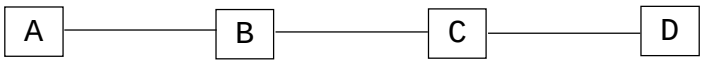
# CSE/EE 461 Lecture 8
# Routing Scalability

Tom Anderson
tom@cs.washington.edu

## Routing Recap So Far



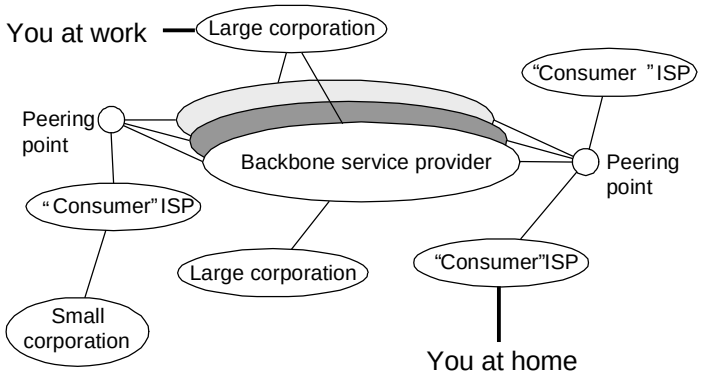

- Distance Vector (RIP)
  - Each node stores table of best next hop to each host
  - At A, get to D via B; at B, get to D via C; at C, get to D via D
  - Exchange table with neighbors
- Link State (OSPF)
  - Each node stores table of all network edges (A-B, B-C, C-D)
  - Exchange directly connected edges with everyone
- Path Vector (BGP)
  - Each node stores best sequence of routers to each host
  - At A, get to D via B-C-D; at B, get to D via C-D; at C, get to D via D
  - Exchange table with neighbors

# Scalability Concerns

- Size of routing tables
  - Solution: Hierarchical IP addresses
    - Allocate addresses to match network structure
    - Aggregate addresses dynamically
    - Virtual IP addresses
- Unscalable routing algorithms
  - Solution: Interdomain routing (BGP)
    - Separately route inside an organization vs. between domains
    - Explicit policy knobs for crossing organizational boundaries

# Structure of the Internet

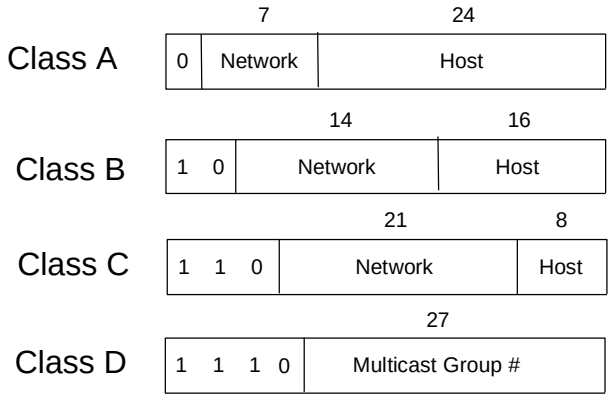
Inter-domain versus intra-domain routing



# How do we reduce router tables?

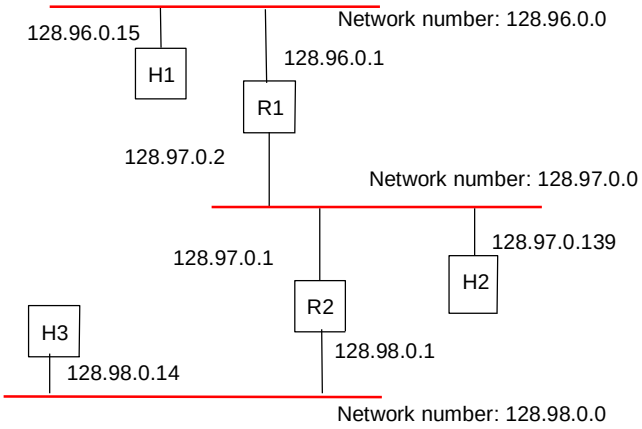
- Assign IP addresses to reflect network structure
  - Each IP address has “network #” and “host #”
  - Network # globally unique; assigned by IANA or ISP
  - Host # locally assigned; all hosts on same network share same network #
- Routing tables only need network #
  - Local delivery inside a network doesn’t involve router

