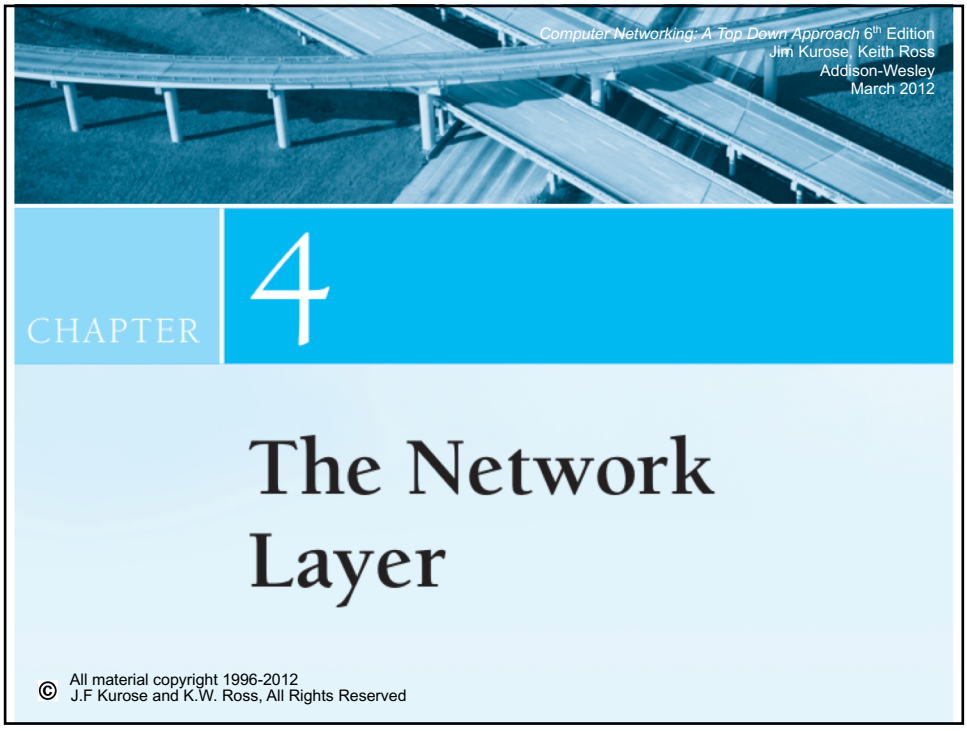
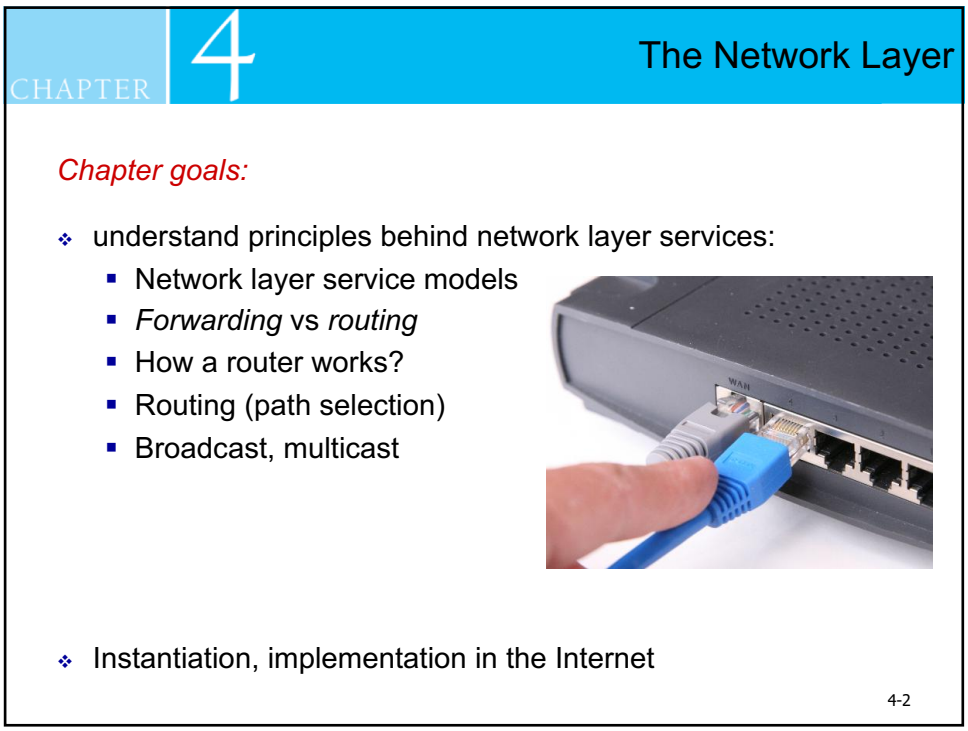

Computer Networking: A Top Down Approach 6th Edition
Jim Kurose, Keith Ross
Addison-Wesley
March 2012
CHAPTER
4
The Network Layer
© All material copyright 1996-2012
J.F Kurose and K.W. Ross, All Rights Reserved


CHAPTER
4
The Network Layer
Chapter goals:
understand principles behind network layer services:
Network layer service models
Forwarding vs routing
How a router works?
Routing (path selection)
Broadcast, multicast
Instantiation, implementation in the Internet
4-2

CHAPTER 4

The Network Layer

Roadmap:

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

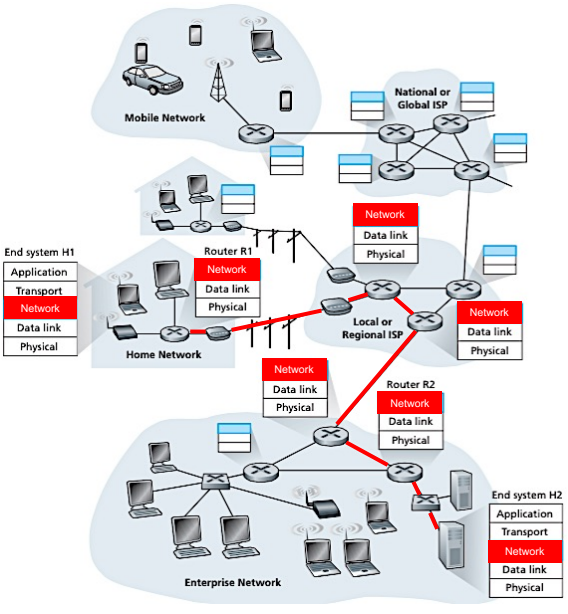
4.6 Routing in the Internet (intra-AS routing)
4.7 Broadcast and multicast routing

4-3

CHAPTER 4

(4.1) Introduction

- Transport *segment* from sending to receiving host
- on **sending** side encapsulates segments into _____
- on **receiving** side, delivers _____ to transport layer
- network layer protocols in *every* host and router
- router examines header fields in all IP *datagrams* passing through it



CHAPTER 4

Forwarding and Routing

(Two Key Network-Layer Functions)

Two Functions

1st:

Move packets from router's input to appropriate router output.

Analogy:
Process of getting through single interchange.

2nd:

Determine route taken by packets from source to destination.

Analogy:
Process of planning trip from source to destination.

4-5

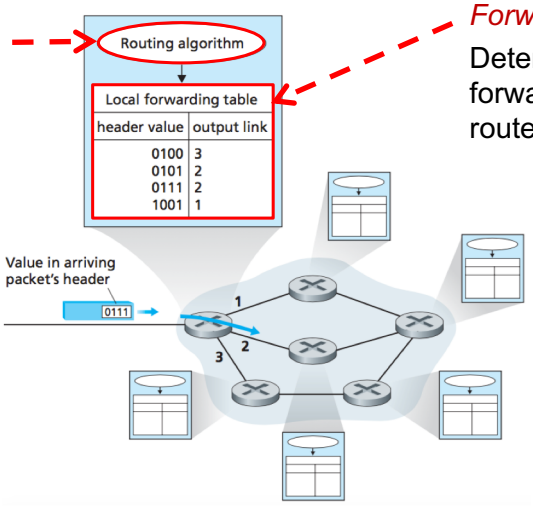
CHAPTER 4

Forwarding and Routing

Interplay between forwarding and routing

Routing:

Determines end-to-end path through network



Forwarding:

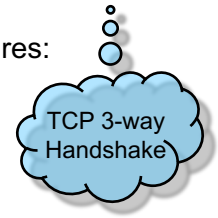
Determines local forwarding for this router

Figure: Routing algorithm determine values in forwarding tables

CHAPTER 4 Forwarding and Routing

3rd: Connection Setup

- 3rd important function in *some* network architectures:
  - Applied in ATM, frame relay, X.25
  - Not applied in Internet



- Before datagrams flow, two end *hosts* **and** intervening *routers* establish _____ (VC)
  - routers get involved
- Network vs Transport layer connection service:
  - *Network:* between two hosts (may also involve intervening routers in case of VCs)
  - *Transport:* between two processes

ATM (Asynchronous Transfer Mode)

4-7

CHAPTER 4 Network Service Models

Q: What *service model* for "channel" transporting *datagrams* from sender to receiver?

Example 1:

Services for individual datagrams:

- guaranteed delivery
- guaranteed delivery with less than 40 *msec* delay

Example 2:

Services for a flow of datagrams:

- in-order datagram delivery
- guaranteed minimum bandwidth to flow
- restrictions on changes in inter-packet spacing

4-8

CHAPTER 4

Network Service Models

Table: Internet and ATMs service models

Network Architecture	Service Model	Bandwidth Guarantee	No-Loss Guarantee	Ordering	Timing	Congestion Indication
Internet	Best Effort	None	None	Any order possible	Not maintained	None
ATM	CBR	Guaranteed constant rate	Yes	In order	Maintained	Congestion will not occur
ATM	ABR	Guaranteed minimum	None	In order	Not maintained	Congestion indication provided

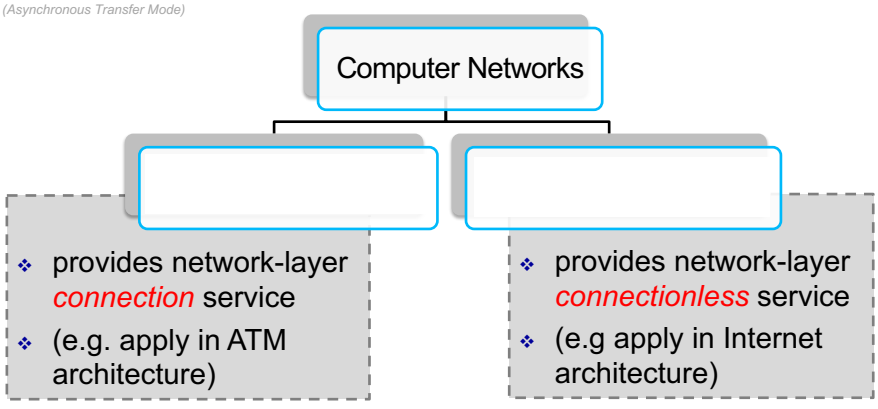
Asynchronous Transfer Mode (ATM)
CBR (Constant Bit Rate)
ABR (Available Bit Rate)

4-9

CHAPTER 4

(4.2) Virtual Circuit & Datagram Networks

ATM (Asynchronous Transfer Mode)



Analogous to TCP/UDP connection-oriented / connectionless transport-layer services, but:

- *service:* host-to-host
- *no choice:* network provides one or the other
- *implementation:* in network core

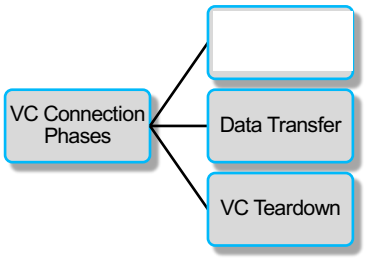
4-10

CHAPTER 4

(a) Virtual-Circuit (VC) Networks

"Source-to-destination path behaves much like telephone circuit"
- performance-wise
- network actions along source-to-destination path

- each packet carries VC identifier (not destination host address)
- *every* router on source-destination path maintains "state" for each passing connection



- link, router resources (bandwidth, buffers) may be *allocated* to VC (dedicated resources = predictable service)

4-11

CHAPTER 4

(a) Virtual-Circuit (VC) Networks

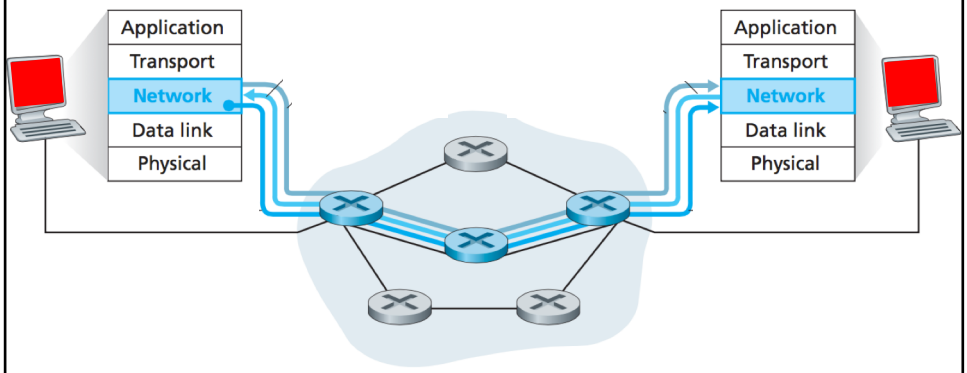


Figure: Virtual-Circuit setup.

