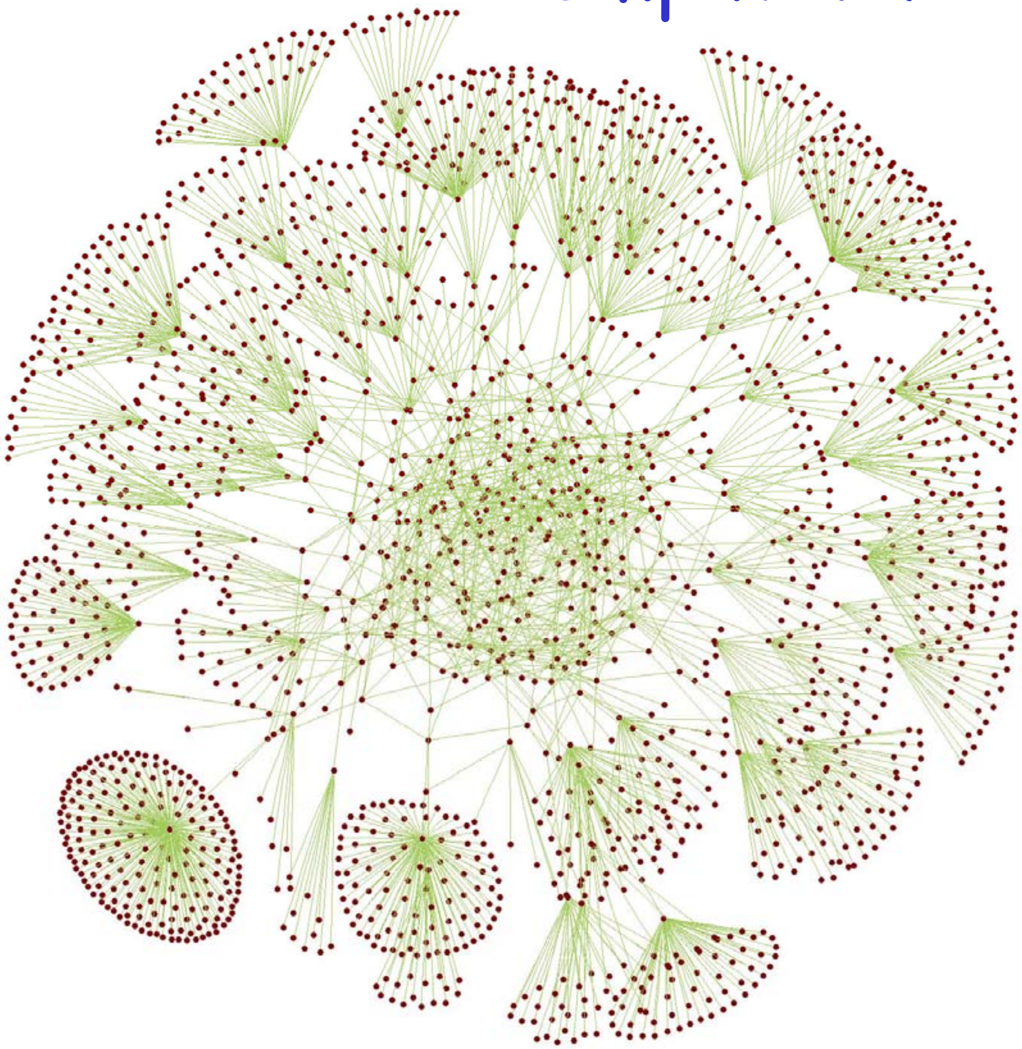
CSCE 560 Introduction to Computer Networking



Dr. Barry Mullins
AFIT/ENG
Bldg 642, Room 209
255-3636 x7979

Chapter 5: Outline

- 5.1 Introduction
- 5.2 Routing protocols
  - Link state
  - Distance vector
- 5.3 Intra-As routing in the Internet: OSPF
- 5.4 Routing among the ISPs: BGP
- 5.5 The SDN control plane
- 5.6 ICMP: The Internet Control Message Protocol
- 5.7 Network management and SNMP

Network-layer Functions

Recall: two network-layer functions

- Forwarding:
  - Move packets from router's input to appropriate router output

Data plane
Chap 4

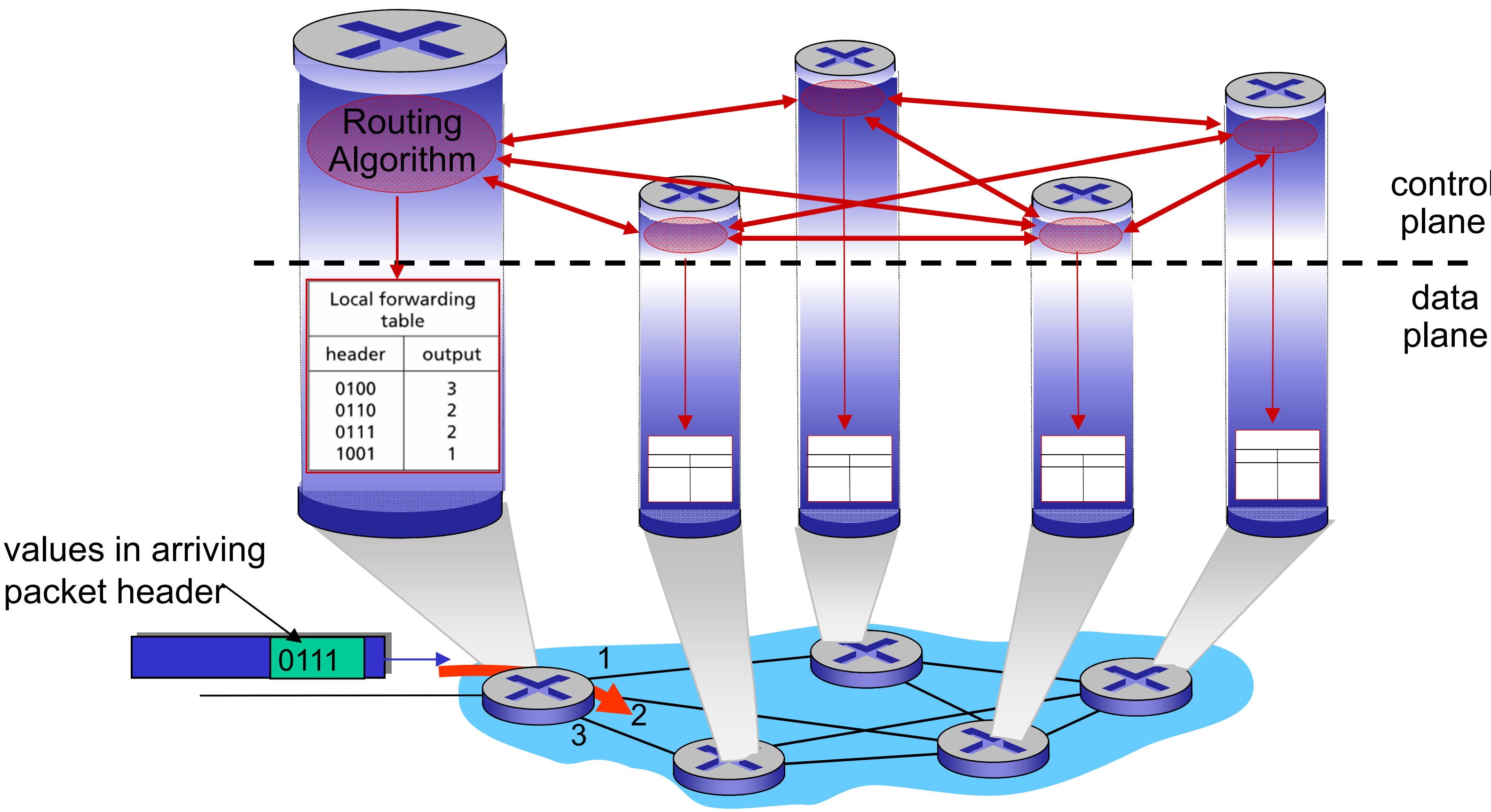
- Routing:
  - Determine route taken by packets from source to destination

Control plane
Chap 5

- Two approaches to structuring network control plane:
  - Traditional: Per-router control
  - SDN: Logically centralized control

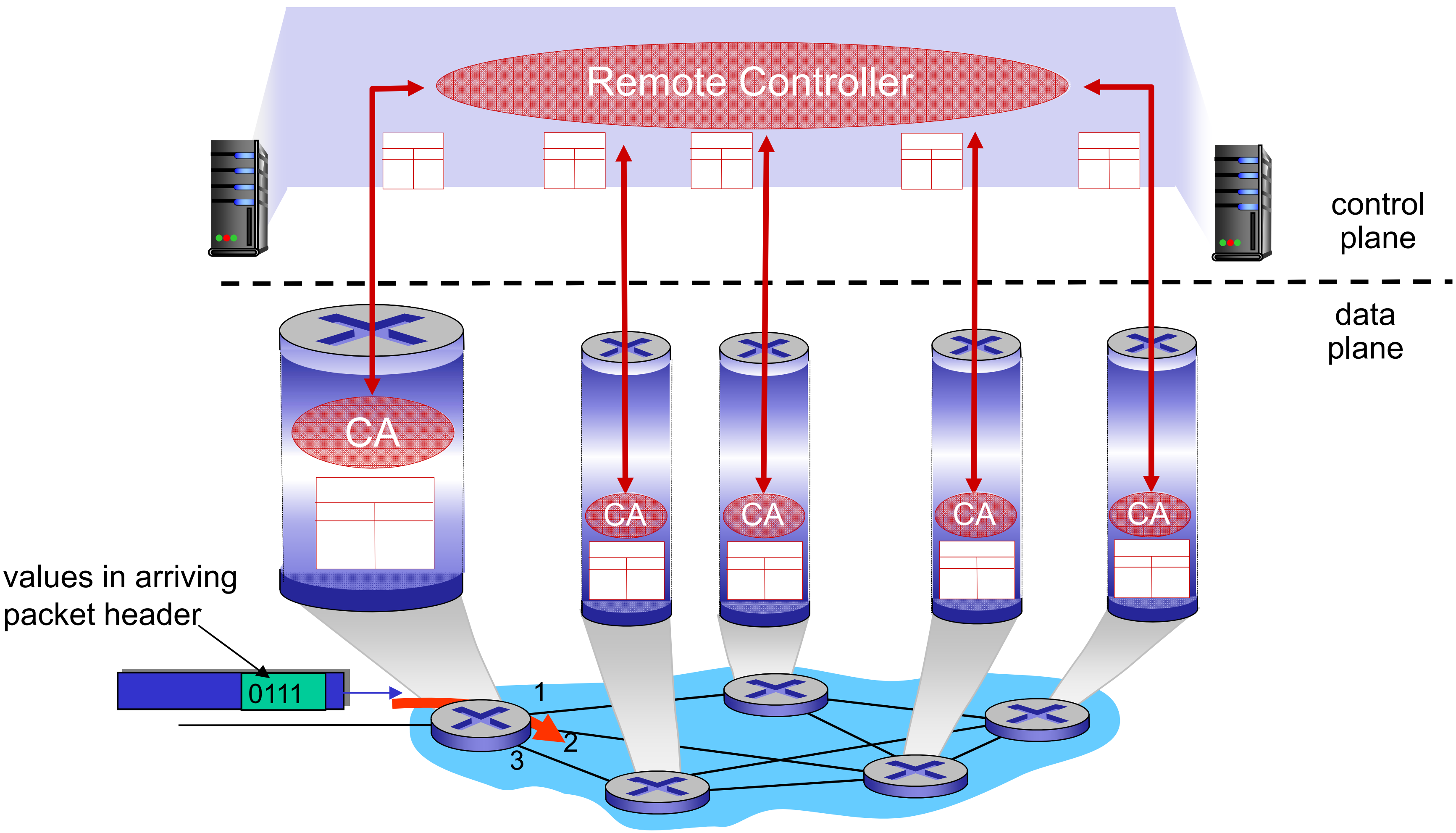
Per-Router Control Plane

- Individual routing algorithm components in each and every router interact in the control plane to compute forwarding tables



