Network Layer

Chapter 4

Network Layer
Chapter 4: network layer

Chapter goals:

- Understand principles behind network layer services:
  - Broadcast, multicast
  - Routing (path selection)
  - How a router works
  - Forwarding versus routing
  - Network layer service models

- Instantiation, implementation in the Internet
Network Layer 4-3

Chapter 4: Outline

4.1 Introduction
4.2 Virtual Circuit and Datagram Networks
4.3 What’s Inside a Router
4.4 IP: Internet Protocol
4.5 Routing Algorithms
4.6 Routing in the Internet
4.7 Broadcast and Multicast

Routing

IPv6
ICMP
IPv4 Addressing
Datagram Format

- BGP
- OSPF
- RIP
- Link State
- Distance Vector
- Hierarchical Routing
Network Layer 4-4

Router examines header fields in all IP datagrams passing through it in every host and router.

Network Layer protocols segment to transport layer.

Receiving side delivers transport segments into IP datagrams.

Sending side encapsulates sending to receiving host.

Routes packets from sending to receiving host.
Two key network-layer functions

- **Forwarding**: Process of moving packets from router's input to appropriate output
- **Routing**: Process of determining appropriate output from router's input to packets from source to destination

Routing algorithms:
- Route packets from source to destination
- Determine route
- Appropriate output
- Move packets

Network Layer

4-5
Interplay between routing and forwarding

Routing algorithm determines end-to-end path through network
Local forwarding table determines local forwarding at this router
Packet's header value in arriving packet
Routing algorithm

<table>
<thead>
<tr>
<th>Header Value</th>
<th>Output Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>1001</td>
<td>1</td>
</tr>
<tr>
<td>1110</td>
<td>2</td>
</tr>
<tr>
<td>1011</td>
<td>2</td>
</tr>
<tr>
<td>0110</td>
<td>3</td>
</tr>
<tr>
<td>1000</td>
<td></td>
</tr>
</tbody>
</table>
Connection setup: an important function in some network architectures:

- ATM, frame relay, X.25 (but not Internet)
- Connection-oriented service:
  - Network vs. transport layer: before datagrams flow, two end hosts and on-path routers establish virtual connection.

Transport vs. network layer connection service:

- Between two processes
Q: What service model for "channel" transporting datagrams from sender to receiver?

Network service model

- packet switching
- bounded variance in inter-packet spacing
- guaranteed delivery
- guaranteed delivery with less than 40 msec delay

Example services for a flow of datagrams:
- in-order delivery
- guaranteed minimum flow bandwidth
- bounded variance in inter-packet spacing

Example services for individual datagrams:
- guaranteed delivery
- guaranteed delivery with less than 40 msec delay
Network layer service models:

<table>
<thead>
<tr>
<th>Bandwidth (rate)</th>
<th>Loss</th>
<th>Order</th>
<th>Timing</th>
<th>Feedback</th>
<th>Congestion</th>
<th>Guarantees</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimum</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>must</td>
</tr>
<tr>
<td>guaranteed rate</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>must</td>
</tr>
<tr>
<td>guaranteed rate</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>must</td>
</tr>
<tr>
<td>constant</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>must</td>
</tr>
<tr>
<td>none</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>must</td>
</tr>
<tr>
<td>Internet best effort</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>must</td>
</tr>
<tr>
<td>ATM</td>
<td>none</td>
<td>none</td>
<td>none</td>
<td>none</td>
<td>none</td>
<td>must</td>
</tr>
<tr>
<td>ATM ABR</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>must</td>
</tr>
<tr>
<td>ATM VBR</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>must</td>
</tr>
<tr>
<td>ATM CBR</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>must</td>
</tr>
</tbody>
</table>

Network layer service models:
Chapter 4: Outline

4.1 Introduction

4.2 Virtual Circuit vs.

4.3 What's inside a router

datagram networks

4.4 IP: Internet Protocol

IPv4 addressing

IPv4 datagram format

IPv6

ICMP

RIP

OSPF

BGP

Routing in the Internet

Hierarchical routing

Distance vector

Link state

Routing algorithms

4.5 Routing algorithms

4.6 Routing in the Internet

4.7 Broadcast and Multicast

Network Layer 4-10
Network Layer

Implementation: primarily in network core

Service: host-to-host

Connectionless transport-layer services, but:

- analogous to TCP/UDP
- LCAP

Layer connection service
- virtual-circuit network service
- datagram network service

Connection, connection-less service
Virtual circuits (e.g., ATM)

Network layer 4-12

Virtual circuits (e.g., ATM)

allocated to VC (dedicated resources = predictable service)

Link, router resources (bandwidth, buffers) may be

Each passing connection

Every router on source-dest path maintains “state” for

call setup, teardown, for each call before data can flow

Network actions along source-to-dest path

Performance-wise

Source-to-dest path behaves like telephone circuit
Virtual circuit implementation

VC consists of:

1. path from source to destination
2. VC numbers, one number for each link along path
3. entries in forwarding tables in routers along path
4. VC number typically changed on each link
5. new VC number comes from forwarding table

Packet belonging to VC carries VC number (rather than dest address)
VC routers maintain connection state information!

<table>
<thead>
<tr>
<th>VC number</th>
<th>Incoming interface</th>
<th>Outgoing VC #</th>
<th>Outgoing interface</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>87</td>
<td>...</td>
<td>3</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>17</td>
<td>...</td>
<td>2</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>18</td>
<td>...</td>
<td>1</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>22</td>
<td>...</td>
<td>3</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>97</td>
<td>...</td>
<td>7</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>63</td>
<td>...</td>
<td>2</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>12</td>
<td>...</td>
<td>1</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Virtual circuit forwarding table in northwest router:
Network Layer

1. initiate call
2. incoming call
3. accept call
4. call connected
5. data flow begins
6. receive data

How does this compare to datagram networks like the Internet?

- Virtual circuit signaling
  - Virtual circuits are not used in today’s Internet
  - Used in ATM, frame-relay, X.25
  - Used to set up, maintain, and tear down VC

Virtual Circuit Signaling
Datagram networks (e.g., Internet)

- no call setup at network layer
- no network-level concept of "connection"
- no state about end-to-end connections
- routers: no state about datagrams
- packets forwarded using destination host address
Network Layer

Datagram forwarding table

<table>
<thead>
<tr>
<th>dest address</th>
<th>output link</th>
</tr>
</thead>
<tbody>
<tr>
<td>address-range 1</td>
<td>1</td>
</tr>
<tr>
<td>address-range 2</td>
<td>2</td>
</tr>
<tr>
<td>address-range 3</td>
<td>2</td>
</tr>
<tr>
<td>address-range 4</td>
<td>3</td>
</tr>
</tbody>
</table>

4 Billion IP addresses, so rather than list individual destination address, list range of addresses (aggregate table entries).

Routing algorithm

arriving packet's header IP destination address in

(DataGram forwarding table)
Q: but what happens if ranges don’t divide up so nicely?

<table>
<thead>
<tr>
<th>Link Interface</th>
<th>Destination Address Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111</td>
</tr>
<tr>
<td>2</td>
<td>11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111</td>
</tr>
<tr>
<td>1</td>
<td>11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111</td>
</tr>
<tr>
<td>0</td>
<td>11001000 00010111 00011111 11111111 through 11001000 00010111 00011111 11111111</td>
</tr>
</tbody>
</table>

**Datagram Forwarding Table**
# Longest Prefix Matching

When looking for forwarding table entry for given destination address, use **longest address prefix that matches destination address**, use `longest prefix matching` that matches destination address, use `longest prefix matching` that matches destination address.

<table>
<thead>
<tr>
<th>Link Interface</th>
<th>Destination Address Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>11001000 00010111 00011000 10101010</td>
</tr>
<tr>
<td>2</td>
<td>11001000 00010111 00011111 10001111</td>
</tr>
<tr>
<td>1</td>
<td>11001000 00010111 00011111 00011111</td>
</tr>
<tr>
<td>0</td>
<td>11001000 00010111 00011111 00011110</td>
</tr>
</tbody>
</table>

Note: `longest prefix matching`
Datagram or VC network: Why?

Network Layer

4-20

Datagram or VC network: why?

