CSCI-1680
Network Layer:
Wrap-up

Rodrigo Fonseca

Based partly on lecture notes by David Mazières, Phil Levis, John Jannotti

Today

- Snowcast feedback
- Multicast
- IPv6

Snowcast Feedback

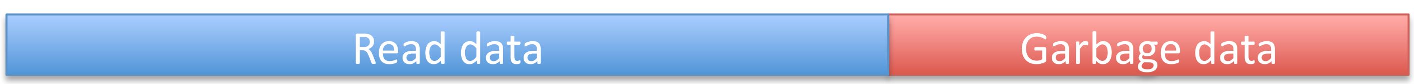
- **Results should be in early next week**
- **Current average: 74**
- **Common mistakes**
  - Byte order: ntoh* and hton*
  - Not checking bytes read/written
  - Using printf for non-textual data
  - Memory and socket leaks
  - Rate calculation: carefully read the spec!

Byte order issues

- In-memory short/long -> hton*() -> network
- Network -> ntoh*() -> short/long

Read/write

- **printf is for strings: *0-terminated* sequence of bytes**
  - Use read/write for binary data
- **Check the return of functions!**
- `ssize_t read (int fd, void *buf, int nbytes);`
  - Returns number of bytes read
  - Returns 0 bytes at end of file, or -1 on error
- `ssize_t write (int fd, void* buf, int nbytes);`
  - Returns number of bytes written, -1 on error
- **Common example:**
  - Read from file in chunks of 1400B.
  - Last chunk < 1400.



  - Send 1400 bytes

Common Ways to Sanity Check/Debug

- **Wireshark: will let you see what goes on the wire**
- **netcat: easy way to send / receive data to servers**
- **Valgrind: will find memory leaks in your program**
  - Forget to free allocated memory
  - Double-free a region
  - Access to unitialized memory
- **gdb: allow you inspect all aspects of your program while running/after a crash**
  - break/watchpoints
  - variable contents, lets you follow pointers, etc…
  - useful after segmentation fault, will tell you where/why

Different IP Service Models

- **Broadcast: send a packet to *all* nodes in some subnet. "One to all"**
  - 255.255.255.255 : all hosts within a subnet, *never* forwarded by a router
  - "All ones host part": broadcast address
    - Host address | (255.255.255.255 & ~subnet mask)
    - E.g.: 128.148.32.143 mask 255.255.255.128
    - ~mask = 0.0.0.127 => Bcast = 128.148.32.255
- **Example use: DHCP**
- **Not present in IPv6**
  - Use multicast to link local all nodes group

Anycast

- **Multiple hosts may share the same IP address**
- **"One to one of many" routing**
- **Example uses: load balancing, nearby servers**
  - DNS Root Servers (e.g. f.root-servers.net)
  - Google Public DNS (8.8.8.8)
  - IPv6 6-to-4 Gateway (192.88.99.1)

Anycast Implementation

- **Anycast addresses are /32s**
- **At the BGP level**
  - Multiple ASs can advertise the same prefixes
  - Normal BGP rules choose one route
- **At the Router level**
  - Router can have multiple entries for the same prefix
  - Can choose among many
- **Each packet can go to a different server**
  - Best for services that are fine with that (connectionless, stateless)

Multicast

- **Send messages to many nodes: "one to many"**
- **Why do that?**
  - Snowcast, Internet Radio, IPTV
  - Stock quote information
  - Multi-way chat / video conferencing
  - Multi-player games
- **What's wrong with sending data to each recipient?**
  - Link stress
  - Have to know address of all destinations

Multicast Service Model

- **Receivers join a multicast group G**
- **Senders send packets to address G**
- **Network routes and delivers packets to all members of G**
- **Multicast addresses: class D (start 1110)**

