Electrical Engineering E6761
Computer Communication Networks
Lecture 6
Routing:
Routing Algorithms

Professor Dan Rubenstein
Tues 4:10-6:40, Mudd 1127
Course URL:
http://www.cs.columbia.edu/~danr/EE6761

Overview

- HW#2
- Midterm
  - extra office hours Fri 2-4
- Routing
  - virtual circuits (ATM) vs. best-effort
  - algorithms (theory)
    - link-state (Dijkstra's Algorithm)
    - distance vector
    - + poisoned reverse
  - Fragmentation & ICMP
  - Intra-Domain
    - RIP
    - OSPF
  - Inter-Domain: BGP

Midterm

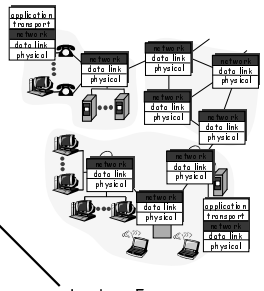
- All material through today's lecture
- Focus more on theory than on application
- Topics (NOTE: not necessarily all-inclusive list)
  - LAN hardware, bridge v. hub
  - IP addressing / CIDR /lookups / using tries
  - Routing: Distance Vector and Link-State
  - Basic queueing questions (similar to HW#2)
    - Little's Law
    - Steady state results for M/M/1/K (focus more on concept than on formula)
    - FIFO, Priority, Round-Robin, WFQ
  - Transport:
    - high level socket programming (when to bind? accept?)
    - reliability (state diagrams, go-back-N, sel. repeat)
    - flow & congestion control
  - Application: DNS

Network layer functions

- transport packet from sending to receiving hosts
- network layer protocols in *every* host, router

three important functions:

- *path determination:* route taken by packets from source to dest. *Routing algorithms*
- *switching:* move packets from router's input to appropriate router output
- *call setup:* some network architectures require router call setup along path before data flows



Lecture 5

Network service model

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

service abstraction

- guaranteed bandwidth?
- preservation of inter-packet timing (no jitter)?
- loss-free delivery?
- in-order delivery?
- congestion feedback to sender?

The most important abstraction provided by network layer:

virtual circuit or datagram?

Virtual circuits

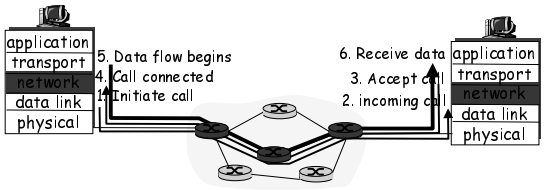
"source-to-dest path behaves much like telephone circuit"

- performance-wise
- network actions along source-to-dest path

- call setup, teardown for each call *before* data can flow
- each packet carries VC identifier (not destination host OD)
- *every* router on source-dest paths maintain "state" for each passing connection
  - NOTE: transport-layer connection only involved two end systems
- link, router resources (bandwidth, buffers) may be *allocated* to VC to get circuit-like performance

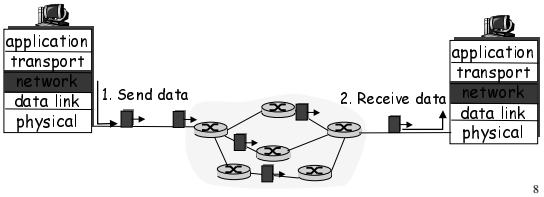
Virtual circuits: signaling protocols

- used to setup, maintain teardown VC
- used in ATM, frame-relay, X.25
- not used in today's Internet



Datagram networks: the Internet model

- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of "connection"
- packets typically routed using destination host ID
  - packets between same source-dest pair may take different paths



Network layer service models:

Network Architecture	Service Model	Guarantees ?				Congestion feedback
		Bandwidth	Loss	Order	Timing	
Internet	best effort	none	no	no	no	no (inferred via loss)
ATM	CBR	constant rate	yes	yes	yes	no congestion
ATM	VBR	guaranteed rate	yes	yes	yes	no congestion
ATM	ABR	guaranteed minimum	no	yes	no	yes
ATM	UBR	none	no	yes	no	no

- Internet model being extented: Intserv, Diffserv
  - to be covered in a few weeks…

Datagram or VC network: why?