Internet (datagram)

Internet (datagram)

ATM (VC)

ATM (VC)

complexity at edge

simple inside network

error recovery

can adapt, perform control,

smart end systems

uniform service difficult

different characteristics

many link types

timing requirement

elastic service, no strict

computers

data exchange amount

human conversation:

evolved from telephony

dumb end systems

need for guaranteed service

strict timing, reliability
Chapter 4: outline

4.1 Introduction
4.2 Virtual Circuit and Datagram Networks
4.3 What’s Inside a Router
4.4 IPv4: Internet Protocol
   - IPv4 Addressing
   - Datagram Format
4.5 Routing Algorithms
   - Link State
   - Distance Vector
   - Hierarchical Routing
4.6 Routing in the Internet
4.7 Broadcast and Multicast
   - RIP
   - OSPF
   - BGP

Network Layer 4.21
Two key router functions:

- Running routing protocols (RIP, OSPF, BGP)
- Forwarding datagrams from incoming to outgoing link
Network Layer

4-23

Input port functions

line termination

link layer protocol (receive)

lookup, forwarding, queueing

switch fabric

physical layer:

bit-level reception

data link layer:

e.g., Ethernet

see chapter 5

decentralized switching:

given datagram dest, lookup output port using forwarding table in input port memory ("match plus action")
goal: complete input port processing at line speed, forwarding rate into switch fabricqueuing: if datagrams arrive faster than forwarding rate into switch fabric

queuing: if datagrams arrive faster than forwarding rate into switch fabric
Network Layer: 4-24

Switching fabrics:

- **Crossbar**: Transfer packet from input to correct output buffer.
- **Bus**: Switching rate: rate at which packets can be transferred from inputs to outputs.
- **Memory**: Often measured as multiple of input/output line rate.

Three types of switching fabrics:

- N inputs: Switching rate N times line rate desirable.
- Transfer rate: Rate at which packets can be transferred from input to correct output buffer.

Switching fabrics
Switching via memory

First generation routers:

- Packet copied to system’s memory
- Packet copied to CPU
- Traditional computers with switching under direct control
- Speed limited by memory bandwidth (2 bus crossings per datagram)
Switching via a bus

Switching speed limited by bus bandwidth

Datagram transferred from input port memory to output port memory

Bus contention

32 Gbps bus, Cisco 5600: sufficient speed for access/enterprise routers

Bus

Switching via interconnection network

- Cisco 12000: switches 60 Gbps
- Switched cells through the fabric
- Datagram into fixed length cells
- Advanced design: fragmenting
- Multiprocessor
- Developed to connect processors in
- Interconnection nets initially
- Crossbar, banyan networks, other
- Overcome bus bandwidth limitations

Initial interconnection nets initially
Network Layer

4-28

Output ports

for transmission scheduling policy chooses among queued datagrams

fabric faster than the transmission rate buffering required when datagrams arrive from

for transmission scheduling policy chooses among queued datagrams

fabric faster than the transmission rate buffering required when datagrams arrive from

Output ports

Line termination

Layer (send) protocol

Link buffer datagram queueing

Switch fabric
buffering when arrival rate via switch exceeds output line speed!

queueing (delay) and loss due to output port buffer overflow!

one packet time later

from input to output

at t' packets move

Output port queuing
How much buffering?

RFC 3439 rule of thumb: average buffering equal to typical RTT times link capacity C.

Recent recommendation with N flows, buffering reduced by \( \frac{1}{\sqrt{N}} \) suffices:

\[
\sqrt{N} \cdot \frac{C}{RTT}
\]

E.g., C=10Gbps, RTT=250ms \( \Rightarrow \) 2.5 Gbit buffer

To "typical" RTT times link capacity C.
Network Layer 4-31

**Input port queuing**

- Fabric slower than input ports combined
- Queueing may occur at input queues
- Delay and loss due to input buffer overflow
- Head-of-the-Line (HOL) blocking
  - Queued datagram at front prevents others from moving forward
- Lower red datagram can be transferred only if one red datagram can be
- Output port contention:
  - Only one red packet can be transferred
  - Lower red packet is blocked

```
Input port queuing
```
Q1: Connection/-less service

Which of the following is connection-oriented?
Pick one.

A. TCP, datagram network
B. UDP, virtual circuit network
C. TCP, virtual circuit network
D. UDP, datagram network
E. TCP, virtual circuit network, datagram network

Which of the following is connection-less service?
Q2: Connection state

Which of the following relies on connection state in routers in the network? Pick one.

A. TCP
B. Internet
C. Virtual circuit network
D. UDP
E. A and C

Network Layer 4-33
Q3: Virtual circuit vs. datagram

How many entries do virtual circuit forwarding tables and datagram network forwarding tables respectively have?

A. 1, 2
B. 2, 4
C. 2, 2
D. 4, 2
E. 2, 1
Which of the following is true? Pick one.

A. A virtual circuit uses a different VC number for each link along a route.
B. A virtual circuit uses the same VC number for all packets in a connection.
C. A virtual circuit router uses the destination address (among other fields) in order to determine the outgoing interface.
D. A and C
Q5: Longest Prefix Matching

On which outgoing interface will a packet destined to 11011001 be forwarded?

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>111</td>
<td>C</td>
</tr>
<tr>
<td>11</td>
<td>B</td>
</tr>
<tr>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>Default</td>
<td>D</td>
</tr>
</tbody>
</table>
Q6: Router architecture

Where will queuing delay or loss occur if the link bandwidth (but not the switching fabric) is the bottleneck?

A. Input port
B. Output port
Q7: Router architecture

Where does head-of-the-line blocking occur?

A. Input port
B. Output port
Q8: Router architecture

The amount of buffering memory required in a router increases/decreases with the number of simultaneous TCP flows flowing through it.

A. Increases
B. Decreases

The amount of buffering memory required in a router increases/decreases with the number of simultaneous TCP flows flowing through it.
Chapter 4: outline

4.1 introduction
4.2 virtual circuit and datagram networks
4.3 what’s inside a router
donetcast and multicast
4.4 Internet Protocol
IPv4 addressing
IPv6
ICMP
datagram format
RIP
OSPF
BGP
link state
distance vector
hierarchical routing
Routing
Routing algorithms
4.6 Routing in the Internet
4.7 broadcast and multicast
IPv6
4.5 Routing algorithms
The Internet network layer

Host, router network layer functions:

- **Routing Protocols**:
  - RIP, OSPF, BGP
  - Path selection

- **IP Protocol**:
  - Datagram format
  - Addressing conventions
  - Packet handling conventions

- **ICMP Protocol**:
  - Error reporting
  - Router “signalling”

**Network Layer**

- **Forwarding**

**Link Layer**

- **IP Protocol**

**Physical Layer**
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol version</td>
<td>IP protocol version number</td>
</tr>
<tr>
<td>Header length</td>
<td>Upper layer protocol length</td>
</tr>
<tr>
<td>Total datagram length</td>
<td>Upper layer payload length</td>
</tr>
<tr>
<td>Type of data for</td>
<td>Fragmentation/reassembly</td>
</tr>
<tr>
<td>Data</td>
<td>IP datagram format</td>
</tr>
<tr>
<td>IP address source</td>
<td>32-bit IP address source</td>
</tr>
<tr>
<td>IP address destination</td>
<td>32-bit IP address destination</td>
</tr>
<tr>
<td>Time to live</td>
<td>Max number remaining hops</td>
</tr>
<tr>
<td>Fragment offset</td>
<td>Fragment offset in bytes</td>
</tr>
<tr>
<td>Fragment ID</td>
<td>16-bit fragment ID</td>
</tr>
<tr>
<td>Flag(s)</td>
<td>Flags related to fragmentation/reassembly</td>
</tr>
<tr>
<td>Options (if any)</td>
<td>Options field</td>
</tr>
<tr>
<td>Length</td>
<td>Total datagram length</td>
</tr>
<tr>
<td>Length</td>
<td>Header length</td>
</tr>
<tr>
<td>Length</td>
<td>Time to live</td>
</tr>
<tr>
<td>Length</td>
<td>Upper layer payload length</td>
</tr>
<tr>
<td>Length</td>
<td>IP datagram format</td>
</tr>
<tr>
<td>Length</td>
<td>Protocol version</td>
</tr>
</tbody>
</table>

Network Layer 4-42

IP datagram format
IP fragmentation, reassembly

- IP header bits used to identify, order fragments
- Final destination "reassembled" only at reassembly point

Different MTU's
- Different link types have different MTU's
- Physical link limits layer frame size (max. transfer unit) that network links have MTU

Network links have MTU

Network layer 4-43
Network Layer

4-44

ID = x
offset = 0
fragflag = 0
length = 4000

offset = 0
fragflag = 1
length = 1500

offset = 185
fragflag = 1
length = 1500

offset = 370
fragflag = 0
length = 1040

one large datagram becomes several smaller datagrams

Example:

MTU = 1500 bytes
4000 byte datagram

IP fragmentation, reassembly
4.1 Introduction

4.2 Virtual Circuit and datagram networks

4.3 What’s inside a router

4.4 IP: Internet Protocol
- IPv6
- ICMP
- IPv4 addressing
- Datagram format

4.5 Routing algorithms
- link state
- distance vector
- hierarchical routing

4.6 Routing in the Internet

4.7 Broadcast and multicast
- BGP
- OSPF
- RIP
Network Layer 4-46

**IP addressing: introduction**

- IP address: 32-bit identifier for interface on host or router
- Interface: connection between host/router and physical link
- Host or router's typically have many interfaces
- Routers typically have two interfaces (e.g., wired Ethernet, wireless 802.11)
- Host typically has one or two interfaces (e.g., wired Ethernet or wireless 802.11)

**IP addresses associated with each interface**

- 223.1.1.1
- 223.1.2.1
- 223.1.3.1
- 223.1.4.1
- 223.1.2.9
- 223.1.3.27
- 223.1.1.1 = 11011111 00000001 00000001 00000001
- 223.1.2.1 = 11011111 00000000 00000001 00000001
- 223.1.3.1 = 11011111 00000000 00000000 00000001
- 223.1.4.1 = 11011111 00000000 00000000 00000000

**Interface: identification for interface on host or router**

- IP address: 32-bit
IP addressing: introduction

Q: how are interfaces actually connected?
A: we'll learn about that in chapter 5, 6.

Q: how are interfaces connected?
A: wired Ethernet interfaces connected by Ethernet switches

Q: let's not worry about how one interface is connected to another (with no intervening router)

A: wireless WiFi interfaces connected by WiFi base station

For now:
network consisting of 3 subnets

- 223.1.1.2
- 223.1.3.1
- 223.1.4.1

subnet: high order bits
host part: low order bits

subnet part: specifies subnet
netmask specifies subnet

Interface with same subnet part of IP address
physically inter-reachable without intervening router

what is a subnet?

IP address: 223.1.1.1

Subnets
Network Layer 4.49

Subnet mask: /24

223.1.1.0/24
223.1.2.0/24
223.1.3.0/24

Subnet

Each isolated network creating islands of from its host or router, detach each interface recipe to determine subnets

Subnets
Network Layer

223.1.1.1
223.1.1.3
223.1.1.4
223.1.2.2
223.1.2.1
223.1.2.6
223.1.3.2
223.1.3.1
223.1.3.27
223.1.1.2
223.1.7.0
223.1.7.1
223.1.8.0
223.1.8.1
223.1.9.1
223.1.9.2

Subnets

how many?
Network Layer 4-51

IP addressing: CIDR

CIDR: Classless InterDomain Routing

200.23.16.0/23

subnet portion of address of arbitrary length

address format: a.b.c.d/x, where x is # bits in

subnet portion of address of arbitrary length

IP addressing: CIDR
Network Layer 4-52

**Q:** How does a host get IP address?

**IP addresses: how to get one?**

- Hard-coded by system administrator in a file
- "Plug-and-play" dynamically get address from a server
- **DHCP: Dynamic Host Configuration Protocol:**
  - UNIX: `/etc/rc.config`
  - IP Properties
  - Windows: Control-Panel>Network-Configuration>TCP/IP
DHCP: Dynamic Host Configuration Protocol

DHCP overview:
- Supports for mobile users to join network (more shortly)
- Allows reuse of addresses while connected
- Can renew its lease on address in use
- Server when it joins network
- Goal: allow host to dynamically obtain its IP address from network

DHCP: Dynamic Host Configuration Protocol
DHCP Client-Server Scenario

DHCP server

DHCP client

Network address in this network needs arriving DHCP

Network Layer 4-54

DHCP client-server scenario

223.1.1.0/24

223.1.2.0/24

223.1.3.0/24

223.1.1.1

223.1.1.3

223.1.1.4

223.1.2.9

223.1.2.2

223.1.3.2

223.1.3.1

223.1.1.2

223.1.3.27

223.1.1.1

223.1.2.1

223.1.4.9

223.1.3.27
DHCP server: 223.1.2.5

DHCP discover
src : 0.0.0.0, 68
dest.: 255.255.255.255, 67
yiaddr: 0.0.0.0
transaction ID: 654
lifetime: 3600 secs

DHCP offer
src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddr: 223.1.2.4
transaction ID: 655
lifetime: 3600 secs

DHCP request
src: 0.0.0.0, 68
dest:: 255.255.255.255, 67
yiaddr: 223.1.2.4
transaction ID: 655
lifetime: 3600 secs

DHCP ACK
src: 223.1.2.5, 67
dest: 255.255.255.255, 68
yiaddr: 223.1.2.4
transaction ID: 655
lifetime: 3600 secs

DHCP client-server scenario
DHCP can return more than just allocated IP addresses:
- Address of first-hop router for client
- Name and IP address of DNS server
- Network mask (indicating network versus host portion)

DHCP: more than IP addresses
A laptop needs to connect to the network. It uses DHCP to get an IP address and other configurations.

1. **DHCP** encapsulates the DHCP request in a UDP packet, which is then encapsulated in an IP packet and then encapsulated in an 802.11 Ethernet frame.
2. The Ethernet frame is broadcast (dest: FFFFFFFF) on the LAN.
3. The Ethernet frame is received at the router running the DHCP server.
4. The Ethernet frame is demultiplexed to the IP layer, and the IP packet is demultiplexed to the UDP layer, then to the DHCP layer.
5. The router sends a DHCP response to the laptop.
6. The DNS server address and the first-hop router address are provided by DHCP.
7. The laptop connects to the network using the configured IP address.
DHCP: Wireshark
output
Network Layer
DHCP: Wireshark
Home output (home LAN)
DHCP: Wireshark
Network Layer
DHCP: Wireshark
**IP addresses: how to get one?**

**Q:** how does network get subnet part of IP address?

**A:** gets portion of its provider ISP’s address space

<table>
<thead>
<tr>
<th>Organization</th>
<th>IPv4 Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11001000 00010111 00010000 00000000</td>
</tr>
<tr>
<td>1</td>
<td>11001000 00010111 00010000 00000000</td>
</tr>
<tr>
<td>2</td>
<td>11001000 00010111 00010000 00000000</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>7</td>
<td>11001000 00010111 00011110 00000000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Organization</th>
<th>IPv4 Block</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11001000 00010111 00010000 00000000</td>
</tr>
<tr>
<td>1</td>
<td>11001000 00010111 00010000 00000000</td>
</tr>
<tr>
<td>2</td>
<td>11001000 00010111 00010000 00000000</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>7</td>
<td>11001000 00010111 00011110 00000000</td>
</tr>
</tbody>
</table>

- **ISP’s block**
Hierarchical addressing allows efficient advertisement of routing information.
Hierarchical addressing: more specific routes

ISPs-R-Us has a more specific route to Organization 1

Send me anything with addresses beginning 200.23.18.0/23

Send me anything with addresses beginning 199.31.0.0/16

Send me anything with addresses beginning 200.23.31.0/23

Send me anything with addresses beginning 200.23.16.0/20

Send me anything with addresses beginning 200.23.30.0/23

Send me anything with addresses beginning 200.23.20.0/23

Send me anything with addresses beginning 200.23.18.0/23

Send me anything with addresses beginning 200.23.20.0/23

Send me anything with addresses beginning 200.23.30.0/23

Send me anything with addresses beginning 200.23.31.0/23

Send me anything with addresses beginning 200.23.16.0/20
IP addressing: the last word...

Q: How does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Numbers and Names http://www.icann.org/

- allocates addresses
- manages DNS
- assigns domain names, resolves disputes

...
Which of the following fields not part of either a TCP or UDP header?

A. Source port
B. Source IP address
C. Receive window
D. Length
E.Checksum

Which of the following fields not part of either a TCP or UDP header?
How many IP addresses belong to the subnet 128.119.254.0/25?
Q3 How many subnets are there in this network?
NAT: network address translation
Network Layer

Network Address Translation (NAT)

- 10.0.0.1
- 10.0.0.2
- 10.0.0.3
- 10.0.0.4

138.76.29.7

Local network (e.g., home network)

10.0.0/24

Rest of Internet

All datagrams leaving local network have the same single source NAT IP address: 138.76.29.7, different source port numbers.

Datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual).

