Router and switch architecture

Martin Heusse
Contenu

- Router architecture
- Routing table data structure
What is routing?

• Packet reception
  ✓ Interface FIFO (ring buffer?) holds groups of bits as they arrive
  ✓ Packet queued until treated by central CPU or interface card CPU (throw interrupt)
  ✓ Check CRC, is there space in memory…
  ✓ Packet classification (Dropped? Accepted? Switching method?)
  ✓ Moved to input hold queue

• Packet forwarding
  ✓ Look up routing table
  ✓ Rewrite header (Ethernet, NAT?, TTL, checksum…)
  ✓ Packet moved to output hold queue
Input and Output queues

• Input queues absorb transient forwarding subsystem saturation Configurable
• Output queues holds burst of packets directed to one interface

• Generally, queues hold a given number of packets (not bytes)
  How would you implement a queue? Ring? Chained list? What is the storage unit (MTU size bin, packet, particle…)
  ✓ There can be several queues in parallel (various priorities)
Shared memory — first generation

- Ex.: conventional PC, Cisco 2800, HP ProCurve 7xxx
- Everything stored in same memory space

- Limiting factor: memory access
Shared memory — first generation (cont.)

Cisco 25xx (1993)
Shared memory — first generation (cont.)

Cisco 7200

[Diagram of Cisco 7200 router architecture]
Shared memory — first generation (cont.)

Juniper M40

• Decoupling of *control plane* and *forwarding plane*
  —forwarding by a dedicated ASIC

• 1998 — 40Gb/s

• JunOS based on FreeBSD
Shared memory — first generation (cont.)

PIC: Physical Interface Card
Intelligent line cards — 2d generation

• Ex.: Cisco 7500
• Line cards have some intelligence, write into each other’s memory

• Limiting factor: 1 shared bus (needs to be N times faster than each of N interfaces)
• Central processor dedicated only to control plane
  Distinct from Forwarding plane
Intelligent line cards — 2d generation (cont.)

Cisco 7500

Diagram showing components such as Daily UART, Boot ROM, NVRAM, PCMCIA, Boot Flash, I/O Bus, RSP, Sys Ctrl ASICs, DRAM, Layer 2 Cache, CPU Bus, MemD Ctrl ASICs, Register FPGA, Diag Bus FPGA, SRAM, QA ASIC, Diag Bus, IP/VIP, Cy Bus 0, Cy Bus 1, Cy Bus Arbiter.
Intelligent line cards — 2d generation (cont.)

Versatile Interface Processors (1/interface)
Intelligent line cards + crossbar switch
3d generation

- Ex. Cisco 7600, juniper T-series, HP ProCurve Switch 4200vl...
- Crossbar switch:

- Routing of N simultaneous packet (or cell)
Head of line blocking

- Crossbar needs to be $N$ times faster than each line or need one buffer / output on each input (i.e. one buffer per crosspoint)
- What goes through the crossbar?
  - ✓ ATM cells
  - ✓ particles? (→ packet reassembly)
  - ✓ packets
## Cisco router performance (packets/s)

<table>
<thead>
<tr>
<th>Router</th>
<th>Process switching</th>
<th>Fast switching</th>
</tr>
</thead>
<tbody>
<tr>
<td>2500</td>
<td>800</td>
<td>4400</td>
</tr>
<tr>
<td>2801</td>
<td>3000</td>
<td>90.000</td>
</tr>
<tr>
<td>7200-NPE-G1</td>
<td>79.000</td>
<td>1.018.000</td>
</tr>
<tr>
<td>7600-dCEF720</td>
<td></td>
<td>48.000.000 per slot</td>
</tr>
</tbody>
</table>
A step further

- Check Cisco CEF (Cisco express forwarding)
- Banyan switch
- MPLS: packets carry an identifier of their processing
Routing table

- Static entries, routing protocols, ARP
- Can be large!
- Entries in use are cached (on interface cards, if applicable)
  - the cache holds a small subset of known destinations
Longest match lookup — Routing table storage

- Source: Ruiz-Sanchez, M.A.; Biersack, E.W.; Dabbous, W., ”Survey and taxonomy of IP address lookup algorithms,” Network, IEEE, vol.15, no.2, pp.8-23, Mar/Apr 2001
Path-compressed trie

Prefixes
a 0*
b 01000*
c 011*
d 1*
e 100*
f 1100*
g 1101*
h 1110*
i 1111*

• Useful for sparsely populated space. But many prefixes used in IPv4
• Backtracking necessary: after reaching e and finding out that it does not match, need to go back to d (for 101... for example)
We have seen that prefixes can overlap (see figure 4). In a trie, when two prefixes overlap, one of them is itself a prefix of the other, see figures 7 and 8. Since prefixes represent intervals of contiguous addresses, when two prefixes overlap this means that one interval of addresses contains another interval of addresses, see figures 4 and 8. In fact, that is why an address can be matched to several prefixes. If several prefixes match, the longest prefix match rule is used in order to find the most specific forwarding information. One way to avoid the use of the longest prefix match rule and to still find the most specific forwarding information is to transform a given set of prefixes into a set of disjoint prefixes. Disjoint prefixes do not overlap and thus no address prefix is itself prefix of another one. A trie representing a set of disjoint prefixes will have prefixes at the leaves but not at internal nodes. To obtain a disjoint-prefix binary trie, we simply add leaves to nodes that have only one child. These new leaves are new prefixes that inherit the forwarding information of the closest ancestor marked as a prefix. Finally, internal nodes marked as prefixes are unmarked. For example, figure 10 shows the disjoint-prefix binary trie that corresponds to the trie in figure 7. Prefixes a, a, a have inherited the forwarding information of the original prefix a, which now has been suppressed. Prefix d has been obtained in a similar way. Since prefixes at internal nodes are expanded or pushed down to the leaves of the trie, this technique has been called leaf pushing by Srinivasan et al. [14]. Figure 11 shows the disjoint intervals of addresses that correspond to the disjoint-prefix binary trie of figure 10.

- Disjoint prefixes do not overlap
There are other techniques!
Sources

• S. Keshav; “An engineering approach to computer networking”
• Cisco Router Architecture
• Ross & Kurose “Computer Networking”
• ...