15-744: Computer Networking

L-4 Routers

Routers

- How do routers process IP packets
- How do you build a router
- Assigned reading
  - [P+98] A 50 Gb/s IP Router
  - [D+97] Small Forwarding Tables for Fast Routing Lookups

Forwarding vs. Routing

- Forwarding: the process of moving packets from input to output
  - The forwarding table
  - Information in the packet
- Routing: process by which the forwarding table is built and maintained
  - One or more routing protocols
  - Procedures (algorithms) to convert routing info to forwarding table.

Outline

- Alternative methods for packet forwarding
- IP packet routing
- Variable prefix match
- IP router design
- Routing protocols – distance vector

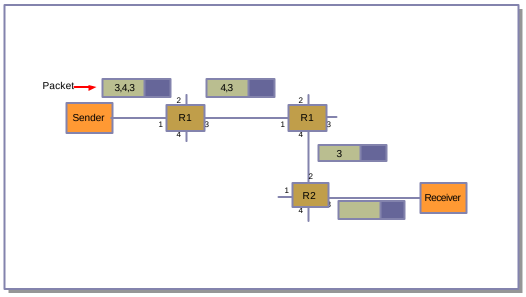
Techniques for Forwarding Packets

- Source routing
  - Packet carries path
- Table of virtual circuits
  - Connection routed through network to setup state
  - Packets forwarded using connection state
- Table of global addresses (IP)
  - Routers keep next hop for destination
  - Packets carry destination address

Source Routing

- List entire path in packet
  - Driving directions (north 3 hops, east, etc..)
- Router processing
  - Examine first step in directions
  - Strip first step from packet
  - Forward to step just stripped off

Source Routing Example



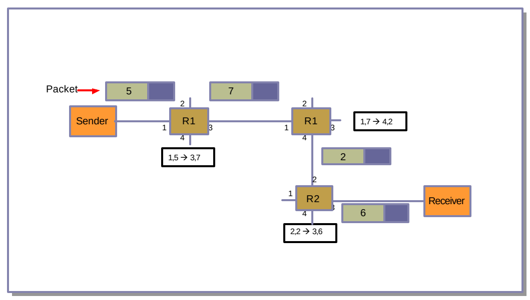
Source Routing

- Advantages
  - Switches can be very simple and fast
- Disadvantages
  - Variable (unbounded) header size
  - Sources must know or discover topology (e.g., failures)
- Typical use
  - Ad-hoc networks (DSR)
  - Machine room networks (Myrinet)

Virtual Circuits/Tag Switching

- Connection setup phase
  - Use other means to route setup request
  - Each router allocates flow ID on local link
  - Creates mapping of inbound flow ID/port to outbound flow ID/port
- Each packet carries connection ID
  - Sent from source with 1st hop connection ID
- Router processing
  - Lookup flow ID – simple table lookup
  - Replace flow ID with outgoing flow ID
  - Forward to output port

Virtual Circuits Examples



Virtual Circuits

- Advantages
  - More efficient lookup (simple table lookup)
  - More flexible (different path for each flow)
  - Can reserve bandwidth at connection setup
  - Easier for hardware implementations
- Disadvantages
  - Still need to route connection setup request
  - More complex failure recovery – must recreate connection state
- Typical uses
  - ATM – combined with fix sized cells
  - MPLS – tag switching for IP networks

IP Datagrams on Virtual Circuits

- Challenge – when to setup connections
  - At bootup time – permanent virtual circuits (PVC)
    - Large number of circuits
  - For every packet transmission
    - Connection setup is expensive
  - For every connection
    - What is a connection?
    - How to route connectionless traffic?

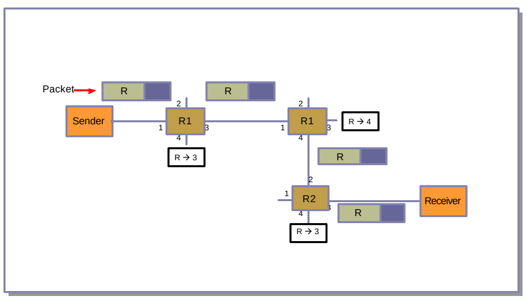
IP Datagrams on Virtual Circuits

- Traffic pattern
  - Few long lived flows
  - Flow – set of data packets from source to destination
  - Large percentage of packet traffic
  - Improving forwarding performance by using virtual circuits for these flows
- Other traffic uses normal IP forwarding