NAT: network address translation
Network Layer

motivation:

visible by outside world (a security plus)

devices inside local net not explicitly addressable,

- devices in local network
  - can change ISP without changing addresses of
    - can change addresses of devices in local network
      - IP address for all devices
        - range of addresses not needed from ISP: just one
          - as outside world is concerned:

NAT: network address translation

- range of addresses not needed from ISP: just one IP address as far as
Network Layer

Implementation: NAT router must:

- outgoing datagrams: replace (source IP address, port #) of every outgoing datagram with corresponding (NAT IP address, new port #)

- incoming datagrams: replace (source IP address, new port #) to (NAT IP address, destination port #) translation pair stored in NAT table

- incoming datagrams: replace (source IP address, port #) in each incoming datagram with corresponding (NAT IP address, new port #) stored in NAT table

- outgoing datagrams: replace (source IP address, port #) of every outgoing datagram with corresponding (NAT IP address, destination port #)

NAT: network address translation
**Network Layer**

1: Host 10.0.0.1 sends datagram to 128.119.40.186, 80

2: NAT router changes datagram source addr from 10.0.0.1, 3345 to 128.119.40.186, 80

3: Reply arrives dest address: 138.76.29.7, 5001

4: NAT router changes datagram dest addr from 138.76.29.7, 5001 to 10.0.0.1, 3345

---

**NAT**: network address translation

---

**NAT Translation Table**

<table>
<thead>
<tr>
<th>WAN side addr</th>
<th>LAN side addr</th>
</tr>
</thead>
<tbody>
<tr>
<td>138.76.29.7, 5001</td>
<td>10.0.0.1, 3345</td>
</tr>
</tbody>
</table>

---

**10.0.0.1**

**128.119.40.186, 80**

---

**10.0.0.2**

---

**10.0.0.3**

---

**10.0.0.4**

---

**S: 10.0.0.1, 3345**

**D: 128.119.40.186, 80**

---

**S: 10.0.0.1, 3345**

**D: 128.119.40.186, 80**

---

**S: 128.119.40.186, 80**

**D: 10.0.0.1, 3345**

---

**S: 138.76.29.7, 5001**

**D: 128.119.40.186, 80**

---

**S: 128.119.40.186, 80**

**D: 10.0.0.1, 3345**

---

**S: 10.0.0.1, 3345**

**D: 128.119.40.186, 80**

---

**S: 128.119.40.186, 80**

**D: 10.0.0.1, 3345**

---

**S: 138.76.29.7, 5001**

**D: 128.119.40.186, 80**

---

**S: 128.119.40.186, 80**

**D: 10.0.0.1, 3345**

---

**S: 10.0.0.1, 3345**

**D: 128.119.40.186, 80**

---
Network Layer

- NAT: network address translation
  - 16-bit port-number field:
    - ~65K simultaneous connections with a single WAN-side address
    - Possible to allow ~65K connections to each WAN-side address
    - 64K simultaneous connections with a single WAN-side address

- NAT is controversial:
  - NAT possibility must be taken into account by app designers, e.g., P2P applications
  - NAT violates end-to-end argument

- Address shortage ought to be solved by IPv6
  - Protocols (e.g., UDP, TCP) must be taken into account by app designers

- NAT is controversial:
  - External destination [IP, port]

- Routers should only process up to layer 3
NAT traversal problem

A client wants to connect to a server with address 10.0.0.1. The server address 10.0.0.1 is local to the LAN (client cannot use it as a destination address), and there is only one externally visible NATed server address: 138.76.29.7.

Solution 1: Statically configure NAT to forward incoming connection requests at a given port to the server's local port. For example, configure NAT to always forward incoming port 2500 connections to the server's local port 25000.

From the diagram:
- The client's address is 10.0.0.4.
- The router/NAT is located at the address 138.76.29.7.
- The server's address is 10.0.0.1.

- Learn public IP address (with lease times)
- Add/remove port mappings
- Automate static NAT port map configuration

NAT traversal problem

10.0.0.1

NAT router

IGD
NAT traversal problem

Solution 3: Relaying (used in Skype)

1. Connection to relay initiated by NAT-ed host
2. Connection to relay initiated by client
3. Relaying established

Relay bridges packets between to connections
External client connects to relay
NAT-ed client establishes connection to relay

NAT traversal problem
Chapter 4: outline

Network Layer 4.1

4.1 Introduction

4.2 Virtual Circuit and Datagram Networks

4.3 What’s inside a router

4.4 IP: Internet Protocol

4.5 Routing Algorithms

4.6 Routing in the Internet

4.7 Broadcast and Multicast

IPv6

ICMP

IPv4 Addressing

Datagram Format

RIP

OSPF

BGP

Link State

Distance Vector

Hierarchical Routing

Routing

IPv4 datagram format

What's inside a router

Routing in the Internet

Broadcast and Multicast

Introduction
Network Layer

ICMP: internet control message protocol

- Used by hosts & routers to communicate network-level information
  - Error reporting: unreachable host, network, port, protocol
  - Echo request/reply (used by ping)

ICMP message: type, code

<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>echo reply (ping)</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>destination unreachable error</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>destination network unreachable</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>destination host unreachable</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>routing protocol unreachable</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>destination port unreachable</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>source quench (congestion control)</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>network unreachable</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>protocol unreachable</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>destination host unreachable</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>network unreachable</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>protocol unreachable</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>destination unreachable</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>echo request/reply (used by ping)</td>
</tr>
</tbody>
</table>

ICMP messages carried in IP datagrams

Level 3: network-layer "above" IP:
- Ping, Echo request/reply (used by hosts & routers)
- Source quench (congestion control)
- Destination unreachable errors
- Protocol unreachable errors
- Port unreachable errors
- Network unreachable errors
Trace route and ICMP

Source stops

message (type 3, code 3)
port unreachable
destination returns ICMP

UDP segment eventually

stopping criteria:

RTTs
source records, source records
arrives at destination host
arrives eventually

ICMP messages includes
ICMP messages (type 11, code 0)
and sends source ICMP

route discards datagrams

arrows to nth router:

When nth set of datagrams

unlike port number
second set has TTL = 2, etc.
first set has TTL = 1
UDP segments to dest
source sends series of

Network Layer 4.77

Traceroute
and ICMP
IPv6: motivation

- initial motivation: 32-bit address space soon to be completely allocated.
- additional motivation: header changes to facilitate QoS
- no fragmentation allowed
- fixed-length 40 byte header

IPv6 datagram format:

- header format helps speed processing/forwarding
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Address</td>
<td>Destination address</td>
<td>128 bits</td>
</tr>
<tr>
<td>Payload Len</td>
<td>Hop Limit</td>
<td></td>
</tr>
<tr>
<td>Next Header</td>
<td>Flow Label</td>
<td></td>
</tr>
<tr>
<td>Ver Pri</td>
<td>Flow priority: identity priority among datagrams in flow</td>
<td>8 bits</td>
</tr>
</tbody>
</table>

IPv6 datagram format

Network Layer

IPv6 datagram format
Other changes from IPv4

- Multicast group management functions
- Additional message types, e.g. "Packet Too Big"
- ICMPv6: new version of ICMP
  - Options: allowed, but outside of header, indicated
  - Options: allowed, but outside of header, indicated
  - Time at each hop
  - Checksum: removed entirely to reduce processing

Network Layer 4.80
Transition from IPv4 to IPv6

Transition from IPv4 to IPv6

Network Layer

IPv4 datagram

IPv6 datagram

IPv4 datagram among IPv4 routers

IPv6 datagram carried as payload in IPv4

tunneling: IPv6 datagram carried as payload in IPv4

IPv6 routers?

IPv4 routers?

how will network operate with mixed IPv4 and IPv6?

no "flag days"

not all routers can be upgraded simultaneously

![Diagram showing IPv4 and IPv6 datagrams and tunneling process.](image-url)
Network Layer 4-82

Tunneling

Physical view:

Logical view:

IPv4 tunnel connecting IPv6 routers
The two subnets 128.119.245.4/26 and 128.119.245.129/25 have overlapping IP addresses.

A. True
B. False

Q11P addressing: subnets
What transport protocol does DHCP use?

A. UDP
B. TCP
C. IP
D. HTTP
A host with a private IP address 192.168.0.2 opens a TCP socket on its local port 4567 and connects to a web server at 34.5.6.7. The NAT's public IP address is 22.33.44.55. Which of the following mapping entries could the NAT create as a result?