  224.x.x.x to 229.x.x.x

  – 28 bits left for group address

LAN Multicast

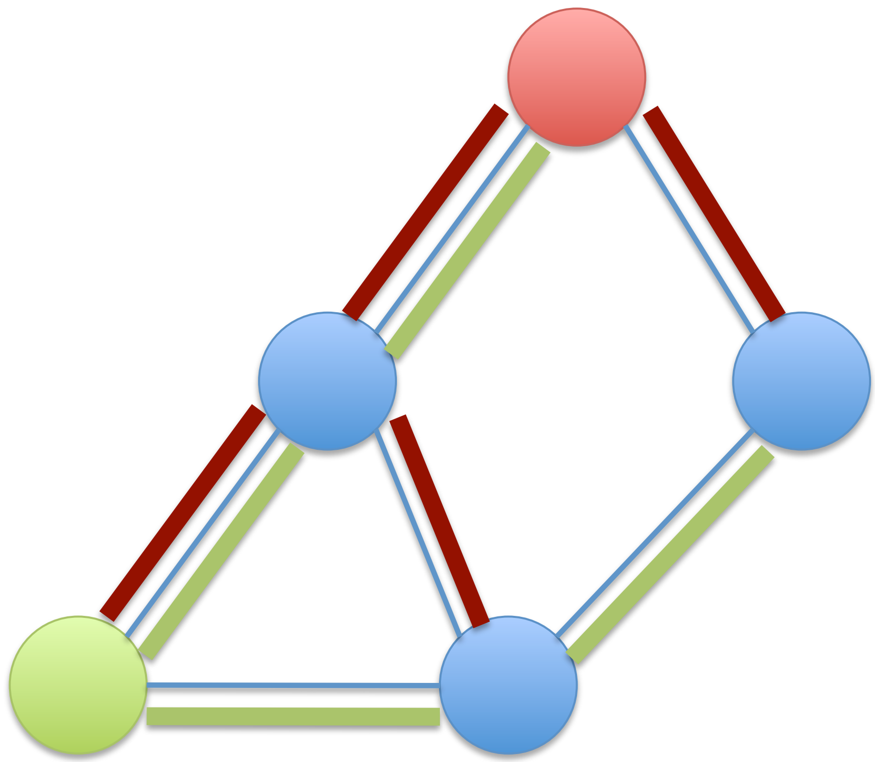
- **Easy on a shared medium**
- **Ethernet multicast address range:**
  - 01:00:57:00:00:00 to 01:00:57:7f:ff:ff
- **Set low 23 bits of Ethernet address to low bits of IP address**
  - (Small problem: 28-bit group address -> 23 bits)

How about on the Internet?

Use Distribution Trees

- **Source-specific trees:**
  - Spanning tree over recipients, rooted at each source
  - Best for each source
- **Shared trees:**
  - Single spanning tree among all sources and recipients
  - Hard to find one shared tree that's best for many senders
- **State in routers much larger for source-specific**

Source vs Shared Trees



Building the Tree: Host to Router

- **Nodes tell their local routers about groups they want to join**
  - IGMP, Internet Group Management Protocol (IPv4)
  - MLD, Multicast Listener Discovery (IPv6)
- **Router periodically polls LAN to determine memberships**
  - Hosts are not required to leave, can stop responding

Building the Tree across networks

- **Routers maintain multicast routing tables**
  - Multicast address -> set of interfaces, or
  - <Source, Multicast address> -> set of interfaces
- **Critical: only include interfaces where there are downstream recipients**

Using Link State

- Augment update message (LSP) to include set of groups that have members on a particular network
- Each router uses Djiktra's algorithm to compute shortest path spanning tree for each source/group pair
- Very expensive!