Global Addresses (IP)

- Each packet has destination address
- Each switch has forwarding table of destination → next hop
  - At v and x: destination → east
  - At w and y: destination → south
  - At z: destination → north
- Distributed routing algorithm for calculating forwarding tables

Global Address Example



Router Table Size

- One entry for every host on the Internet
  - 100M entries,doubling every year
- One entry for every LAN
  - Every host on LAN shares prefix
  - Still too many, doubling every year
- One entry for every organization
  - Every host in organization shares prefix
  - Requires careful address allocation

Outline

- Alternative methods for packet forwarding
- IP packet routing
- Variable prefix match
- IP router design
- Routing protocols – distance vector

Original IP Route Lookup

- Address classes
  - A: 0 | 7 bit network | 24 bit host (16M each)
  - B: 10 | 14 bit network | 16 bit host (64K)
  - C: 110 | 21 bit network | 8 bit host (255)
- Address would specify prefix for forwarding table
  - Simple lookup

Original IP Route Lookup – Example

- www.cmu.edu address 128.2.11.43
  - Class B address – class + network is 128.2
  - Lookup 128.2 in forwarding table
  - Prefix – part of address that really matters for routing
- Forwarding table contains
  - List of class+network entries
  - A few fixed prefix lengths (8/16/24)
- Large tables
  - 2 Million class C networks

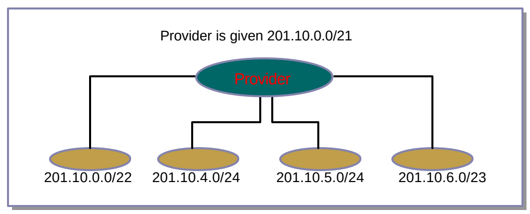
CIDR Revisited

- Supernets
  - Assign adjacent net addresses to same org
  - Classless routing (CIDR)
- How does this help routing table?
  - Combine routing table entries whenever all nodes with same prefix share same hop

CIDR Example

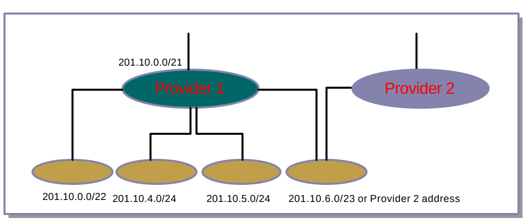
- Network provide is allocated 8 class C chunks, 201.10.0.0 to 201.10.7.255
  - Allocation uses 3 bits of class C space
  - Remaining 21 bits are network number, written as 201.10.0.0/21
- Replaces 8 class C routing entries with 1 combined entry
  - Routing protocols carry prefix with destination network address
  - Longest prefix match for forwarding

CIDR Illustration



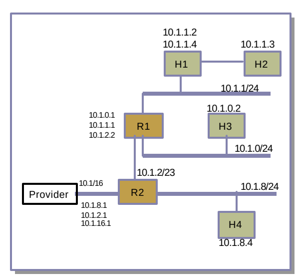
CIDR Shortcomings

- Multi-homing
- Customer selecting a new provider



Routing to the Network

- Packet to 10.1.1.3 arrives
- Path is R2 – R1 – H1 – H2

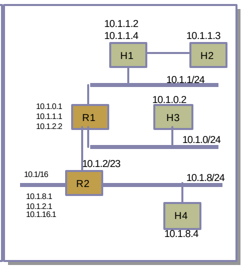


Routing Within the Subnet

- Packet to 10.1.1.3
- Matches 10.1.0.0/23

Routing table at R2

Destination	Next Hop	Interface
127.0.0.1	127.0.0.1	lo0
Default or 0/0	provider	10.1.16.1
10.1.8.0/24	10.1.8.1	10.1.8.1
10.1.2.0/23	10.1.2.1	10.1.2.1
10.1.0.0/23	10.1.2.2	10.1.2.1