Logically Centralized Control Plane (SDN)

- A distinct (typically remote) controller interacts with local control agents (CAs)

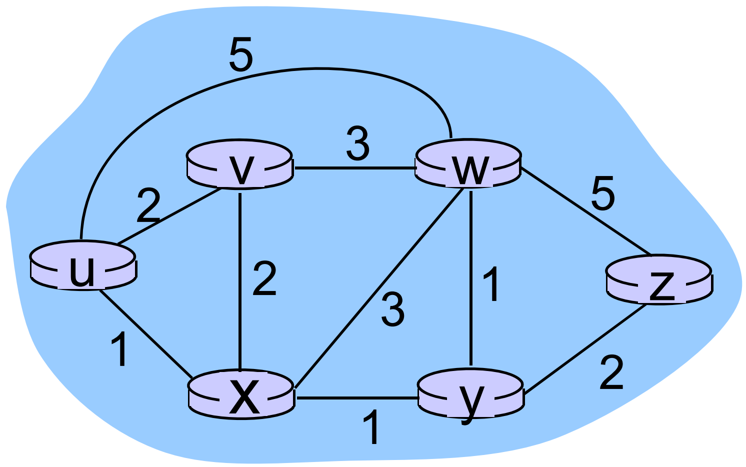


Chapter 5: Outline

- 5.1 Introduction
- 5.2 Routing protocols
  - Link state
  - Distance vector
- 5.3 Intra-As routing in the Internet: OSPF
- 5.4 Routing among the ISPs: BGP
- 5.5 The SDN control plane
- 5.6 ICMP: The Internet Control Message Protocol
- 5.7 Network management and SNMP

Graph Abstraction

- Abstract network as a graph
  - Routers and end systems are nodes
  - Links and channels are edges



Graph: G = (N,E)

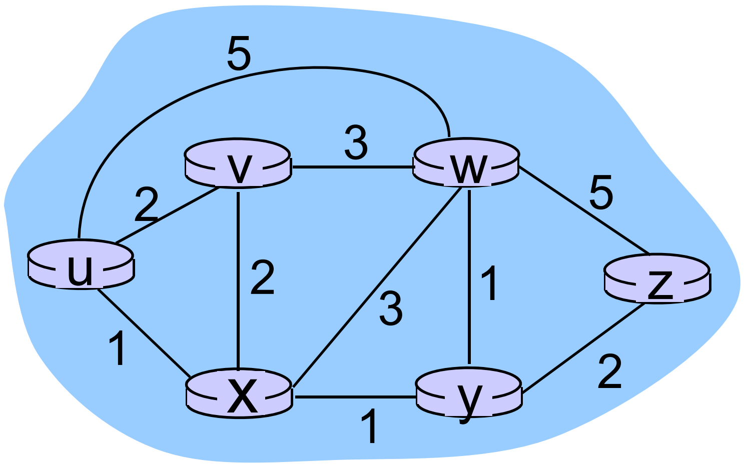
N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

Remark: Graph abstraction is useful in other network contexts

Example: P2P, where N is set of peers and E is set of TCP connections

Graph Abstraction: Costs



- $c(x,x')$ = cost of link (x,x')

  - e.g., $c(w,z) = 5$

- Cost could always be 1, or inversely related to bandwidth

Cost of path $(x_1, x_2, x_3, \ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Question: What's the least-cost path between u and z ?

Routing algorithm: algorithm that finds least-cost path

Routing Algorithm Classification

Static or dynamic?

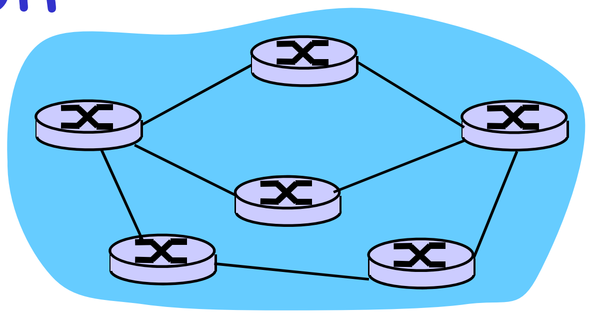
Static:

- Routes change very slowly over time
- A human updates the forwarding table ☺

Dynamic:

- Routes change more quickly
  - Periodic update
  - In response to link cost changes

Routing Algorithm Classification



Global or decentralized information?

Global:

- All routers have complete global topology and link cost info
- Router has a map of the entire network before algorithm is run
- "**Link state**" algorithms
  - Flood (broadcast) routing (link) information to all nodes
  - Each router sends only the portion of the forwarding table that describes the state of its own links
  - Each router builds a picture of the entire network in its routing tables

Chapter 5: Outline

- 5.1 Introduction
- 5.2 Routing protocols
  - Link state
  - Distance vector
- 5.3 Intra-AS routing in the Internet: OSPF
- 5.4 Routing among the ISPs: BGP
- 5.5 The SDN control plane
- 5.6 ICMP: The Internet Control Message Protocol
- 5.7 Network management and SNMP

A Link-State Routing Algorithm

Dijkstra's algorithm

- Global net topology and link costs known to all nodes
  - All nodes have same info
  - Accomplished via "link state broadcast"
- Computes least cost paths from itself ("source") to all other nodes
  - Generates forwarding table for itself
- Iterative: after k iterations, know least cost path to k destinations

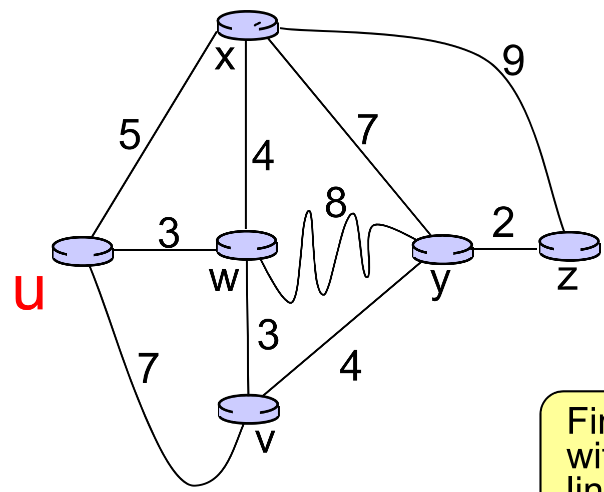
Notation:

- c(x,y): link cost from node x to y
  - ∞ if not direct neighbors
- D(v): current value of cost of path from source to dest. v
- p(v): predecessor node along path from source to v
- N': set of nodes whose least cost path definitively known
  - This set grows until all nodes are included

Dijkstra's Algorithm for Source Node u

Initialization:
N' = {u}
for all nodes b
  if b adjacent to u
    then D(b) = c(u,b)
  else D(b) = ∞

Loop
find e not in N' such that D(e) is a minimum
add e to N'
update D(b) for all b adjacent to e and not in N' :
    D(b) = min(D(b), D(e) + c(e,b))
until all nodes in N'



Find neighbor with smallest link cost

New cost to b is either
- old cost from u to b

or
- known shortest path cost from u to e plus cost from e to b

Dijkstra's Algorithm: Example

Step	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

Loop

find e not in N' such that D(e) is a minimum

add e to N'

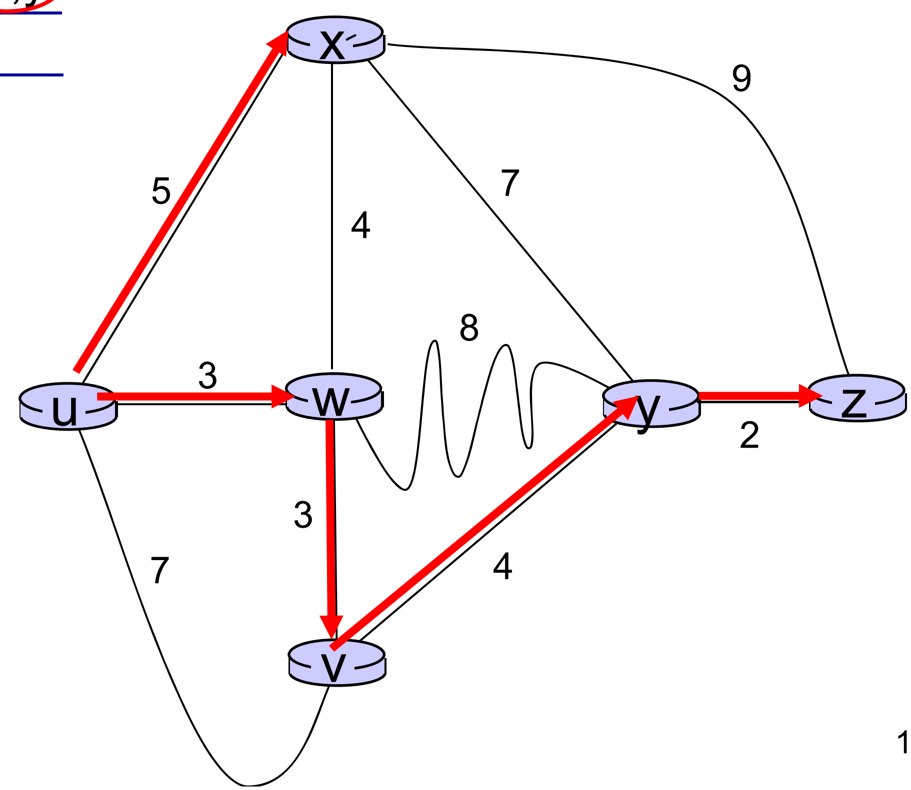
update D(b) for all b adjacent to e and not in N' :

D(b) = min(D(b), D(e) + c(e,b))

until all nodes in N'

Notes:

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



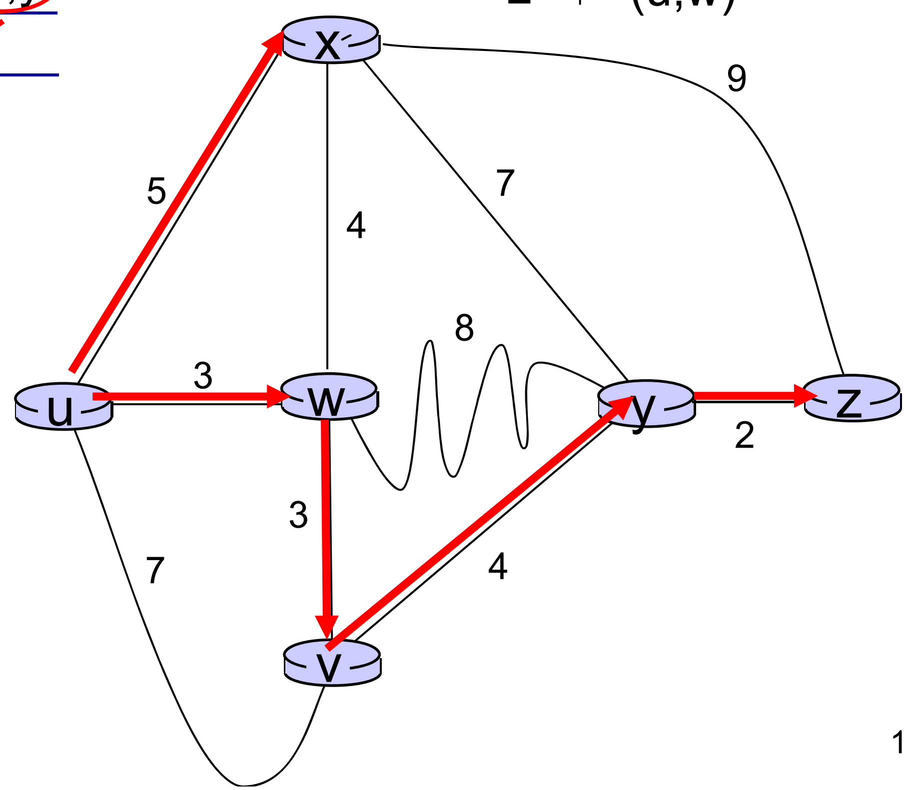
Dijkstra's Algorithm: Example

Step	N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					

Resulting forwarding table in u:

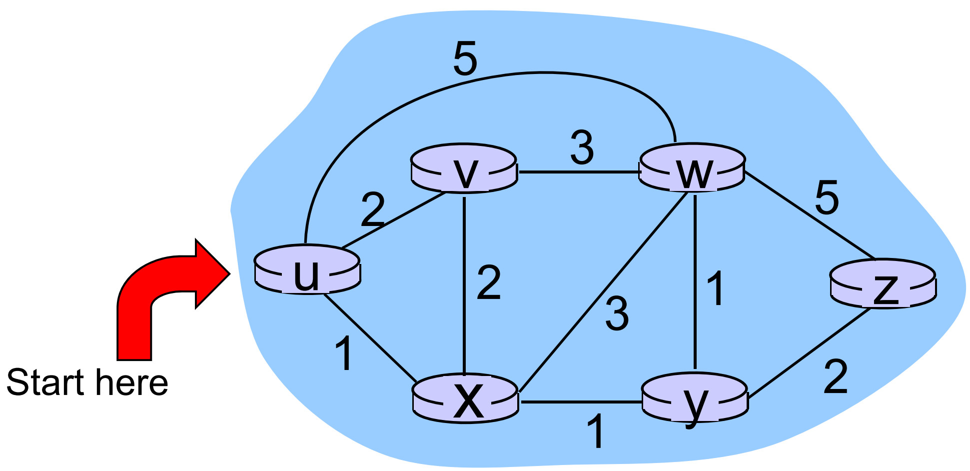
Destination	Link
v	(u,w)
x	(u,x)
y	(u,w)
w	(u,w)
z	(u,w)

Follow the predecessor nodes to discover the appropriate links



Dijkstra's Algorithm: Another Example

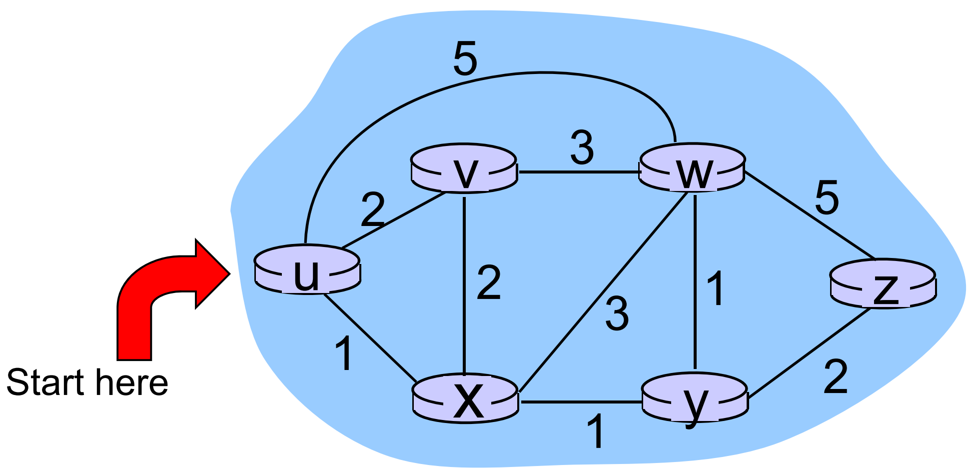
Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞



Initialization:
```
N' = {u}
for all nodes b
  if b adjacent to u
      then D(b) = c(u,b)
  else D(b) = ∞
```

Dijkstra's Algorithm: Another Example

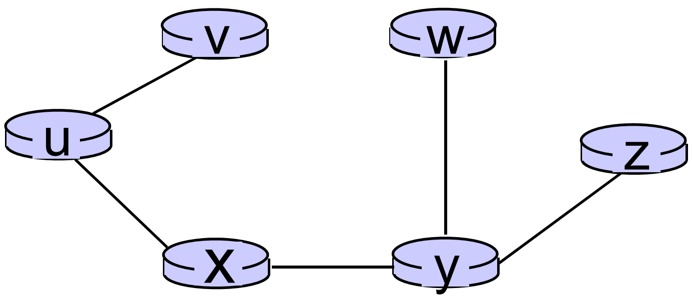
Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					



Loop
find e not in N' such that D(e) is a minimum
add e to N'
update D(b) for all b adjacent to e and not in N' :
D(b) = min(D(b), D(e) + c(e,b))
until all nodes in N'

Dijkstra's Algorithm: Resulting Table

Resulting shortest-path tree from u:



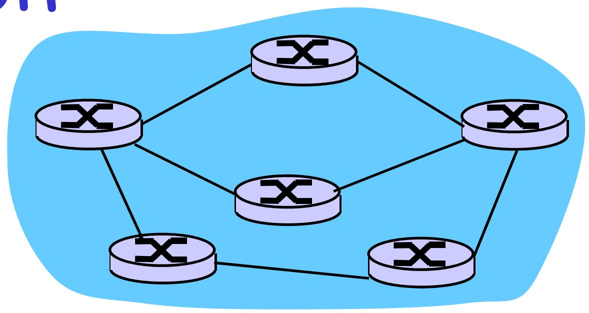
Resulting forwarding table in u:

Destination	Link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

Chapter 5: Outline

- 5.1 Introduction
- 5.2 Routing protocols
  - Link state
  - Distance vector
- 5.3 Intra-As routing in the Internet: OSPF
- 5.4 Routing among the ISPs: BGP
- 5.5 The SDN control plane
- 5.6 ICMP: The Internet Control Message Protocol
- 5.7 Network management and SNMP

Routing Algorithm Classification



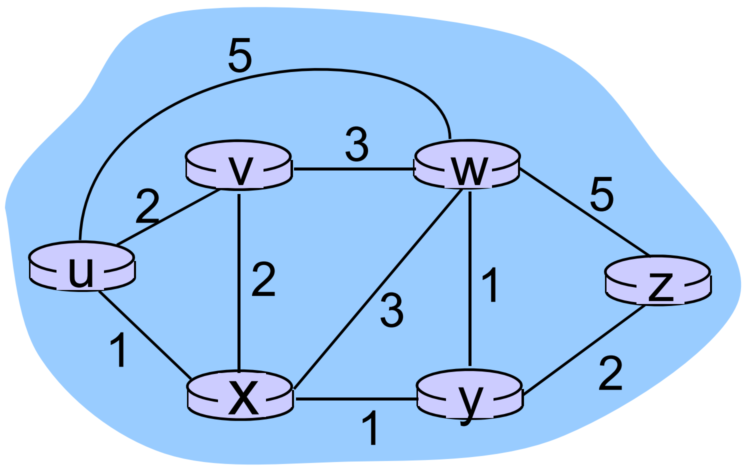
Global or decentralized information?

Decentralized:

- Router knows physically-connected neighbors and link costs to neighbors
- Iterative, distributed process of computation by exchanging info with neighbors
- "**Distance vector**" algorithms
  - Each router sends all or some portion of its routing table only to its neighbors

Bellman-Ford Equation

Define $d_x(y)$:= <u>actual</u> cost of least-cost **path** from x to y



$d_u(z) = \min_a \{ c(u,a) + d_a(z) \}$

Known: $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

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

Node that achieves minimum is next hop in shortest path ➜ add to forwarding table

Distance Vector Algorithm

- $D_x(y)$ = estimate of least cost from x to y
- Distance vector ($\mathbf{D}_x$)
  - Cost estimates from x to all other nodes y in N

- Node x maintains the following data
  1. Cost to each neighbor v: $c(x,v)$
  2. Its distance vector (path costs to nodes that x knows about)
     - $\mathbf{D}_x = [D_x(y): y \in N]$
  3. Its neighbors' distance vectors
     - $\mathbf{D}_v = [D_v(y): y \in N]$

Distance Vector Algorithm

Basic idea:

- Each node periodically sends its own distance vector (DV) estimate to neighbors
- When a node x receives new DV estimate from neighbor, it updates its own DV using B-F equation:

$$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\} \quad \text{for each node } y \in N$$

- Node x sends its DV to its neighbors if the DV changed
- Under natural conditions, the estimate $D_x(y)$ converges to the actual least cost $d_x(y)$

Distance Vector Algorithm

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 msg from neighbor)

↓

recompute estimates

↓

if DV to any dest has changed, *notify* neighbors

Distance Vector Algorithm

At all nodes, x:

Initialization:

for all destinations y in N:

$D_x(y) = c(x,y)$ /* if y is not a neighbor then c(x,y) = infinity */

for each neighbor w

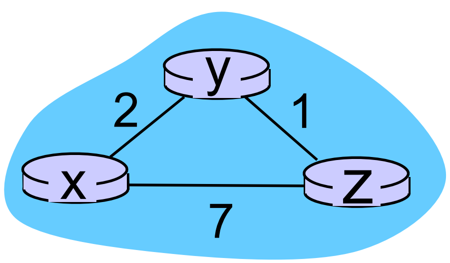
$D_w(y)$ = infinity for all destinations y in N

for each neighbor w

send distance vector $\mathbf{D_x} = [D_x(y)$: y in N] to w

Distance vector →

		cost to		
		x	y	z
from	x	0	2	7
	y	∞	∞	∞
	z	∞	∞	∞



Distance Vector Algorithm

loop
  wait (until I see a link cost change to neighbor w
    or until I receive update from neighbor w)
  For each y in N:
    $D_x(y) = \min_v\{c(x,v) + D_v(y)\}$
  If $D_x(y)$ changed for any destination y
    send distance vector $D_x = [D_x(y)\text{: y in N}]$ to all neighbors
forever

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

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

node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

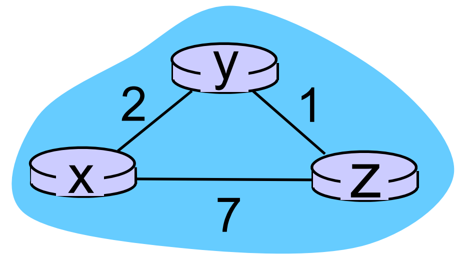
from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

node y table

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

node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0



time

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

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

node x table

from \ cost to	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

from \ cost to	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

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

node y table

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

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

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

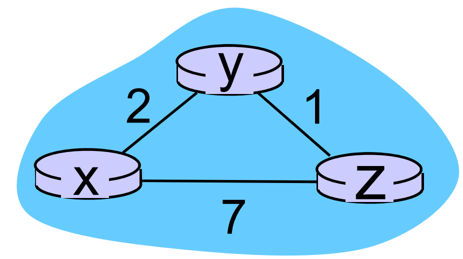
node z table

from \ cost to	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

from \ cost to	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

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

time



Resulting forwarding table in x:

Destination	Link
y	(x,y)
z	(x,y)

Chapter 5: Outline

- 5.1 Introduction
- 5.2 Routing protocols
  - Link state
  - Distance vector
- 5.3 Intra-AS routing in the Internet: OSPF
- 5.4 Routing among the ISPs: BGP
- 5.5 The SDN control plane
- 5.6 ICMP: The Internet Control Message Protocol
- 5.7 Network management and SNMP

Making Routing Scalable

Our routing study thus far → idealization

- All routers identical
- Network "flat"

  … not true in practice

Administrative autonomy

- Internet = network of networks
- Each network admin may want to control routing in own network

Scale:

With billions of destinations:

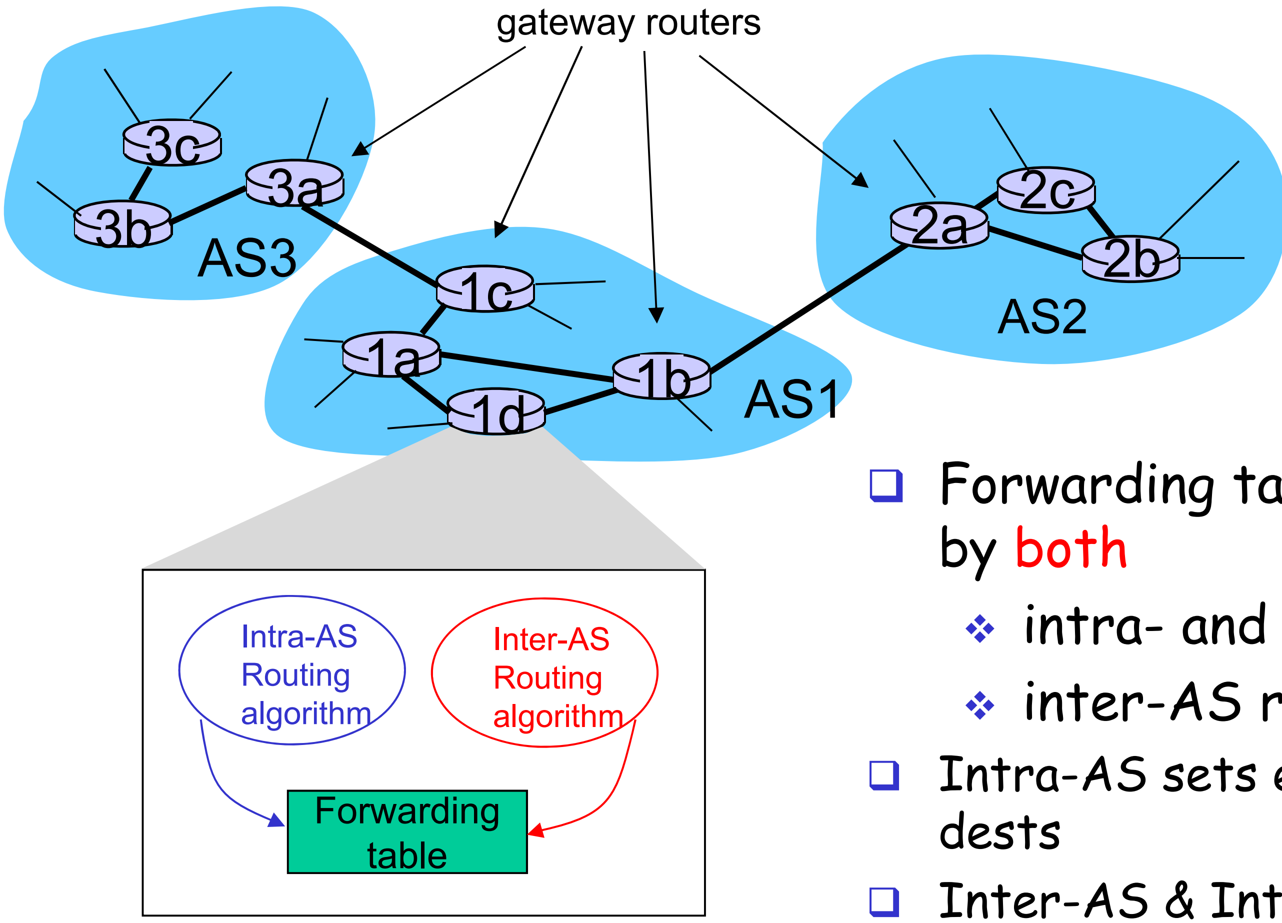
- Can't store all dest's in routing tables!
- Routing table exchange would swamp links!

Internet Approach to Scalable Routing

- Aggregate routers into regions called "autonomous systems" (AS) (aka domains)
  - Each AS is assigned an AS Number (ASN) by IANA
- Intra-AS routing
  - Routers in same AS run same routing protocol
    - "INTRA-AS" routing protocol
  - Routers in different AS can run different intra-AS routing protocol
  - An intra-AS router only needs to know about the other routers in the AS
- Inter-As routing
  - Routing among AS's
  - Gateways perform inter-domain routing (as well as intra-domain routing)
- Gateway router
  - At "edge" of its own AS
  - Direct link to router in another AS
- AFIT ASN = AS133
  - Not currently in use
- WPAFB ASN = AS132

IANA - Internet Assigned Numbers Authority - a department of ICANN

Interconnected ASes



- Forwarding table is configured by both
  - intra- and
  - inter-AS routing algorithms
- Intra-AS sets entries for internal dests
- Inter-AS & Intra-AS sets entries for external dests

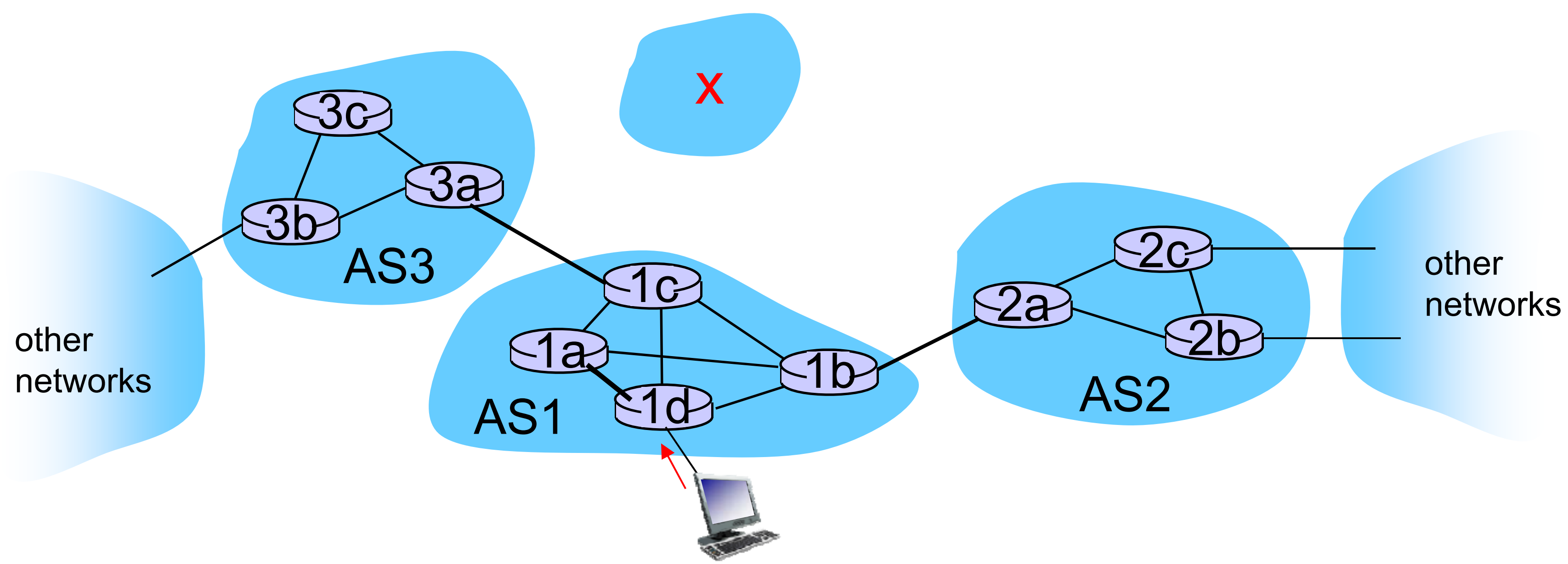
Inter-AS tasks

- Suppose router 1d in AS1 receives datagram for a dest outside of AS1 → subnet x
  - 1d should forward packet towards one of the gateway routers, but which one?

