

Lecture 9:

Networks to Internetworks

CSE 123: Computer Networks

Aaron Schulman





Lecture 9 Overview

- Finish Bridging and Spanning Tree
- Internetworking
 - ◆ Routing
 - ◆ Internet Protocol

Project 1a

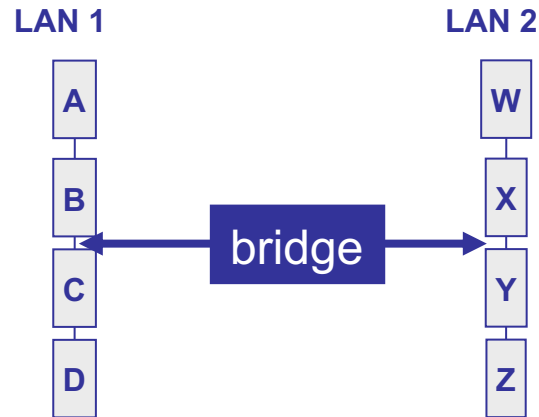


- Autograder is working, several instances of 0's though
 - ◆ If you see "Killed" contact the TA in a PM on Gradescope, this means you ran out of mem.
 - ◆ If you see any Python errors also let us know
- Project 1b builds on Project 1a, so you need to get 1a done.
 - ◆ In Project 2a, you will get half credit for completing Project 1a tests that failed
 - ◆ So if you got a 0% on Project 1a, you can now get up to a 50% Project 1a
- Regrade requests due this Friday
 - ◆ Submit on Gradescope



Selective Forwarding

- Only rebroadcast a frame to the LAN where its destination resides
 - ◆ If *A* sends packet to *X*, then bridge must forward frame
 - ◆ If *A* sends packet to *B*, then bridge shouldn't





Forwarding Tables

- Need to know “destination” of frame
 - ◆ Destination address in frame header (48bit in Ethernet)
- Need know which destinations are on which LANs
 - ◆ One approach: statically configured by hand
 - » Table, mapping address to output port (i.e. LAN)
 - ◆ But we’d prefer something automatic and dynamic...
- Simple algorithm:
Receive frame *f* on port *q*
Lookup *f.dest* for output port /* know where to send it? */
If *f.dest* found
then if output port is *q* then drop /* *already delivered* */
 else forward *f* on output port;
else flood *f*;
/* forward on all ports but the one where frame arrived*/

Host	Port
A	1
B	1
C	1
D	1
W	2
X	2
Y	2
Z	2

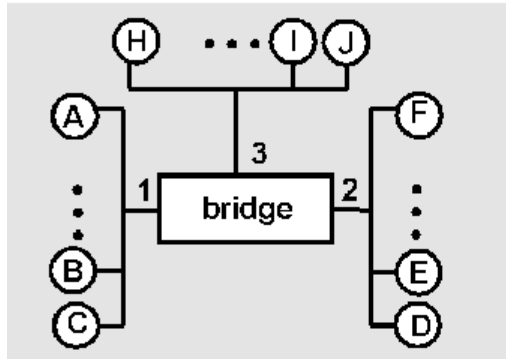


Learning Bridges

- Eliminate manual configuration by learning which addresses are on which LANs
- Basic approach
 - ◆ If a frame arrives on a port, then associate its **source** address with that port
 - ◆ As each host transmits, the table becomes accurate
- What if a node moves? Table aging
 - ◆ Associate a timestamp with each table entry
 - ◆ Refresh timestamp for each new packet with same source
 - ◆ If entry gets too stale, remove it