4-12

CHAPTER 4

(b) Datagram Networks

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

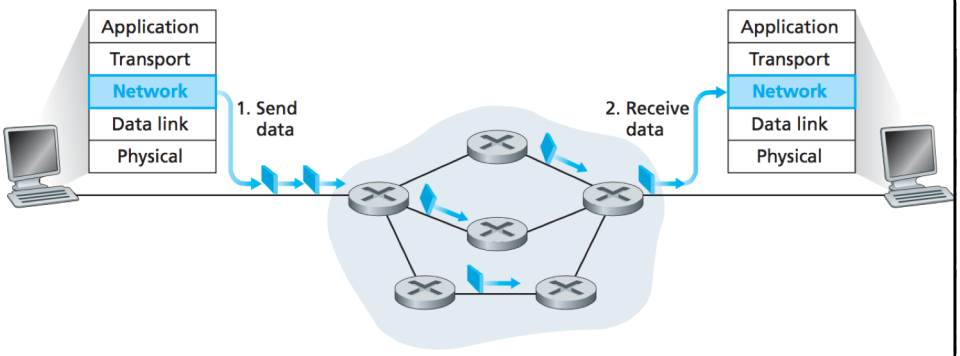


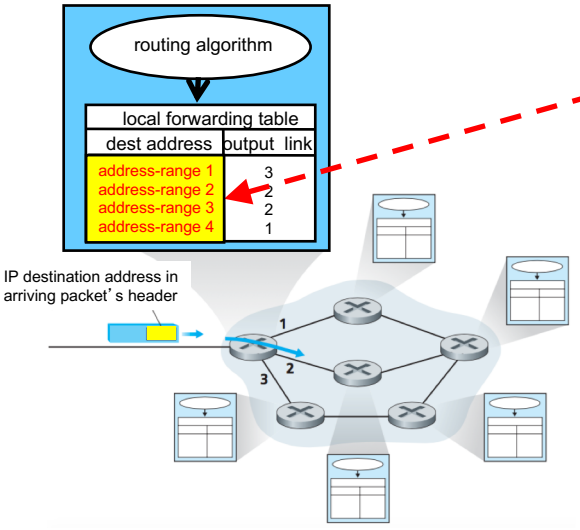
Figure: Datagram network.

4-13

CHAPTER 4

(b) Datagram Networks

Datagram forwarding table



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

IPv4: 2^{32} = 4,294,967,296

4-14

CHAPTER 4

(b) Datagram Networks

Datagram forwarding table

	Destination Address Range	Link Interface
200.23.16.0 200.23.23.255	11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
200.23.24.0 200.23.24.255	11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
200.23.25.0 200.23.31.255	11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
	otherwise	3

Table: Four routers has four links interfaces and packets to be forwarded.

CHAPTER 4

Datagram vs VC Networks: Why?

ATM (Asynchronous Transfer Mode)

Internet *(datagram)*

- data exchange among computers
  - "elastic" service, no strict timing requirement
- many link types:
  - different characteristics
  - uniform service difficult
- "smart" end systems (computers)
  - can adapt, perform control, error recovery

_____ *inside network, complexity at "edge"*

ATM *(VC)*

- evolved from telephony
- human conversation:
  - strict timing, reliability requirements
  - need for guaranteed service
- "dumb" end systems
  - telephones

_____ *inside network*

4-16

CHAPTER 4

(4.3) What's Inside a Router?

TWO key router functions:

- run *routing algorithms* / protocol (RIP, OSPF, BGP)
- *forwarding* datagrams from incoming to outgoing link

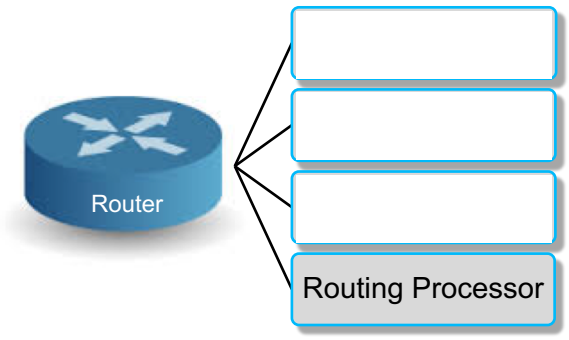


Figure: Four router components

RIP (Routing Information Protocol)
OSPF (Open-Shortest Path First)
BGP (Border Gateway Protocol)

4-17

CHAPTER 4

Router Architecture

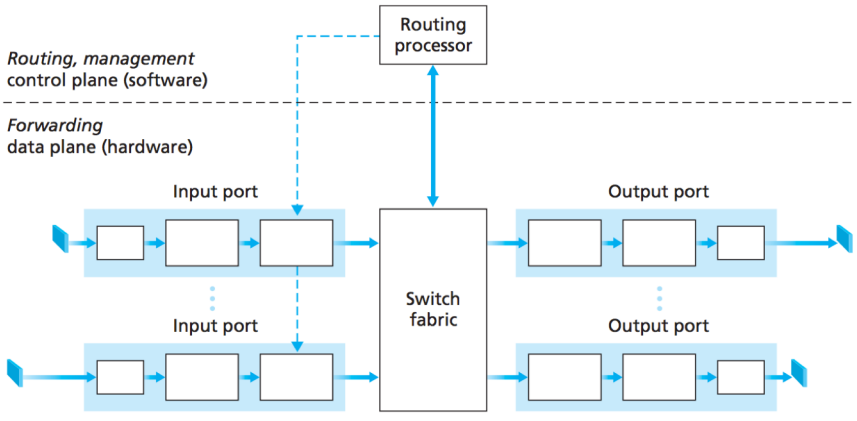


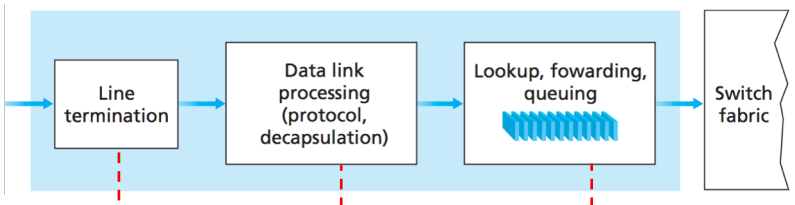
Figure: A high-level view of a generic router architecture

4-18

CHAPTER 4

Input Processing

Input port functions



Physical layer: bit-level reception

Data link layer: e.g., Ethernet (see chapter 5)

Decentralized switching:

- given datagram destination, lookup output port using forwarding table in input port memory *("match plus action")*
- **Goal**: complete input port processing at 'line speed'
- queuing: if datagrams arrive faster than forwarding rate into switch fabric

CHAPTER 4

Switching Fabrics

- Transfer packet from **input buffer** to appropriate **output buffer**
- Switching rate: rate at which packets can be transfer from inputs to outputs
  - often measured as multiple of input/output line rate
  - *N* inputs: switching rate = *N* times line rate desirable

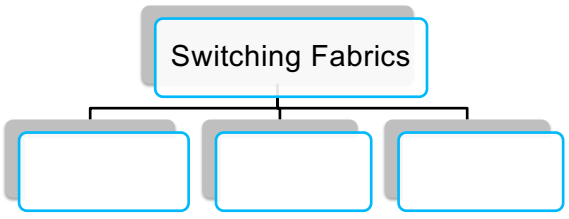


Figure: Three type of switching fabrics techniques

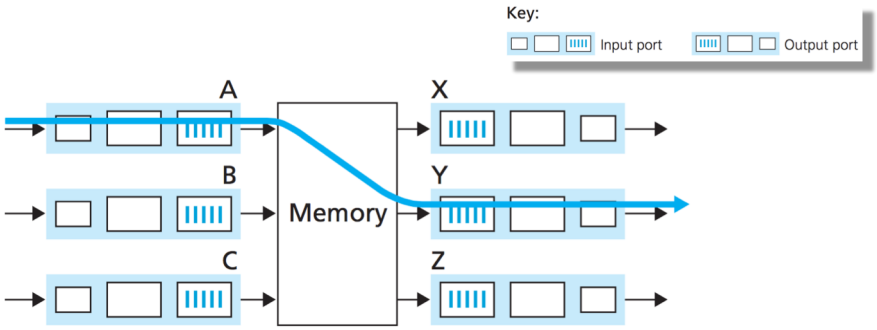
4-20

CHAPTER 4 Switching Fabrics

(i) Memory

First generation routers:

- traditional computers with switching under direct control of CPU
- packet copied to system's memory
- speed limited by memory bandwidth (2 bus crossings per datagram)

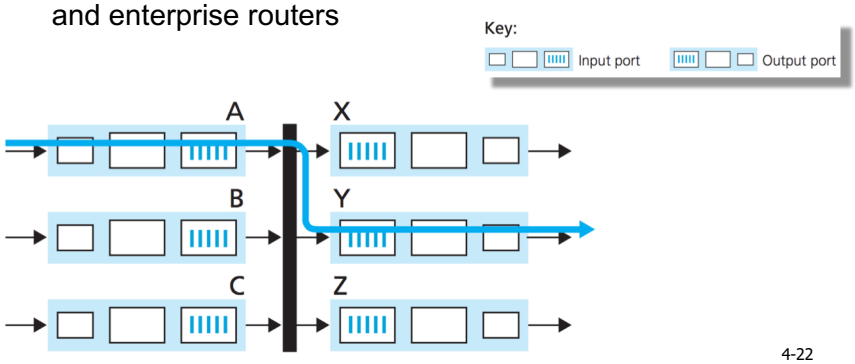


4-21

CHAPTER 4 Switching Fabrics

(ii) Bus

- Datagram from input port memory to output port memory via a single shared bus
- *Bus contention (conflict):* switching speed limited by bus bandwidth
  - e.g. 32 Gbps bus, Cisco 5600: sufficient speed for access and enterprise routers



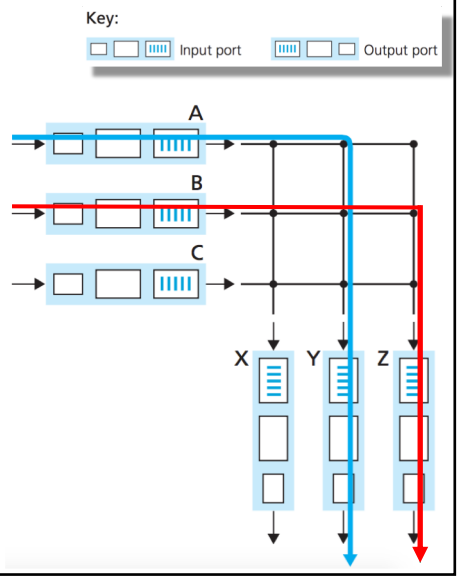
4-22

CHAPTER 4

Switching Fabrics

(iii) Crossbar

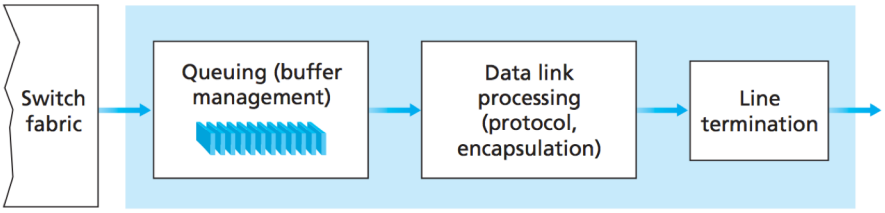
- Overcome bus bandwidth limitations (single shared bus)
- *Banyan networks*, *crossbar*, other interconnection networks initially developed to connect processors in multiprocessor
- advanced design (to speedup switching)
  - ✓ _____ into fixed length cells, switch cells through the fabric.
- Cisco 12000: switches 60Gbps through the interconnection network



CHAPTER 4

Output Processing

Output ports



- _____ required when datagrams arrive from fabric faster than the transmission rate
- *scheduling discipline* chooses among queued datagrams for transmission

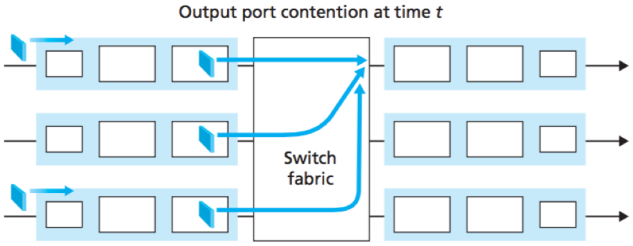
4-24

CHAPTER 4

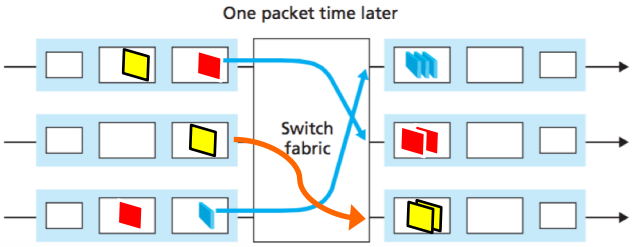
Where Does Queueing Occurs?

Output ports queueing

- buffering when arrival rate via switch exceeds output line speed



- *(delay) and (or drop tail) due to output port buffer overflow!*
- *QoS will be affected*



QoS (Quality-of-Services))

CHAPTER 4

Where Does Queueing Occurs?

Buffer size needed

- RFC3439 rule of thumb:
  - average buffering size = RTT * link capacity C

- Example:
  RTT = 250 *msec* ("typical"), C = 10 *Gbps* link
  → average buffering size = 250 *msec* * 10 *Gbps*
  = 2.5 *Gbit* buffer

4-26

CHAPTER 4

Exercise 4.0

Supposed the link capacity for a router is 20 *Mbps* and RTT for a packet is 400 *msec*.

(a) Calculate the average buffering size.

(b) If the TCP flow of packets is 16, calculate the buffering needed for the router.

CHAPTER 4

Solution 4.0

Solution:

- Link capacity, C = 20 *Mbps*
- RTT = 400 *msec*
- TCP flows, N = 16

(a) Average Buffer size,	(b) Buffer size,

CHAPTER 4

Where Does Queueing Occurs?

Input ports queueing

- Fabric slower than input ports combined → queueing may occur at input queues
  - *Queueing delay and loss due to input buffer overflow!*

_____ (HOL) blocking:

Queued datagram at front of queue prevents others in queue from moving forward

4-29

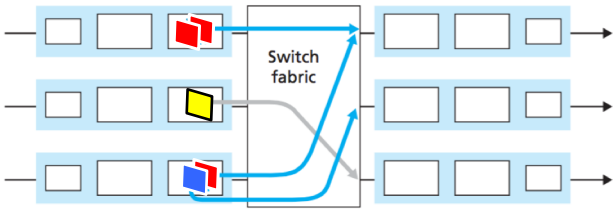
CHAPTER 4

Where Does Queueing Occurs?

Input ports queueing

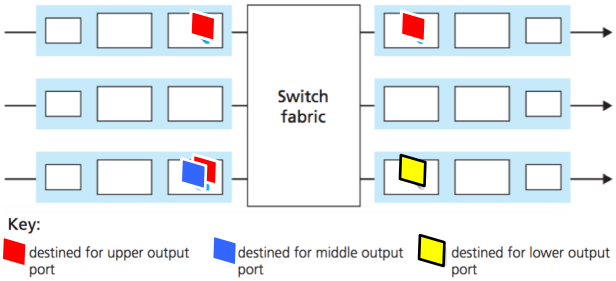
Output port contention (conflict):

- Only one red datagram can be transferred.
- lower red packet is blocked.