Internet
- data exchange among computers
  - "elastic" service, no strict timing req.
- "smart" end systems (computers)
  - can adapt, perform control, error recovery
  - simple inside network, complexity at "edge"
- many link types
  - different characteristics
  - uniform service difficult

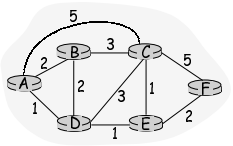
ATM
- evolved from telephony
- human conversation:
  - strict timing, reliability requirements
  - need for guaranteed service
- "dumb" end systems
  - telephones
  - complexity inside network

Routing

Routing protocol

Goal: determine "good" path (sequence of routers) thru network from source to dest.

Graph abstraction for routing algorithms:
- graph nodes are routers
- graph edges are physical links
  - link cost: delay, $ cost, or congestion level



- "good" path:
  - typically means minimum cost path
  - other def's possible
  - in practice: typically each edge assigned weight of 1

Routing Algorithm classification

Global or decentralized information?

Global:
- all routers have complete topology, link cost info
- "link state" algorithms

Decentralized:
- router knows physically-connected neighbors, link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- "distance vector" algorithms

Static or dynamic?

Static:
- routes change slowly over time

Dynamic:
- routes change more quickly
  - periodic update
  - in response to link cost changes

Shortest Path Algorithms

Notation:

- c(i,j): link cost from node i to j. cost infinite 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, that is next v
- N: set of nodes whose least cost path definitively known

Key Idea

- Let D(A,B) = min dist from any node A to any node B
- Let adj(A) = set of nodes w/ edges from A (adjacent)
- $D(A,B) = \min_{C \in adj(A)} \{c(A,C) + D(C,B)\}$
- shortest paths computed recursively from other shortest paths

13

Dijsktra's Algorithm

```
1  Initialization:
2   N = {A}
3   for all nodes v
4    if v adjacent to A
5      then D(v) = c(A,v)
6      else D(v) = infty
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 all v adjacent to 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    shortest path cost to w plus cost from w to v */
15  until all nodes in N
```

15

A Link-State Routing Algorithm

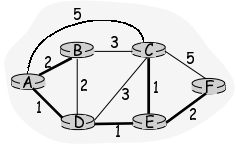
Dijkstra's algorithm

- net topology, link costs known to all nodes
  - accomplished via "link state broadcast"
  - all nodes have same info
- computes least cost paths from one node ("source") to all other nodes
  - gives routing table for that node
- iterative: after k iterations, know least cost path to k dest.'s

14

Dijkstra's algorithm: example

Step	start N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)	D(F),p(F)
→0	A	2,A	5,A	1,A	infinity	infinity
→1	AD	2,A	4,D		2,D	infinity
→2	ADE	2,A	3,E			4,E
→3	ADEB		3,E			4,E
→4	ADEBC					4,E
5	ADEBCF					



16

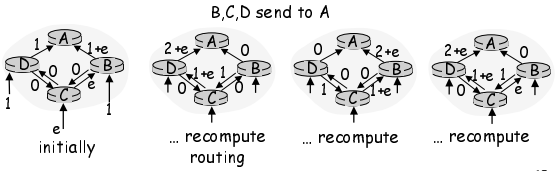
Dijkstra's algorithm, discussion

Algorithm complexity: n nodes

- each iteration: need to check all nodes, w, not in N
- $n*(n+1)/2$ comparisons: $O(n^2)$
- more efficient implementations possible: $O(n \log n)$

Oscillations possible:

- e.g., link cost = amount of carried traffic



Distance Vector Routing Algorithm

iterative:

- continues until no nodes exchange info.
- *self-terminating*: no "signal" to stop

asynchronous:

- nodes need *not* exchange info/iterate in lock step!

distributed:

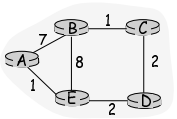
- each node's info communicates *only* to directly-attached neighbors

Distance Table data structure

- each node has its own
  - row for each possible destination
  - column for each directly-attached neighbor to node
- example: in node X, for dest. Y via neighbor Z:

$$D^X(Y,Z) = \text{distance } \textit{from } X \textit{ to } Y, \textit{ via } Z \text{ as next hop} = c(X,Z) + \min_w\{D^Z(Y,w)\}$$

Distance Table: example



$D^E(C,D) = c(E,D) + \min_w\{D^D(C,w)\}$
$= 2+2 = 4$

$D^E(A,D) = c(E,D) + \min_w\{D^D(A,w)\}$
$= 2+3 = 5$ loop!