Host	Port
A	1
B	1
C	1
D	1
W	2
X	2
Y	2
Z	2

Learning Example

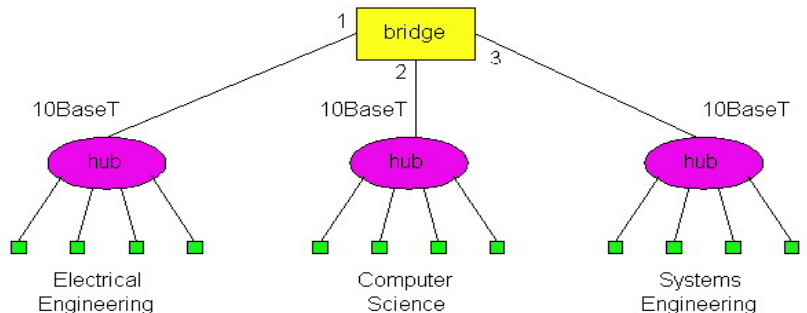
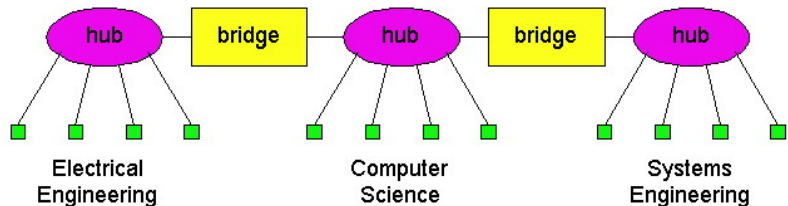


- If *D* generates a reply to *C*, it sends it on its LAN
 - ◆ bridge sees frame from *D*
 - ◆ bridge notes that *D* is on port 2
 - ◆ bridge knows *C* on port 1 (from previous slide), so **selectively** forwards frame via port 1



Network Topology

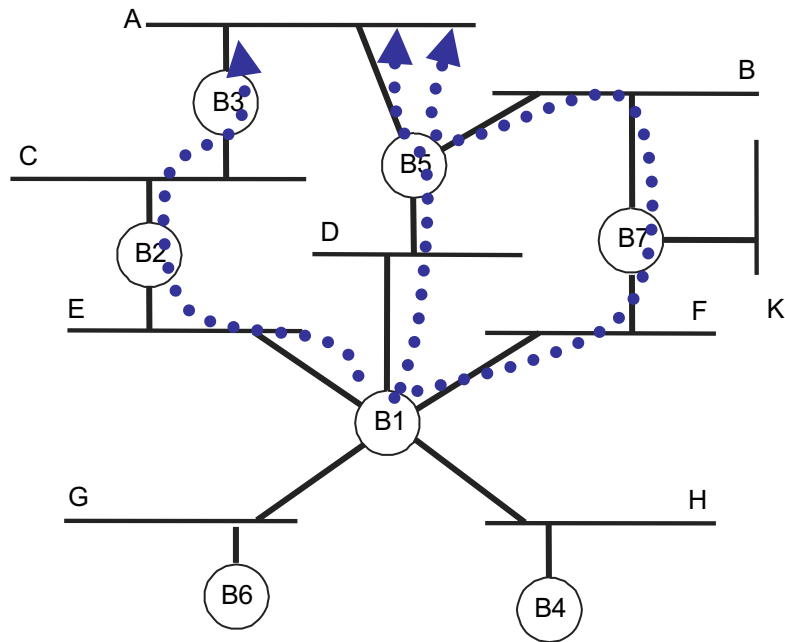
- Linear organization
 - ◆ Inter-bridge hubs (e.g. CSE) are single points of failure
 - ◆ Unnecessary transit (e.g. ECE $\leftrightarrow$ SE must traverse CSE)
- Backbone/tree
 - ◆ Can survive LAN failure
 - ◆ Manages all inter-LAN communication
 - ◆ Requires more ports





An Issue: Cycles

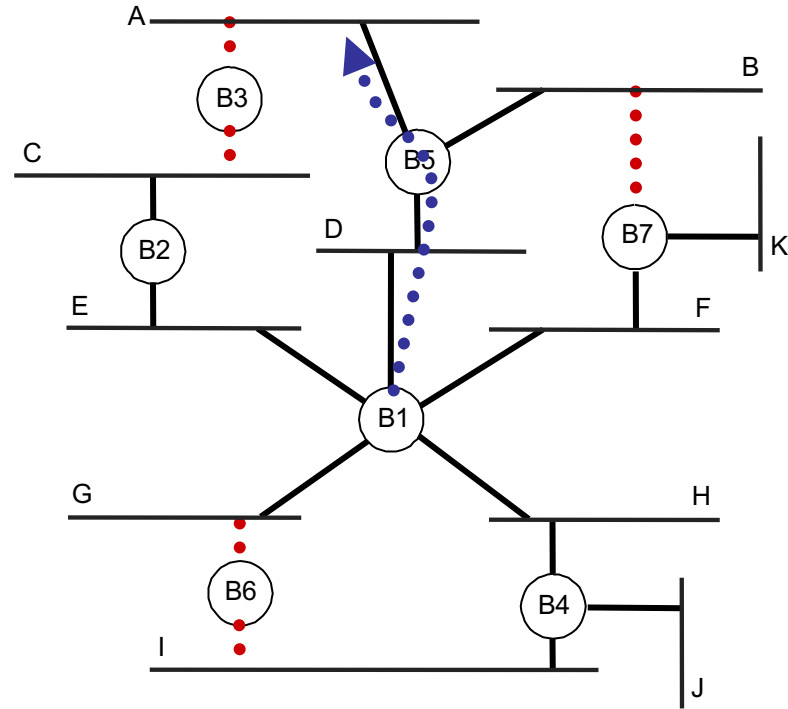
- Learning works well in tree topologies
- But trees are fragile
 - ◆ Net admins like redundant/backup paths
- How to handle Cycles?
 - ◆ Where should *B1* forward packets destined for LAN *A*?