One packet time later:

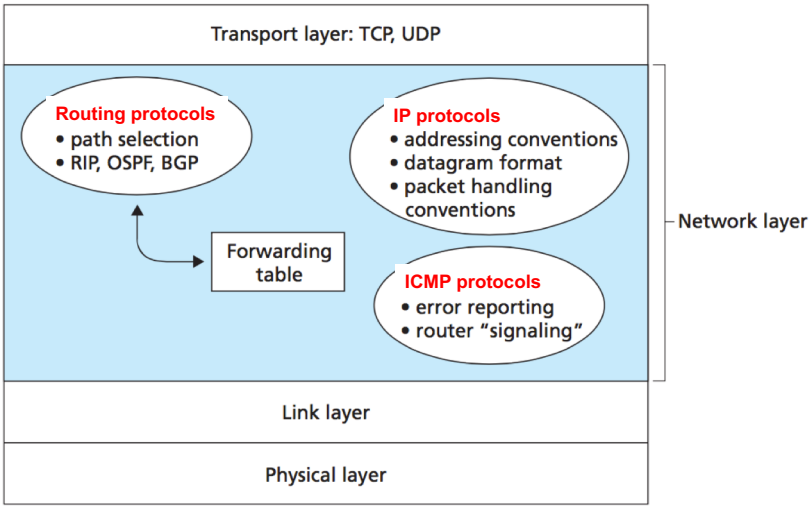
- blue packet experiences HOL blocking



CHAPTER 4

(4.4) The Internet Protocol (IP)

The Internet network layer

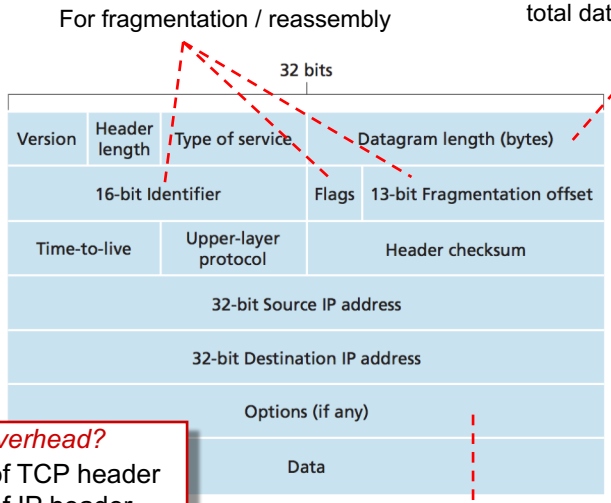


RIP (Routing Information Protocol)
OSPF (Open Shortest path First)
BGP (Border Gateway Protocol)

4-31

CHAPTER 4

IPv4 Datagram Format



How much overhead?
→ 20 bytes of TCP header
→ 20 bytes of IP header
= 40 bytes header + application layer overhead

CHAPTER 4

IPv4 Datagram Format

IP fragmentation, reassembly

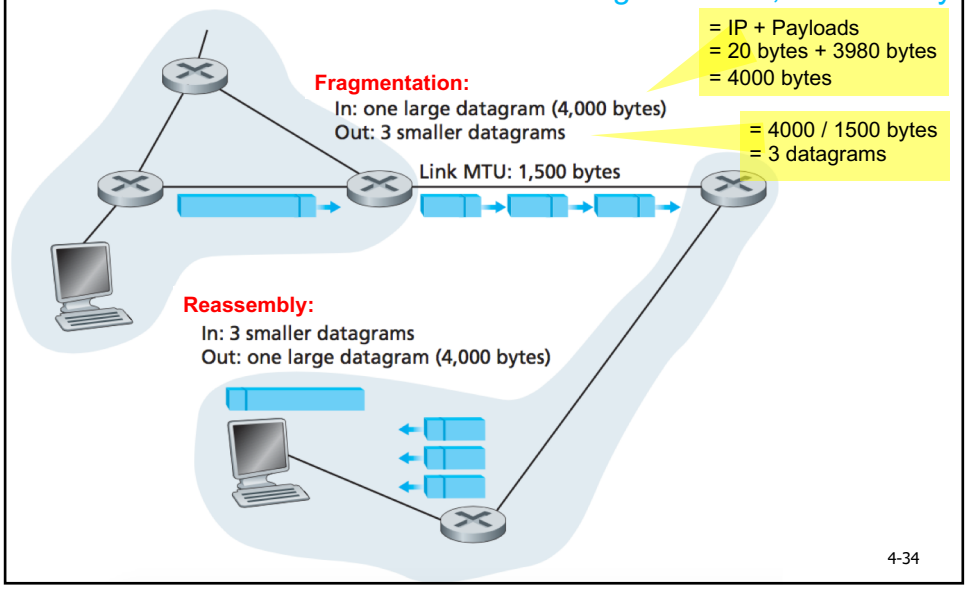
16-bit Identifier	Flags	13-bit Fragmentation offset

- Network links have MTU (_____)
  - largest possible link-level frame
    - different link types, different MTUs
- Large IP datagram divided ("fragmented") within network
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order related fragments

CHAPTER 4

IPv4 Datagram Format

IP fragmentation, reassembly



4-34

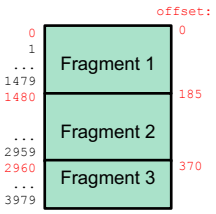
CHAPTER 4

IPv4 Datagram Format

IP fragmentation, reassembly

Fragment	Bytes	ID	Offset	Flag
1st fragment	1,480 bytes in the data field of the IP datagram	identification = 777	offset = 0 (meaning the data should be inserted beginning at byte 0)	flag = 1 (meaning there is more)
2nd fragment	1,480 bytes of data	identification = 777	offset = 185 (meaning the data should be inserted beginning at byte 1,480. Note that 185 · 8 = 1,480)	flag = 1 (meaning there is more)
3rd fragment	1,020 bytes (= 3,980–1,480–1,480) of data	identification = 777	offset = 370 (meaning the data should be inserted beginning at byte 2,960. Note that 370 · 8 = 2,960)	flag = 0 (meaning this is the last fragment)

Figure: IP fragments



Offset value specified in units of 8-byte chunks

CHAPTER 4

IPv4 Datagram Format

IP fragmentation, reassembly

Fragment #1	length=1500	ID=777	flag=1	offset=0
Fragment #2	length=1500	ID=777	flag=1	offset=185
Fragment #3	length=1040	ID=777	flag=0	offset=370

Figure: IP fragments

CHAPTER 4

Exercise 4.1

A datagram of 6500 bytes arrived at a router and must be forwarded to a link with an MTU of 1900 bytes. Suppose that the original datagram is stamped with an identification number of 743. Draw all IP fragments generated after fragmentation that reflect the requirement of original payload data in the datagram.

Datagram

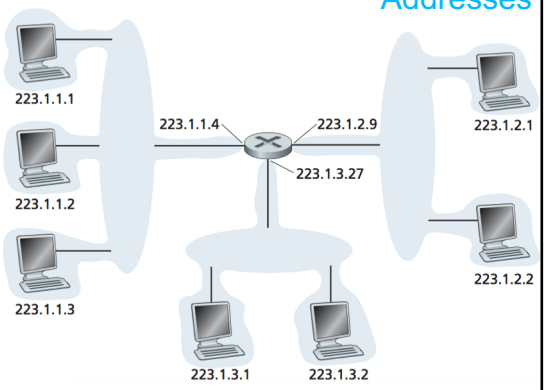
length=6500	ID=743	flag=0	offset=0

CHAPTER 4

IPv4 Addressing

- *IP address:* 32-bit identifier for host, router *interface*

Addresses



223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

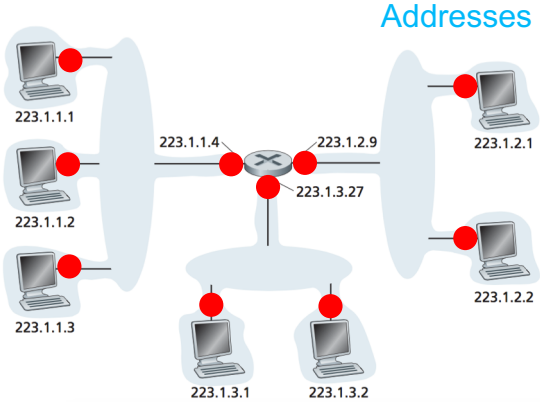
Dotted-decimal notation

CHAPTER 4

IPv4 Addressing

Addresses

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



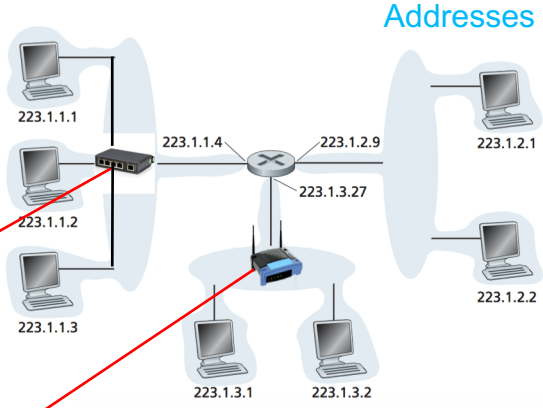
One IP address associated with each interface !

CHAPTER 4

IPv4 Addressing

Addresses

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



A: wired Ethernet interfaces connected by Ethernet switches (Link Layer)

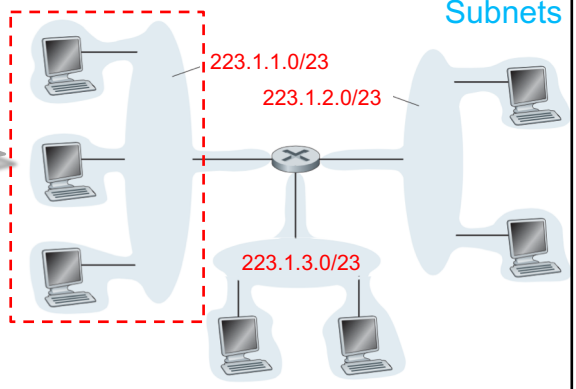
A: wireless WiFi interfaces connected by WiFi base station

For now: Don't need to worry about how one interface is connected to another (with no intervening router)

CHAPTER 4 IPv4 Addressing

Subnets

Interconnecting three host interfaces and one router interface forms a _____.



What's a **subnet** ?

- Subnet or subnetwork is a logical, visible subdivision of an *IP network*.
- The practice of dividing a network into two or more networks is called _____.

CHAPTER 4 IPv4 Addressing

Subnets

Recipe:

- to determine the subnets, detach each interface from its host or router, creating islands of isolated networks
- each isolated network is called a *subnet*

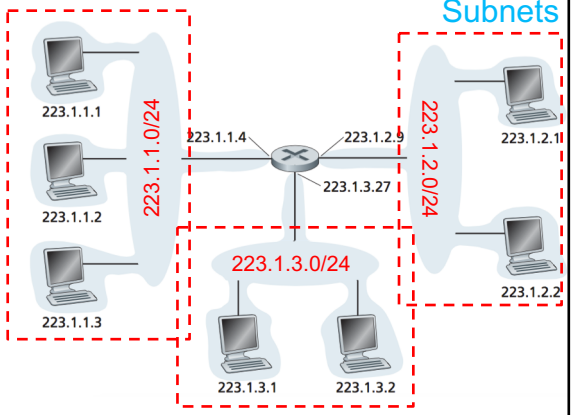


Figure: Network consisting of 3 subnet addresses

223.1.3.0 / 24

Subnet part → ← Host part

11011111.00000001.00000011.00000000
⋮
11011111.00000001.00000011.11111111

Subnet mask: /24

CHAPTER 4 IPv4 Addressing

Subnets Mask

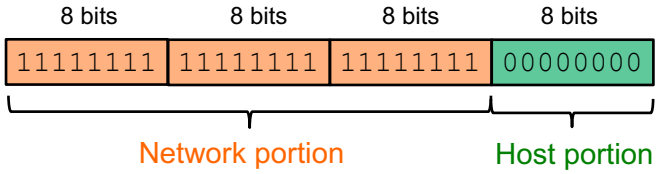
- A computer (or a router, which is simply a specialized computer) must be able to identify whether a computer with a given IP address is on its subnet or not.
- The _____ is:
  - used to separate the network portion of an IP address from the host portion.
  - a set of 32 bits which the bits in the network portion of the address are set to 1s and the host portion is set to 0s.

4-43

CHAPTER 4 IPv4 Addressing

Subnets Mask

- Subnet mask : `255.255.255.0`



- This is called a `/24` address.

Example:
IP address `192.168.1.100/24` would be separated into:
- subnet portion → `192.168.1`
- host portion → `100`

4-44

Self-Test

IPv4 Addressing

Subnets Mask

Find the subnet mask address in decimal values.

(a) /8

(b) /16

(c) /24

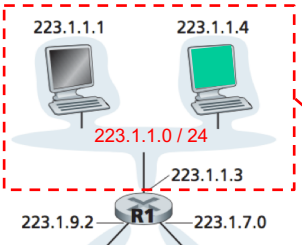
(d) /32

4-45

CHAPTER 4

IPv4 Addressing

Subnets Mask



Subnet address:
IP address AND subnet mask

Example:
If the subnet mask is 255.255.255.0, define the subnet address.

```
      223.1.1.4
AND   255.255.255.0

11011111.00000001.00000001.00000100
11111111.11111111.11111111.00000000
11011111.00000001.00000001.00000000
```

223.1.1.0/24

CHAPTER 4

IPv4 Addressing

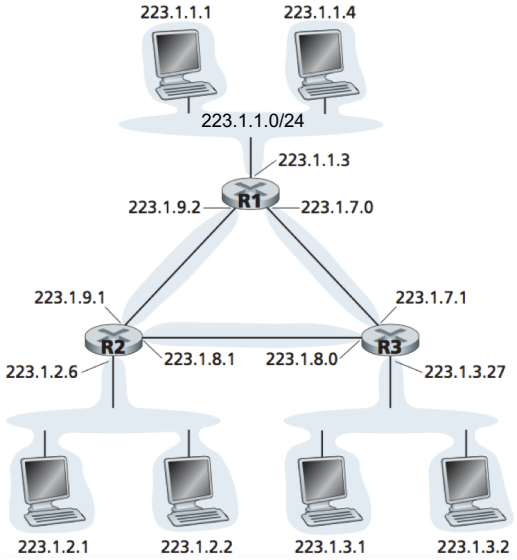


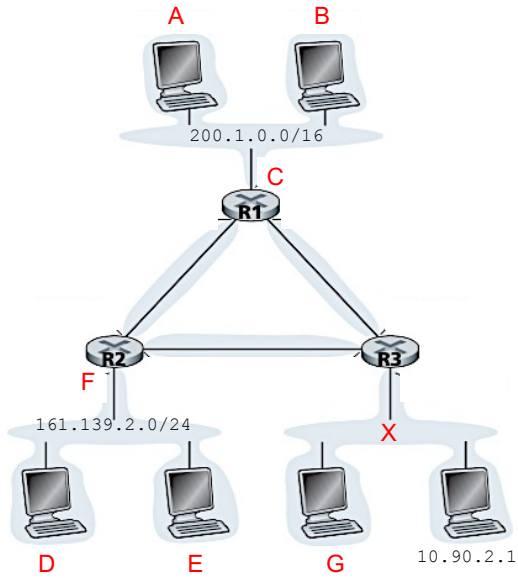
Figure: Three routers interconnecting six subnets

Self-Test

If the subnet mask is `255.255.255.0`, define all the subnet addresses with the subnet mask.