Distance Vector (DVMRP)

- **Reverse path broadcast**
  - Each router already knows shortest path to S is through neighbor N
  - When receive multicast packet *from* S, forward on all outgoing links (except the one it came from), iff packet came from N
- **Eliminate duplicate broadcast packets by letting only one router per LAN ("parent") forward**
  - Router on shortest path from S
  - Break ties with smallest address
- **Problem: so far, this is *broadcast !***

Distance Vector (cont)

- **Goal: prune networks that have no hosts in group G**
- **If LAN is a leaf (e.g., no other routers), easy:**
  - Use IGMP
- **Otherwise, propagate "no members of G here"**
  - Only happens when multicast address becomes active
- **"*Flood-and-Prune*"**

Scaling issues

- **What if you have very few recipients spread on many networks?**
  - Flood and prune highly inefficient
- **PIM-SM (Protocol-independent multicast, Sparse Mode)**
  - Name a Rendezvous Point (RP) router for a domain
  - Send a JOIN(*,G) message to RP
  - Routers note the JOIN in their routing table
  - Sender S sends unicast packet to RP, which multicasts it to the tree
  - Optimization 1: RP sends JOIN(S,G) to S
  - Optimization 2: Recipients send JOIN(S,G) to S

Inter-Domain

- **MSDP connects RPs from different domains together over TCP**
  - Mesh of MSDP uses reverse path broadcast
- **Other examples (e.g. BGMP)**

Practical Considerations

- **Multicast protocols end up being quite complex**
- **Introduce a lot of router state**
- **Turned off on most routers**
- **Mostly used within domains**
  - In the department: Ganglia monitoring infrastructure
  - IPTV on campus
- **Alternative: do multicast in higher layers**

IPv6

- **Main motivation: IPv4 address exhaustion**
- **Initial idea: larger address space**
- **Need new packet format:**
  - REALLY expensive to upgrade all infrastructure!
  - While at it, why don't we fix a bunch of things in IPv4?
- **Work started in 1994, basic protocol published in 1998**

IPv6 Key Features

- **128-bit addresses**
  - Autoconfiguration
- **Simplifies basic packet format through *extension headers***
  - 40-byte base header (fixed)
  - Make less common fields optional
- **Security and Authentication**

IPv6 Address Representation

- **Groups of 16 bits in hex notation**

  47cd:1244:3422:0000:0000:fef4:43ea:0001

- **Two rules:**
  - Leading 0's in each 16-bit group can be omitted

    47cd:1244:3422:0:0:fef4:43ea:1

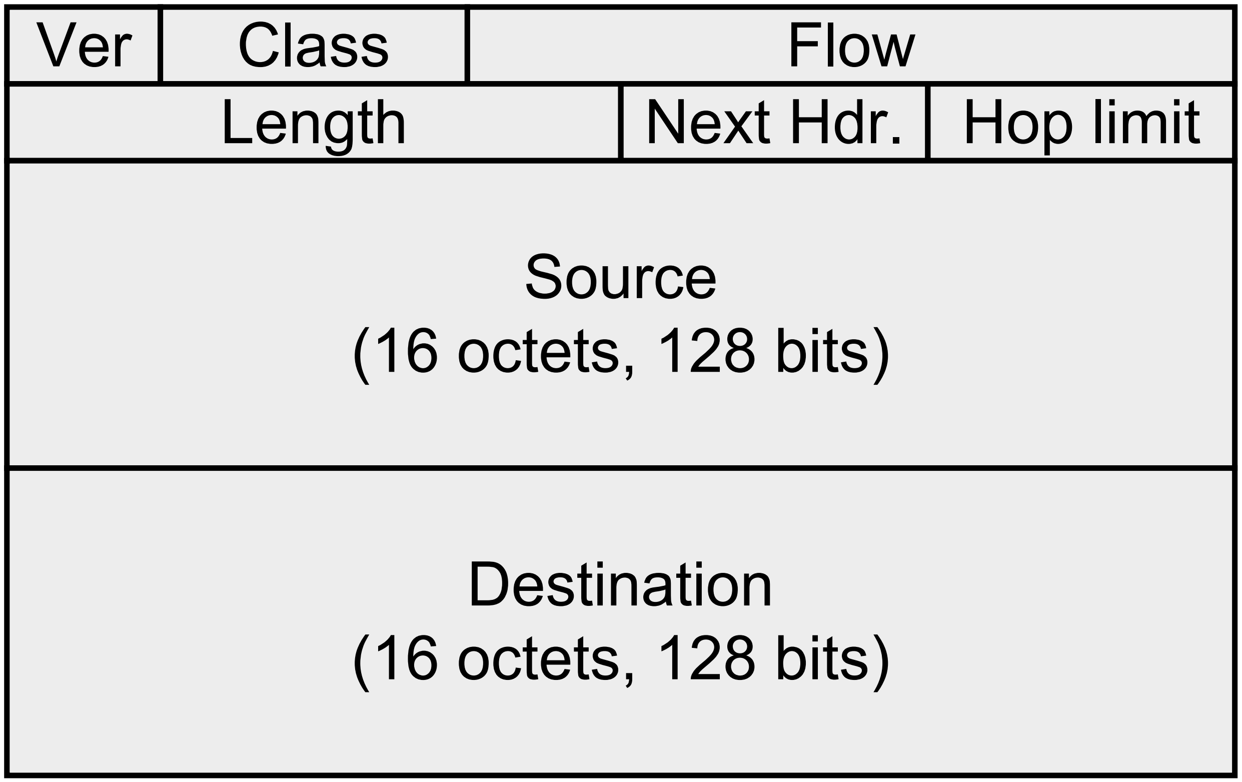
  - One contiguous group of 0's can be compacted