# IPv4 Address Formats




- 32 bits written in “dotted quad” notation
  - Example: 18.31.0.135

# Network Example




# Getting an IP address

- "Static" IP addresses
  - IP address assigned to each machine; sysadmin must configure
- Dynamic Host Configuration Protocol (DHCP)
  - One DHCP server with the bootstrap info
    - Host address, gateway address, subnet mask, …
    - Find DHCP server using LAN broadcast
  - Addresses are leased; renew periodically
- "Stateless" Autoconfiguration (in IPv6)
  - Get rid of server – reuse Ethernet addresses for lower portion of address (uniqueness) and learn higher portion from routers

## Updated Forwarding Routine

Addresses have network and host portions

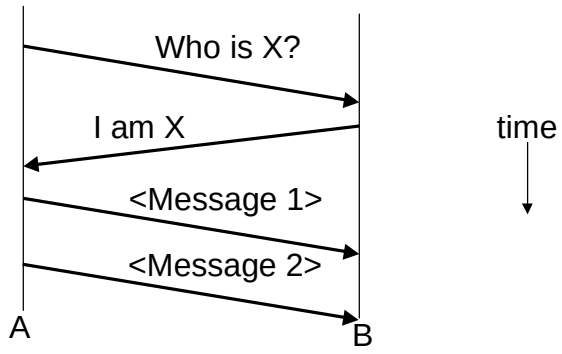
- At sender:
  - If destination network is the same as the host network, then deliver locally (without router).
  - Otherwise send to the router
- At router:
  - If destination network is directly attached then deliver locally (using ARP)
  - Otherwise look up destination network in routing table to find next hop and send to next router.

## Address Resolution Protocol (ARP)

- Need LAN address to send packet on last hop
  - Requires mapping from IP to MAC addresses
- ARP is a dynamic approach to learn mapping
  - Node A sends broadcast query for IP address X
  - Node B with IP address X replies with its MAC address M
  - A caches (X, M); old information is timed out (~15 mins)
  - Also: B caches A's MAC and IP addresses, other nodes refresh

# ARP Example

- To send first message use ARP to learn MAC address
- For later messages (common case), consult ARP cache




# IP Address Issues

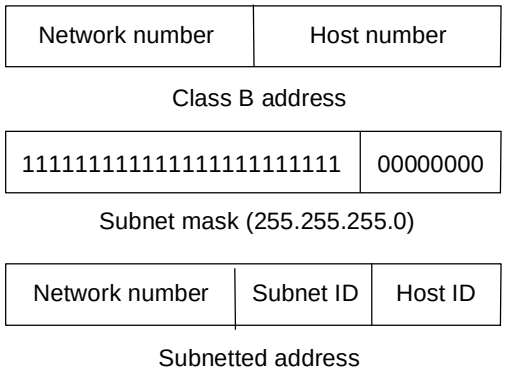
- What if we run out of IP addresses?
  - 4B possible IP addresses
  - 20B+ microprocessors fabricated in 2001
- Rigid class A/B/C structure causes internal fragmentation
  - Means we will run out of addresses faster!
- Router tables still too large
  - 2M class C networks!
  - Need to be able to aggregate networks

# Efficient IP Address Allocation

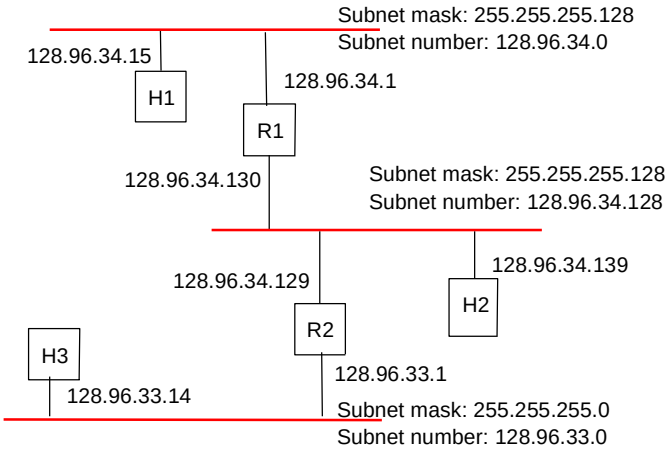
- Subnets
  - split net addresses between multiple sites
- Supernets
  - assign adjacent net addresses to same organization
  - classless routing (CIDR)
    - combine routing table entries whenever all nodes with same prefix share same hop
  - B-tree “search” for matching prefix
- Hardware support for fast prefix lookup

# Subnetting – More Hierarchy

- Split up one network number into multiple physical networks
- Internal structure isn’t propagated
- Helps allocation efficiency



