# Chapter 5

# The Network Layer

# Network Layer Design Isues

- Store-and-Forward Packet Switching
- Services Provided to the Transport Layer
- Implementation of Connectionless Service
- Implementation of Connection-Oriented Service
- Comparison of Virtual-Circuit and Datagram Subnets

# Store-and-Forward Packet Switching

The environment of the network layer protocols.

# Implementation of Connectionless Service

A's table

| Dest. | Line (initially) |
|---|---|
| A | – |
| B | B |
| C | C |
| D | B |
| E | C |
| F | C |

| Dest. | Line (later) |
|---|---|
| A | – |
| B | B |
| C | C |
| D | B |
| E | B |
| F | B |

C's table

| Dest. | Line |
|---|---|
| A | A |
| B | A |
| C | – |
| D | D |
| E | E |
| F | E |

E's table

| Dest. | Line |
|---|---|
| A | C |
| B | D |
| C | C |
| D | D |
| E | – |
| F | F |

Routing within a diagram subnet.

# Implementation of Connection-Oriented Service

A's table

| In | | Out | |
|---|---|---|---|
| H1 | 1 | C | 1 |
| H3 | 1 | C | 2 |

C's table

| | | | |
|---|---|---|---|
| A | 1 | E | 1 |
| A | 2 | E | 2 |

E's table

| | | | |
|---|---|---|---|
| C | 1 | F | 1 |
| C | 2 | F | 2 |

Routing within a virtual-circuit subnet.

# Comparison of Virtual-Circuit and

| Issue | Datagram subnet | Virtual-circuit subnet |
|---|---|---|
| Circuit setup | Not needed | Required |
| Addressing | Each packet contains the full source and destination address | Each packet contains a short VC number |
| State information | Routers do not hold state information about connections | Each VC requires router table space per connection |
| Routing | Each packet is routed independently | Route chosen when VC is set up; all packets follow it |
| Effect of router failures | None, except for packets lost during the crash | All VCs that passed through the failed router are terminated |
| Quality of service | Difficult | Easy if enough resources can be allocated in advance for each VC |
| Congestion control | Difficult | Easy if enough resources can be allocated in advance for each VC |

# Routing Algorithms

- The Optimality Principle
- Shortest Path Routing
- Flooding
- Distance Vector Routing
- Link State Routing
- Hierarchical Routing
- Broadcast Routing
- Multicast Routing
- Routing for Mobile Hosts
- Routing in Ad Hoc Networks

# Routing Algorithms (2)

Conflict between fairness and optimality.

# The Optimality Principle

(a) A subnet. (b) A sink tree for router B.

# Shortest Path Routing

The first 5 steps used in computing the shortest path from A to D. The arrows indicate the working node.

# Flooding

```
#define MAX_NODES 1024                    /* maximum number of nodes */
#define INFINITY 1000000000               /* a number larger than every maximum path */
int n, dist[MAX_NODES][MAX_NODES];/* dist[i][j] is the distance from i to j */

void shortest_path(int s, int t, int path[])
{ struct state {                          /* the path being worked on */
      int predecessor;                    /* previous node */
      int length;                         /* length from source to this node */
      enum {permanent, tentative} label;  /* label state */
  } state[MAX_NODES];

  int i, k, min;
  struct state *p;

  for (p = &state[0]; p < &state[n]; p++) {  /* initialize state */
      p->predecessor = –1;
      p->length = INFINITY;
      p->label = tentative;
  }
  state[t].length = 0;  state[t].label = permanent;
  k = t;                                  /* k is the initial working node */
```

Dijkstra's algorithm to compute the shortest path through a graph.

# Flooding (2)

```
do {                                        /* Is there a better path from k? */
    for (i = 0; i < n; i++)                 /* this graph has n nodes */
        if (dist[k][i] != 0 && state[i].label == tentative) {
            if (state[k].length + dist[k][i] < state[i].length) {
                state[i].predecessor = k;
                state[i].length = state[k].length + dist[k][i];
            }
        }

    /* Find the tentatively labeled node with the smallest label. */
    k = 0; min = INFINITY;
    for (i = 0; i < n; i++)
        if (state[i].label == tentative && state[i].length < min) {
            min = state[i].length;
            k = i;
        }
    state[k].label = permanent;
} while (k != s);

/* Copy the path into the output array. */
i = 0;  k = s;
do {path[i++] = k; k = state[k].predecessor; } while (k >= 0);
}
```

Dijkstra's algorithm to compute the shortest path through a graph.