AS1 needs:

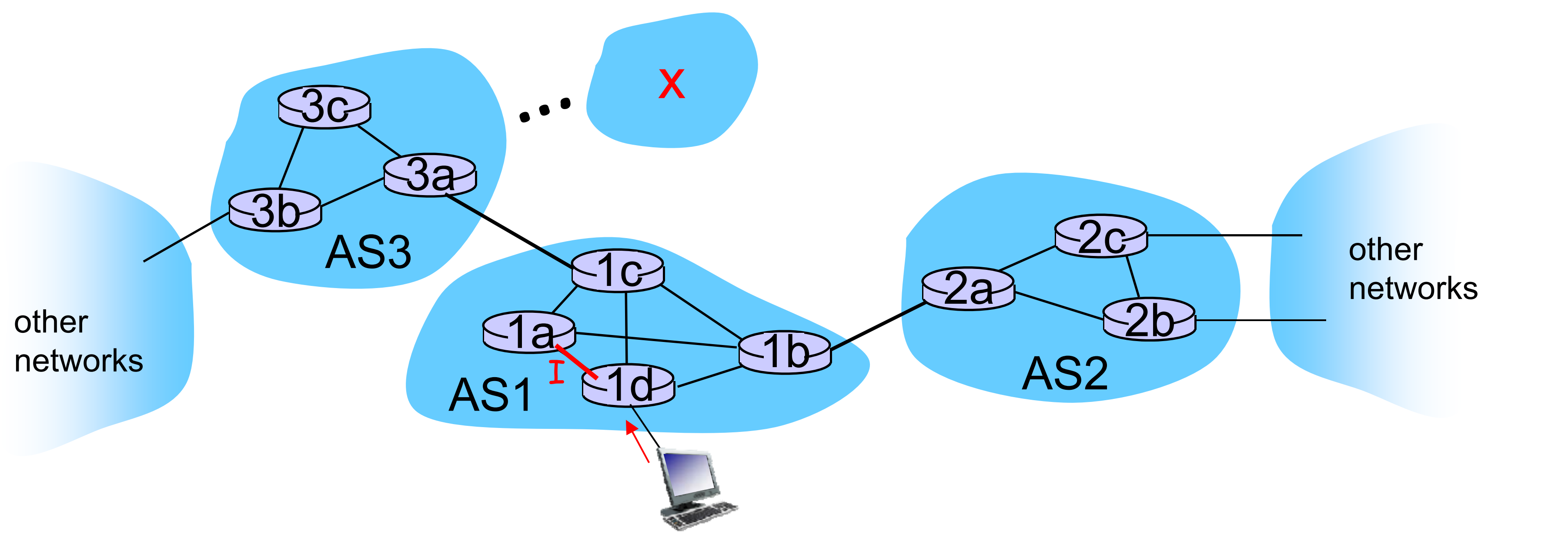
1. To learn which dests are reachable through AS2 and AS3
2. To propagate this reachability info to all routers in AS1

Job of inter-AS routing!



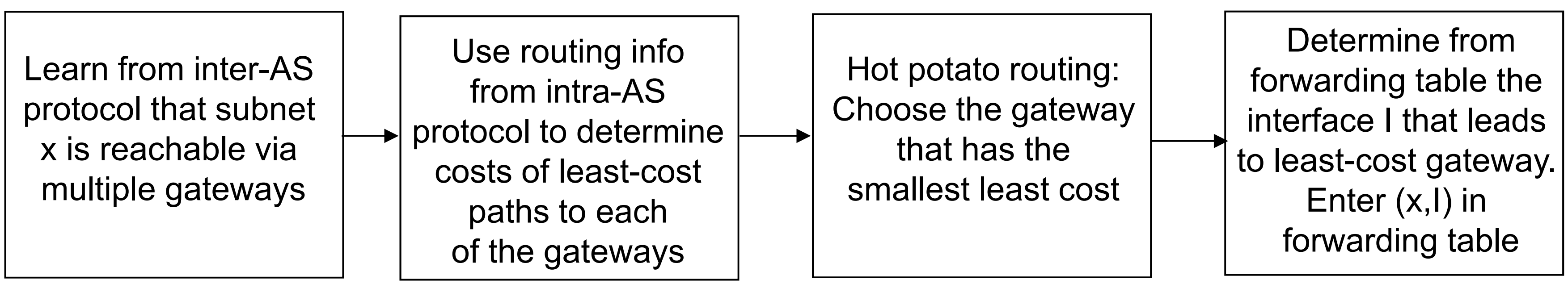
Example: Setting Forwarding Table in Router 1d

- Suppose AS1 learns from the inter-AS protocol that subnet x is reachable through AS3 (gateway 1c) but not from AS2
- Inter-AS protocol propagates reachability info to internal routers
- Router 1d determines from intra-AS routing info that its interface I is on the least cost path to 1c
- Puts in forwarding table entry (x,I)



Example: Choosing Among Multiple ASes

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



Intra-AS Routing

- Also known as Interior Gateway Protocols (IGP)

- Most common Intra-AS routing protocols:
  - OSPF: Open Shortest Path First
    - Dijkstra (Link State)
  - RIP: Routing Information Protocol
    - Bellman-Ford (Distance Vector)
  - IGRP: Interior Gateway Routing Protocol (Cisco proprietary)

OSPF (Open Shortest Path First)

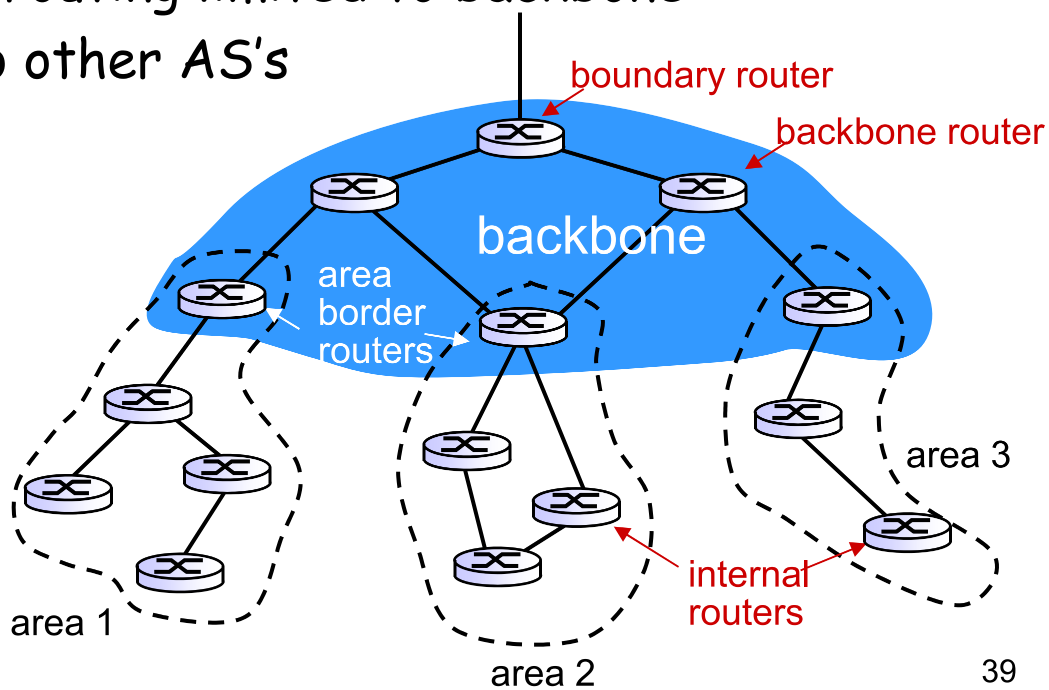
- "Open": publicly available
- Uses Link State algorithm
- OSPF advertisement carries one entry (link state) per neighbor router
- Advertisements disseminated to entire AS or area (via flooding)
  - Carried in OSPF messages directly over IP (upper layer = 89)
  - Sent at least every 30 minutes
  - Routers detect each other using HELLO OSPF protocol packet broadcasts
  - Routers then select a designated router (DR) which acts as a hub to reduce traffic between routers
    - DR maintains a complete topology table of the network and sends the updates to the other routers via multicast
    - Every time a router sends an update, it sends it to the DR
      - DR then sends the update out to all other routers in the area

OSPF "Advanced" Features

- Link costs set by administrator instead of all links = 1 (as in RIP)
- Security: all OSPF messages authenticated (to prevent malicious intrusion)
- Multiple same-cost paths allowed (only one path in RIP)
- Support for Hierarchical OSPF in large domains
- Typically deployed in upper-tier ISPs

Hierarchical OSPF

- Two-level hierarchy: 1. local area, 2. backbone
  - Link-state advertisements only in area
  - Each node has detailed area topology; only know direction (shortest path) to nets in other areas
- Area border routers: "summarize" distances to nets in own area, advertise to other area border routers
- Backbone routers: run OSPF routing limited to backbone
- Boundary routers: connect to other AS's



Chapter 5: Outline

- 5.1 Introduction
- 5.2 Routing protocols
  - Link state
  - Distance vector
- 5.3 Intra-AS routing in the Internet: OSPF
- 5.4 Routing among the ISPs: BGP
- 5.5 The SDN control plane
- 5.6 ICMP: The Internet Control Message Protocol
- 5.7 Network management and SNMP

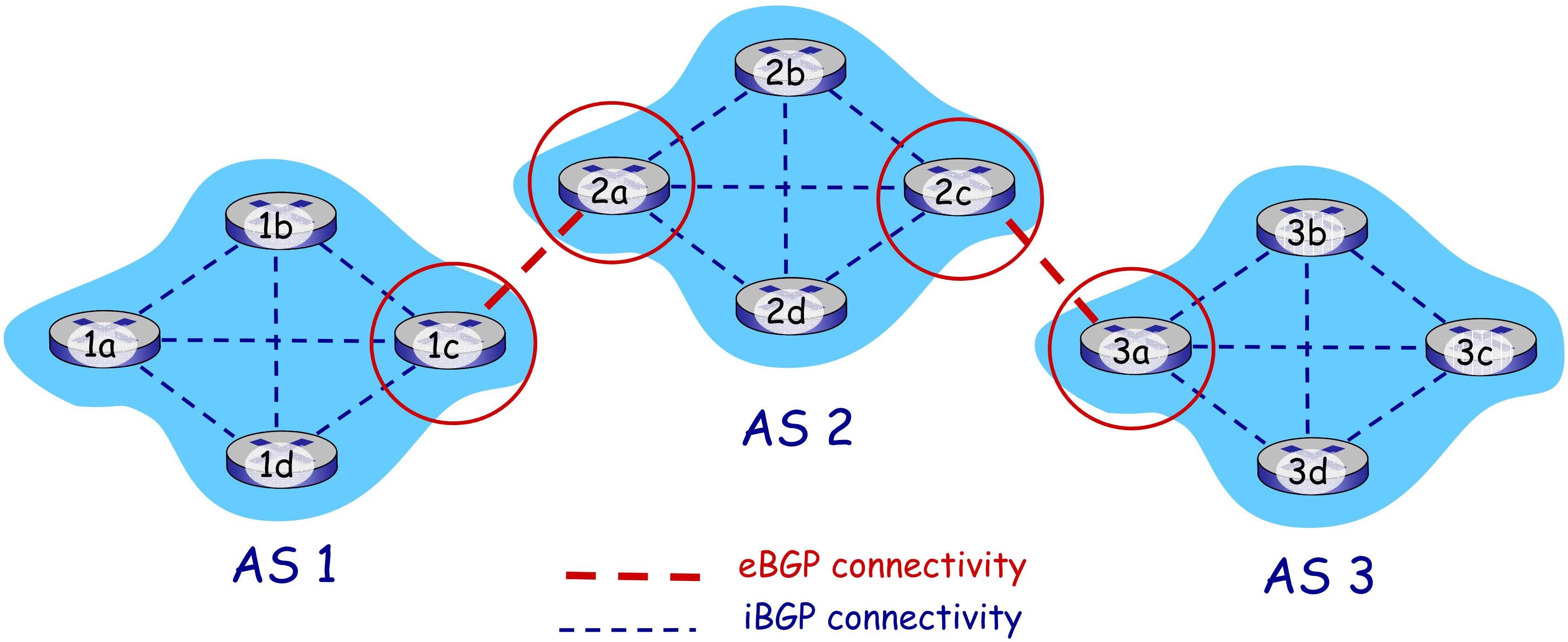
Internet INTER-AS Routing: BGP

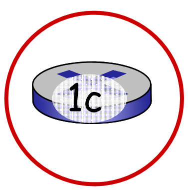
- BGP (Border Gateway Protocol): *the* de facto standard

