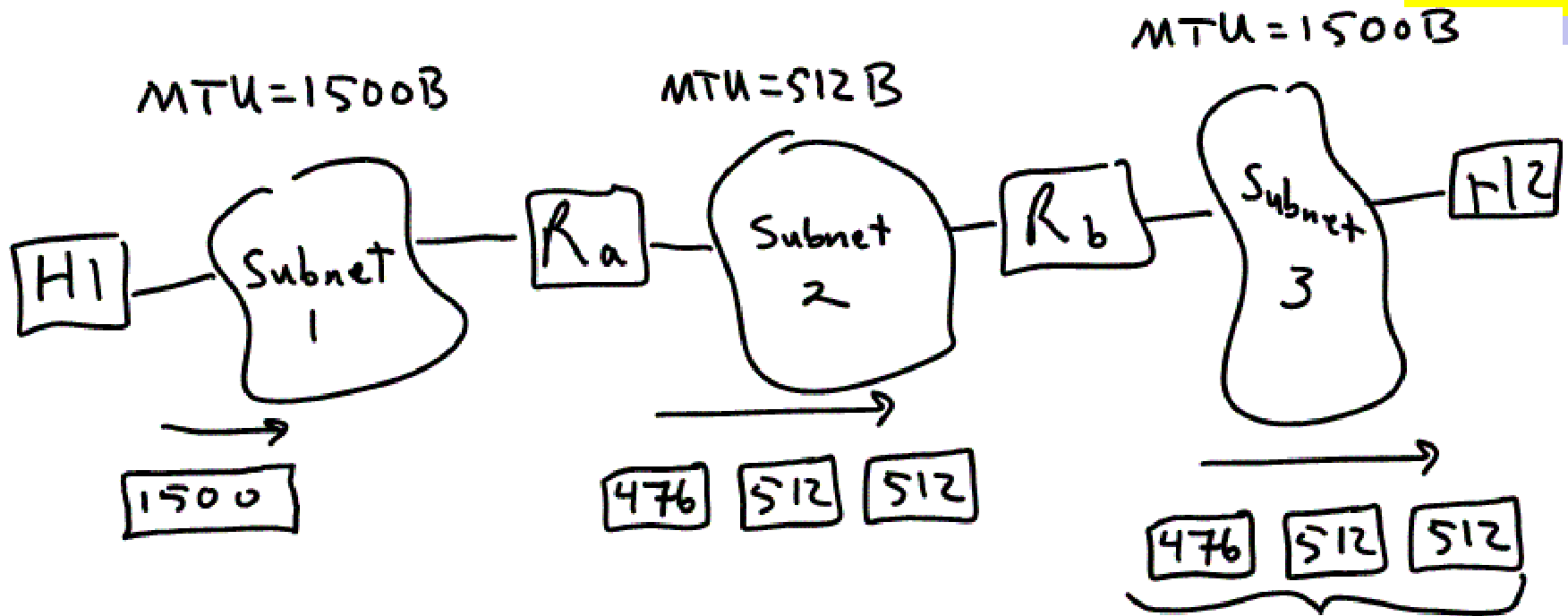
# Three basic approaches

CS519

1. Use a routing protocol that conveys Minimum MTU of the path
  - Host picks the right MTU size from the start
2. Router drops packet and sends an error message to the Host
3. Router fragments packet into smaller packets
  - IP uses a combination of 2 and 3