$D^E(A,B) = c(E,B) + \min_w\{D^B(A,w)\}$
$= 8+6 = 14$ loop!

$D^E()$ — cost to destination via

destination	A	B	D
A	(1)	14	5
B	7	8	(5)
C	6	9	(4)
D	4	11	(2)

Distance table gives routing table

$D^E()$ — cost to destination via

destination	A	B	D
A	(1)	14	5
B	7	8	(5)
C	6	9	(4)
D	4	11	(2)

destination	Outgoing link to use, cost
A	A,1
B	D,5
C	D,4
D	D,4

Distance table → Routing table

Distance Vector Routing: overview

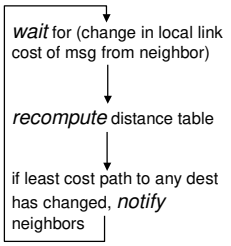
Iterative, asynchronous:
each local iteration caused by:

- local link cost change
- message from neighbor: its least cost path change from neighbor

Distributed:

- each node notifies neighbors *only* when its least cost path to any destination changes
  - neighbors then notify their neighbors if necessary

Each node:



Distance Vector Algorithm:

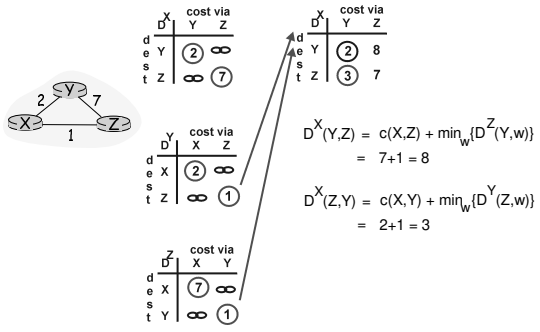
At all nodes, X:

```
1  Initialization:
2  for all adjacent nodes v:
3      D^X(*,v) = infty       /* the * operator means "for all rows" */
4      D^X(v,v) = c(X,v)
5  for all destinations, y
6      send min_w D^X(y,w) to each neighbor  /* w over all X's neighbors */
```

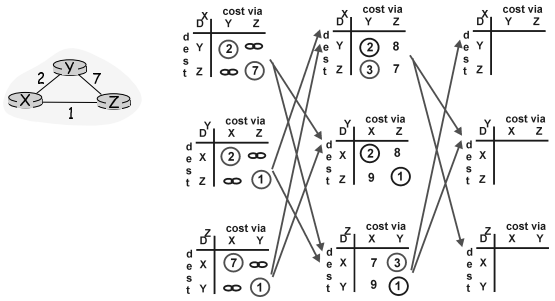

Distance Vector Algorithm (cont.):

```
8  loop
9    wait (until I see a link cost change to neighbor V
10        or until I receive update from neighbor V)
11
12   if (c(X,V) changes by d)
13     /* change cost to all dest's via neighbor v by d */
14     /* note: d could be positive or negative */
15     for all destinations y:  D^X(y,V) = D^X(y,V) + d
16
17   else if (update received from V wrt destination Y)
18     /* shortest path from V to some Y has changed  */
19     /* V has sent a new value for its  min_w DV(Y,w) */
20     /* call this received new value is "newval"     */
21     for the single destination y: D^X(Y,V) = c(X,V) + newval
22
23   if we have a new min_w D^X(Y,w)for any destination Y
24     send new value of min_w D^X(Y,w) to all neighbors
25
26 forever
```


Distance Vector Algorithm: example



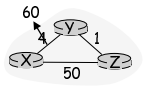
Distance Vector Algorithm: example

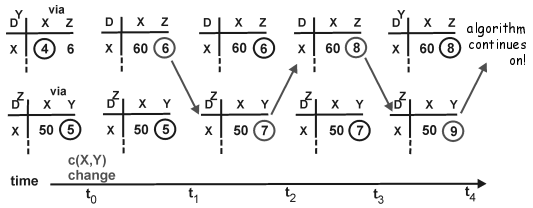


Distance Vector: link cost changes

Link cost changes:

- good news travels fast
- bad news travels slow - "count to infinity" problem!
- Due to lack of a global view!

