1 # EECS 122, Lecture 12

Kevin Fall

kfall@cs.berkeley.edu

2 ## IP Forwarding

- Hosts and routers may need to perform indirect delivery (through a router)
- Often hosts have a single router to use (“default router ”), but in general both routers and hosts can make use of multiple routers
- Question: Which router to use?

3 ## Which Router to Use?

- Really two questions:
  - what algorithm and data structures are needed at a router to properly forward an IP datagram to its next hop
  - how do the routers become aware of the optimal paths (next hops) to use
- Focus on the first question for now…

4 ## Routing Tables

- Hosts and routers maintain tables containing routing information
- More precisely, *routing tables* are used to compute optimal routes and *forwarding tables* are used to dictate immediate next-hops for each destination
- Forwarding tables are updated by routing protocols based on the routing tables

5 ## Typical Configuration

6 Forwarding Tables (FIBs)

- Also called “forwarding information base,” (FIB) generally contains the following:

7 Using Forwarding Tables

- Table entries (routes) are selected based on destination address of packet to route
- So, to route to destination D, we compute (D & M1) and compare with D1. If they match, H1 is a possible next hop.

8 Using Forwarding Tables

- By comparing (D & M) for each route entry *i*, we get a set of matching route entries
- The entry with the longest prefix (most number of ‘1’ bits in the mask field) is the single winner
- Algorithm called “longest” or “best” prefix match, and is used by all IP routers

9 FIB Example

10 FIB Example

11 FIB Example

12 Implementation

- Longest Prefix Match Algorithm

    - must be executed for each packet
    - must work with the entire Internet routing table (2/25/99: 42869 entries at MAE-E, 16 and 24 bit prefixes dominate [~28k])
- Two Common Software Approaches
    - Hash-based lookup
    - Tree-based lookup

13 Tree-Based Lookups

- Tree search appears useful, but should not be based on direct key compares (we are using bit prefixes, not full keys)
- Rather, observe each key (destination address) is composed of a series of bits
- Each tree depth will map one or more bits in the search key

14 Search Tries (from reTRIEval)

- M-ary tree, where each node is an M-place vector with components corresponding to characters or digits
- Each node on level *l* represents all valid keys beginning with a certain sequence of *l* characters
- Each node specifies an M-way branch, depending on the (M+1)st character/digit

15 Simple Trie Example

- Alphabet: {A,B,D}, words: ADA, ADD, BAA, BAD

16 Patricia Trees

- Keys are represented by binary strings
- To avoid 1-way branching that would happen in a

simple binary search trie, include a *skip* value, which indicates how many bits to skip before examining the next bit at each node

- Sometimes called "path compression"

17 Simple Patricia Example

- Keys: 11001, 111011, 10110111110

18 Patricia Tree Properties

- N-1 internal nodes, N external nodes
- search performance insensitive to insertion order, but very sensitive to distribution of digits
- assuming random distribution, approx # of digits/bits inspected is about log N, for M-ary trie with N nodes [see Knuth vol3 for the (nontrivial) analysis]

19 Simple P-Tree for IP Routing

- (from BSD, circa 1990)
- Keys: 0/0, 128.32.0.0/16, 128.32.8.0/24, 18.0.0.0/8

20 Observations

- This (simple) scheme places common prefixes together in the same leaf (using a linked list)
- Parent pointers in nodes are used to back up in the tree in case of a mis-match (default router or non-contiguous mask)
- Various new approaches have been developed: DP-Tries, LC-Tries, etc.

21 Hashing-Based Approaches

- Original BSD routing support included hash on network routed and host routes, but this was before LPM was required

- Newer hash based scheme provides 32 different hash tables, and looks for exact match
- Simplest: perform linear search for the right hash table, starting w/longest

22 Reaction to LPM

- Three main approaches to problem of IP lookups:
  - faster software algorithms
  - hardware acceleration (e.g. CAMs)
  - protocol modifications
- Faster algorithms have called into question the protocol approaches

23 Hardware Approaches

- Content Addressable Memory (CAM)
  - exact-matching CAMs can be used 1 per prefix length
  - 32 CAMs for IPv4, 128 for IPv6 (expensive)
  - Max size about 8k right now

24 Protocol Approaches

- IP and Tag/Layer 3 “Switching”
  - similar to notion of VCID in circuit-switched networks (and may use them!)
  - requires separate protocol to provide address/TAG mappings
- Basically uses IP packets and IP routing as “signaling” for circuit set-up

25 L3 Switching

- Normal forwarding requires LPM lookup on destination address at each hop

- In L3 Switching, assign a short “tag” to each packet as it enters the network (same domain)
- Tags are rewritten at each hop, and might be carried in L2 (e.g. ATM VC’s)

26 L3 Switching Claimed Benefits

- Label provides fast lookup (1 cycle)
- Can incorporate additional policy or characteristics at ingress router, so that even pkts w/same dest go along different paths
- Provides some functions similar to source routing without the routing cost

27 L3 Switching Complications

- Requires adoption of “label distribution” protocol, which may not span admin. domains
- Probably not worth it on a per-flow basis (web flows)
- Does not completely avoid LPM search
- Time to Live (TTL) question…

28 The TTL Question

- In regular IP forwarding, TTL zero implies discard (good if loops arise)
- No TTL decrementing per hop here…
  - so, decrement it by the amount it should have been decremented along the complete path
  - requires egress LS router to know path length

29 Why L3 Routing Still Needed

- So, can we use L3 switching for everything?

- – No, hard to support these things:
  - world-wide label assignment
  - packet filtering/firewalls
  - non L3-switching hosts
  - non L3-switching routers

30 Multi-Protocol Label Switching (MPLS)

- IETF Internet Drafts addressing L3 switching
- Latest architecture document 2/99
  - – take advantage of ATM switching
  - – be compatible with RSVP/Int Serv, existing routing protocols
  - – support aggregation, multicast, good OA&M

31 MPLS Architecture Features

- Downstream assignment of labels for unicast traffic, driven by either control or data traffic, multiple label dist. protocols
- LIFO “stack” of labels for hierarchy
- Both “normal” and source-routing
- Variable label granularity, label merging
- TTL “fixup”

32 Label Merging

- Suppose you have >1 packets destined for the same IP address but that arrive with different labels
- The ability to assign the same outgoing label for both incoming packets is called “label merging” (some ATM switches)
- Useful to avoid maintenance of a large number of VCs between switches

33 Label Merging Example

34 Label Merging Example

35 Label Merging Example

36 Merging on ATM

- Problem: cells within ATM/AAL5 contain VPI/VCI (VC) numbers. Hardware either carries each VPI/VCI as a label, or each VPI as a label.
- VC Merging: have to reconstruct AAL frames to avoid interleaving cells
- VP Merging: use common VPI with different VCIs to demultiplex