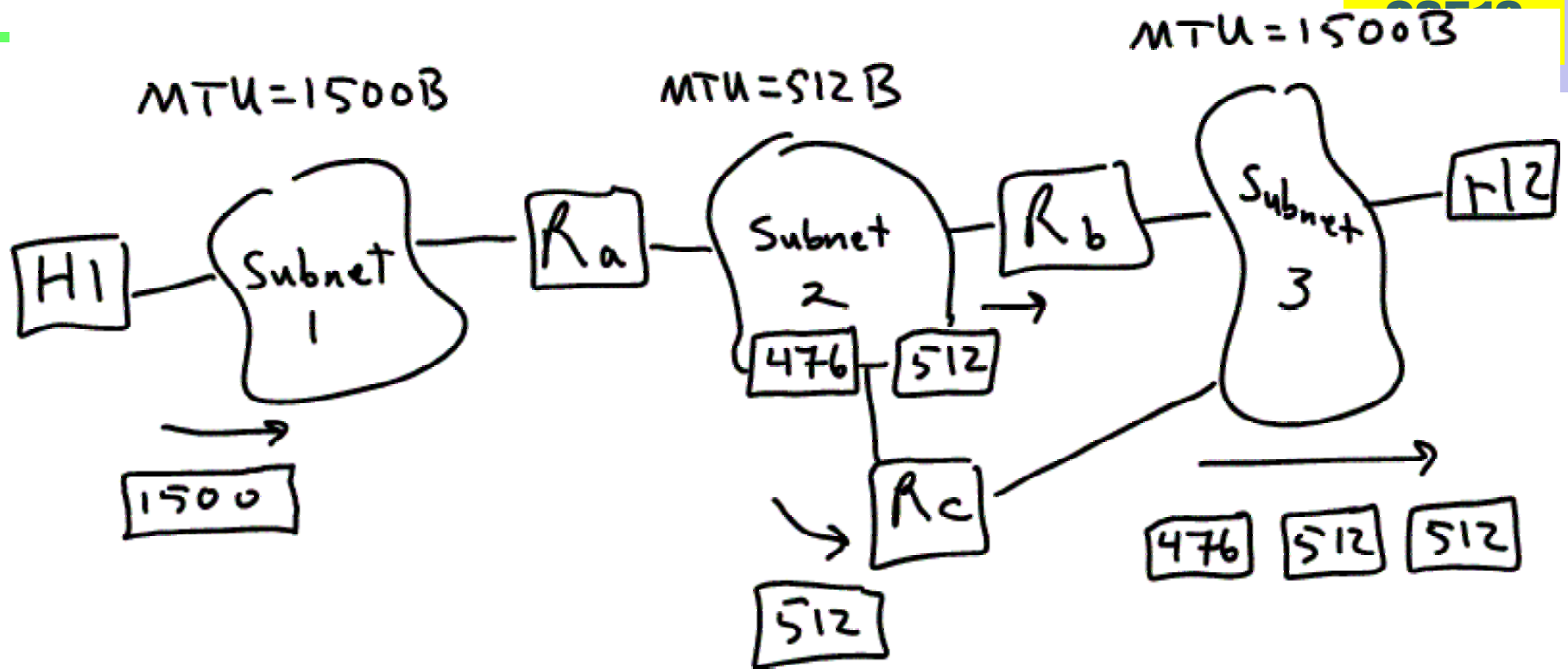
# Fragmentation and reassembly

CS519



H2 IP layer buffers these and reassembles them into a single packet

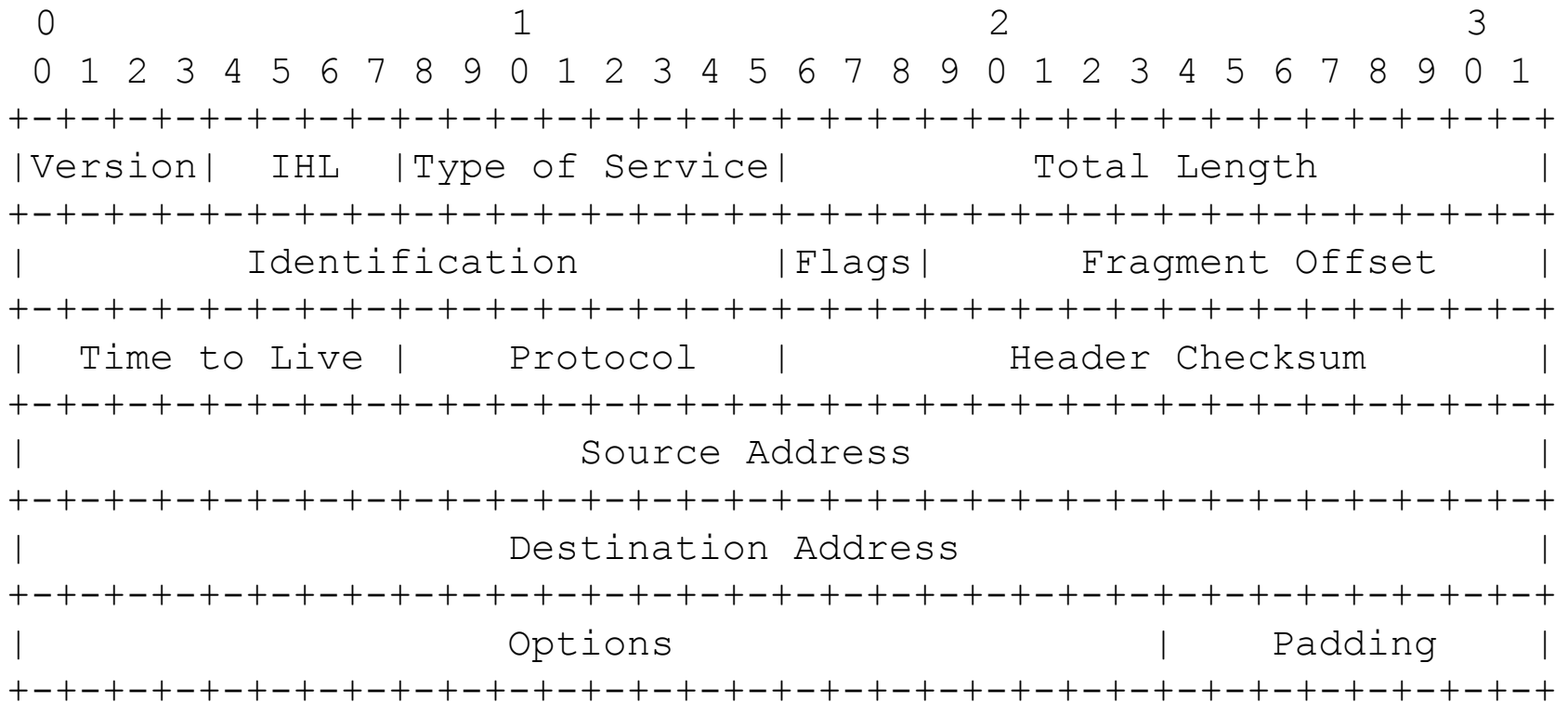
# Why doesn't Rb reassemble the packet?