# Subnet Example

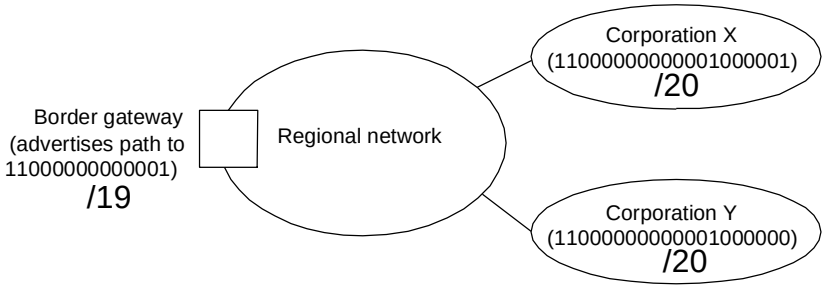



# CIDR (Supernetting)

- CIDR = Classless Inter-Domain Routing
- Aggregate adjacent advertised network routes
  - Ex: ISP has class C addresses 192.4.16 through 192.4.31
  - Really like one larger 20 bit address class …
  - Advertise as such (network number, prefix length)
  - Reduces size of routing tables
- But IP forwarding is more involved
  - Find longest matching prefix

# CIDR Example

- X and Y routes can be aggregated because they form a bigger contiguous range.




- But aggregation isn't always possible. Why?

# IP Forwarding Revisited

- IP address still has network #, host #
  - With class A/B/C, split was obvious from first few bits
  - Now split varies as you traverse the network!
- Routing table contains variable length "prefixes"
  - IP address and length indicating what bits are fixed
  - Next hop to use for each prefix
- To find the next hop:
  - There can be multiple matches; take the longest matching prefix

# IPv6 – 128 bit addresses


- With IPv4, number of hosts limited to 4B
- IPv6 allows every device (PDA, toaster) to have its own IP address
- Modifies packet format
  - How do IPv4 systems communicate with IPv6 ones?
  - How do we switch over?

# Network Address Translation

- Every network, organization, or ISP can have its own private IPv4 address space
  - Example: hosts assigned 10.01, 10.02, …
  - Internal communication occurs normally
  - All external communication goes through NAT
  - NAT transforms each packet to maintain illusion of global Internet addresses

## NAT Mechanics

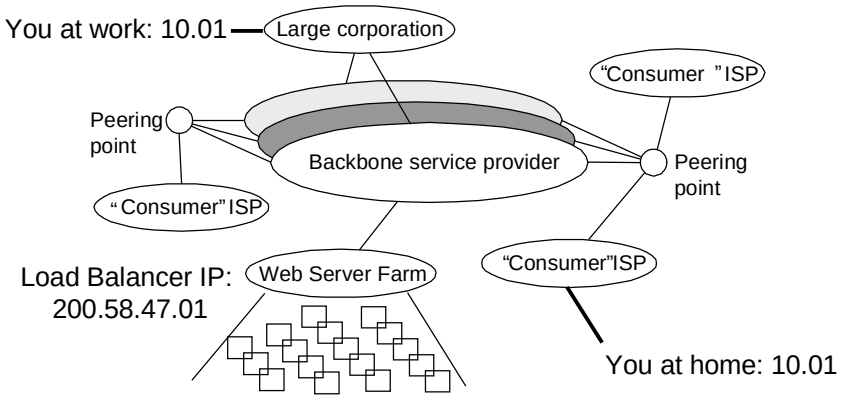
- Host wants to access an external web service
  - Sends request; From: 10.01; To: 200.58.47.01
- NAT transforms outgoing packets
  - Rewrite source address so reply comes back
    - From: 128.80.40.01, port 5736; To: 200.58.47.01
  - Return address is dynamically allocated on connection setup; torn down on connection termination
- NAT transforms incoming packets
  - Match incoming reply to internal host making request
  - Rewrite destination address so reply goes to host
    - From: 200.58.47.01; To: 10.01

## Load Balancers

- A highly available, scalable web service can require thousands of servers
- Original approach:
  - Use DNS to translate web service name to IP addresses of individual servers
  - Load balance by “round robin” – give different clients different server addresses
- Want server failures to be transparent
  - Use DNS to translate name to a single address
  - Load balancer forwards incoming requests to servers
  - Translates each incoming/outgoing packet, as in NAT

# NAT/Load Balancer Example

Private IP address spaces; no more global connectivity except for visible services (e.g., yahoo, kmart.com)




# Summary

- Hierarchical address allocation helps routing scale
  - Addresses are constrained by topology
  - Hide internal structure within a domain via subnets
  - Only need to advertise and compute routes for network aggregates
  - Keep hosts simple and let routers worry about routing
- ARP learns the mapping from IP to MAC address