Spanning Tree

- Spanning tree uses *subset* of bridges so there are no cycles
 - ◆ Prune some ports
 - ◆ Only **one** tree
- Q: How do we find a spanning tree?
 - ◆ Automatically!
 - ◆ Elect root, find paths





Spanning Tree Algorithm

- Each bridge sends periodic configuration messages
 - ◆ (RootID, Distance to Root, BridgeID)
 - ◆ All nodes think they are root initially
- Each bridge updates route/Root upon receipt
 - ◆ Smaller root address is better
 - ◆ Select port with lowest cost to root as “root port”
 - To break ties, bridge with smaller address is better
- Rebroadcast new config to ports for which we’re “best”
 - ◆ Don’t bother sending config to LANs with better options
 - ◆ Add 1 to distance, send on ports that haven’t told us about a shorter path to the root
- Only forward packets on ports for which we’re on the shortest path to root (prunes edges to form tree)

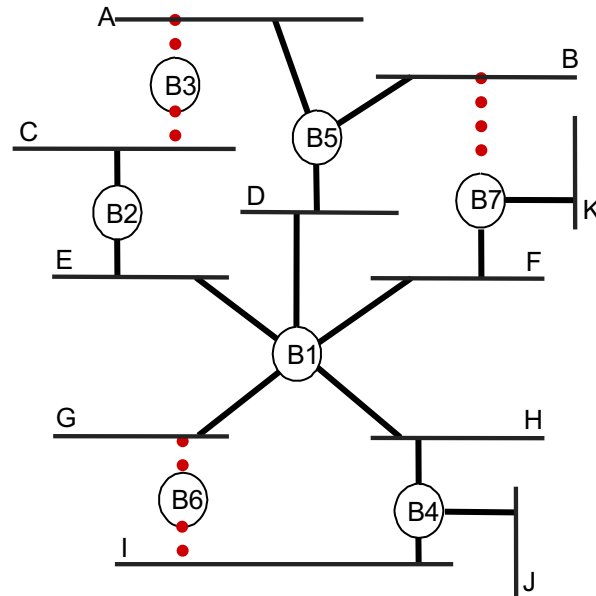
Spanning Tree Example



- Sample messages to and from B3:

1. B3 sends (B3, 0, B3) to LAN C (B2) and LAN A (B5)
2. B3 receives (B2, 0, B2) and (B5, 0, B5) and accepts B2 as root
3. B3 sends (B2, 1, B3) to LAN A (B5)
4. B3 receives (B1, 1, B2) and (B1, 1, B5) and accepts B1 as root
5. B3 wants to send (B1, 2, B3) but doesn't as its nowhere "best"
6. B3 receives (B1, 1, B2) and (B1, 1, B5) again and again...