CHAPTER 4

Exercise 4.2



Given a network with hosts and interconnecting routers.

(a) What are the subnet mask addresses for the network that consists of host A, B, D, and E?

(b) What is the subnet address of network X if the subnet mask address is `255.0.0.0`?

(c) Get a valid IP address for each host A, B, D, E and G.

(d) Get a valid IP address for each router's interface at C and F.

CHAPTER 4

IPv4 Addressing

Classful addressing

Class A
Subnet Mask
/8

Netwok	Host	Host	Host
255	0	0	0

Class B
Subnet Mask
/16

Netwok	Network	Host	Host
255	255	0	0

Class C
Subnet Mask
/24

Netwok	Network	Network	Host
255	255	255	0

www.smartPCtricks.com

Example:

- An organization needs 2000 hosts and apply class B
- **Class B** allocated 65534 interfaces: leaving more than 63000 unused.

Problem:

- Not optimized and wasted addresses

Solution: CIDR

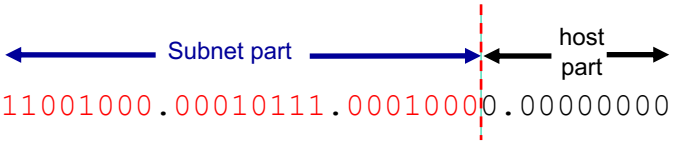
CIDR (Classless InterDomain Routing)

CHAPTER 4

IPv4 Addressing

Classless InterDomain Routing (CIDR)

- more flexible than original system of Internet Protocol (IP) address scheme i.e. classful addressing
- can avoid situations where large numbers of IP addresses are unused
- subnet portion of address of arbitrary length
- address format: `a.b.c.d/x`, where `x` is # bits in subnet portion of address often referred to as `prefix`



200.23.16.0/23

CHAPTER 4

IPv4 Addressing

Subnets Mask

128	64	32	16	8	4	2	1		
1	0	0	0	0	0	0	0	=	128
1	1	0	0	0	0	0	0	=	192
1	1	1	0	0	0	0	0	=	224
1	1	1	1	0	0	0	0	=	240
1	1	1	1	1	0	0	0	=	248
1	1	1	1	1	1	0	0	=	252
1	1	1	1	1	1	1	0	=	254
1	1	1	1	1	1	1	1	=	255

Example:

Given an IP address `161.132.3.1/20`
Calculate the subnet mask address.

`11111111.11111111.1111`0000`.00000000`

20 bits

Subnet mask address : `255.255.240.0`

CHAPTER 4

Self-Test

Define the subnet mask address in decimal values.

(a) `/27`

(b) `/24`

(c) `/16`

(d) `/28`

(e) `/22`

4-52

CHAPTER 4 IPv4 Addressing

Determine the number of hosts in subnets

Example IP address: 172.20.0.0/16

Network portion	Host portion	
10101100 00010100	00000000 00000000	→ 1
10101100 00010100	00000000 00000001	→ 2
10101100 00010100	00000000 00000010	→ 3
...	...	
10101100 00010100	11111111 11111101	→ 65,534
10101100 00010100	11111111 11111110	→ 65,535
10101100 00010100	11111111 11111111	→ 65,536

N = 16 bits

Total hosts $= 2^N - 2 = 2^{16} - 2 = 65,534$

$= 65,536 - 2$
$= 65,534$

CHAPTER 4 IPv4 Addressing

Obtaining a block of addresses

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

A: Gets allocated portion of its provider ISP's address space

Example: 200.23.16.0/20 divided into 8 subnets (2^3).

ISP's block:
200.23.16.0/20 → 11001000.00010111.00010000.00000000

Subnet address #**0**:
200.23.16.0/23 → 11001000.00010111.00010000.00000000

Subnet address #**1**:
200.23.18.0/23 → 11001000.00010111.00010010.00000000

Subnet address #**2**:
200.23.20.0/23 → 11001000.00010111.00010100.00000000

CHAPTER 4

IPv4 Addressing

Obtaining a block of addresses

Subnet address #**3**:
200.23.22.0/20 → 11001000.00010111.00010110.00000000

Subnet address #**4**:
200.23.24.0/20 → 11001000.00010111.00011000.00000000

Subnet address #**5**:
200.23.26.0/23 → 11001000.00010111.00011010.00000000

Subnet address #**6**:
200.23.28.0/23 → 11001000.00010111.00011100.00000000

Subnet address #**7**:
200.23.30.0/23 → 11001000.00010111.00011110.00000000

CHAPTER 4

Exercise 4.3

Given an IP address as 200.23.0.0/20. The network need to be divided into 5 subnets with first subnet label as subnet#0.

(a) What is the subnet mask address for the given IP address?
(b) How many hosts can be supported?
(c) List all subnets addresses.
(d) What is the new subnet mask address for the each subnet?
(e) How many hosts can be supported for each subnet?
(f) List the first 5 valid IP addresses for the subnet address #3.
(g) How many subnet remain unused and can be utilized in future?

CHAPTER 4

IPv4 Addressing

Obtaining a block of addresses

Q: How does an ISP get block of addresses?

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

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

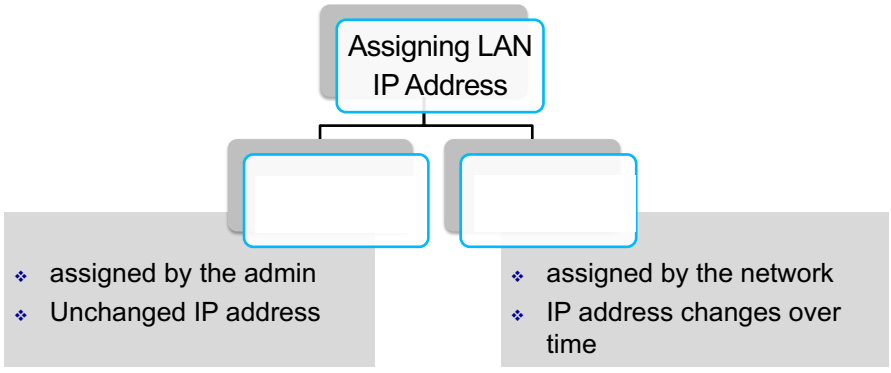
CHAPTER 4

IPv4 Addressing

Obtaining a host addresses

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

- From local organization (e.g. UTM)



CHAPTER 4

IPv4 Addressing

Obtaining a host addresses: Static IP

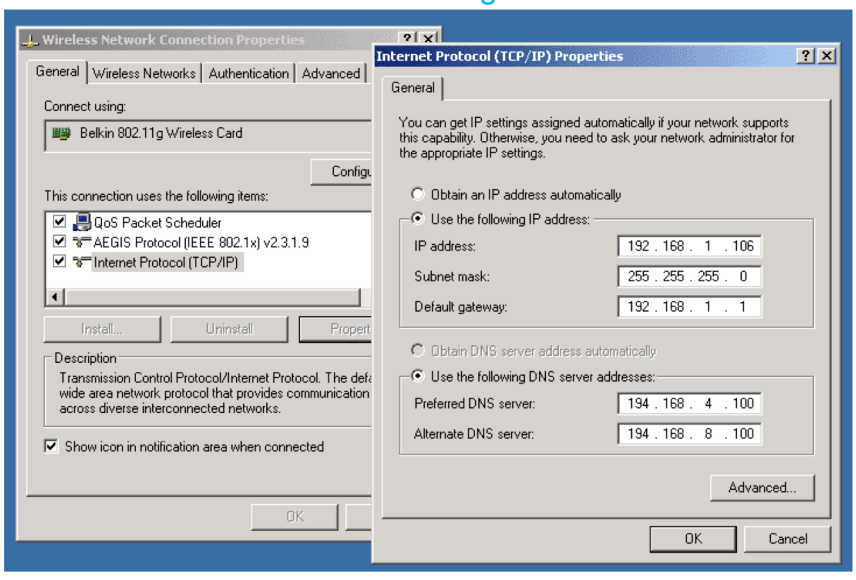
- Hard-coded by system admin in a file
  - Windows:
    control-panel → network → configuration → TCP/IP → properties
  - UNIX: `/etc/rc.config`

4-59

CHAPTER 4

IPv4 Addressing

Obtaining a host addresses: Static IP



http://www.abyssunderground.co.uk/images/router/ip2.gif

CHAPTER 4

IPv4 Addressing

Self-Test

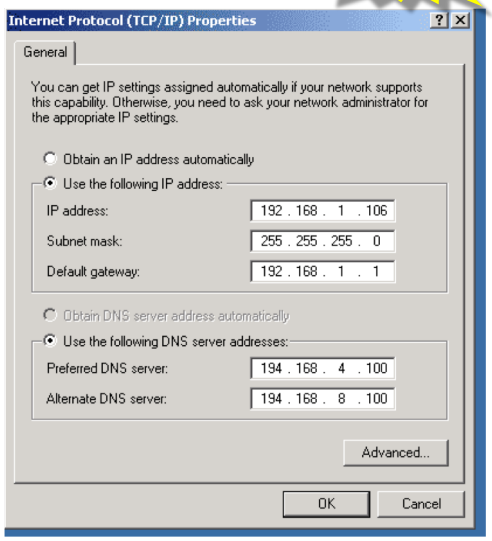
Given an Internet Protocol (TCP/IP) configuration.

(a) What is the subnet address?

(b) How many hosts can be supported in the subnet?

(c) What is the total subnets have been created if the network IP `192.168.0.0/20`

(d) What is the new subnet mask address if the IP address assigned as `192.168.3.106/22`



http://www.abyssunderground.co.uk/images/router/ip2.gif

CHAPTER 4

IPv4 Addressing

Obtaining a host addresses: DHCP

DHCP: Dynamic Host Configuration Protocol:
- Dynamically get address from as server
- "plug-and-play" protocol
- Temporarily IP address

Goal:

Allow host to *dynamically* obtain its IP address from network server when it joins network

- can renew its lease on address in use
- allows reuse of addresses (only hold address while connected/"on")
- support for mobile users who want to join network (more shortly)

4-62

CHAPTER 4 IPv4 Addressing

Obtaining a host addresses: DHCP

Figure: Four-step process of DHCP

DHCP overview:

- host broadcasts "DHCP discover" message [optional]
- DHCP server responds with "DHCP offer" message [optional]
- host requests IP address: "DHCP request" message
- DHCP server sends address: "DHCP ACK" message

4-63

IPv4 Addressing

Obtaining a host addresses: DHCP

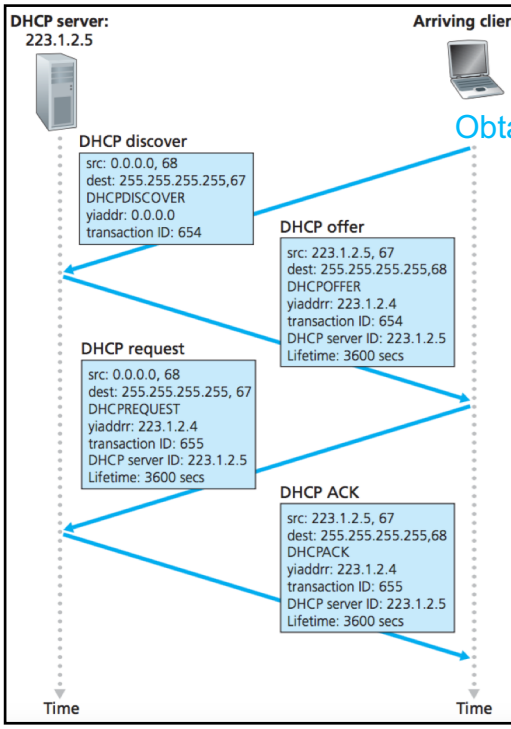
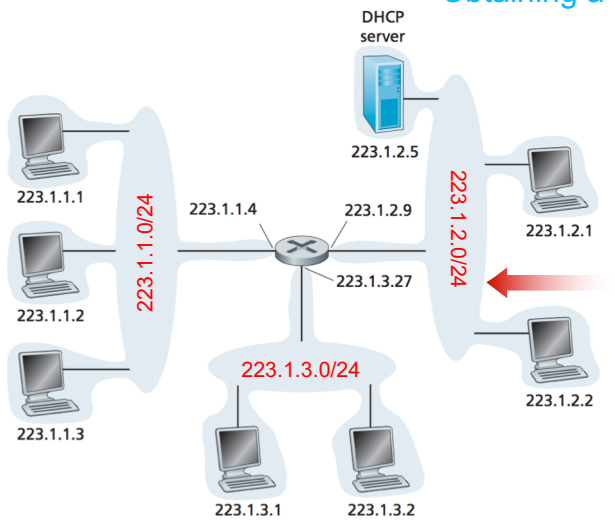


Figure: DHCP client-server interaction

yiaddr (your internet address)