Different fragments might take different paths, besides it is complex for the routers.

# IP Header (RFC 791)

CS519







# IP Header

CS519

- Version = 4
- IHL = IP Header Length (in units of 32 bit words)
  - Usually 5
- ToS: RFC791 is out of date
- Total Length = packet length (max 65K bytes)
- Identification, Flags, and Fragment Offset
  - Controls fragmentation
- TTL: decrement at each hop, drop packet if 0
  - In case of routing loop



# IP Header

CS519

- Protocol: identifies the next layer
  - The “where” of IP
  - TCP=6, UCP=17
  - About ½ assigned
- Header Checksum
  - A simple checksum of the IP header
  - Router checks and modifies at TTL decrement
- Source and Destination address
  - Ex: 128.42.33.58
- Options
  - Source route
  - Timestamp
  - Security, etc.
  - Not used in practice
- Padding
  - Must pad to integral 32-bit boundary





# Why no fragmentation and checksum in IPv6?

---

CS519

- Actually there is fragmentation (as an “option”), but only the source host can fragment
  - Routers cannot fragment, because this is too costly
  - Indeed fragmentation is rare in IPv4
- Checksum is expensive in routers, and not a disaster if header corrupted
  - It just gets misrouted and eventually dropped



# IP has a related control protocol

---

CS519

- ICMP: Internet Control Message Protocol (RFC 792)
- Three primary purposes:
  1. Give routing directives to hosts
  2. Debug routing problems
  3. Give error feedback to hosts
- ICMP runs over IP



# Primary ICMP messages



**CS519**

- Redirect
  - Tell host to use another router
- Destination Unreachable
  - Tells host that packet can't be delivered for various reasons
  - Dest subnet, host, protocol, port unavailable
  - Don't fragment (DF) Flag set, but needed to fragment
- TTL exceeded
- "Ping" (Echo and Echo Reply)
  - Destination host replies...good for aliveness checking

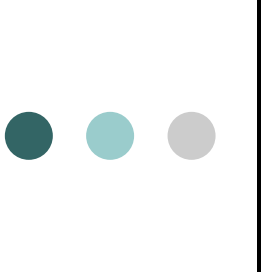


# Not used ICMP messages

---

CS519

- Source quench
  - This was the attempt at getting congestion control from subnets (as well as routers)
- Parameter problem
- Timestamp
- Information Request



# Traceroute is a clever use of ICMP TTL Exceeded message

---

CS519

- Traceroute discovers the path from source to destination
  - But not from destination to source!
- It also discovers where in the path delay is taking place
  - Or where in the path a failure occurs
- (tracert in windows)





# How traceroute works



CS519

- It sends out a series of ping packets with increasing TTL (1, 2, ... )
- When TTL=1, the first router returns an ICMP TTL exceeded message
  - Now we know who the first router is
- When TTL=2, the second router returns an ICMP TTL exceeded message
  - Now we know who the second router is!
- And so on