    47cd:1244:3422::fef4:43ea:1

IPv6 Addresses

- **Break 128 bits into 64-bit network and 64-bit interface**
  - Makes autoconfiguration easy: interface part can be derived from Ethernet address, for example
- **Types of addresses**
  - All 0's: unspecified
  - 000…1: loopback
  - ff/8: multicast
  - fe8/10: link local unicast
  - fec/10: site local unicast
  - All else: global unicast

IPv6 Header


Ver
Class
Flow
Length
Next Hdr.
Hop limit
Source
(16 octets, 128 bits)
Destination
(16 octets, 128 bits)

IPv6 Header Fields

- **Version: 4 bits, 6**
- **Class: 8 bits, like TOSS in IPv4**
- **Flow: 20 bits, identifies a *flow***
- **Length: 16 bits, datagram length**
- **Next Header, 8 bits: …**
- **Hop Limit: 8 bits, like TTL in IPv4**
- **Addresses: 128 bits**

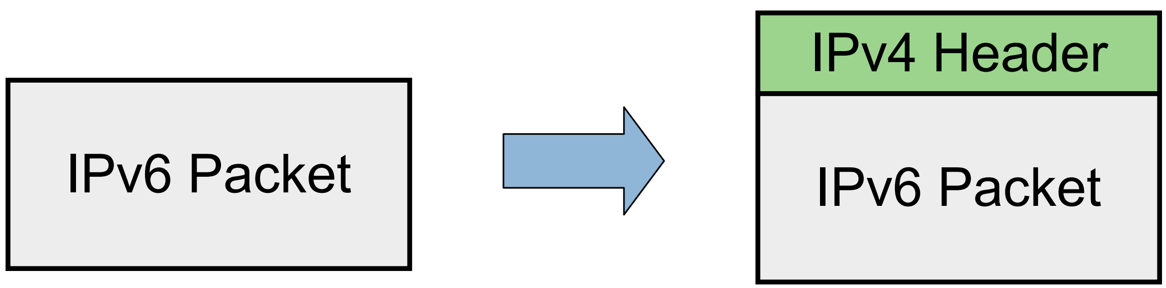
- **No options, *no checksum***

Interoperability

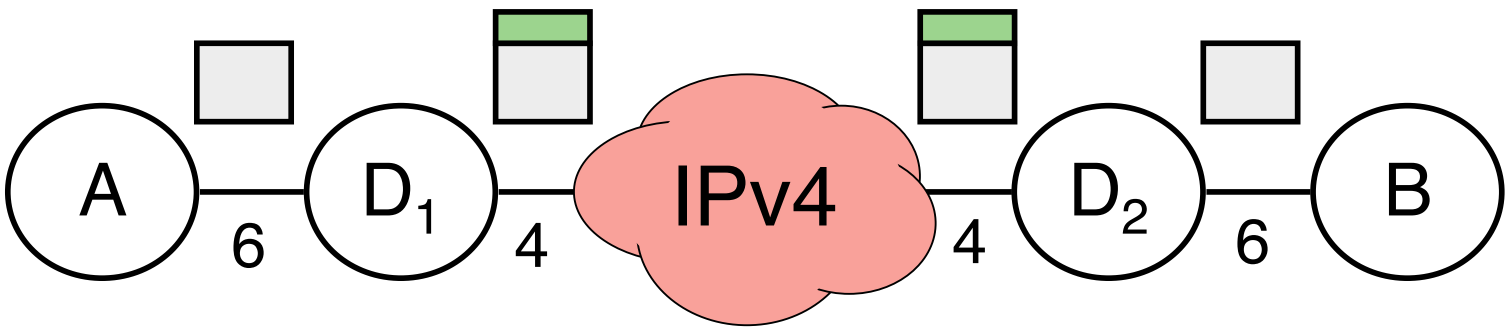
- **RFC 4291**
- **Every IPv4 address has an associated IPv6 address**
- **Simply prefix 32-bit IPv4 address with 96 bits of 0**
  - E.g., ::128.148.32.2
- **Two IPv6 endpoints must have IPv6 stacks**
- **Transit network:**
  - v6 – v6 – v6 : ✔
  - v4 – v4 – v4 : ✔
  - v4 – v6 – v4 : ✔
  - v6 – v4 – v6 : ✗!!

IP Tunneling

- Encapsulate an IP packet inside another IP packet
- Makes an end-to-end path look like a single IP hop



IPv6 in IPv4 Tunneling



- **Key issues: configuring the tunnels**
  - Determining addresses
  - Determining routes
  - Deploying relays to encapsulate/forward/decapsulate
- **6to4 is a standard to automate this**
  - Deterministic address generation
  - Anycast 192.88.99.1 to find gateway into IPv6 network