CHAPTER 4 IPv4 Addressing

Obtaining a host addresses: DHCP



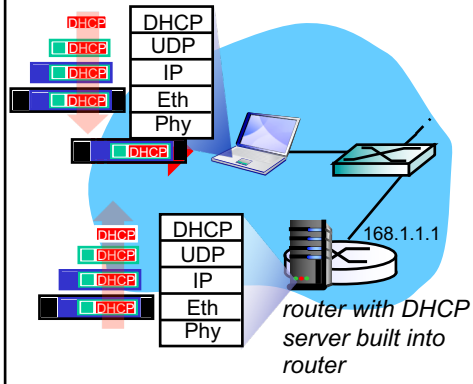
DHCP can return more than just allocated IP address on subnet:

- Address of first-hop router for client;
- Name and IP address of DNS server;
- Network mask (indicating network versus host portion of address)

Figure: DHCP client-server scenario

CHAPTER 4 IPv4 Addressing

Obtaining a host addresses: DHCP



- connecting laptop needs its IP address, address of first-hop router, address of DNS server: use DHCP
- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.11 Ethernet
- Ethernet frame broadcast (Destination: `255.255.255.255`) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

Figure: DHCP example

4-66

CHAPTER 4 IPv4 Addressing

Obtaining a host addresses: DHCP

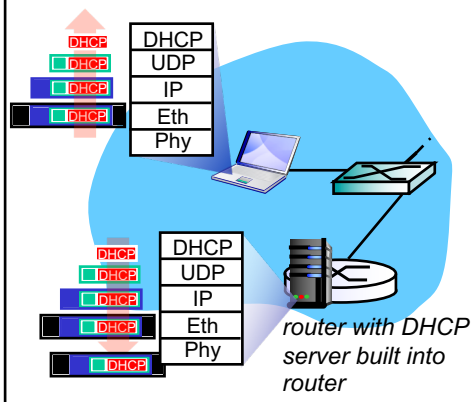


Figure: DHCP example

- DHCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulation of DHCP server, frame forwarded to client, demuxing up to DHCP at client
- client now knows its IP address, name and IP address of DNS server, IP address of its first-hop router

CHAPTER 4 IPv4 Addressing

Obtaining a host addresses: DHCP

Request

Message type: **Boot Request (1)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) **DHCP Message Type = DHCP Request**
Option: (61) Client identifier
Length: 7; Value: 010016D323688A;
Hardware type: Ethernet
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Option: (t=50,l=4) Requested IP Address = 192.168.1.101
Option: (t=12,l=5) Host Name = "nomad"
Option: (55) Parameter Request List
Length: 11; Value: 010F03062C2E2F1F21F92B
1 = Subnet Mask; 15 = Domain Name
3 = Router; 6 = Domain Name Server
44 = NetBIOS over TCP/IP Name Server
……

Reply

Message type: **Boot Reply (2)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 192.168.1.101 (192.168.1.101)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 192.168.1.1 (192.168.1.1)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) DHCP Message Type = DHCP ACK
Option: (t=54,l=4) Server Identifier = 192.168.1.1
Option: (t=1,l=4) Subnet Mask = 255.255.255.0
Option: (t=3,l=4) Router = 192.168.1.1
Option: (6) Domain Name Server
Length: 12; Value: 445747E2445749F244574092;
IP Address: 68.87.71.226;
IP Address: 68.87.73.242;
IP Address: 68.87.64.146
Option: (t=15,l=20) Domain Name = "hsd1.ma.comcast.net."

Figure: DHCP example Wireshark output (home LAN)

CHAPTER 4 IPv4 Addressing

Obtaining a host addresses: DHCP

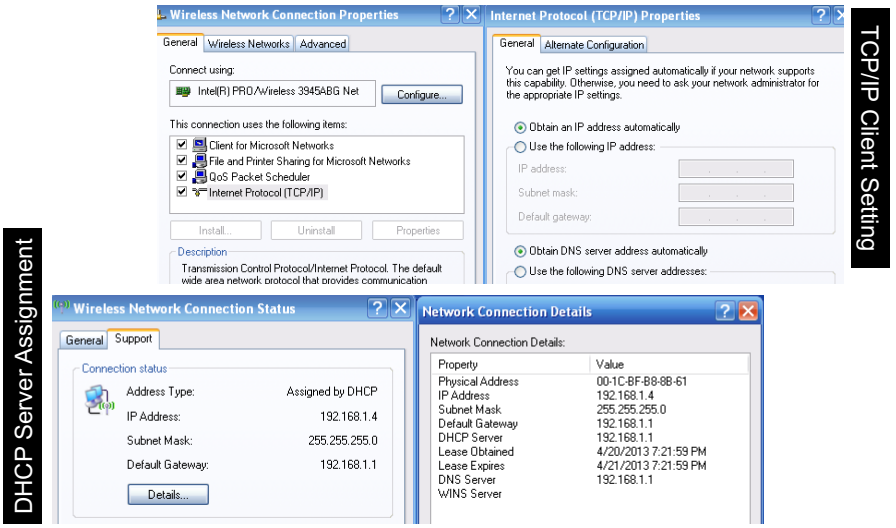
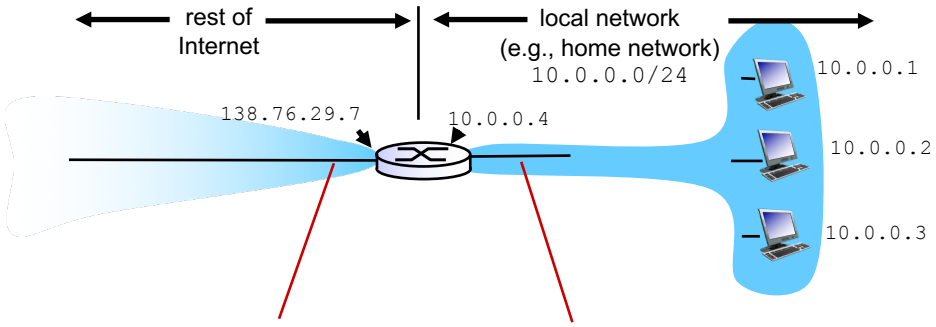


Figure: DHCP example assignment (home LAN)

CHAPTER 4 IPv4 Addressing

Network Address Translation (NAT)



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

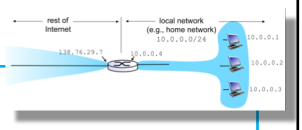
Datagrams with source or destination in this network have 10.0.0.0/24 address for source, destination (as usual)

4-70

CHAPTER 4 IPv4 Addressing

Network Address Translation (NAT)

Motivation: local network uses just one IP address as far as outside world is concerned:



- range of addresses not needed from ISP
  - just one IP address for all devices
- can change ISP without changing addresses of devices in local network
- can change addresses of devices in local network without notifying outside world
- devices inside local network not explicitly (precisely) addressable, visible by outside world (a security plus)

4-71

CHAPTER 4 IPv4 Addressing

Network Address Translation (NAT)

Implementation: NAT router must:

- *Outgoing datagrams: replace* (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
  . . . remote clients/servers will respond using (NAT IP address, new port #) as destination address
- *Remember (in NAT translation table)* every (source IP address, port #) to (NAT IP address, new port #) translation pair
- *Incoming datagrams: replace* (NAT IP address, new port #) in destination fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

CHAPTER 4

IPv4 Addressing

Network Address Translation (NAT)

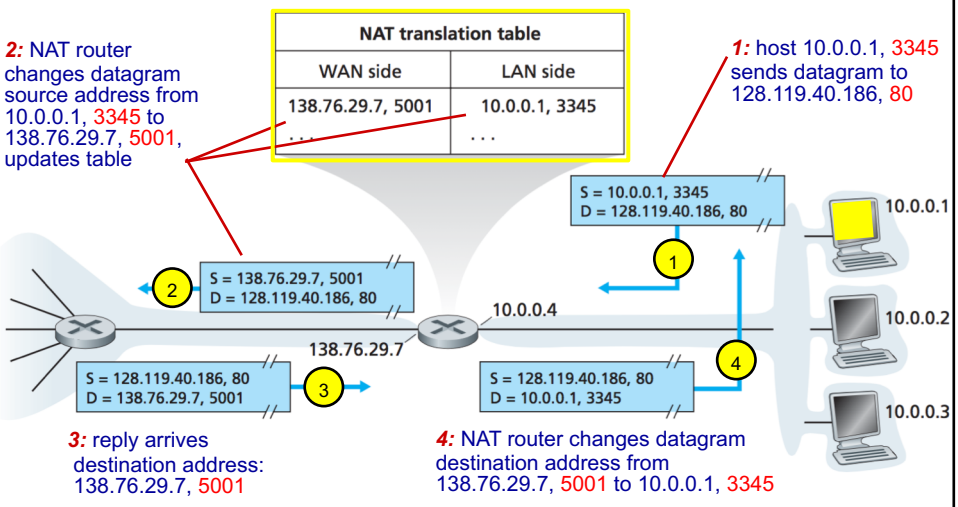


Figure: Network address translation

4-73

CHAPTER 4

Internet Control Message Protocol (ICMP)

- used by hosts & routers to communicate network-level information
  - error reporting: unreachable host, network, port, protocol
  - echo request/reply (used by ping)
- network-layer "above" IP:
  - ICMP message carried in IP datagrams
- ICMP message: type, code plus first 8 bytes of IP datagram causing error

Type	Code	Description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Figure: ICMP message types

4-74

CHAPTER 4

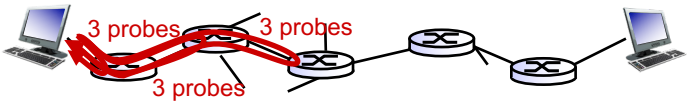
Internet Control Message Protocol (ICMP)

Traceroute and ICMP

- source sends series of UDP segments to destination:
  - first set has `TTL =1`
  - second set has `TTL=2`, etc.
  - unlikely port number
- when *n*th set of datagrams arrives to *n*th router:
  - router discards datagrams
  - and sends source ICMP messages (type 11, code 0)
  - ICMP messages includes name of router & IP address
- when ICMP messages arrives, source records RTTs

Stopping criteria:

- UDP segment eventually arrives at destination host
- destination returns ICMP "port unreachable" message (type 3, code 3)
- source stops



CHAPTER 4

IPv6

Motivation

Initial motivation:

- 32-bit address space soon to be completely allocated.

Additional motivation:

- header format helps speed processing/forwarding
- header changes to facilitate QoS

IPv6 datagram format:

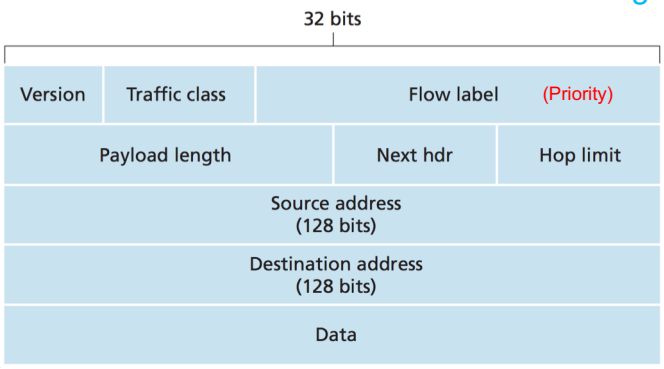
- fixed-length 40 byte header
- no fragmentation allowed

4-76

http://www.ibmsystemsmag.com/getattachment/11b1fdc8-09c3-4c2a-af55-4425e0a0cdfa/

CHAPTER 4 IPv6

Datagram format



Priority : identify priority among datagrams in flow

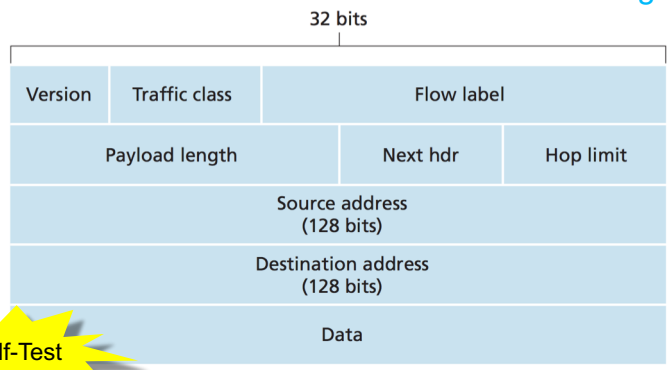
Flow Label : identify datagrams in same "flow." (concept of"flow" not well defined).

Next header : identify upper layer protocol for data

4-77

CHAPTER 4 IPv6

Datagram format



Self-Test

Proof the size of IPv6 datagram header is 40 bytes.