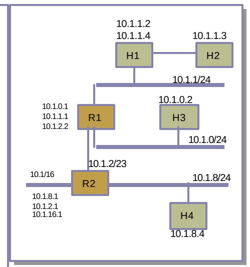


Routing Within the Subnet

- Packet to 10.1.1.3
- Matches 10.1.1.1/31
  - Longest prefix match

Routing table at R1

Destination	Next Hop	Interface
127.0.0.1	127.0.0.1	lo0
Default or 0/0	10.1.2.1	10.1.2.2
10.1.0.0/24	10.1.0.1	10.1.0.1
10.1.1.0/24	10.1.1.1	10.1.1.4
10.1.2.0/23	10.1.2.2	10.1.2.2
10.1.1.2/31	10.1.1.2	10.1.1.2

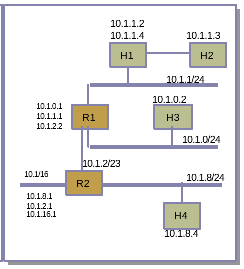


Routing Within the Subnet

- Packet to 10.1.1.3
- Direct route
  - Longest prefix match

Routing table at H1

Destination	Next Hop	Interface
127.0.0.1	127.0.0.1	lo0
Default or 0/0	10.1.1.1	10.1.1.2
10.1.1.0/24	10.1.1.2	10.1.1.1
10.1.1.3/31	10.1.1.2	10.1.1.2



Global Addresses

- Advantages
  - Stateless – simple error recovery
- Disadvantages
  - Every switch knows about every destination
    - Potentially large tables
  - All packets to destination take same route

Comparison

	Source Routing	Global Addresses	Virtual Circuits
Header Size	Worst	OK – Large address	Best
Router Table Size	None	Number of hosts (prefixes)	Number of circuits
Forward Overhead	Best	Prefix matching	Pretty Good
Setup Overhead	None	None	Connection Setup
Error Recovery	Tell all hosts	Tell all routers	Tell all routers and Tear down circuit and re-route

How do we set up Routing Tables?

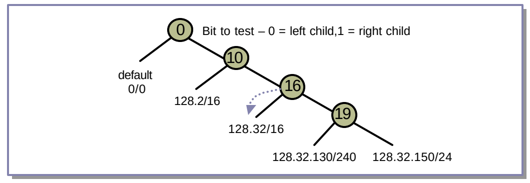
- Graph theory to compute "shortest path"
  - Switches = nodes
  - Links = edges
  - Delay, hops = cost
- Need to adapt to changes in topology

Outline

- Alternative methods for packet forwarding
- IP packet routing
- Variable prefix match
- IP router design
- Routing protocols – distance vector

How To Do Variable Prefix Match

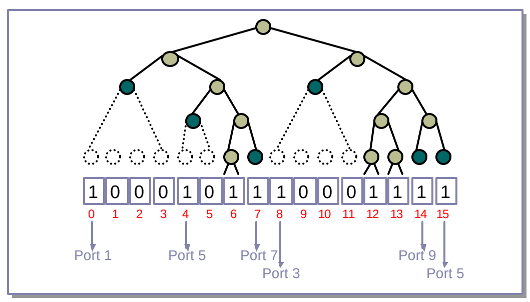
- Traditional method – Patricia Tree
  - Arrange route entries into a series of bit tests
- Worst case = 32 bit tests
  - Problem: memory speed is a bottleneck



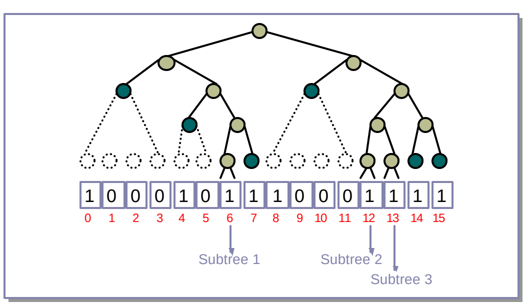
Speeding up Prefix Match (P+98)

- Cut prefix tree at 16 bit depth
  - 64K bit mask
  - Bit = 1 if tree continues below cut (root head)
  - Bit = 1 if leaf at depth 16 or less (genuine head)
  - Bit = 0 if part of range covered by leaf

Prefix Tree



Prefix Tree



Speeding up Prefix Match (P+98)

- Each 1 corresponds to either a route or a subtree
  - Keep array of routes/pointers to subtree
  - Need index into array – how to count # of 1s
  - Keep running count to 16bit word in base index + code word (6 bits)
  - Need to count 1s in last 16bit word
    - Clever tricks
- Subtrees are handled separately