# Distance Vector Routing

(a)

| To | A | I | H | K | New estimated delay from J | Line |
|---|---|---|---|---|---|---|
| A | 0 | 24 | 20 | 21 | 8 | A |
| B | 12 | 36 | 31 | 28 | 20 | A |
| C | 25 | 18 | 19 | 36 | 28 | I |
| D | 40 | 27 | 8 | 24 | 20 | H |
| E | 14 | 7 | 30 | 22 | 17 | I |
| F | 23 | 20 | 19 | 40 | 30 | I |
| G | 18 | 31 | 6 | 31 | 18 | H |
| H | 17 | 20 | 0 | 19 | 12 | H |
| I | 21 | 0 | 14 | 22 | 10 | I |
| J | 9 | 11 | 7 | 10 | 0 | – |
| K | 24 | 22 | 22 | 0 | 6 | K |
| L | 29 | 33 | 9 | 9 | 15 | K |
| | JA delay is 8 | JI delay is 10 | JH delay is 12 | JK delay is 6 | | |

Vectors received from J's four neighbors

New routing table for J

(b)

(a) A subnet. (b) Input from A, I, H, K, and the new routing table for J.

# Distance Vector Routing (2)

The count-to-infinity problem.

# Link State Routing

Each router must do the following:

1. Discover its neighbors, learn their network address.
2. Measure the delay or cost to each of its neighbors.
3. Construct a packet telling all it has just learned.
4. Send this packet to all other routers.
5. Compute the shortest path to every other router.

# Learning about the Neighbors

(a) Nine routers and a LAN. (b) A graph model of (a).

# Measuring Line Cost

A subnet in which the East and West parts are connected by two lines.

# Building Link State Packets

(a)

Link State Packets

| A | |
|---|---|
| Seq. | |
| Age | |
| B | 4 |
| E | 5 |

| B | |
|---|---|
| Seq. | |
| Age | |
| A | 4 |
| C | 2 |
| F | 6 |

| C | |
|---|---|
| Seq. | |
| Age | |
| B | 2 |
| D | 3 |
| E | 1 |

| D | |
|---|---|
| Seq. | |
| Age | |
| C | 3 |
| F | 7 |

| E | |
|---|---|
| Seq. | |
| Age | |
| A | 5 |
| C | 1 |
| F | 8 |

| F | |
|---|---|
| Seq. | |
| Age | |
| B | 6 |
| D | 7 |
| E | 8 |

(b)

(a) A subnet. (b) The link state packets for this subnet.

# Distributing the Link State Packets

| Source | Seq. | Age | Send flags | | | ACK flags | | | Data |
|---|---|---|---|---|---|---|---|---|---|
| | | | A | C | F | A | C | F | |
| A | 21 | 60 | 0 | 1 | 1 | 1 | 0 | 0 | |
| F | 21 | 60 | 1 | 1 | 0 | 0 | 0 | 1 | |
| E | 21 | 59 | 0 | 1 | 0 | 1 | 0 | 1 | |
| C | 20 | 60 | 1 | 0 | 1 | 0 | 1 | 0 | |
| D | 21 | 59 | 1 | 0 | 0 | 0 | 1 | 1 | |

The packet buffer for router B in the previous slide (Fig. 5-13).

# Hierarchical Routing

(a)

Full table for 1A

| Dest. | Line | Hops |
|---|---|---|
| 1A | – | – |
| 1B | 1B | 1 |
| 1C | 1C | 1 |
| 2A | 1B | 2 |
| 2B | 1B | 3 |
| 2C | 1B | 3 |
| 2D | 1B | 4 |
| 3A | 1C | 3 |
| 3B | 1C | 2 |
| 4A | 1C | 3 |
| 4B | 1C | 4 |
| 4C | 1C | 4 |
| 5A | 1C | 4 |
| 5B | 1C | 5 |
| 5C | 1B | 5 |
| 5D | 1C | 6 |
| 5E | 1C | 5 |

(b)

Hierarchical table for 1A

| Dest. | Line | Hops |
|---|---|---|
| 1A | – | – |
| 1B | 1B | 1 |
| 1C | 1C | 1 |
| 2 | 1B | 2 |
| 3 | 1C | 2 |
| 4 | 1C | 3 |
| 5 | 1C | 4 |

(c)

Hierarchical routing.