4-78

CHAPTER 4 IPv6

Other changes from IPv4

- *Checksum* : Removed entirely to reduce processing time at each hop
- *Options* : Allowed, but outside of header, indicated by "Next Header" field

- *ICMPv6* : New version of ICMP
  - » additional message types, e.g. "`Packet Too Big`"
  - » multicast group management functions

4-79

http://blogs.salleurl.edu/networking-and-internet-technologies/files/2013/05/IPv6-IPv4coexistence.jpg

CHAPTER 4 IPv6

Transitioning from IPv4 to IPv6

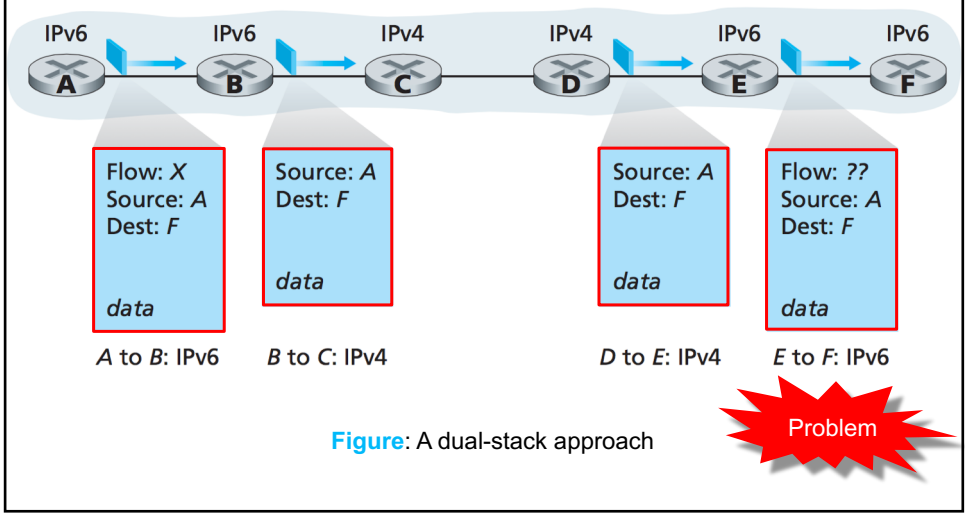


Figure: A dual-stack approach

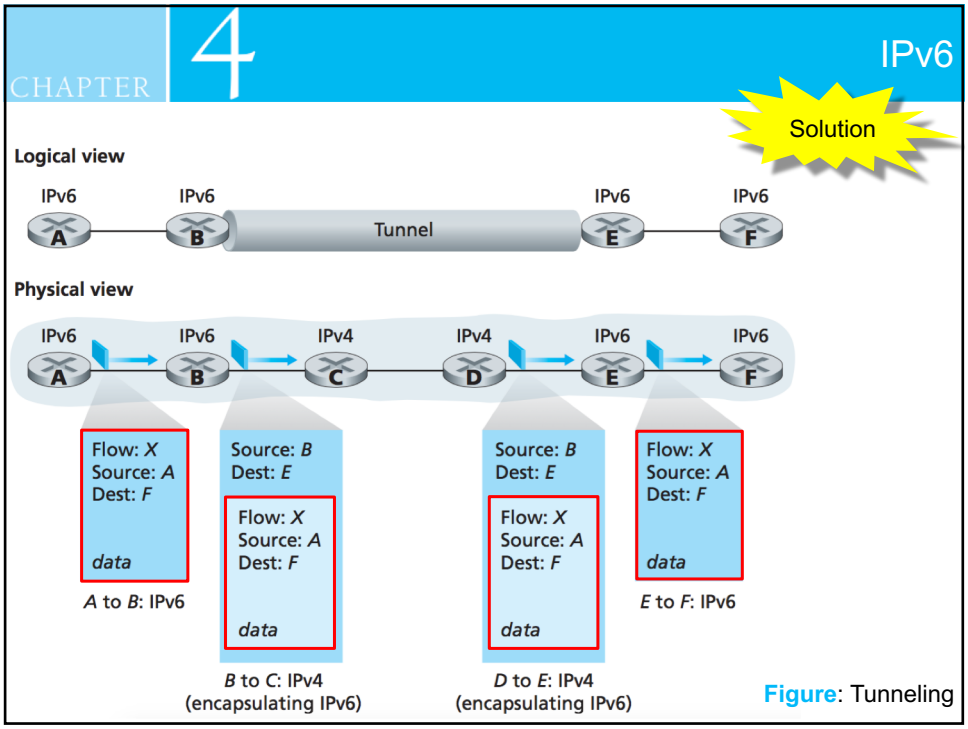


Figure: Tunneling

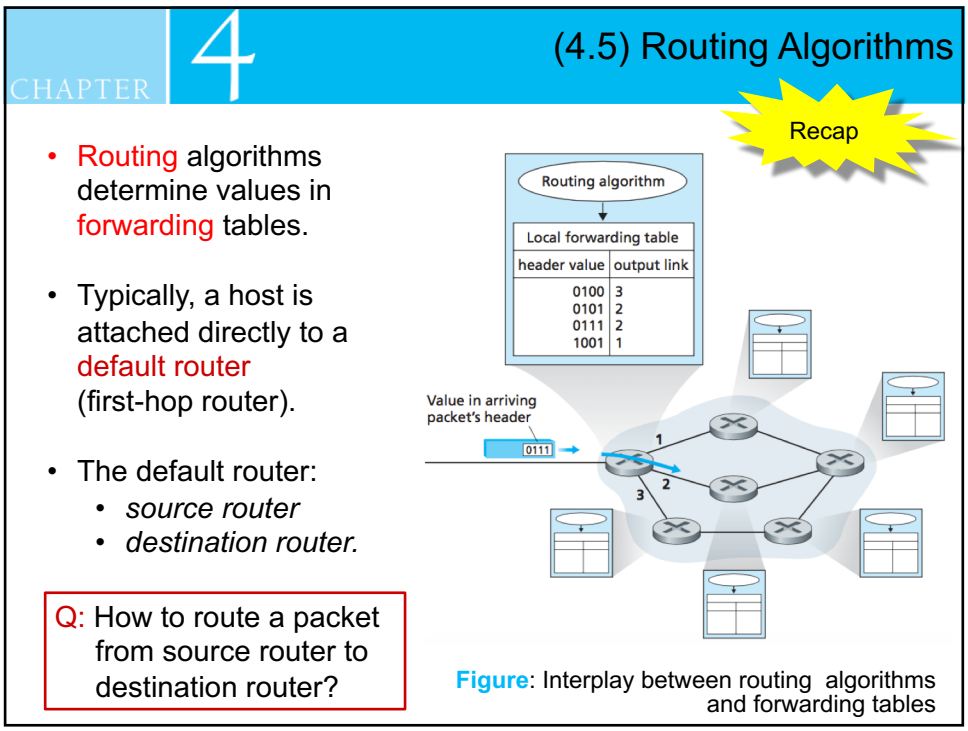
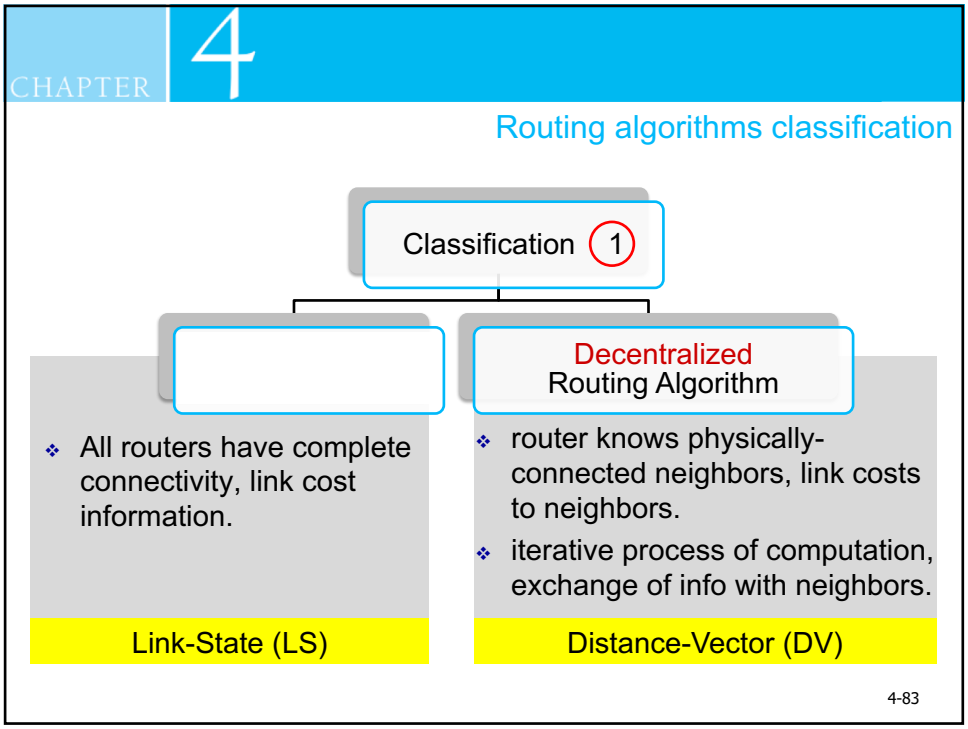
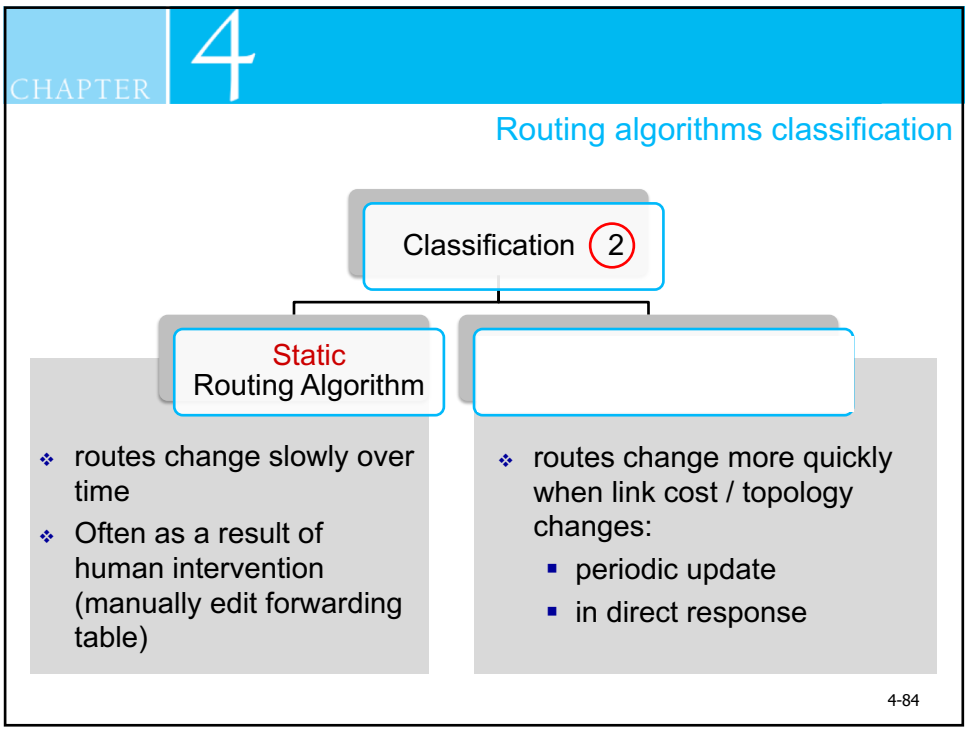


Figure: Interplay between routing algorithms and forwarding tables


CHAPTER 4
Routing algorithms classification
Classification 1
Decentralized
Routing Algorithm
All routers have complete connectivity, link cost information.
router knows physically-connected neighbors, link costs to neighbors.
iterative process of computation, exchange of info with neighbors.
Link-State (LS)
Distance-Vector (DV)
4-83

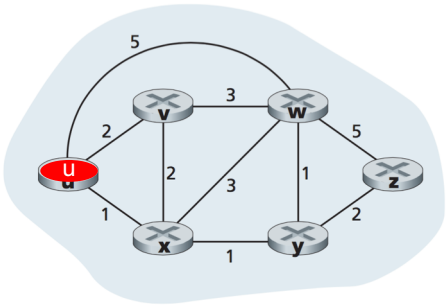

CHAPTER 4
Routing algorithms classification
Classification 2
Static
Routing Algorithm
routes change slowly over time
Often as a result of human intervention (manually edit forwarding table)
routes change more quickly when link cost / topology changes:
periodic update
in direct response
4-84

CHAPTER 4

The Link-State (LS)

Dijkstra's Algorithm

- Network topology, link costs known to all nodes
  - accomplished via "link state broadcast"
  - all nodes have same information



- Computes least cost paths from one node (example: "source" as u) to all other nodes (v, w, x, y and z)
  - gives *forwarding table* for that node u
- Iterative: after k iterations, know least cost path to k destination nodes

4-85

CHAPTER 4

The Link-State (LS)

Dijkstra's Algorithm

```
1  Initialization:
2    N' = {u}
3    for all nodes v
4      if v is a neighbor of u
5        then D(v) = c(u,v)
6      else D(v) = ∞
7
8  Loop
9    find w not in N' such that D(w) is a minimum
10   add w to N'
11   update D(v) for each neighbor v of w and not in N':
12         D(v) = min( D(v), D(w) + c(w,v) )
13   /* new cost to v is either old cost to v or known
14    least path cost to w plus cost from w to v */
15 until N'= N
```

Notation:

$c(x,y)$	: link cost from node x to y; $= \infty$ if not direct neighbors
$D(v)$	: current value of cost of path from source to destination v
$p(v)$	: predecessor node along path from source to v
N'	: set of nodes whose least cost path definitively known

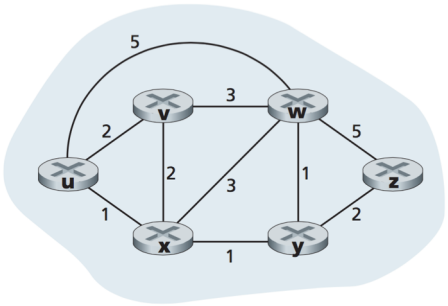
4-86

CHAPTER 4

The Link-State (LS)

Dijkstra's Algorithm

Example:
Routers in nodes *u, v, w, x, y* and *z* have been assigned with link cost value as stated in the diagram.



(a) Construct table least cost paths from node *u* to node *z*
- construct shortest path tree by tracing predecessor nodes

(b) Computes least cost paths from node *u* to node *z*

(c) Produce forwarding table for node *u*

4-87

CHAPTER 4

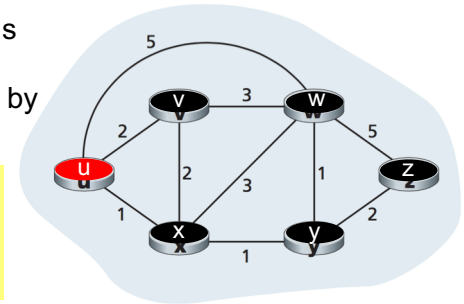
The Link-State (LS)

Dijkstra's Algorithm

Solution:

(a) Construct table least cost paths from node *u* to node *z*
- construct shortest path tree by tracing predecessor nodes

$$
\begin{aligned}
D(v) &= \min(\text{link cost for possible route nodes } p(v)) \\
&= \min(D(v), D(x) + c(x,v)) \\
&= \min\{(2,u),(1+2,x)\} \\
&= \min\{(2,u),(3,x)\} = 2,u
\end{aligned}
$$



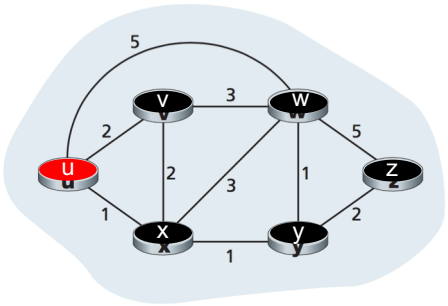
step	*N'*	$D(v),p(v)$	$D(w),p(w)$	$D(x),p(x)$	$D(y),p(y)$	$D(z),p(z)$
0	*u*	2,*u*	5,*u*	**1, *u***	∞	∞
1	*ux*	2,*u*	4,*x*	–	**2, *x***	∞
2	*uxy*	**2, *u***	3,*y*	–	–	4,*y*
3	*uxyv*	–	**3, *y***	–	–	4,*y*
4	*uxyvw*	–	–	–	–	**4, *y***
5	*uxyvwz*	–	–	–	–	–

CHAPTER 4 The Link-State (LS)

Dijkstra's Algorithm

Self-Test

Try to complete the table if router v is chosen instead of router y at step 1.