- BGP provides each AS a means to:
  1. eBGP: Obtain subnet reachability info from neighboring (external) ASs
  2. iBGP: Propagate reachability info to all routers internal to the AS
  3. Determine "good" routes to subnets based on reachability information and policy

- Allows a subnet to advertise its existence to rest of the Internet: *"I am here"*

eBGP, iBGP Connections

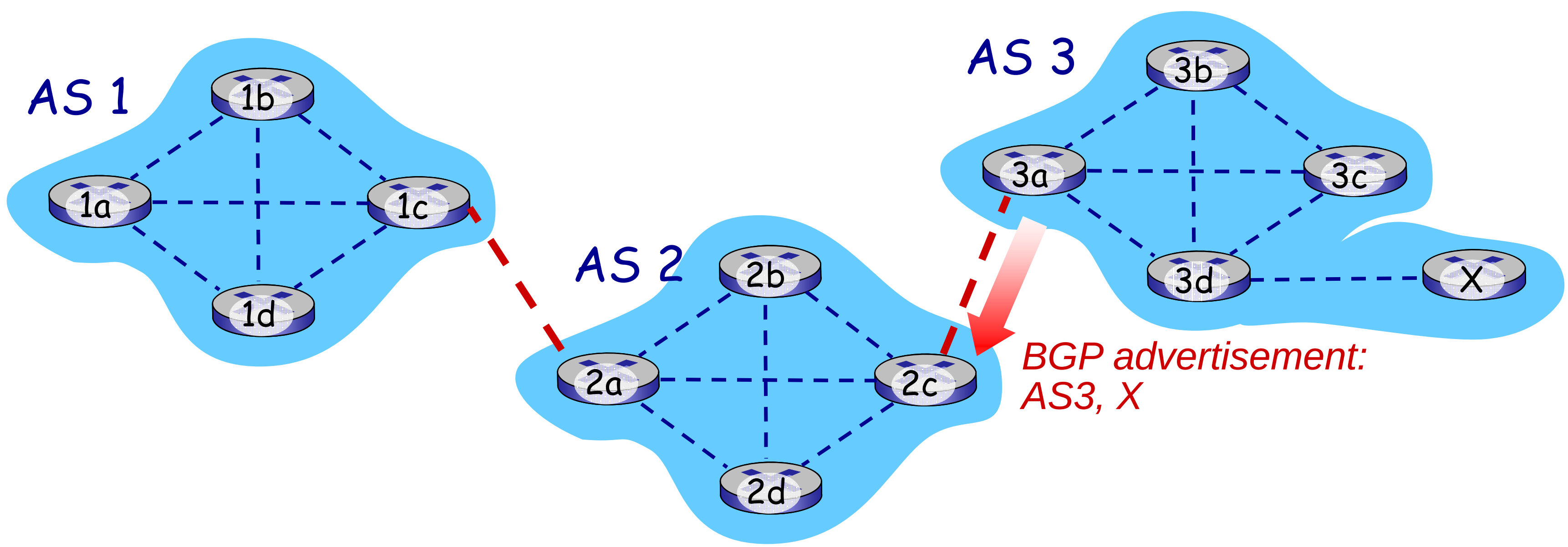




gateway routers run both eBGP and iBGP protocols

BGP Basics

- **BGP session**: Pairs of routers (BGP peers) exchange routing info over semi-permanent TCP connections (port 179)
  - BGP sessions do not correspond to physical links
  - Keep-alive msgs sent every 30 seconds to maintain connection
- When AS3 gateway router 3a advertises path AS3,X to AS2 gateway router 2c:
  - AS3 promises to AS2 it will forward datagrams towards X



Path Attributes and BGP Routes

- Advertised prefix includes BGP attributes
  - prefix + attributes = "route"
- Three mandatory "well known" attributes:
  - Origin – The origin of a BGP route
    - "i" (code 0) - internal - route aggregation (**network** command used)
    - "e" (code 1) – external – obsolete
    - "?" (code 2) – origin in unknown or route was redistributed/ aggregated/incomplete
  - AS-PATH: contains the ASs through which the advertisement for the prefix passed: AS67 AS17
    - Useful for detecting loops
  - NEXT-HOP: IP address of the next-hop router
    - Sending router sets the next-hop field to its own IP address
    - Used to
      - establish BGP TCP sessions
      - determine least-cost path to closest link by internal routing algorithms

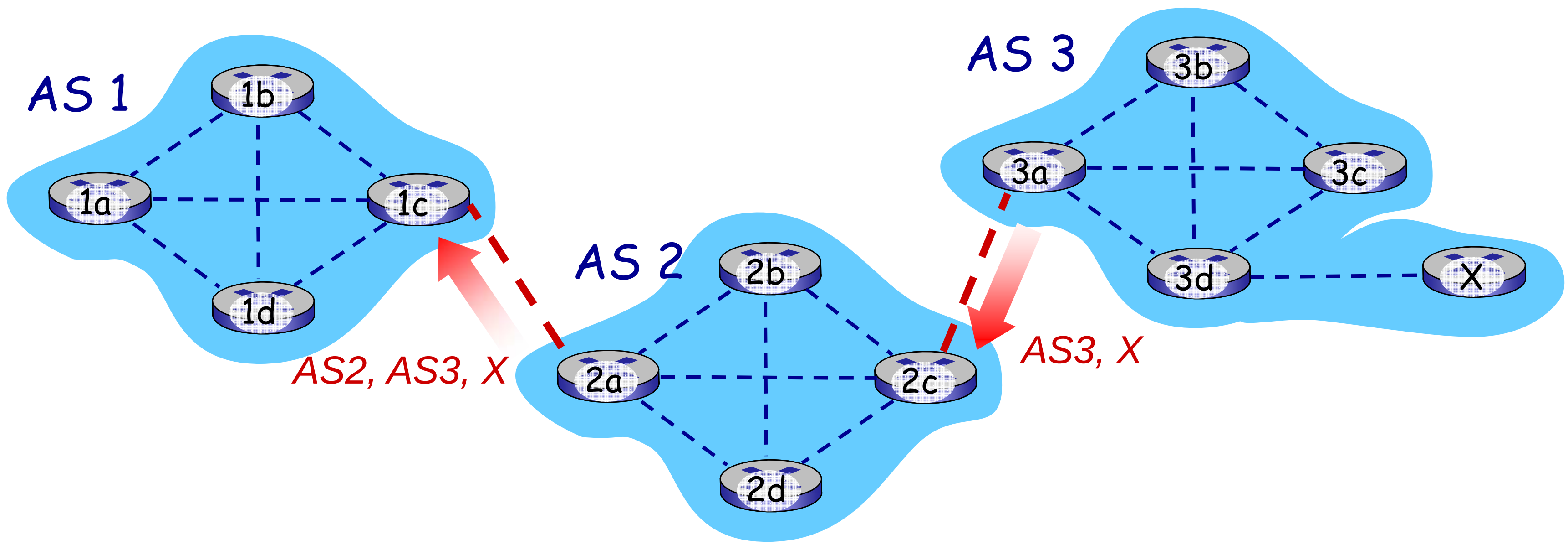
http://docwiki.cisco.com/wiki/Border_Gateway_Protocol

Path Attributes and BGP Routes

- Gateway router receiving route advertisement uses import policy to accept/decline
  - Why decline?
    - The AS may not want to send traffic through one of the ASs listed in AS-PATH or
    - It detected a loop

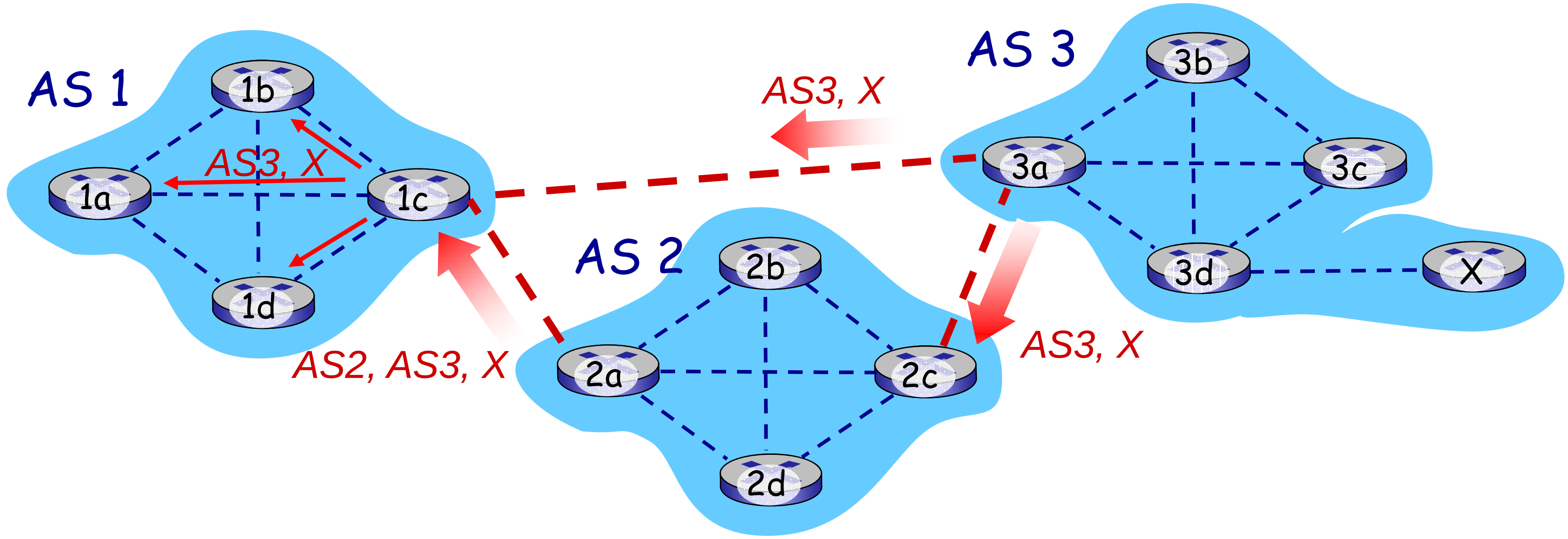
- *Policy-based* routing

BGP Path Advertisement



- AS2 router 2c receives path advertisement AS3,X (via eBGP) from AS3 router 3a
- Based on AS2 policy, AS2 router 2c accepts path AS3,X, propagates (via iBGP) to all AS2 routers
- Based on AS2 policy, AS2 router 2a advertises (via eBGP) path AS2, AS3, X to AS1 router 1c