Other uses for tunneling

- Virtual Private Networks
- Use case: access CS network from the outside
- Set up an encrypted TCP connection between your computer and Brown's OpenVPN server
- Configure routes to Brown's internal addresses to go through this connection
- Can connect two remote sites securely

Extension Headers

- **Two types: hop-by-hop and end-to-end**
- **Both have a next header byte**
- **Last next header also denotes transport protocol**
- **Destination header: intended for IP endpoint**
  - Fragment header
  - Routing header (loose source routing)
- **Hop-by-hop headers: processed at each hop**
  - Jumbogram: packet is up to 2^{32} bytes long!

Example Next Header Values

- 0: Hop by hop header
- 1: ICMPv4
- 4: IPv4
- 6:TCP
- 17: UDP
- 41: IPv6
- 43: Routing Header
- 44: Fragmentation Header
- 58: ICMPv6

Fragmentation and MTU

- **Fragmentation is supported only on end hosts!**
- **Hosts should do MTU discovery**
- **Routers will not fragment: just send ICMP saying packet was too big**
- **Minimum MTU is 1280-bytes**
  - If some link layer has smaller MTU, must interpose fragmentation reassembly underneath

Current State

- **IPv6 Deployment has been slow**
- **Most end hosts have dual stacks today (Windows, Mac OSX, Linux, *BSD, Solaris)**
- **2008 Google study:**
  - Less than 1% of traffic in any country
- **Requires all parties to work!**
  - Servers, Clients, DNS, ISPs, all routers
- **IPv4 and IPv6 will coexist for a long time**

Coming Up

- IP handins: please pay attention to the issues we discussed today, good luck!
- Next week: Transport Layer