# Broadcast Routing

Reverse path forwarding. (a) A subnet. (b) a Sink tree. (c) The tree built by reverse path forwarding.

# Multicast Routing

(a) A network. (b) A spanning tree for the leftmost router.
(c) A multicast tree for group 1. (d) A multicast tree for group 2.

# Routing for Mobile Hosts

A WAN to which LANs, MANs, and wireless cells are attached.

# Routing for Mobile Hosts (2)

Packet routing for mobile users.

# Routing in Ad Hoc Networks

Possibilities when the routers are mobile:

1. Military vehicles on battlefield.
   - No infrastructure.
2. A fleet of ships at sea.
   - All moving all the time
3. Emergency works at earthquake .
   - The infrastructure destroyed.
4. A gathering of people with notebook computers.
   - In an area lacking 802.11.

# Route Discovery

a) (a) Range of A's broadcast.

b) (b) After B and D have received A's broadcast.

c) (c) After C, F, and G have received A's broadcast.

d) (d) After E, H, and I have received A's broadcast.

Shaded nodes are new recipients. Arrows show possible reverse routes.

# Route Discovery (2)

| Source address | Request ID | Destination address | Source sequence # | Dest. sequence # | Hop count |
|---|---|---|---|---|---|

Format of a ROUTE REQUEST packet.

# Route Discovery (3)

| Source address | Destination address | Destination sequence # | Hop count | Lifetime |
|---|---|---|---|---|

Format of a ROUTE REPLY packet.

# Route Maintenance

| Dest. | Next hop | Distance | Active neighbors | Other fields |
|---|---|---|---|---|
| A | A | 1 | F, G | |
| B | B | 1 | F, G | |
| C | B | 2 | F | |
| E | G | 2 | | |
| F | F | 1 | A, B | |
| G | G | 1 | A, B | |
| H | F | 2 | A, B | |
| I | G | 2 | A, B | |

(a)

(b)

(a) D's routing table before G goes down.
(b) The graph after G has gone down.

# Node Lookup in Peer-to-Peer Networks

(a)

| Start | IP addr of successor |
|---|---|
| 2 | 4 |
| 3 | 4 |
| 5 | 7 |
| 9 | 12 |
| 17 | 20 |

Node 1's finger table

| Start | IP addr of successor |
|---|---|
| 5 | 7 |
| 6 | 7 |
| 8 | 12 |
| 12 | 12 |
| 20 | 20 |

Node 4's finger table

| Start | IP addr of successor |
|---|---|
| 13 | 15 |
| 14 | 15 |
| 16 | 20 |
| 20 | 20 |
| 28 | 1 |

Node 12's finger table

(b)

(a) A set of 32 node identifiers arranged in a circle. The shaded ones correspond to actual machines. The arcs show the fingers from nodes 1, 4, and 12. The labels on the arcs are the table indices.

(b) Examples of the finger tables.

# Congestion Control Algorithms

- General Principles of Congestion Control
- Congestion Prevention Policies
- Congestion Control in Virtual-Circuit Subnets
- Congestion Control in Datagram Subnets
- Load Shedding
- Jitter Control

# Congestion

When too much traffic is offered, congestion sets in and performance degrades sharply.

# General Principles of Congestion Control

1. Monitor the system .
   - detect when and where congestion occurs.
2. Pass information to where action can be taken.
3. Adjust system operation to correct the problem.

# Congestion Prevention Policies

| Layer | Policies |
|---|---|
| Transport | • Retransmission policy<br>• Out-of-order caching policy<br>• Acknowledgement policy<br>• Flow control policy<br>• Timeout determination |
| Network | • Virtual circuits versus datagram inside the subnet<br>• Packet queueing and service policy<br>• Packet discard policy<br>• Routing algorithm<br>• Packet lifetime management |
| Data link | • Retransmission policy<br>• Out-of-order caching policy<br>• Acknowledgement policy<br>• Flow control policy |

Policies that affect congestion.

# Congestion Control in Virtual-Circuit Subnets

(a) A congested subnet. (b) A redrawn subnet, eliminates congestion and a virtual circuit from A to B.

# Hop-by-Hop Choke Packets

(a) A choke packet that affects only the source.