A. [22.33.44.55, 3333] → [192.168.0.2, 80]
B. [34.5.6.7, 80] → [22.33.44.55, 4567]
C. [192.168.0.2, 80] → [22.33.44.55, 3333]
D. [22.33.44.55, 3967] → [192.168.0.2, 4567]
A host with a private IP address 192.168.0.2 opens a TCP socket on its local port 4567 and connects to a web server at 34.5.6.7. The NAT's public IP address is 22.33.44.55. Suppose the NAT created the mapping [22.33.44.55, 3967] → [192.168.0.2, 4567] as a result.

What are the source and destination port numbers in the SYNACK response from the server?

A. 80, 3967
B. 4567, 80
C. 3967, 80
D. 3967, 4567

A4 IP addressing: NATs
Q5: IP addressing: NATs

Which of the following is false about NATs?

A. NATs are network layer devices but process both network and transport layer headers.

B. The number of possible simultaneous connections through a NAT is limited by the size of the port number space.

C. The number of possible simultaneous connections through a NAT is limited by the size of the private (LAN-side) IP address space.

D. Outgoing packets from a NAT originating from different LAN-side machines can not have the same srcIP, srcPort.
Which of the following is not true?

A. IPv6 increases the size of the IP address space from $2^{32}$ to $2^{128}$.

B. IPv6 removes checksums and fragmentation compared to IPv4.

C. IPv6 has fixed length headers.

D. IPv6 adds reliability at the network layer.

Which of the following is not true?
Interplay between routing, forwarding

Routing algorithm

Local forwarding table

Routing algorithm determines end-end path through network

Local forwarding table determines forwarding at this router

IP destination address in arriving packet's header

Routing algorithm determines end-end path through network

Local forwarding table determines forwarding at this router

Network Layer 4-91
Graph abstraction is useful in other network contexts, e.g.,

$p2p$, where $N$ is set of peers and $E$ is set of TCP connections.

Aside: $N$ is set of peers and $E$ is set of TCP connections.

Graph: $G = (N, E)$

$E = \{ (z', y') (z', w') (w', y') (w', x') (x', w') (x', y') (y', z') (z', u') (u', v') (v', y') (w', y') \}$

$N = \{ u, v, w, x, y, z \}$
Network Layer

Graph abstraction: costs

Routing algorithm: algorithm that finds the least cost path

key question: what is the least-cost path between u and z?

\[
\text{cost of path } (x_1, x_2, x_3, \ldots, x_p) = c(x_1, x_2) + c(x_2, x_3) + \cdots + c(x_{p-1}, x_p)
\]

Congestion

or inversely related to

inversely related to bandwidth,

cost could always be 1, or

\[
c(x, x') = \begin{cases} 5 & \text{cost of link } (x,x') \\ 1 & \text{e.g., } c(w', z) = 5 \end{cases}
\]
Routing algorithm classification

- "Distance vector" algorithms
  - Information exchanged among neighbors
  - Computation, exchange of link state information
  - Iterative process of costs to neighbors
  - Connected neighbors, links
  - Router knows physically-connected neighbors
  - Decentralized

- "Link state" algorithms
  - Link state, keep topology
  - Link cost information
  - All routers have complete information
  - Global

Q: Global or decentralized?
Chapter 4: outline

4.1 Introduction

4.2 Virtual circuit and datagram networks

4.3 What's inside a router

4.4 IP: Internet Protocol

   - IPv6
   - ICMP
   - IPv4 addressing
   - Datagram format

4.5 Routing algorithms

   - Link state
   - Distance vector
   - Hierarchical routing

4.6 Routing in the Internet

4.7 Broadcast and multicast

   - BGP
   - OSPF
   - RIP

Routing
A Link-State Routing Algorithm

Dijkstra’s algorithm

known
least cost path definitely
set of nodes whose
along path from source to v
predecessor node
of path from src to dest v
current value of cost

D(v): current value of cost from src to dest v
p(v): predecessor node along path from source to v
N': set of nodes whose costs known to all other nodes
c(v,y): link cost from node x to y; = ∞ if not direct neighbors
N: set of nodes whose costs known to all nodes
via “link state broadcast”

Interattribute: after k
iterations, know least cost path to k dest’s.
Gives node forwarding table to all other nodes
computes least cost paths from one node (‘source’) to all other nodes
costs known to all nodes network topology, link

notation:
notation:

Dijkstra's Algorithm

Initialization:

\[ N' = \{ u \} \]

for all nodes v

\[ D(v) = \begin{cases} c(u,v) & \text{if } v \text{ adjacent to } u \\ \infty & \text{otherwise} \end{cases} \]

Loop

until all nodes in N'

notation:

\[ c(x,y) \]: link cost from node x to y

\[ p(v) \]: predecessor node along path from source to v

\[ D(v) \]: current value of cost of path from source to v

\[ N' \]: set of nodes whose least cost path definitively known

\[ x \prec y \]: not direct neighbors

\[ \infty \]: if not direct neighbors

\[ D(v) = \begin{cases} \infty & \text{else} \\ \text{if } v \text{ adjacent to } u & \text{then} \end{cases} \]

\[ D(v) = c(u,v) \]

\[ N' = \{ n \} \] for all nodes v

network layer 4-97
**Algorithm:**

**Initialization:**

1. $N = \{u\}$
2. for all nodes $v$
3. if $v$ adjacent to $u$
4. then $D(v) = c(u,v)$
5. else $D(v) = \infty$

**Loop:**

6. while all nodes in $N$'
7. $p(v) = \infty$ if $D(v)$ changed above
8. shortest path cost to $w$ plus cost from $w$ to $v$
9. $D(v) = \min(D(v), D(w) + c(w,v))$
10. update $D(v)$ for all $v$ adjacent to $w$ and not in $N$'
11. add $w$ to $N$'
12. find $w$ not in $N$' such that $D(w)$ is a minimum
13. until all nodes in $N$'
14. notation:
15. $c(v,w)$: link cost from node $x$ to $y$.
16. $D(v)$: current value of cost from src to dest $v$
17. $p(v)$: predecessor node along path from src to dest $v$
18. $N$'
19. set of nodes whose least cost path definitively known

Notation:

- $N$': set of nodes whose least cost path from source to $v$
- $p(v)$: predecessor node along path from src to dest $v$
- $D(v)$: current value of cost of path from src to dest $v$
- $\infty$ if not direct neighbors
- $c(v,w)$: link cost from node $x$ to $y$.
Dijkstra’s algorithm: example

**Notes:**
- Construct shortest path tree by tracing predecessor nodes.
- Ties can exist (can be broken arbitrarily).

<table>
<thead>
<tr>
<th>Step</th>
<th>N</th>
<th>(z)</th>
<th>(d)</th>
<th>(x)</th>
<th>(w)</th>
<th>(v)</th>
<th>(y)</th>
<th>(d)</th>
<th>(z)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>7</td>
<td>n</td>
<td>3</td>
<td>w</td>
<td>6</td>
<td>w</td>
<td>2</td>
<td>4</td>
<td>y</td>
</tr>
<tr>
<td>2</td>
<td>11</td>
<td>11</td>
<td>11</td>
<td>10</td>
<td>10</td>
<td>14</td>
<td>14</td>
<td>12</td>
<td>x</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Network Layer 4-99
Dijkstra's algorithm: another example

Network Layer 4-100

Step

Dijkstra's algorithm: another example
Dijkstra’s algorithm: example (2)

Resulting shortest-path tree from u:

\[(x, n) \quad z\]
\[(x, n) \quad w\]
\[(x, n) \quad y\]
\[(x, n) \quad x\]
\[(\wedge, n) \quad \wedge\]

Resulting forwarding table in u:

\[
\begin{array}{c|c}
\text{Link} & \text{Destination} \\
\hline
\wedge & v \\
\hline
x & x \\
\wedge & \wedge \\
\end{array}
\]
Network Layer 4-102

Dijkstra's algorithm, discussion

algorithm complexity: \( n \text{ nodes} \)

each iteration: need to check all nodes, \( \mathcal{O}(n^2) \)

more efficient implementations possible: \( \mathcal{O}(n \log n) \)

oscillations possible:

- e.g., support link cost equals amount of carried traffic:

Initially

resulting in new costs

resulting in new costs

resulting in new costs

resulting in new costs

\[ \begin{array}{cccc}
A & D & C & B \\
0 & 0 & 2+e & 1+e \\
1+e & 0 & 2+e & 1 \\
2+e & 1 & 1+e & 0 \\
1+e & 2+e & 0 & 0 \\
\end{array} \]
Chapter 4: outline