Data forwarding at B3 is turned off for both LAN A and LAN C





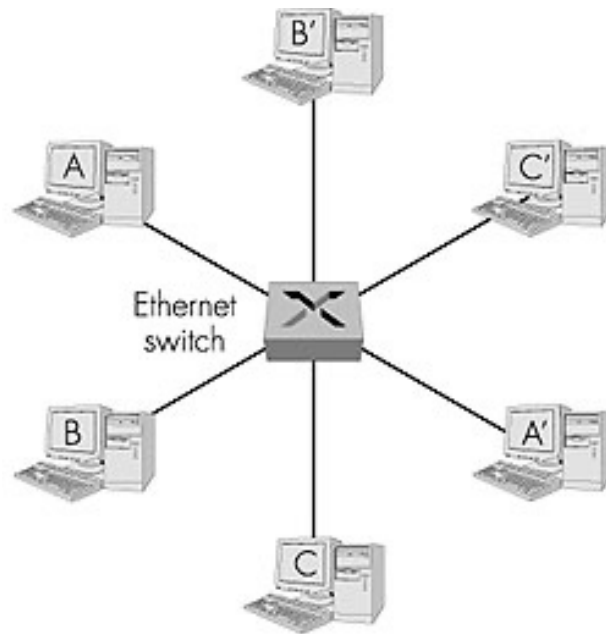
Important Details

- What if root bridge fails?
 - ◆ Age configuration info
 - » If not refreshed for MaxAge seconds then delete root and recalculate spanning tree
 - » If config message is received with a more recent age, then recalculate spanning tree
 - ◆ Applies to all bridges (not just root)
- Temporary loops
 - ◆ When topology changes, takes a bit for new configuration messages to spread through the system
 - ◆ Don't start forwarding packets immediately -> wait some time for convergence

Switched Ethernet



- Hosts directly connected to a bridge
 - ◆ learning + spanning tree protocol
- Switch supports parallel forwarding
 - ◆ A-to-B and A'-to-B' simultaneously
 - ◆ Generally **full duplex** as well
- **Switch** backplane capacity varies
 - ◆ Ideally, **non-blocking**
 - ◆ I.e., can run at full line rate on all ports
- No longer any shared bus
 - ◆ Each link is its own collision domain





Layer-2 Forwarding

- Create spanning tree across LANs
 - ◆ Learn which ports to use to reach which addresses
- Benefits
 - ◆ Higher link bandwidth (point-to-point links)
 - ◆ Higher aggregate throughput (parallel communication)
 - ◆ Improved fault tolerance (redundant paths)
- Limitations
 - ◆ Requires homogeneous link layer (e.g. all Ethernet)
 - ◆ Harder to control forwarding topology
- What if we want to connect different link layers?



Combing Networks

- Main challenge is heterogeneity of link layers:
 - ◆ Addressing
 - » Each network media has a different addressing scheme
 - ◆ Bandwidth
 - » Modems to terabits
 - ◆ Latency
 - » Seconds to nanoseconds
 - ◆ Frame size
 - » Dozens to thousands of bytes
 - ◆ Loss rates
 - » Differ by many orders of magnitude
 - ◆ Service guarantees
 - » “Send and pray” vs reserved bandwidth



IP Philosophy

- Impose few demands on network
 - ◆ Make few assumptions about what network can do
 - ◆ No QoS, no reliability, no ordering, no large packets
 - ◆ No persistent state about communications; no connections
- Manage heterogeneity at hosts (not in network)
 - ◆ Adapt to underlying network heterogeneity
 - ◆ Re-order packets, detect errors, retransmit lost messages...
 - ◆ Persistent network state only kept in hosts (fate-sharing)
- Service model: **best effort**, a.k.a. *send and pray*