Speeding up Prefix Match (P+98)

- Scaling issues
  - How would it handle IPv6
- Other possiblities
  - Why were the cuts done at 16/24/32 bits?
  - Improve data structure by shuffling bits

Speeding up Prefix Match - Alternatives

- Route caches
  - Temporal locality
  - Many packets to same destination
- Other algorithms
  - Waldvogel – Sigcomm 97
    - Binary search on hash tables
    - Works well for larger adresses
  - Bremler-Barr – Sigcomm 99
    - Clue = prefix length matched at previous hop
    - Why is this useful?

Speeding up Prefix Match - Alternatives

- Content addressable memory (CAM)
  - Hardware based route lookup
  - Input = tag, output = value associated with tag
  - Requires exact match with tag
    - Multiple cycles (1 per prefix searched) with single CAM
    - Multiple CAMs (1 per prefix) searched in parallel
  - Ternary CAM
    - 0,1,don't care values in tag match
    - Priority (I.e. longest prefix) by order of entries in CAM

Outline

- Alternative methods for packet forwarding
- IP packet routing
- Variable prefix match
- IP router design
- Routing protocols – distance vector

What Does a Router Look Like?

- Line cards
  - Network interface cards
- Forwarding engine
  - Fast path routing (hardware vs. software)
- Backplane
  - Switch or bus interconnect
- Network controller
  - Handles routing protocols, error conditions

Router Processing (P+88)

- Packet arrives arrives at inbound line card
- Header transferred to forwarding engine
  - 24/56 bytes of packet + link layer info
- Forwarding engine transmits result to line card
- Packet copied to outbound line card

Forwarding Engine (P+88)

- General purpose processor + software
- 8KB L1 Icache
  - Holds full forwarding code
- 96KB L2 cache
  - Forwarding table cache
- 16MB L3 cache
  - Full forwarding table x 2 - double buffered for updates

Forwarding Engine (P+88)

- Checksum updated but not checked
- Options handled by network proc
- Fragmentation handed by network processor
- Multicast packets are copied on input line card
- Packet trains help route hit rate
  - Packet train = sequence of packets for same/similar flows

Network Processor

- Runs routing protocol and downloads forwarding table to forwarding engines
  - Two forwarding tables per engine to allow easy switchover
- Performs "slow" path processing
  - Handles ICMP error messages
  - Handles IP option processing

Switch Design Issues

- Have N inputs and M outputs
  - Multiple packets for same output – output contention
  - Switch contention – switch cannot support arbitrary set of transfers
    - Crossbar
    - Bus
      - High clock/transfer rate needed for bus
    - Banyan net
      - Complex scheduling needed to avoid switch contention
- Solution – buffer packets where needed

Switch Buffering

- Input buffering
  - Which inputs are processed each slot – schedule?
  - Head of line packets destined for busy output blocks other packets
- Output buffering
  - Output may receive multiple packets per slot
  - Need speedup proportional to # inputs
- Internal buffering
  - Head of line blocking
  - Amount of buffering needed

Line Card Interconnect (P+88)

- Virtual output buffering
  - Maintain per output buffer at input
  - Solves head of line blocking problem
  - Each of MxN input buffer places bid for output
- Crossbar connect
- Challenge: map of bids to schedule for crossbar

Switch Scheduling (P+88)

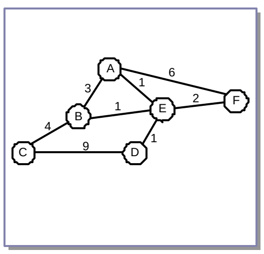
- Schedule for 128 byte slots
  - Greedy mapping of inputs to outputs
- Fairness
  - Order of greedy matching permuted randomly
  - Priority given to forwarding engine in schedule (why?)
- Parallelized
  - Check independent paths simultaneously

Outline

- Alternative methods for packet forwarding
- IP packet routing
- Variable prefix match
- IP router design
- Routing protocols – distance vector

Factors Affecting Routing

- Routing algorithms view the network as a graph
- Problem: find lowest cost path between two nodes
- Factors
  - Static topology
  - Dynamic load
  - Policy