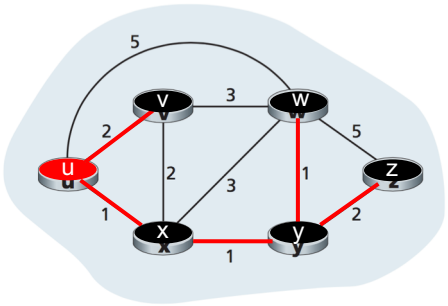


step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x	–	2,x	∞
2						
3						
4						
5						

CHAPTER 4 The Link-State (LS)

Dijkstra's Algorithm

(b) Computes least cost paths from node u to node z

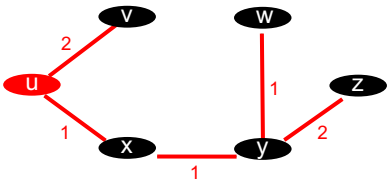


step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1, u	∞	∞
1	ux	2,u	4,x	–	2, x	∞
2	uxy	2, u	3,y	–	–	4,y
3	uxyv	–	3, y	–	–	4,y
4	uxyvw	–	–	–	–	4, y
5	uxyvwz	–	–	–	–	–

CHAPTER 4 The Link-State (LS)

Dijkstra's Algorithm

(b) Computes least cost paths from node u to node z



(c) Produce forwarding table for node u

Destination	Link (next u)	Least cost
v	(u, v)	2
w	(u, x)	3
x	(u, x)	1
y	(u, x)	2
z	(u, x)	4

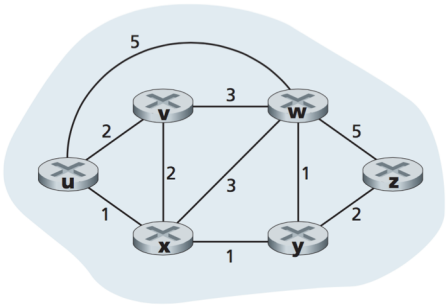
Network Layer 4-92

CHAPTER 4

Exercise 4.5

Dijkstra's Algorithm

Routers in nodes *u, v, w, x, y* and *z* have been assigned with link cost value as stated in the diagram.



(a) Construct table least cost paths from node *z* to node *u*
- construct shortest path tree by tracing predecessor nodes

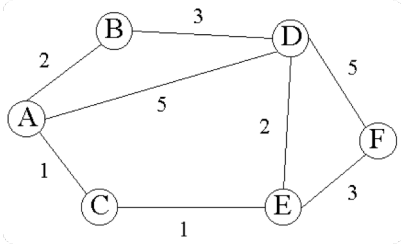
(b) Computes least cost paths from node *z* to node *u*

(c) Produce forwarding table for node *z*

CHAPTER 4

Exercise 4.6

Routers in nodes *A, B, C, D, E* and *F* have been assigned with link cost value as stated in the diagram.



(a) Construct table least cost paths from node *A* to node *F* using Dijkstra's algorithm

(b) Draw least cost paths from node *A* to node *F*

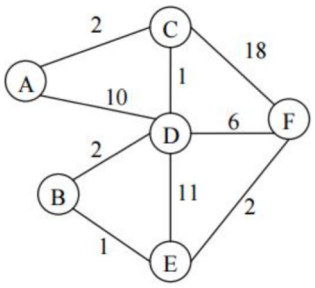
(c) Produce forwarding table for node *A*

http://www4.ncsu.edu/~chou/course/Diagrams/distance-vector.gif

CHAPTER 4

Exercise 4.7

Routers in nodes *A, B, C, D, E,* and *F* have been assigned with link cost value as stated in the diagram.



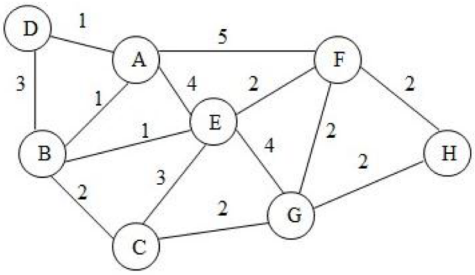
(a) Construct table least cost paths from node *D* to node *H* using Dijkstra's algorithm
(b) Draw least cost paths from node *D* to node *H*
(c) Produce forwarding table for node *D*

https://i.stack.imgur.com/bcz2a.jpg

CHAPTER 4

Exercise 4.8

Routers in nodes *A, B, C, D, E, F, G* and *H* have been assigned with link cost value as stated in the diagram.



(a) Construct table least cost paths from node *D* to node *H* using Dijkstra's algorithm
(b) Draw least cost paths from node *D* to node *H*
(c) Produce forwarding table for node *D*

http://s3.amazonaws.com/answer-board-image/6bfef7b5-05b9-4210-afb5-712510e57a37.jpeg

CHAPTER 4

The Distance Vector (DV)

Bellman-Ford

Key idea:

- from time-to-time, each node sends its own distance vector estimate to neighbors
- when x receives new DV estimate from neighbor, it updates its own DV using B-F equation:

Bellman-Ford equation (dynamic programming)

let $d_x(y)$ → cost of least-cost path from x to y

then $d_x(y) = \min_v\{c(x,v) + d_v(y)\}$ for each node $y \in N$

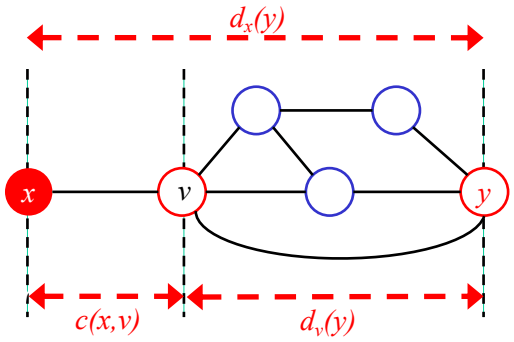
min taken over all neighbors v of x | cost to neighbor v | cost from neighbor v to destination y

4-97

CHAPTER 4

The Distance Vector (DV)

Bellman-Ford



$$d_x(y) = \min_v\{c(x,v) + d_v(y)\}$$

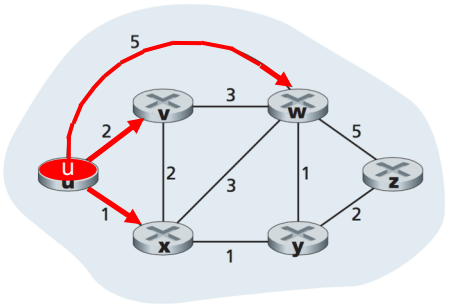
- under minor, natural conditions, the estimate $d_x(y)$ *converge to the actual least cost* $d_x(y)$

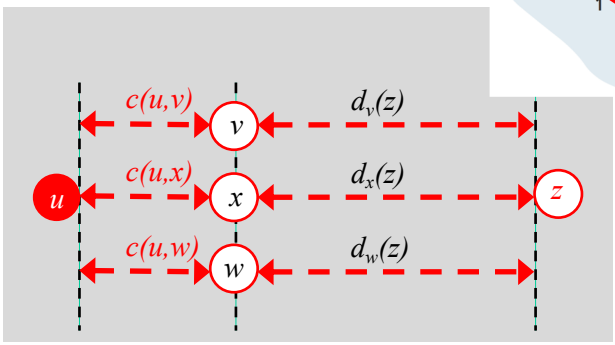
4-98

CHAPTER 4 The Distance Vector (DV)

Bellman-Ford

Example:
Neighbor to u are v, x, and w;



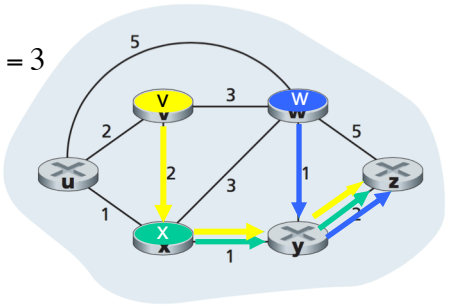


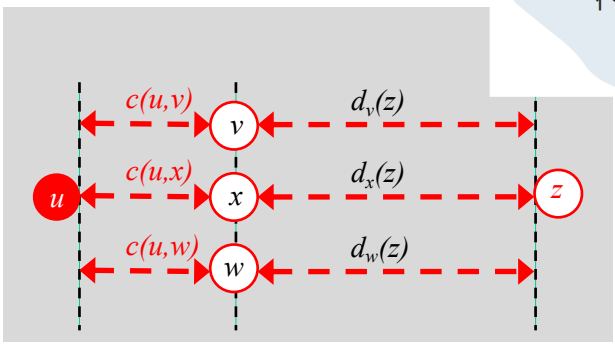
4-99

CHAPTER 4 The Distance Vector (DV)

Bellman-Ford

Clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$





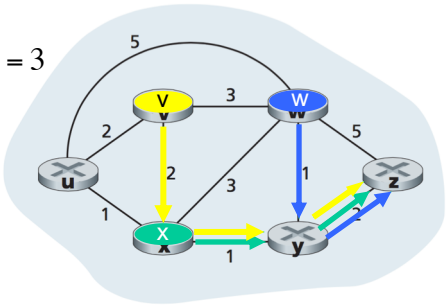
4-100

CHAPTER 4

The Distance Vector (DV)

Bellman-Ford

Clearly, $d_v(z) = 5, \quad d_x(z) = 3, \quad d_w(z) = 3$



B-F equation says:

$$
\begin{aligned}
d_u(z) &= \min\{c(u,v) + d_v(z), \\
&\qquad c(u,x) + d_x(z), \\
&\qquad c(u,w) + d_w(z)\} \\
&= \min\{2+5, \\
&\qquad 1+3, \\
&\qquad 5+3\} \\
&= \min\{7,4,8\} = 4
\end{aligned}
$$

4-101

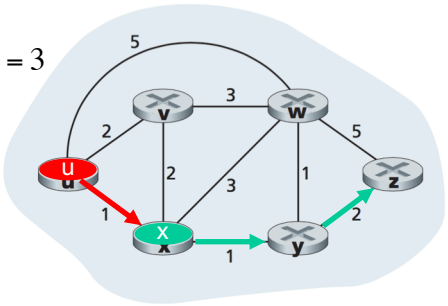
CHAPTER 4

The Distance Vector (DV)

Bellman-Ford

Example:
Neighbor to *u* are *x*, *v*, and *w*;

Clearly, $d_v(z) = 5, \quad d_x(z) = 3, \quad d_w(z) = 3$



B-F equation says:

$$
\begin{aligned}
d_u(z) &= \min\{c(u,v) + d_v(z), \\
&\qquad c(u,x) + d_x(z), \\
&\qquad c(u,w) + d_w(z)\} \\
&= \min\{2+5, \\
&\qquad 1+3, \\
&\qquad 5+3\} \\
&= \min\{7,4,8\} = 4
\end{aligned}
$$

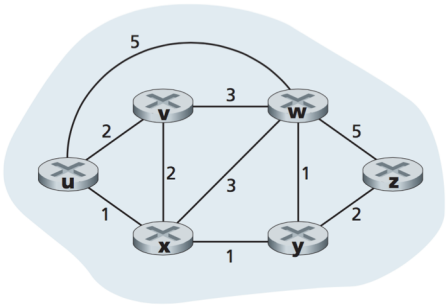
Node achieving minimum is next hop in shortest path, used in forwarding table.

4-102

CHAPTER 4

Exercise 4.9

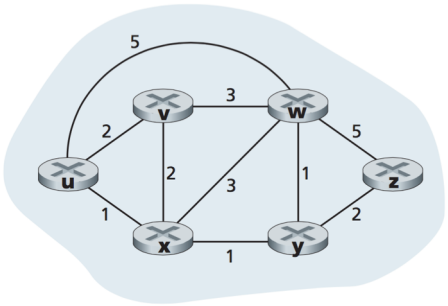
Calculate the cost from z to u by using the Bellman-Ford algorithm.



CHAPTER 4

Exercise 4.10

Calculate the cost from x to w by using the Bellman-Ford algorithm.