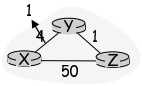




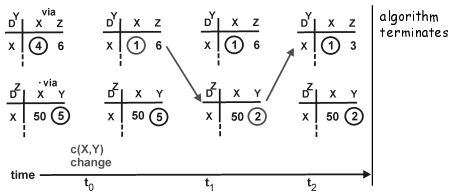
Distance Vector: link cost changes

Link cost changes:

- node detects local link cost change
- updates distance table (line 15)
- if cost change in least cost path, notify neighbors (lines 23,24)



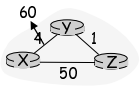
"good news travels fast"

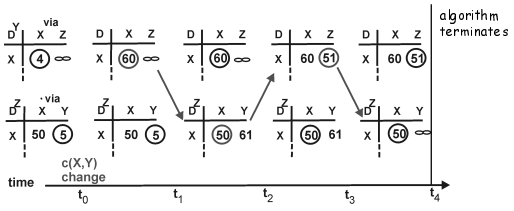


Distance Vector: poisoned reverse

If Z routes through Y to get to X :

- Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)
- will this completely solve count to infinity problem?





Comparison of LS and DV algorithms

Message complexity
- LS: with n nodes, E links, O(nE) msgs sent each
- DV: exchange between neighbors only
  - convergence time varies

Speed of Convergence
- LS: $O(n^2)$ algorithm requires O(nE) msgs
  - may have oscillations
- DV: convergence time varies
  - may be routing loops
  - count-to-infinity problem

Robustness: what happens if router malfunctions?

LS:
- node can advertise incorrect *link* cost
- each node computes only its *own* table

DV:
- DV node can advertise incorrect *path* cost
- each node's table used by others
  - errors propagate thru network

Hierarchical Routing

Our routing study thus far - idealization
- all routers identical
- network "flat"
- … *not* true in practice

scale: with 50 million destinations:
- can't store all dest's in routing tables!
- routing table exchange would swamp links!

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

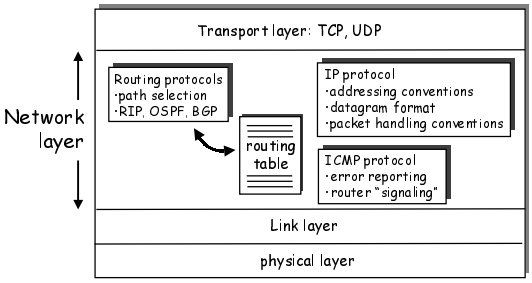
Hierarchical Routing

- aggregate routers into regions, "autonomous systems" (AS)
- routers in same AS run same routing protocol
  - "inter-AS" routing protocol
  - routers in different AS can run different inter-AS routing protocol

gateway routers
- special routers in AS
- run inter-AS routing protocol with all other routers in AS
- *also* responsible for routing to destinations outside AS
  - run *intra-AS routing* protocol with other gateway routers

The Internet Network layer

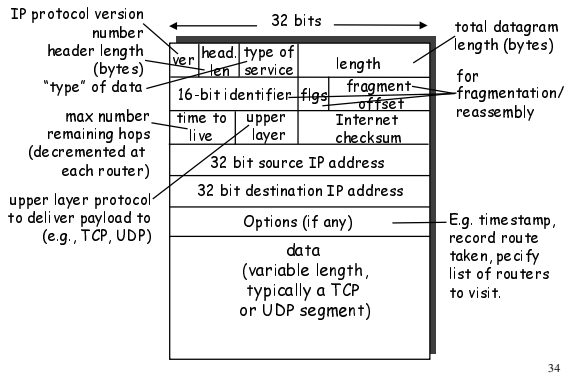
Host, router network layer functions:



Routing in the Internet

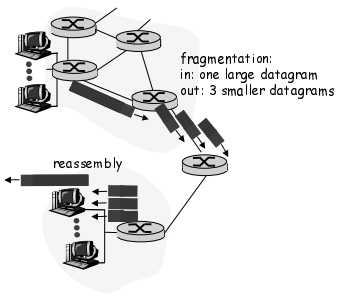
- The Global Internet consists of Autonomous Systems (AS) interconnected with each other:
  - **Stub AS**: small corporation
  - **Multihomed AS**: large corporation (no transit)
  - **Transit AS**: provider
- Two-level routing:
  - **Intra-AS**: administrator is responsible for choice
  - **Inter-AS**: unique standard

IP datagram format

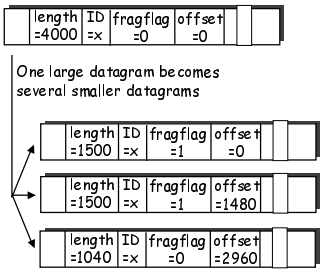


IP Fragmentation & Reassembly

- network links have MTU (max.transfer size) - largest possible link-level frame.
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination (keeps routers' jobs simpler)
  - IP header bits used to identify, order related fragments



IP Fragmentation and Reassembly

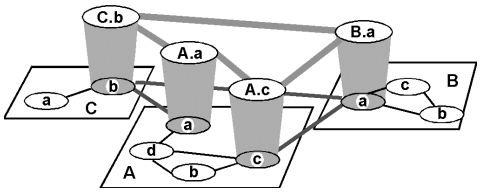


ICMP: Internet Control Message Protocol

- used by hosts, routers, gateways to communication 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
- 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

Internet AS Hierarchy



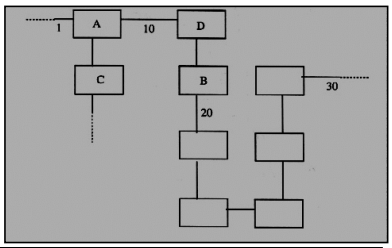
Intra-AS Routing

- Also known as Interior Gateway Protocols (IGP)
- Most common IGPs:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IGRP: Interior Gateway Routing Protocol (Cisco propr.)

RIP (Routing Information Protocol)

- Distance vector type scheme
- Included in BSD-UNIX Distribution in 1982
- Distance metric: # of hops (max = 15 hops)
  - *Can you guess why there is a limit?*
- Distance vector: exchanged every 30 sec via a Response Message (also called **Advertisement**)
- Each Advertisement contains up to 25 destination nets

RIP (Routing Information Protocol)



Destination Network	Next Router	Num. of hops to dest.
1	A	2
20	B	2
30	B	7
10	--	1
….	….	….

4

RIP Table processing

- RIP routing tables managed by an **application** process called route-d (daemon)
- Advertisements encapsulated in UDP packets (no reliable delivery required; advertisements are periodically repeated)

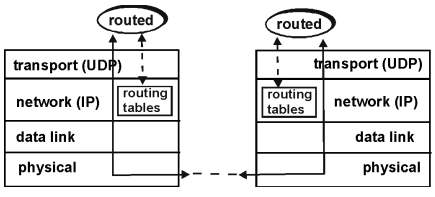
43

RIP: Link Failure and Recovery

- If no advertisement heard after 180 sec, neighbor/link dead
- Routes via the neighbor are invalidated; new advertisements sent to neighbors
- Neighbors in turn send out new advertisements if their tables changed
- Link failure info quickly propagates to entire net
- Poison reverse used to prevent ping-pong loops (infinite distance = 16 hops)

42

RIP Table processing



44

RIP Table example (continued)

RIP Table example
(at router *giroflee.eurocom.fr*):

- Three attached class C networks (LANs)
- Router only knows routes to attached LANs
- Default router used if destination addr has no matching interface
- Route multicast address: 224.0.0.0
- Loopback interface (for debugging): packet sent is delivered as received

RIP Table example

Destination	Gateway	Flags	Metric	Use	Interface
127.0.0.1	127.0.0.1	UH	0	26492	lo0
192.168.2.	192.168.2.5	U	2	13	fa0
193.55.114.	193.55.114.6	U	3	58503	le0
192.168.3.	192.168.3.5	U	2	25	qaa0
224.0.0.0	193.55.114.6	U	3	0	le0
default	193.55.114.129	UG	0	143454	