BGP Path Advertisement



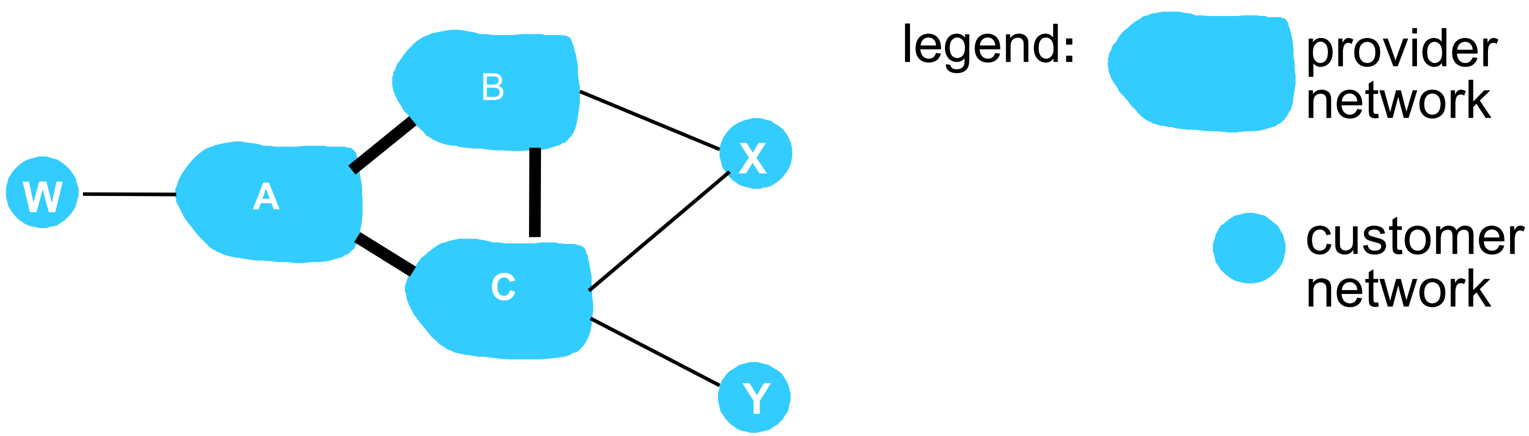
- Gateway router (e.g., 1c) may learn about multiple paths to destination
  - AS2,AS3,X from 2a
  - AS3,X from 3a
  - Based on policy, AS1 gateway router 1c chooses path AS3,X, and advertises path within AS1 via iBGP

BGP Route Selection

- Router may learn about more than 1 route to destination AS, selects route based on:
  1. Local preference value attribute: policy decision
  2. Shortest AS-PATH
  3. Closest NEXT-HOP router: hot potato routing
  4. "Additional criteria"

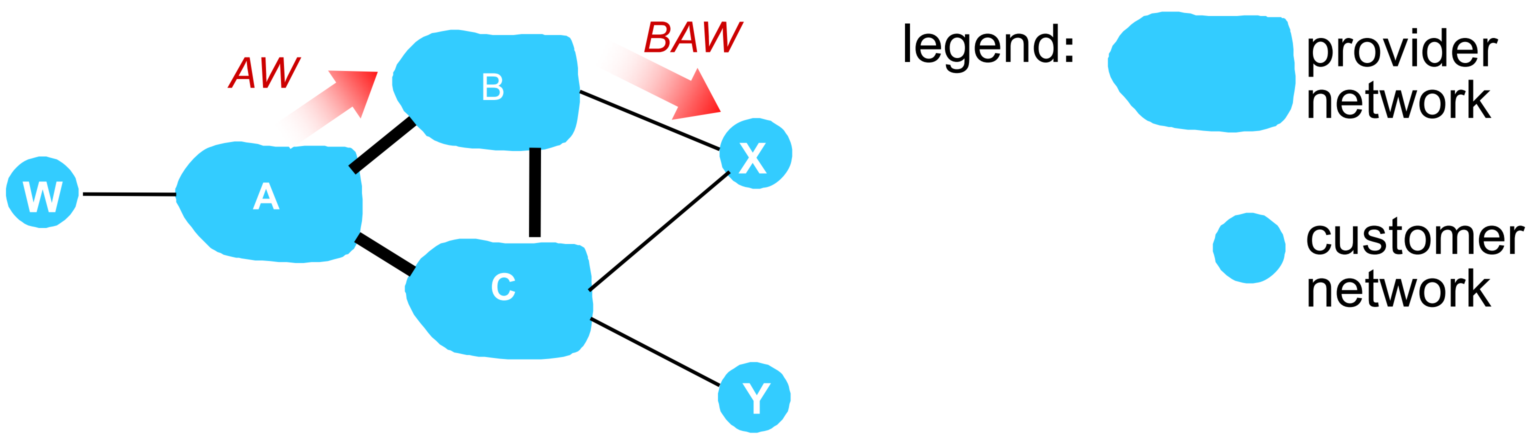
BGP Routing Policy

- A,B,C are provider networks (ASs)
- W,X,Y are customers (of provider networks) and also ASs
- X is dual-homed: attached to two networks
  - X does not want to route from B to C
  - .. so X will not advertise to B a route to C



BGP Routing Policy

- A advertises to B the path AW
- B advertises to X the path BAW
- Should B advertise to C the path BAW?
  - No way! B gets no "revenue" for routing CBAW since neither W nor C are B's customers
  - B wants to force C to route to w via A
  - B wants to route only to/from its customers!
- Rule of thumb – if traffic is for ISP's customers → forward



Why Different Intra-, Inter-AS Routing?

Policy:

- Inter-AS: admin wants control
  - over how its traffic routed and
  - who routes through its net
- Intra-AS: single admin
  - no policy decisions needed

Scale:

- Hierarchical routing saves table size and reduces update traffic

Performance:

- Intra-AS: can focus on performance
- Inter-AS: policy may dominate over performance

Chapter 5: Outline

- 5.1 Introduction
- 5.2 Routing protocols
  - Link state
  - Distance vector
- 5.3 Intra-As routing in the Internet: OSPF
- 5.4 Routing among the ISPs: BGP
- 5.5 The SDN control plane
- 5.6 ICMP: The Internet Control Message Protocol
- 5.7 Network management and SNMP

Historical Approach To Routing

- Internet network layer: historically has been implemented via distributed, per-router approach

- Monolithic router contains
  - Switching hardware,
  - Runs proprietary implementation of Internet standard protocols (RIP, OSPF, BGP) in proprietary router OS (e.g., Cisco IOS)

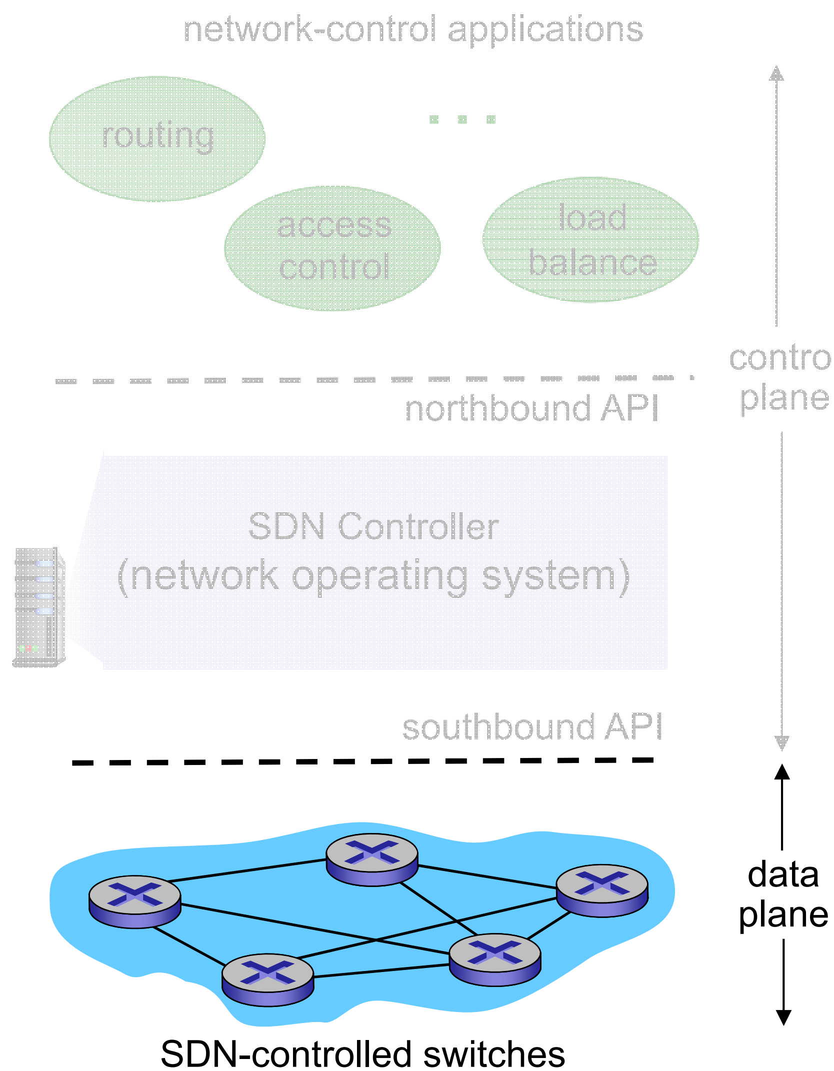
- Different "middleboxes" for different network layer functions
  - Firewalls, routers, switches, load balancers, NAT boxes, …

Why a Logically Centralized Control Plane in SDN?