4.1 Introduction

4.2 Virtual Circuit and Datagram Networks

4.3 What's Inside a Router

4.4 IP: Internet Protocol

4.5 Routing Algorithms

4.6 Routing in the Internet

- Hierarchical Routing
- Distance Vector
- Link State
- Link State

4.7 Broadcast and Multicast

- BGP
- OSPF
- RIP
- IPv6
- ICMP
- IPv4 addressing
- Datagram format

Network Layer 4-103
Distance vector algorithm

Bellman-Ford equation (dynamic programming)

\[
d^x(y) = \min \left\{ (x,v) + d^v(y) \right\} \text{ for all } y \text{ in } \mathcal{V}
\]

let

\[
d^x(y) = \min \text{ cost taken over all neighbors } v \text{ of } x
\]

\[
\text{cost to neighbor } v
\]

\[
\text{cost from neighbor } v \text{ to destination } y
\]
Bellman-Ford example

Bellman-Ford example

node achieving minimum is next hop in shortest path, used in forwarding table

d\text{u}(z) = \min\{ c(\text{u,v}) + d\text{v}(z), c(\text{u,x}) + d\text{x}(z), c(\text{u,w}) + d\text{w}(z) \} = \min\{2 + 5, 1 + 3, 5 + 3\} = 4

\text{clearly, } d\text{v}(z) = 5, d\text{x}(z) = 3, d\text{w}(z) = 3

B-F equation says:

\{ (z)^w d + (w,n)\}
\{ (z)^x d + (x,n)\}
\{ (z)^y d + (y,n) \} \text{ min } (z)^n d

\text{z} = (z)^w d, \text{z} = (z)^x d, \text{z} = (z)^y d

\text{node achieving minimum is next hop in shortest path, used in forwarding table}
Network Layer 4-106

Distance vector algorithm:

\[ d^x(y) = \text{estimate of least cost from } x \text{ to } y \]

node x:

\[ \hat{d}^x = [d^x(y): y \in N] \]

x maintains distance vector \( d^x \) of each neighbor y:

x maintains its neighbors distance vectors. For each neighbor y:

\[ c(x,y) \]

knows cost to each neighbor y:

Distance vector algorithm.
Network Layer

Key Idea: Distance Vector Algorithm

When node \( x \) receives its own DV estimate from a neighbor, it updates its own DV estimate using the B-F equation:

\[
D^x(y) \rightarrow \min_{v \in N} \{c(x,v) + D^v(y)\}
\]

for each node \( v \in N \).

Under minor, natural conditions, the estimate \( D^x(y) \) converges to the actual least cost distance \( d^x(y) \).
Network Layer

Distance vector algorithm

Each node:

Wait for (change in local link cost or msg from neighbor)

Recompute estimates

If DV to any dest has changed, notify neighbors

Notify neighbors

If DV changes

DV changes

Neighbors then notify their neighbors only when its neighbors notify theirs

Each node notifies

Distributed:

Neighbor DV update message from

Local link cost change

Caused by:

Each local iteration

Iterative, asynchronous
$$D_x(y) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(y)\}$$

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$$

$$D_y = \min\{c(x,y) + D_y(z), c(x,z) + D_z(y)\}$$

$$D_z = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$$
\[ D(x,z) = \min\{c(x,y) + D(y), c(x,z) + D(z)\} \]

\[ \begin{align*}
\text{node } x & : & D(x,x) &= 0 \\
& & D(x,y) &= 2 \\
& & D(x,z) &= 7 \\
\text{node } y & : & D(y,x) &= 3 \\
& & D(y,y) &= 0 \\
& & D(y,z) &= 2 \\
\text{node } z & : & D(z,x) &= 1 \\
& & D(z,y) &= 1 \\
& & D(z,z) &= 0 \\
\end{align*} \]
Distance Vector

4-111

Distance vector: link cost changes

\[ t^0: \text{Y detects link-cost change, updates its DV, informs its neighbors.} \]

\[ t^1: \text{Z receives update from Y, updates its table, computes new least cost to X, sends its neighbors its DV.} \]

\[ t^2: \text{Y receives Z's update, updates its distance table.} \]

\[ \text{Y does not change, so does not send a message to Z.} \]

Fast

travels

good

\[ \text{If DV changes, notify neighbors} \]

\[ \text{Distance vector updates routing info, recalculates} \]

\[ \text{Node detects local link cost change} \]

\[ \text{Link cost changes:} \]
Distance vector: Link cost changes

Network Layer

Distance vector: link cost changes

- node detects local link cost change
- bad news travels slow - "count to infinity"
- Count to infinity problem!
- 44 iterations before algorithm stabilizes: see text

Poisoned reverse:
- If Z routes through Y to get to X:
  - Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)

Will this completely solve count to infinity problem?
Comparison of LS and DV algorithms

**Message Complexity**

- **LS:**
  - $O(n^2)$ algorithm requires $O(nE)$ msgs
  - convergence time varies
  - may have oscillations
  - node can advertise incorrect link cost
  - router malfunction:
    - what happens if router malfunctions?
  - robustness:
    - what happens if

- **DV:**
  - $O(nE)$ msgs
  - convergence time varies
  - may have oscillations
  - DV node can advertise incorrect path cost
  - own table
  - each node computes only its
    - link cost
  - node can advertise incorrect

**Speed of Convergence**

- **LS:**
  - convergence time varies
  - only exchange between neighbors
  - $O(nE)$ links, $O(nE)$ msgs sent
  - with $n$ nodes, $E$ links

- **DV:**
  - convergence time varies
  - may be routing loops
  - DV: convergence time varies
  - DV node can advertise incorrect path cost
  - each node's table used by others
  - error propagate thru network

**Robustness**

- **LS:**
  - node can advertise incorrect link cost

- **DV:**
  - DV node can advertise incorrect path cost
  - each node’s table used by others

**Count-to-Infinity Problem**

- **LS:**
  - node can advertise incorrect link cost

- **DV:**
  - DV node can advertise incorrect path cost
  - each node’s table used by others
Intradomain routing

Which of link state and distance vector routing has more predictable convergence time?

1. Link state
2. Distance vector

0/1: Intradomain routing
Intradomain routing

In this link-state routing network running Dijkstra’s algorithm, the set \( N' \) (the set of nodes to which the least cost is definitively known) is initially \{u\}. After two iterations, which nodes belong to \( N' \)?

- A. \( u \)
- B. \( u \), \( x \)
- C. \( w \)
- D. \( u \), \( w \)
- E. \( u \), \( w \), \( x \)

In this link-state routing network running Dijkstra’s algorithm, the set \( N' \) (the set of nodes to which the least cost is definitively known) is initially \{u\}. After two iterations, which nodes belong to \( N' \)?

- A. \( u \)
- B. \( u \), \( x \)
- C. \( w \)
- D. \( u \), \( w \)
- E. \( u \), \( w \), \( x \)
In link state routing, the time for routing to re-converge after a link-cost change does NOT significantly depend on which one of the following?

A. Number of nodes
B. Diameter of the network
C. Whether the link cost increased or decreased
D. Whether routing is load-dependent or not

Q3 Intradomain routing
In this distance vector routing network, roughly how many iterations will the network take to re-converge after the event shown?

In this distance vector routing network, roughly 4 iterations.
Chapter 4: Outline

4.1 Introduction

4.2 Virtual Circuit and Datagram Networks

4.3 What’s Inside a Router

4.4 IP: Internet Protocol

4.5 Routing Algorithms

4.6 Routing in the Internet

4.7 Broadcast and Multicast

4.8 Link State

4.9 Distance Vector

4.10 Hierarchical Routing

Network Layer 4-118

Routing

IPv6

ICMP

IPv4 addressing

Datagram format

RIP

BGP

OSPF

IPv4 datagram format
Hierarchical routing

our routing study thus far - idealization
  ❖ all routers identical
  ❖ network “flat”
  … *not* true in practice

**scale:** with ~1B hosts, millions of routers
  ❖ can’t store the entire Internet graph!
  ❖ routing table exchange would swamp links!

**administrative autonomy**
  ❖ internet = network of networks
  ❖ each network admin may want to control routing in its own network
Network Layer

Aggregate routers into regions, autonomous systems (AS) routers in same AS run same intra-AS routing protocol. Routers in different AS can run different intra-AS protocols.

Gateway router: another AS has link to router in its own AS at “edge” of its own AS systems (AS) autonomous regions, aggregate routers into hierarchical routing.
Intra-AS & inter-AS sets entries for external dests

Inter-AS & intra-AS sets entries for internal dests

Intra-AS and Inter-AS routing algorithm
configured by both Intra-AS and Inter-AS routing algorithm

Forwarding table

Interconnected ASes
Network Layer

Inter-AS tasks

1. Suppose router in AS1 receives datagram destined outside of AS1:
   "router should forward packet to gateway router, but which one?"