Two Main Approaches

- Distance-vector (DV) protocols
- Link state (LS) protocols

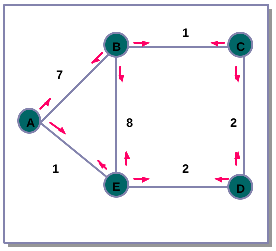
Distance Vector Protocols

- Employed in the early Arpanet
- Distributed next hop computation
- Unit of information exchange
  - Vector of distances to destinations
- Distributed Bellman-Ford Algorithm

Distributed Bellman-Ford

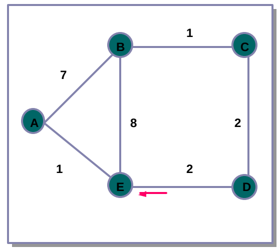
- Start Conditions:
  - Each router starts with a vector of (zero) distances to all directly attached networks
- Send step:
  - Each router advertises its current vector to all neighboring routers
- Receive step:
  - Upon receiving vectors from each of its neighbors, router computes its own distance to each neighbor
  - Then, for every network X, router finds that neighbor who is closer to X than to any other neighbor
  - Router updates its cost to X
  - After doing this for all X, router goes to send step

Example - Initial Distances



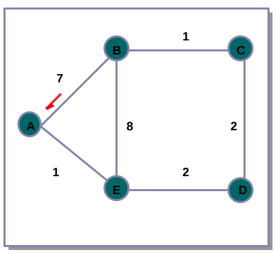
Info at Node	Distance to Node A	B	C	D	E
A	0	7	~	~	1
B	7	0	1	~	8
C	~	1	0	2	~
D	~	~	2	0	2
E	1	8	~	2	0

E Receives D's Routes; Updates Cost



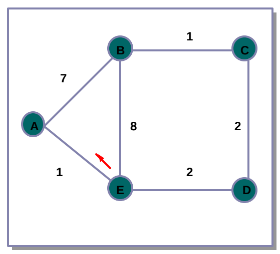
Info at Node	Distance to Node A	B	C	D	E
A	0	7	~	~	1
B	7	0	1	~	8
C	~	1	0	2	~
D	~	~	2	0	2
E	1	8	4	2	0

A receives B's; Updates Cost



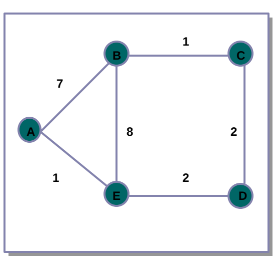
Info at Node	Distance to Node A	B	C	D	E
A	0	7	8	~	1
B	7	0	1	~	8
C	~	1	0	2	~
D	~	~	2	0	2
E	1	8	4	2	0

A receives E's routes; Updates Costs



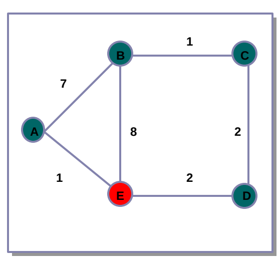
Info at Node	Distance to Node A	B	C	D	E
A	0	7	5	3	1
B	7	0	1	~	8
C	~	1	0	2	~
D	~	~	2	0	2
E	1	8	4	2	0

Final Distances



Info at Node	Distance to Node A	B	C	D	E
A	0	6	5	3	1
B	6	0	1	3	5
C	5	1	0	2	4
D	3	3	2	0	2
E	1	5	4	2	0

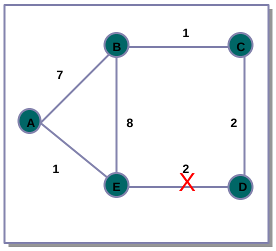
View From a Node



E's routing table

dest	Next hop A	B	D
A	1	14	5
B	7	8	5
C	6	9	4
D	4	11	2

Final Distances After Link Failure



Info at Node	Distance to Node A	B	C	D	E
A	0	7	8	10	1
B	7	0	1	3	8
C	8	1	0	2	9
D	10	3	2	0	11
E	1	8	9	11	0

Next Lecture: Intra-Domain Routing

- Routing algorithms
  - Distance vector routing – challenges
  - Link state routing
- How to make routing adapt to load
- How to make routing scale
- Assigned reading
  - [KZ89] The revised ARPANET routing metric