(b) A choke packet that affects each hop it passes through.

# Start HERE

# Jitter Control

(a) High jitter. (b) Low jitter.

# Quality of Service

- Requirements
- Techniques for Achieving Good Quality of Service
- Integrated Services
- Differentiated Services
- Label Switching and MPLS

# Requirements

| Application | Reliability | Delay | Jitter | Bandwidth |
|---|---|---|---|---|
| E-mail | High | Low | Low | Low |
| File transfer | High | Low | Low | Medium |
| Web access | High | Medium | Low | Medium |
| Remote login | High | Medium | Medium | Low |
| Audio on demand | Low | Low | High | Medium |
| Video on demand | Low | Low | High | High |
| Telephony | Low | High | High | Low |
| Videoconferencing | Low | High | High | High |

How stringent the quality-of-service requirements are.

# Buffering

Smoothing the output stream by buffering packets.

# The Leaky Bucket Algorithm

(a) A leaky bucket with water. (b) a leaky bucket with packets.

# The Leaky Bucket Algorithm

(a) Input to a leaky bucket. (b) Output from a leaky bucket. Output from a token bucket with capacities of (c) 250 KB, (d) 500 KB, (e) 750 KB, (f) Output from a 500KB token bucket feeding a 10-MB/sec leaky bucket.

# The Token Bucket Algorithm

(a) Before. (b) After.

# Admission Control

| Parameter | Unit |
|---|---|
| Token bucket rate | Bytes/sec |
| Token bucket size | Bytes |
| Peak data rate | Bytes/sec |
| Minimum packet size | Bytes |
| Maximum packet size | Bytes |

An example of flow specification.

# Packet Scheduling

(a)

| Packet | Finishing time |
|---|---|
| C | 8 |
| B | 16 |
| D | 17 |
| E | 18 |
| A | 20 |

(b)

(a) A router with five packets queued for line O.
(b) Finishing times for the five packets.

# RSVP-The ReSerVation Protocol

(a) A network, (b) The multicast spanning tree for host 1.
(c) The multicast spanning tree for host 2.

# RSVP-The ReSerVation Protocol (2)

(a) Host 3 requests a channel to host 1. (b) Host 3 then requests a second channel, to host 2. (c) Host 5 requests a channel to host 1.

# Expedited Forwarding

Expedited packets experience a traffic-free network.

# Assured Forwarding

A possible implementation of the data flow for assured forwarding.

# Label Switching and MPLS

Transmitting a TCP segment using IP, MPLS, and PPP.

# Internetworking

- How Networks Differ
- How Networks Can Be Connected
- Concatenated Virtual Circuits
- Connectionless Internetworking
- Tunneling
- Internetwork Routing
- Fragmentation

# Connecting Networks

A collection of interconnected networks.

# How Networks Differ

| Item | Some Possibilities |
|---|---|
| Service offered | Connection oriented versus connectionless |
| Protocols | IP, IPX, SNA, ATM, MPLS, AppleTalk, etc. |
| Addressing | Flat (802) versus hierarchical (IP) |
| Multicasting | Present or absent (also broadcasting) |
| Packet size | Every network has its own maximum |
| Quality of service | Present or absent; many different kinds |
| Error handling | Reliable, ordered, and unordered delivery |
| Flow control | Sliding window, rate control, other, or none |
| Congestion control | Leaky bucket, token bucket, RED, choke packets, etc. |
| Security | Privacy rules, encryption, etc. |
| Parameters | Different timeouts, flow specifications, etc. |
| Accounting | By connect time, by packet, by byte, or not at all |

Some of the many ways networks can differ.

# How Networks Can Be Connected

(a) Two Ethernets connected by a switch.
(b) Two Ethernets connected by routers.

# Concatenated Virtual Circuits

Internetworking using concatenated virtual circuits.

# Connectionless Internetworking

A connectionless internet.

# Tunneling

Tunneling a packet from Paris to London.

# Tunneling (2)

Tunneling a car from France to England.

# Internetwork Routing

(a) An internetwork. (b) A graph of the internetwork.

# Fragmentation

(a) Transparent fragmentation. (b) Nontransparent fragmentation.

# Fragmentation (2)

Fragmentation when the elementary data size is 1 byte.

(a) Original packet, containing 10 data bytes.