2. AS1 must learn which destinations are reachable through AS2, which through AS3.

Propagate this reachability info to all routers in AS1: job of inter-AS routing!

AS1
AS2
AS3

other networks
other networks
other networks

Inter-AS tasks
Example: setting forwarding table in router 1d

Suppose AS1 learns (via inter-AS protocol) that subnet x reachable via AS3 (gateway 1c), but not via AS2.

- router 1d determines from intra-AS routing info that its interface I is on the least cost path to 1c.
- installs forwarding table entry \((x, I)\)

Inter-AS protocol propagates reachability info to all internal routers.
Example: choosing among multiple ASes
Example: choosing among multiple ASes

- Now suppose AS1 learns from inter-AS protocol that subnet x is reachable from AS3 and from AS2.
- This is also job of inter-AS routing protocol to determine towards which gateway it should forward packets for dest x.
- To configure forwarding table, router 1d must determine which gateway it should send packets towards.
- Hot potato routing: send packet towards closest of two routers.

- Use routing info from intra-AS protocol to determine costs of least-cost paths to each of the gateways.
- Hot potato routing: choose the gateway that has the smallest least cost.
- Determine from forwarding table the interface I that leads to least-cost gateway.
- Enter (x, I) in forwarding table.
Chapter 4: Outline

4.1 Introduction

4.2 Virtual Circuit and Datagram Networks

4.3 What’s Inside a Router

4.4 IP: Internet Protocol

IPv6

ICMP

IPv4 Addressing

Datagram Format

4.5 Routing Algorithms

RIP

OSPF

Distance Vector

BGP

4.6 Routing in the Internet

Hierarchical Routing

Link State
Intra-AS Routing

also known as interior gateway protocols (IGP)

most common intra-AS routing protocols:

- RIP: Routing Information Protocol
- OSPF: Open Shortest Path First
- IGRP: Interior Gateway Routing Protocol

also known as interior gateway protocols (IGP)
Network Layer 4-128

RIP (Routing Information Protocol)

Distance vector algorithm
- Distance metric: # hops (max = 15 hops), each link has cost 1
- Each advertisement can list up to 25 destination IP subnets
- DVs exchanged with neighbors every 30 sec (advertisement)
- Included in BSD-UNIX distribution in 1982

From router A to destination subnets:

<table>
<thead>
<tr>
<th>Subnet</th>
<th>Hops</th>
</tr>
</thead>
<tbody>
<tr>
<td>u</td>
<td>1</td>
</tr>
<tr>
<td>v</td>
<td>2</td>
</tr>
<tr>
<td>w</td>
<td>2</td>
</tr>
<tr>
<td>x</td>
<td>3</td>
</tr>
<tr>
<td>y</td>
<td>3</td>
</tr>
<tr>
<td>z</td>
<td>2</td>
</tr>
</tbody>
</table>

Diagram:

```
    y
   /   
  x     z
 /     /  
A     B   D
    /     / 
   w     v 
    /     /
   n     u 
```

Each advertisement can list up to 25 destination IP subnets.
### Network Layer 4-129

**Routing Table in Router D**

<table>
<thead>
<tr>
<th>Destination Subnet</th>
<th>Next Hop Router</th>
<th># Hops to Dest</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>z</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>W</td>
<td>-</td>
<td>1</td>
</tr>
</tbody>
</table>

**A-to-D Advertisement**

RIP: example
RIP: Link Failure, Recovery

Distance = 16 hops

Poison reverse used to prevent ping-pong loops (infinite)

Link failure info quickly (?) propagates to entire network

Changed

Neighbors in turn send out new advertisements (if tables

New advertisements sent to neighbors

Routes via neighbor invalidated

Link declared dead

If no advertisement heard after 180 sec --> neighbor

Network Layer 4-130
Network Layer 4-131

- RIP table processing

RIP routing tables managed by application-level process called route-d (daemon) advertised as periodic UDP packets.

- RIP routing tables managed by application-level

RIP table processing
OSPF (Open Shortest Path First)

- “open”: publicly available
- uses link state algorithm
  - LS packet dissemination
  - full topology map at each node
  - route computation using Dijkstra’s algorithm
- OSPF advertisement carries one entry per neighbor
- advertisements flooded to *entire* AS
  - carried in OSPF messages directly over IP (rather than TCP or UDP)
- *IS-IS routing* protocol: nearly identical to OSPF
Network Layer 4-133

Hierarchical OSPF in large domains.

Multicast OSPF (MOSPF) uses same topology data base as OSPF

- Integrated unicast and multicast support:
  - Best effort TOS: high for real time; low for best effort TOS (e.g., satellite link cost set “low” for different TOS)
  - For each link, multiple cost metrics for same-cost paths allowed (only one in RIP)

- Security: all OSPF messages authenticated (to prevent malicious intrusion)

OSPF “advanced” features (not in RIP)
Hierarchical OSPF
Network Layer

Two-level hierarchy: local area, backbone.

"Link-state advertisements only in area"

Each node has detailed area topology; only knows

Boundary routers: connect to other AS's.

Area border routers: summarize distances to nets in own area, advertise to other Area Border Routers.

Backbone routers: run OSPF limited to backbone.

Boundary routers: run OSPF limited to backbone.

Hierarchical OSPF
Internet: „I am here“
allows subnet to advertise its existence to rest of
reachability information and policy.
determine “good” routes to other networks based on
internal routers.
BGP: propagate reachability information to all AS-
neighboring ASs.
EBGP: obtain subnet reachability information from
BGP provides each AS a means to:
„glue that holds the Internet together“
inter-domain routing protocol
BCP (Border Gateway Protocol): the de facto

Network Layer 4-136

BGP (Border Gateway Protocol): the de facto
BGP basics

AS3 can aggregate prefixes in its advertisement.
AS3 promises it will forward datagrams towards that prefix.
When AS3 advertises a prefix to AS1:

- Advertises paths to destination network prefixes ("path vector protocol")
- Exchange over semi-permanent TCP connections
- Advertises prefix to destination network prefixes

BGP session: two BGP routers ("peers") exchange BGP messages:
BGP basics: distributing path information
Path attributes and BGP routes

ever advertise routes to AS y.
  e.g., select cheaper route; or never route through AS x; or
export policy to re-advertise route
advertisements uses import policy to select/reject route and
and
policy-based

Gateway router receiving route

(x) multiple links may exist from self to next-hop-AS

NEXT-HOP: indicates specific internal-AS router to next-

AS-PATH: contains ASs through which prefix

two important attributes:

prefix + attributes = "route"
advertised prefix includes BGP attributes

-
Network Layer

4-140

BGP route selection (import policy)

1. Destination AS; selects route based on:
   • Router may learn about more than 1 route to

2. Shortest AS-PATH

3. Closest NEXT-HOP router: hot potato routing

4. Additional criteria

Local preference value attribute: policy decision
Q: Which of the above routes are permitted by “valley-free” export policy?

- Examples (arrows indicate flow or customer/provider relationship, else peering):
  - Never advertise peer or provider routes to another
  - Export policies commonly use “valley-free” routing

BGP re-announce (export policy)
Network Layer

BGP messages exchanged between peers over TCP connection

- OPEN: opens TCP connection to peer and authenticates sender
- KEEPALIVE: keeps connection alive in absence of updates; also ACKs OPEN request
- UPDATE: announces, withdraws, or updates path attributes
- NOTIFICATION: reports errors in previous msg; also used to close connection

BGP messages exchanged between peers over TCP
A, B, C are provider networks

X, W, Y are customer networks of provider networks

X is dual-homed: attached to two networks

X does not want to route from B via X to C

.. so X will not advertise to B a route to C

Legend:

network: provider
network: customer

BGP routing policy
Network Layer 4-144

BGP routing policy (2)

A advertises path AW to B
B advertises path BAW to X

Should B advertise path BAW to C?

No way! B gets no revenue for routing CBAW since neither W nor C are B's customers.
B wants to route only to/from its customers!

B wants to force C to route to W via A

W
X
Y
A
B
C

Legend:

- Customer
- Provider
- Network

BGP Routing Policy (2)
Why different Intra-, Inter-AS routing?