- Easier network management
  - Avoid router misconfigurations
  - Greater flexibility of traffic flows
- Table-based forwarding allows "programming" routers
  - Centralized "programming" easier
    - Compute tables centrally and distribute
  - Distributed (historical) "programming" more difficult
    - Compute tables as result of distributed algorithm (protocol) implemented in each and every router
- Open (non-proprietary) implementation of control plane

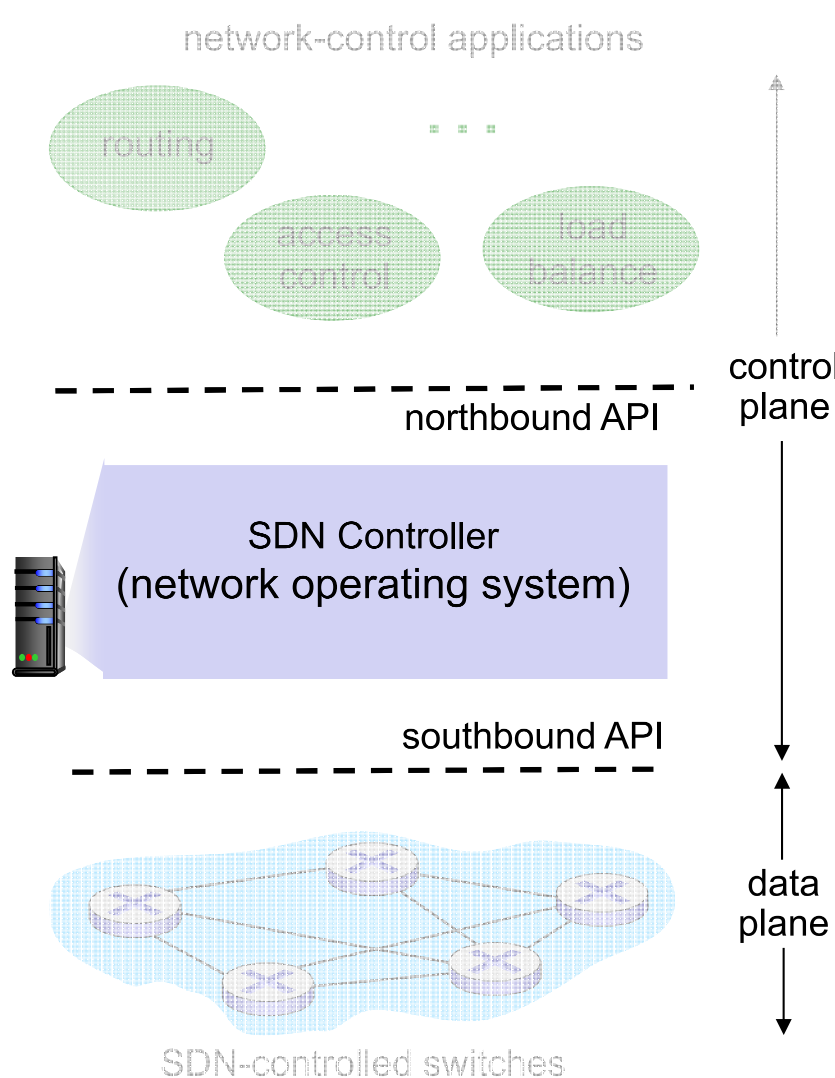
SDN Perspective: Data Plane Switches

- Fast, simple, commodity switches implementing generalized data-plane forwarding in hardware
- Switch flow table computed and installed by controller
- API for table-based switch control (e.g., OpenFlow)
  - Defines what is controllable and what is not
- Protocol for communicating with controller
  - OpenFlow
    - TCP port 6653



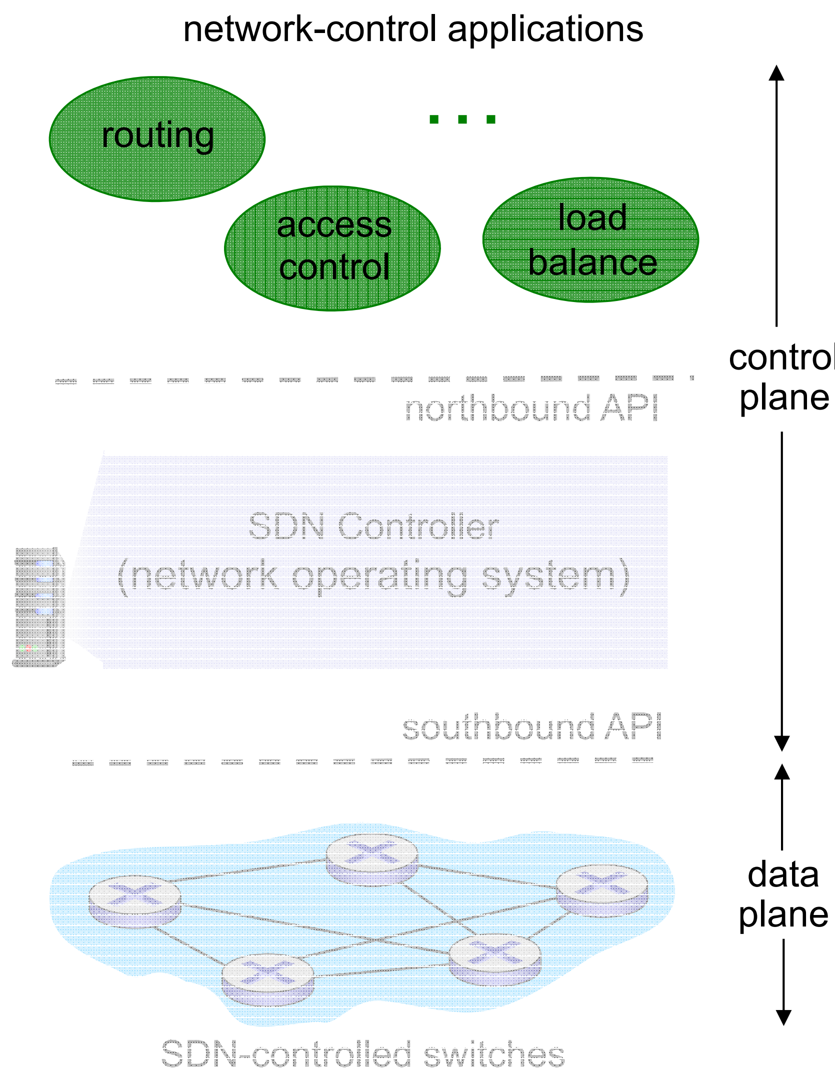
SDN Perspective: SDN Controller

- Maintain network state information
- Interacts with network control applications "above" via northbound API
- Interacts with network switches "below" via southbound API
- Implemented as distributed system for performance, scalability, fault-tolerance, robustness



SDN Perspective: Control Applications

- "Brains" of control:
  implement control functions
  using lower-level services,
  API provided by SDN controller
- Can be provided by 3rd party
- Distinct from routing
  vendor or SDN controller



Chapter 5: Outline

- 5.1 Introduction
- 5.2 Routing protocols
  - Link state
  - Distance vector
- 5.3 Intra-As routing in the Internet: OSPF
- 5.4 Routing among the ISPs: BGP
- 5.5 The SDN control plane
- 5.6 ICMP: The Internet Control Message Protocol
- 5.7 Network management and SNMP

ICMP: Internet Control Message Protocol

- Short messages used to send error & other control information
- 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 msgs carried in IP datagrams

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	4	frag. needed and DF set
3	6	dest network unknown
3	7	dest host unknown
8	0	echo request (ping)
9	0	route advertisement (mobile)
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

8 bits	8 bits	16 bits
Type	Code	Checksum
Depends on message type		
Internet header + 8 bytes of original datagram causing error		

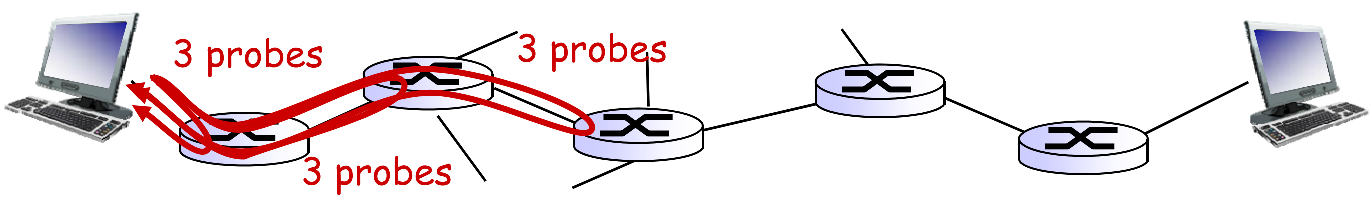
Traceroute and ICMP

- Source sends series (3) of
  - ICMP echo request (Win)
  - UDP segments (Linux) to dest with an unlikely port number
  - 1st TTL = 1; 2nd TTL = 2, etc.
- When nth datagram arrives at nth router:
  - Router discards datagram
  - Sends to source an ICMP "TTL expired" message
  - Message includes name of router & IP address

- When ICMP message arrives, source calculates RTT
- Traceroute does this 3 times

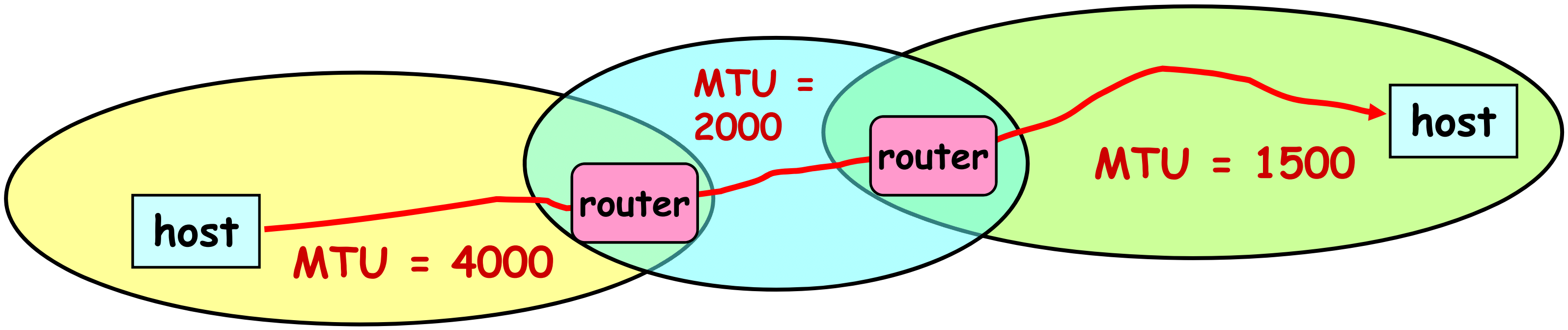
<u>Stopping criterion</u>

- Win: ICMP echo reply
- Linux: UDP segment eventually arrives at destination host
  - Destination returns ICMP "dest port unreachable" packet (type 3, code 3)
- When source gets the ICMP, it stops



Path MTU Discovery with ICMP

8 bits	8 bits	16 bits
Type: 3	Code: 4	Checksum
Unused = 0		Next-hop MTU
Internet header + 8 bytes of original datagram		



- Operation
  - Send max-sized (MTU) packet with "do not fragment" flag set in IP hdr
  - If problem encountered, router returns ICMP message
    - "Destination unreachable: Fragmentation needed"
- Typically, all packets follow same route
  - Makes sense to do MTU discovery if message is large
    - Send series of packets (i.e., a large message) from one host to another after MTU has been discovered to amortize discovery cost
- Enabled by default in Windows
  - To display current MTU
    - **`netsh interface ipv4 show subinterfaces`**