(b) Fragments after passing through a network with maximum packet size of 8 payload bytes plus header.

(c) Fragments after passing through a size 5 gateway.

# The Network Layer in the Internet

- The IP Protocol
- IP Addresses
- Internet Control Protocols
- OSPF – The Interior Gateway Routing Protocol
- BGP – The Exterior Gateway Routing Protocol
- Internet Multicasting
- Mobile IP
- IPv6

# Design Principles for Internet

1. Make sure it works.
2. Keep it simple.
3. Make clear choices.
4. Exploit modularity.
5. Expect heterogeneity.
6. Avoid static options and parameters.
7. Look for a good design; it need not be perfect.
8. Be strict when sending and tolerant when receiving.
9. Think about scalability.
10. Consider performance and cost.

# Collection of Subnetworks

The Internet is an interconnected collection of many networks.

# The IP Protocol

The IPv4 (Internet Protocol) header.

# The IP Protocol (2)

| Option | Description |
|---|---|
| Security | Specifies how secret the datagram is |
| Strict source routing | Gives the complete path to be followed |
| Loose source routing | Gives a list of routers not to be missed |
| Record route | Makes each router append its IP address |
| Timestamp | Makes each router append its address and timestamp |

Some of the IP options.

# IP Addresses

IP address formats.

# IP Addresses (2)

Special IP addresses.

# Subnets

A campus network consisting of LANs for various departments.

# Subnets (2)

A class B network subnetted into 64 subnets.

# CDR – Classless InterDomain Routing

| University | First address | Last address | How many | Written as |
|---|---|---|---|---|
| Cambridge | 194.24.0.0 | 194.24.7.255 | 2048 | 194.24.0.0/21 |
| Edinburgh | 194.24.8.0 | 194.24.11.255 | 1024 | 194.24.8.0/22 |
| (Available) | 194.24.12.0 | 194.24.15.255 | 1024 | 194.24.12/22 |
| Oxford | 194.24.16.0 | 194.24.31.255 | 4096 | 194.24.16.0/20 |

A set of IP address assignments.

# NAT – Network Address Translation

Placement and operation of a NAT box.

# Internet Control Message Protocol

| Message type | Description |
|---|---|
| Destination unreachable | Packet could not be delivered |
| Time exceeded | Time to live field hit 0 |
| Parameter problem | Invalid header field |
| Source quench | Choke packet |
| Redirect | Teach a router about geography |
| Echo request | Ask a machine if it is alive |
| Echo reply | Yes, I am alive |
| Timestamp request | Same as Echo request, but with timestamp |
| Timestamp reply | Same as Echo reply, but with timestamp |

The principal ICMP message types.

# ARP– The Address Resolution Protocol

Three interconnected /24 networks: two Ethernets and an FDDI ring.

# Dynamic Host Configuration Protocol

Operation of DHCP.

# OSPF – The Interior Gateway Routing Protocol

(a) An autonomous system. (b) A graph representation of (a).

# OSPF (2)

The relation between ASes, backbones, and areas in OSPF.

# OSPF (3)

| Message type | Description |
|---|---|
| Hello | Used to discover who the neighbors are |
| Link state update | Provides the sender's costs to its neighbors |
| Link state ack | Acknowledges link state update |
| Database description | Announces which updates the sender has |
| Link state request | Requests information from the partner |

The five types of OSPF messeges.

# BGP – The Exterior Gateway Routing

(a)

Information F receives
from its neighbors about D

From B: "I use BCD"
From G: "I use GCD"
From I: "I use IFGCD"
From E: "I use EFGCD"

(b)

(a) A set of BGP routers. (b) Information sent to F.

# The Main IPv6 Header

The IPv6 fixed header (required).

# Extension Headers

| Extension header | Description |
|---|---|
| Hop-by-hop options | Miscellaneous information for routers |
| Destination options | Additional information for the destination |
| Routing | Loose list of routers to visit |
| Fragmentation | Management of datagram fragments |
| Authentication | Verification of the sender's identity |
| Encrypted security payload | Information about the encrypted contents |

IPv6 extension headers.

# Extension Headers (2)

| | | | |
|---|---|---|---|
| Next header | 0 | 194 | 4 |
| Jumbo payload length | | | |

The hop-by-hop extension header for large datagrams (jumbograms).

# Extension Headers (3)

The extension header for routing.