CHAPTER 4

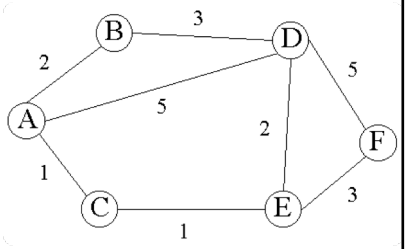
Exercise 4.11

Using the Bellman-Ford algorithm to calculate the cost from:

(a) *A* to *F*

(b) *B* to *E*

(c) *F* to *B*



http://www4.ncsu.edu/~chou/course/Diagrams/distance-vector.gif

CHAPTER 4

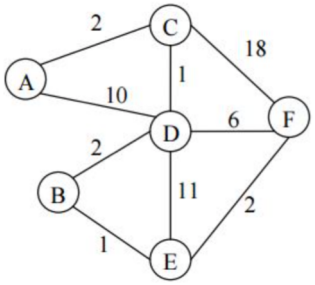
Exercise 4.12

Using the Bellman-Ford algorithm to calculate the cost from:

(a) *A* to *F*

(b) *B* to *C*

(c) *F* to *B*



https://i.stack.imgur.com/bcz2a.jpg

CHAPTER 4

Exercise 4.13

Using the Bellman-Ford algorithm to calculate the cost from:

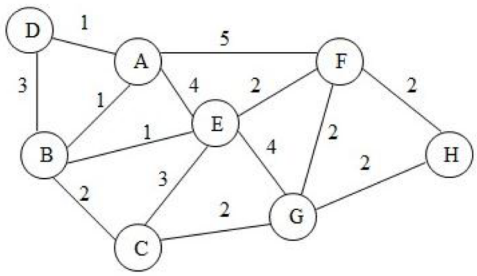
(a) *D* to *E*

(b) *D* to *H*

(c) *H* to *B*

(d) *F* to *D*

(e) *A* to *E*



http://s3.amazonaws.com/answer-board-image/6bfef7b5-05b9-4210-afb5-712510e57a37.jpeg

CHAPTER 4

The Distance Vector (DV)

Computations & Update Generation

Iterative, Asynchronous:

Each local iteration caused by:

- local link cost change
- DV update message from neighbor

Distributed:

Each node notifies neighbors *only* when its DV changes

- neighbors then notify their neighbors if necessary

Each node:

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

↓

Recompute estimates

↓

if DV to any destination has changed, *Notify* neighbors

4-108

CHAPTER 4

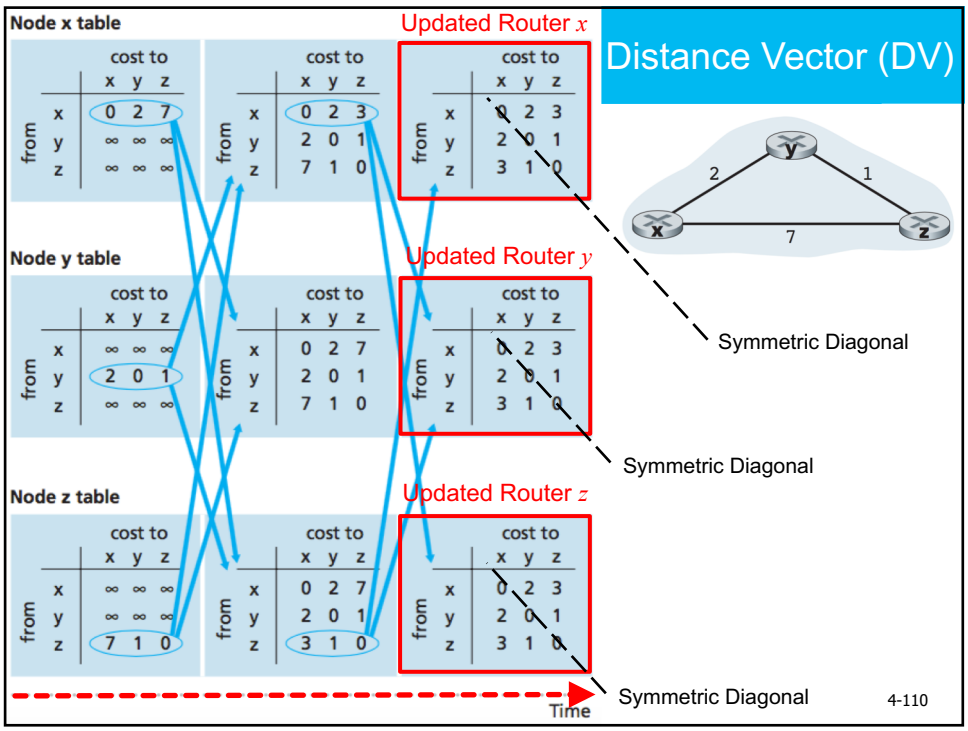
The Distance Vector (DV)

```
1  Initialization:
2     for all destinations y in N:
3        Dx(y) = c(x,y)    /* if y is not a neighbor then c(x,y) = ∞ */
4     for each neighbor w
5        Dw(y) = ? for all destinations y in N
6     for each neighbor w
7        send distance vector Dx = [Dx(y): y in N] to w
8
9  loop
10    wait (until I see a link cost change to some neighbor w or
11          until I receive a distance vector from some neighbor w)
12
13    for each y in N:
14       Dx(y) = minv{c(x,v) + Dv(y)}
15
16    if Dx(y) changed for any destination y
17       send distance vector Dx = [Dx(y): y in N] to all neighbors
18
19 forever
```

Figure: DV algorithm at each node, *x*

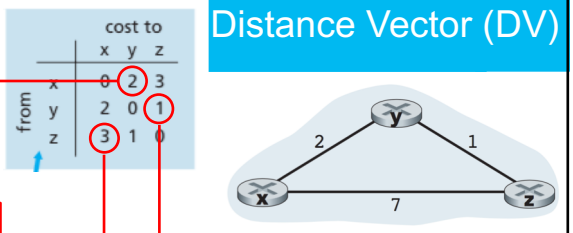
4-109

Distance Vector (DV)



4-110

Distance Vector (DV)



from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

Cost from x to y:

$$D_x(y) = \min\{c(x,y) + D_y(y), \; c(x,z) + D_z(y)\}$$
$$= \min\{2+0, 7+1\}$$
$$= \min\{2,8\} = 2$$

Cost from z to x:

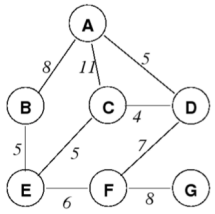
$$D_z(x) = \min\{c(z,x) + D_x(x), \; c(z,y) + D_y(x)\}$$
$$= \min\{7+0, 1+2\}$$
$$= \min\{7,3\} = 3$$

Cost from y to z:

CHAPTER 4

Exercise 4.14

Answer the questions based on the figure and the table with some values that constructed by Bellman-Ford algorithm.



(a) Calculate the cost at w, x and y.
(b) Proof the cost in the blue shaded cells.
(c) Calculate the cost at z. Justify your answer.

from \ cost to	A	B	C	D	E	F	G
A					w		
B				x			
C	y						
D		13					
E				9			
F			11				
G	z						

http://homepages.herts.ac.uk/~comqrgd/docs/network-notes/network-notes-img31.png

CHAPTER 4

Hierarchical Routing

Our routing study thus far - idealization
- all routers identical
- network "flat"

… *not* true in practice

Reason 1

Scale: with 600 million destinations:
- can't store all destination's in routing tables!
- routing table exchange would swamp (flood) links!

Reason 2

Administrative autonomy
- internet = network of networks
- each network admin may want to control routing in its own network

4-113

CHAPTER 4

Hierarchical Routing

(Solution)

- Aggregate routers into regions, "Autonomous Systems (AS)"
- Routers in same AS run same routing protocol
  - "intra-AS" routing protocol
  - routers in different AS can run different intra-AS routing protocol

Gateway router:
- at "edge" of its own AS
- Has link to router in another AS

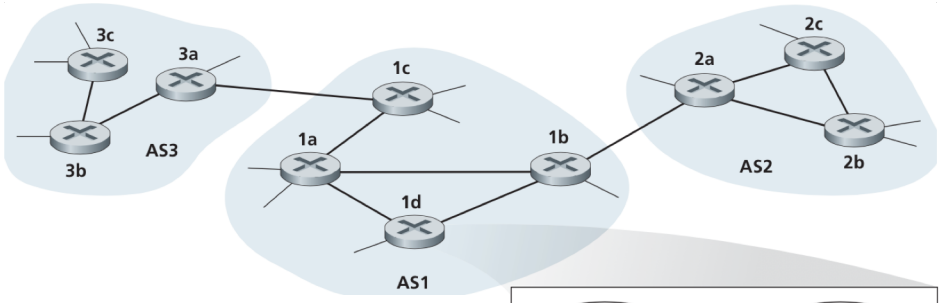
Q: How to Update Forwarding Table Content in "Autonomous System (AS)" router ?

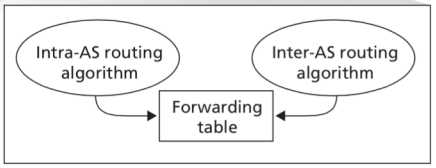
4-114

CHAPTER 4

Hierarchical Routing

Interconnected ASs





- Forwarding table configured by both intra- and inter-AS routing algorithm:
  - intra-AS sets entries for internal destinations
  - inter-AS & intra-AS sets entries for external destinations

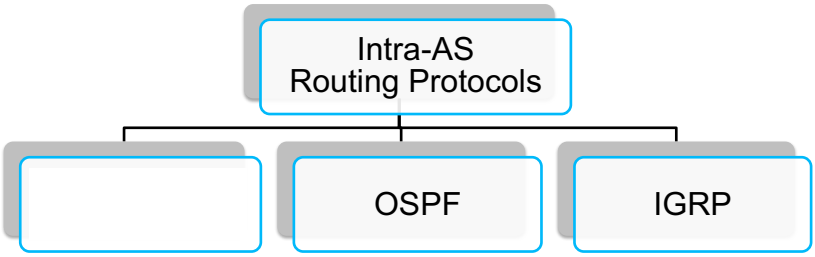
4-115

CHAPTER 4

(4.6) Routing in the Internet

Intra-AS Routing

- also known as *Interior Gateway Protocols (IGP)*
- Most common intra-AS routing protocols:



(Distance Vector)

OSPF: Open Shortest Part First (Link State)

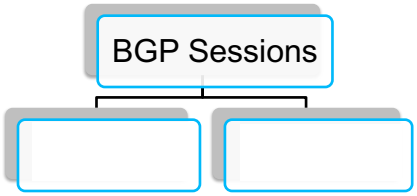
IGRP: Interior Gateway Routing Protocol (Distance Vector with Link State property)

4-116

CHAPTER 4

Inter-AS Routing

- BGP (Border Gateway Protocol): the *de facto* inter-domain routing protocol. It is based on "path vector routing " (different from Distance Vector (DV) and Link-State (LS))
  - "glue that holds the Internet together"
- BGP provides each AS a means to determine "good" routes to other networks based on reachability information and policy



Obtain subnet reachability information from neighboring ASs.

Propagate reachability information to all AS-internal routers.

CHAPTER 4

Inter-AS Routing

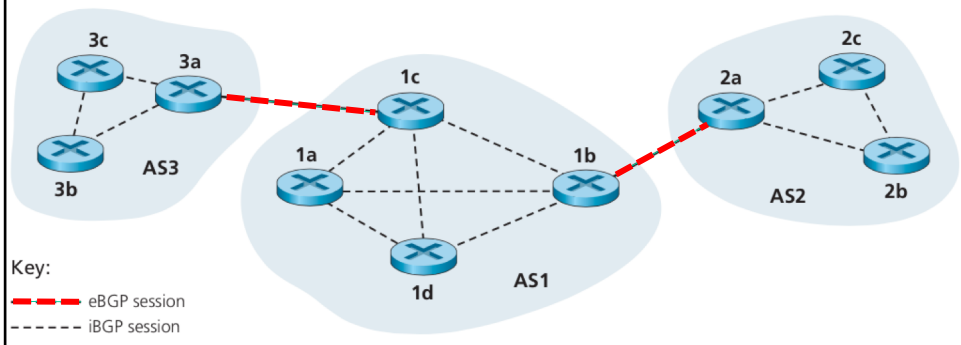


Figure: eBGP and iBGP sessions

4-118

CHAPTER 4

(4.7) Broadcast & Multicast Routing

Intra-AS Routing

- Deliver packets from source to all other nodes
- Source duplication is inefficient:

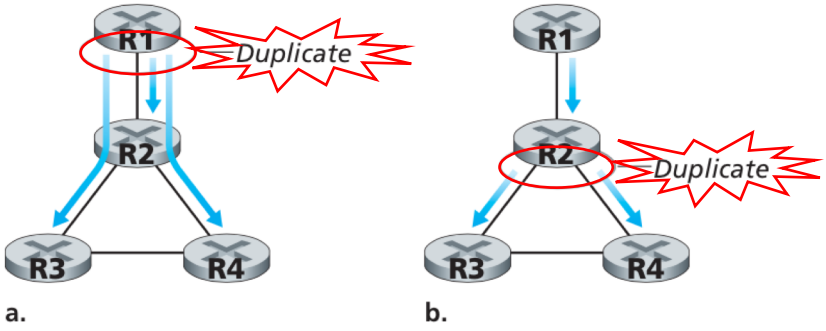


Figure: (a) Source-duplication versus (b) in-network duplication

4-119

CHAPTER 4

Intra-AS Routing

Source duplication: How does source determine recipient addresses? (through in-network duplication)

Uncontrolled Flooding:
when node receives broadcast packet, sends copy to all neighbors
- problems: cycles & broadcast storm

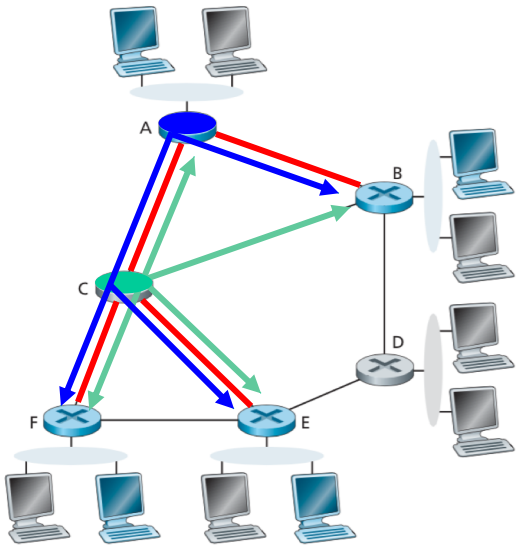
Controlled Flooding:
node only broadcasts packet if it has not broadcast same packet before
- node keeps track of packet *id* already broadcasted
- Or Reverse Path Forwarding (RPF): only forward packet if it arrived on shortest path between node and source

Spanning tree:
- no redundant packets received by any node

4-120

CHAPTER 4 Multicast Routing Algorithms

Problem statement



Goal: Find a tree (or trees) connecting routers that having local *multicast* group members

- *Tree:* not all paths between routers used

Shared-tree: same tree used by all group members

Source-based: different tree from each sender to receivers

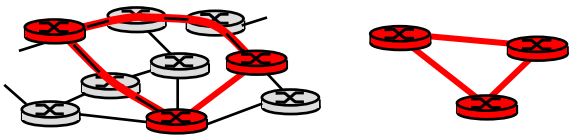
Figure: Multicast hosts, their attached routers, and other routers

4-121

CHAPTER 4 Multicast Routing Algorithms

Tunneling

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



Physical topology Logical topology

- multicast datagram encapsulated inside "normal" (non-multicast-addressed) datagram
- normal IP datagram sent thru "tunnel" via regular IP unicast to receiving multicast router (recall IPv6 inside IPv4 tunneling)
- receiving multicast router un-encapsulates to get multicast datagram

4-122

CHAPTER 4

Summary

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, DHCP, ICMP, NAT, 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

- Understand principles behind network layer services:
  - network layer service models, forwarding versus routing how a router works, routing (path selection), broadcast, multicast
- Instantiation, implementation in the Internet

4-123