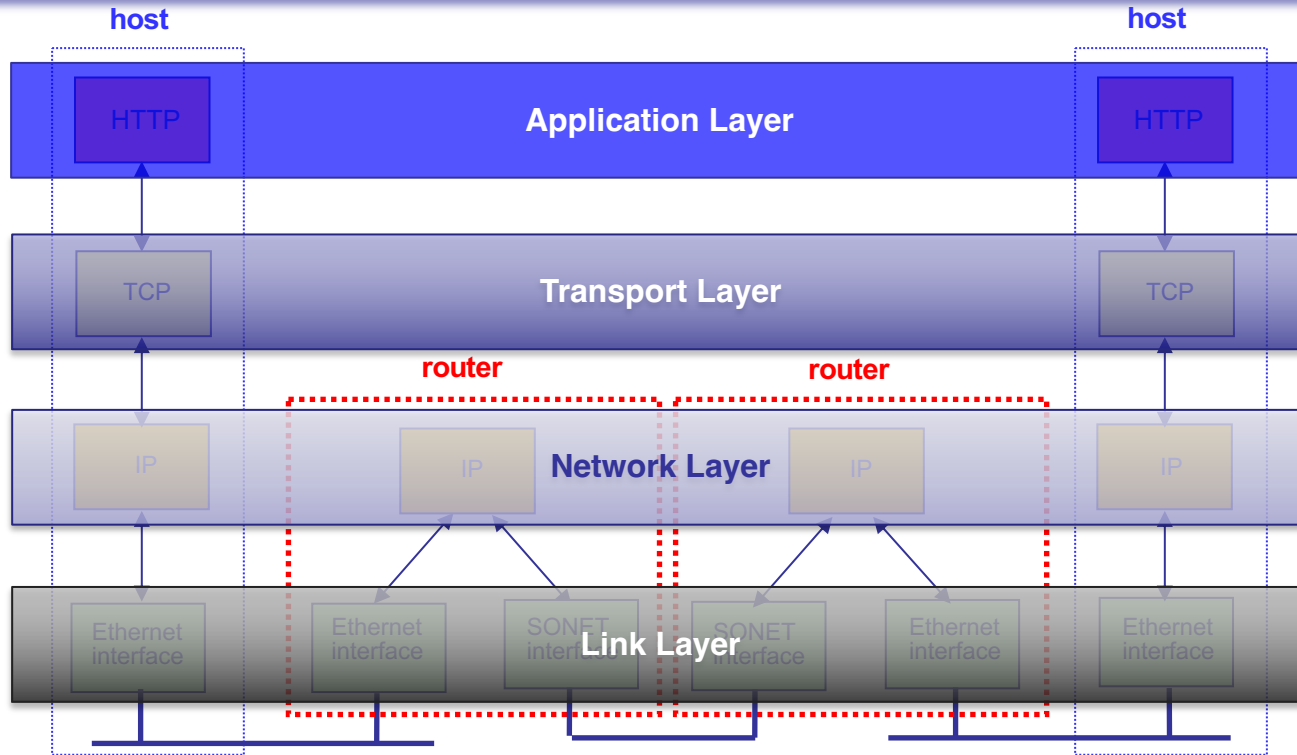
internetworking



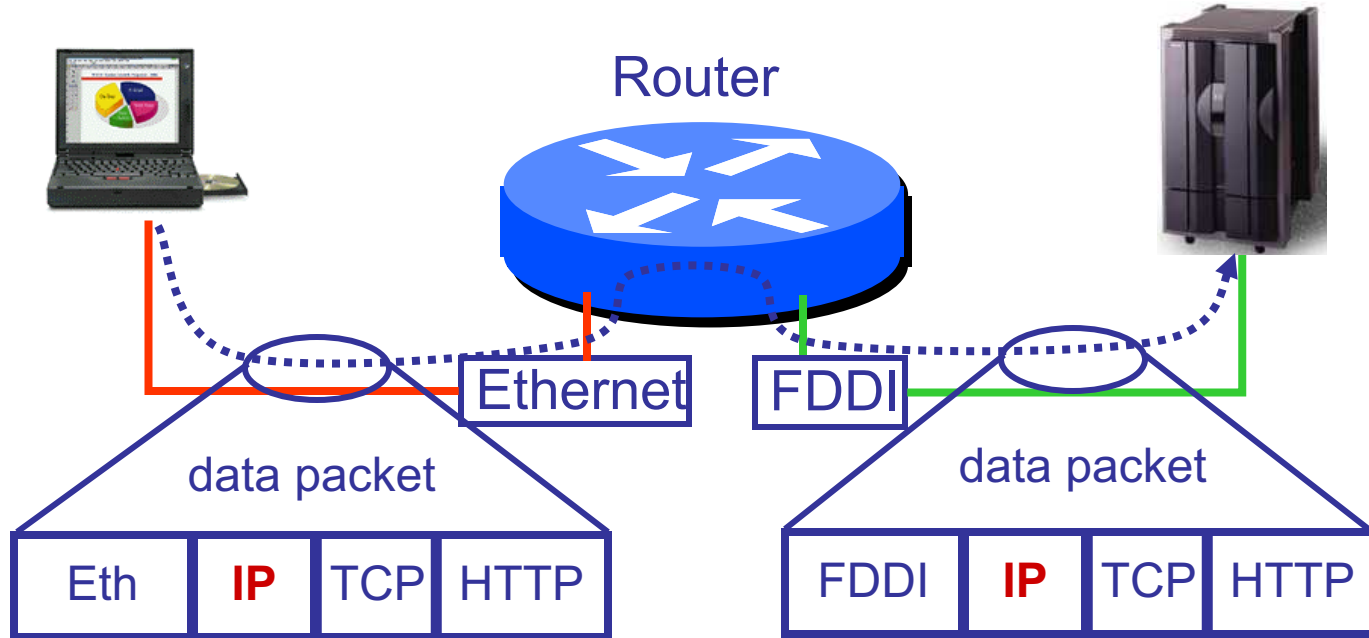
- Cerf & Kahn74, “*A Protocol for Packet Network Intercommunication*”
 - ◆ Foundation for the modern Internet
- Routers forward packets from source to destination
 - ◆ May cross many separate networks along the way
- All packets use a common Internet Protocol
 - ◆ Any underlying data link protocol
 - ◆ Any higher layer transport protocol