- **Network Layer**
- 4-145

- **Inter-AS**: Policy may dominate over performance
- **Intra-AS**: Performance can focus on performance

- **Why different Intra-, Inter-AS routing?**
- **Policy**: 
  - Inter-AS: Single admin, so no policy decisions needed
  - Inter-AS: Admin wants control over how its traffic routed, who routes through its net.
- **Scale**: 
  - Inter-AS: Single admin, so no policy decisions needed
  - Hierarchical routing saves table size, reduced update traffic

- **Inter-AS**: Admin wants control over how its traffic routed, who routes through its net.
Q1 Network-layer functions

Which of the following are respectively used to
(1) assign an IP address, (2) forward packets, (3) compute routes within an AS?

A. DHCP, longest-prefix matching, interdomain routing
B. IP, NAT, BGP
C. IP, memory/bus/interconnection architecture, OSPF
D. TCP, IP, BGP
E. TCP, IP, BGP

Which of the following are respectively used to compute routes within an AS?
(1) assign an IP address, (2) forward packets, (3) 

Q1 Network-layer functions
Q2 Internet routing

What is hot-potato routing used for?

A. To determine whether to accept or decline a route.
B. To determine whether and who to re-advertise a route.
C. To determine the best interdomain route by relying on intradomain routing information.
D. To determine the best interdomain route based on local preferences.
E. To disseminate interdomain routing information to routers within an AS.
What is the most number of iterations that RIP can take to re-converge after a routing event? (Hint: how bad can count-to-infinity be in RIP given its implementation?)
Are the two routes below allowed by valley-free routing? The source is the leftmost node and the destination is the rightmost. Arrows indicate customer-to-provider flow relationship, non-arrows indicate peering.

A. Yes, Yes
B. Yes, No
C. No, No
D. No, Yes

Are the two routes below allowed by valley-free routing?
Chapter 4: outline

4.1 introduction
4.2 virtual circuit and datagram networks
4.3 what’s inside a router
4.4 IP: Internet Protocol
   - datagram format
   - IPv4 addressing
   - ICMP
   - IPv6
4.5 routing algorithms
   - link state
   - distance vector
   - hierarchical routing
4.6 routing in the Internet
   - RIP
   - OSPF
   - BGP
4.7 broadcast and multicast routing
source duplication: how does source determine recipient addresses?

source duplication is inefficient:

deliver packets from source to all other nodes

Broadcast routing
Limiting in-network duplication

Network Layer 4-152

Limiting in-network duplication by any node

- Packets only forwarded along a single shared spanning tree, so no redundant packets received by any node

**Spanning tree:**
- e.g., reverse path forwarding (RPF) (details shortly)
  - Node keeps track of packet ids already broadcasted
  - Hasn’t broadcast same packet before

**Flooding:**
- Node only broadcasts pkt if it
  - Controlled Flooding: Node only broadcasts pkt if it
  - Sends copy to all neighbors
  - When node receives broadcast packet,

Problems: cycles & broadcast storm
Multicast routing: problem statement

Goal: find a tree (or trees) connecting routers having local multicast group members.

Legend:
- **member**: group member
- **group router**: router with a group member
- **not member**: not group member
- **source-based**: different tree from each sender to recipients
- **group-shared**: single spanning tree used by all members
- **tree**: not all paths between routers used

Group-shared spanning tree and source-based forest broad approaches for both broadcast and multicast routing.
Network Layer 4-154

(a) Broadcast initiated at A

(b) Broadcast initiated at D

First construct one spanning tree, then forward/make copies only along that spanning tree.

Shared spanning tree
(a) stepwise construction of spanning tree (center: E)

(b) constructed spanning tree

Shared spanning tree: creation

- pick a center node
- each node sends unicast join message to center
- message forwarded until it arrives at a node already belonging to spanning tree

Shared spanning tree: creation
Network Layer

Source-based shortest path tree

LEGEND

- Source: source
- i: indicates order link added by algorithm
- 1: Link used for forwarding
- i indicates order link added
- Router with attached group member
- Router with no attached group member

E.g., Dijkstra's algorithm + reverse-path forwarding routes from source to all receivers multicast forwarding: tree of shortest path

"Source-based shortest path tree"
If (multicast datagram received on incoming link on shortest path back to source) then flood datagram onto all outgoing links else ignore datagram.

Each router has simple forwarding behavior:
- Source-based approach relying on router's knowledge of unicast shortest path to sender
- Source-based approach relying on router's shortest path to source

Reverse path forwarding
may be a bad choice with asymmetric links.

Result is a source-specific reverse SPT.

Reverse path forwarding: example
Network Layer

Reverse path forwarding: pruning

Forwarding

Links with multicast

Prune message

group member

Router with no attached
group member

Router with attached
group member

LEGEND

s: source

downstream group members

prune“ msg's sent upstream by router with no

no need to forward datagrams down subtree

forwarding tree has subtrees with no multicast group members

Reverse path forwarding: pruning
End of Chapter 4 syllabus
Network Layer

Network Layer 4-161

Shared-tree: steiner tree

- Steiner tree: minimum cost tree connecting all routers with attached group members

- Excellent heuristics exist

- Problem is NP-complete

- Information about entire network needed

- Computational complexity

- Not used in practice:

  - Monolithic: rerun when any router needs to join/leave

Shared-tree: steiner tree
Center-based trees

4-162

Network Layer

this router

path taken by join-msg becomes new branch of tree for

or arrives at center

join-msg either hits existing tree branch for this center, or

forwarded towards center

join-msg processed by intermediate routers and

router

edge router sends unicast join-msg addressed to center

to Join:

one router identified as “center” of tree

single delivery tree shared by all
Center-based trees: example

Suppose R6 chosen as center.
Internet Multicasting Routing: DVMRP

- DVMRP: distance vector multicast routing protocol, RFC1075
- flood and prune: reverse path forwarding, sourcing via RPF
- initial datagram to mcast group flooded everywhere
- no assumptions about underlying unicast
- constructed by communicating DVMRP routers
- RPF tree based on DVMRP's own routing tables
- RPF tree based
- protocol, RFC1075
- DVMRP: distance vector multicast routing

Routers not wanting group: send upstream prune msgs
DVMRP: continued...

- Soft state: DVMRP router periodically (1 min.)
- Forgets branches are pruned:
- Downstream router: re-prune or else continue to receive
- Odds and ends
- Following IGMP join at leaf
- Routers can quickly regraft to tree
- Commonly implemented in commercial router
Network Layer 4.166

**Tunneling**

Q: how to connect islands of multicast routers in a sea of unicast routers?

mcast datagram encapsulated inside normal (non-multicast-addressed) datagram

normal IP datagram sent thru „tunnel“ via regular IP unicast

receiving mcast router unencapsulates to get mcast datagram

multicast datagram inside „normal“ normal IP tunneling

physical topology

logical topology
PIM: Protocol Independent Multicast

- not dependent on any specific underlying unicast routing algorithm (works with all)

Two different multicast distribution scenarios:

- **dense**
  - bandwidth more plentiful
  - group members densely packed, in "close proximity"
  - members small with respect to number of interconnected networks

- **sparse**
  - bandwidth not plentiful
  - group members widely dispersed
  - members small with respect to number of networks

PIM: Protocol Independent Multicast
Consequences of sparse-dense dichotomy:

- Consequences of sparse-dense dichotomy:
  - Router processing of multicast trees (e.g., RPF)
  - Bandwidth and non-group-based
  - Data-driven construction on receiver-driven construction
  - Explicitly join explicitly prune
  - No membership until routers
  - Sparse
  - dense

Router processing profiteering

Consequences of sparse-dense dichotomy:
PIM dense mode

is a leaf-node router
has protocol mechanism for router to detect it
underlying routing algorithm
flood than DVMRP reduces reliance on
less complicated (less efficient) downstream
for incoming datagram
underlying unicast protocol provides RPF into
flood-and-prune RPF: similar to DVMRP but...
PIM - Sparse mode

Network Layer 4.170

PIM - sparse mode

Center-based approach

Router sends join msg to center-based approach

Path concentration, shorter performance: less increased specific tree

After joining via RP, router can switch to source-specific tree

Intermediate routers update state and forward join

Rendezvous point (RP)

All data multicast from rendezvous point

R1

R2

R3

R4

R5

R6

R7

joins
Network Layer

**PIM - Sparse mode**

- RP can send *stop* msg to source if no attached receivers
- RP can extend mcast tree upstream to RP
- RP can extend mcast down RP-rooted tree which distributes unicast data to RP
- “no one is listening!” receivers
- RP sends *stop* msg if no attached receivers

All data multicast from rendezvous point

Join

R1
R2
R3
R4
R5
R6
R7

Network Layer 4-171

sender(s):
Chapter 4: Done!