Defining the Internet



IP Networking



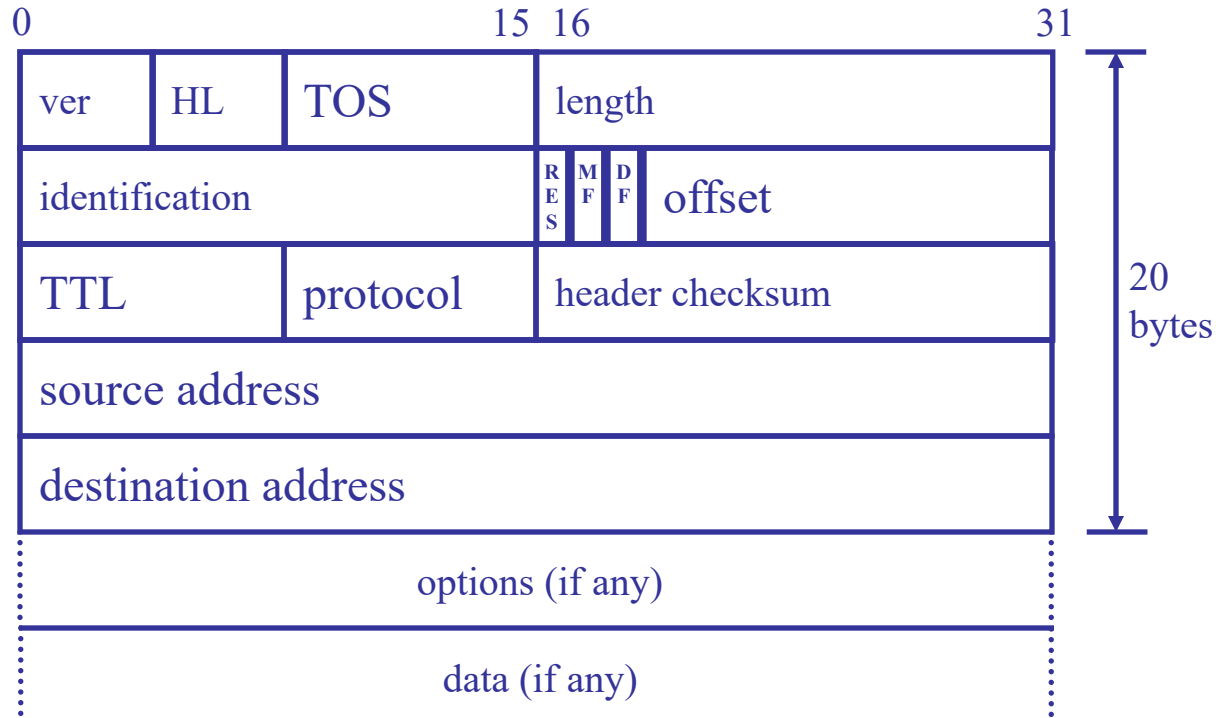
Routers



- A router is a store-and-forward device
 - ◆ Routers are connected to multiple networks
 - ◆ On each network, looks just like another host
 - ◆ A lot like a switch, but supports multiple datalink layers and makes decisions at the network layer
- Must be explicitly addressed by incoming frames (L2)
 - ◆ Not at all like a bridge or switch, which are transparent
 - ◆ Removes link-layer header, parses IP header (L3)
- Looks up next hop, forwards on appropriate network
 - ◆ Each router need only get one step closer to destination



IPv4 Packet Header





For Next Time

- Read 3.3
- HW2 out now; due next Wednesday