Flags:

U = route is up

H = target is a hosthost

G = use gateway

Use: # of lookups for the route

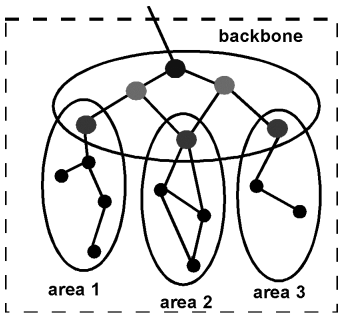
OSPF (Open Shortest Path First)

- "open": publicly available
- Uses the Link State algorithm
  - LS packet dissemination
  - Topology map at each node
  - Route computation using Dijkstra's alg
- OSPF advertisement carries one entry per neighbor router
- Advertisements disseminated to entire Autonomous System (via flooding)

OSPF "advanced" features (not in RIP)

- Security: all OSPF messages are authenticated (to prevent malicious intrusion); TCP connections used
  - Q: how can TCP (transport layer) be used when it depends on the network layer to locate destination? Circular?
- Multiple same-cost paths allowed (only one path in RIP)
- For each link, multiple cost metrics for different TOS (eg, satellite link cost set "low" for best effort; high for real time)
- Integrated uni- and multicast support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF
- Hierarchical OSPF in large domains.

Hierarchical OSPF



e.g., 4 Areas:

one area is "nominated" to be the backbone

49

Hierarchical OSPF

- Two-level hierarchy: local area and backbone.
- Link-state advertisements do not leave respective areas.
- Nodes in each area have detailed area topology; they only know direction (shortest path) to networks in other areas.
- **Area Border routers** "summarize" distances to networks in the area and advertise them to other Area Border routers.
- **Backbone routers** run an OSPF routing alg limited to the backbone.
- **Boundary routers** connect to other ASs.

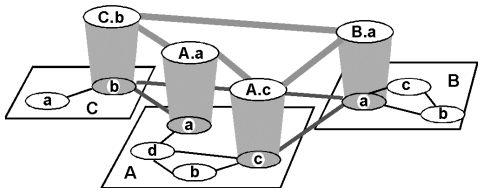
50

IGRP (Interior Gateway Routing Protocol)

- CISCO proprietary (1997); successor of RIP (mid 80s)
- Distance Vector, like RIP
- several cost metrics (delay, bandwidth, reliability, load etc)
- uses TCP to exchange routing updates
- routing tables exchanged only when costs change
- Loop-free routing achieved by using a Distributed Updating Alg. (DUAL) based on *diffused computation*
- In DUAL, after a distance increase, the routing table is *frozen* until all affected nodes have learned of the change.

51

Inter-AS routing



52

Inter-AS routing (cont)

- BGP (Border Gateway Protocol): the de facto standard
- **Path Vector** protocol: an extension of Distance Vector
- Each Border Gateway broadcasts to neighbors (peers) the entire path (ie, sequence of ASs) to destination
- For example, Gateway X may store the following path to destination Z:

Path (X,Z) = X,Y1,Y2,Y3,…,Z

Inter-AS routing (cont)

- Now, suppose Gwy X sends its path to peer Gwy W
- Gwy W may or may not select the path offered by Gwy X, because of cost, policy ($$$$) or loop prevention reasons.
- If Gwy W selects the path advertised by Gwy X, then:

Path (W,Z) = w, Path (X,Z)

Note: path selection based not so much on cost (eg,# of AS hops), but mostly on administrative and policy issues (e.g., do not route packets through competitor's AS)

Inter-AS routing (cont)

- Peers exchange BGP messages using TCP.
- OPEN msg opens TCP connection to peer and authenticates sender
- UPDATE msg advertises new path (or withdraws old)
- KEEPALIVE msg keeps connection alive in absence of UPDATES; it also serves as ACK to an OPEN request
- NOTIFICATION msg reports errors in previous msg; also used to close a connection

Why different Intra- and Inter-AS routing ?

- **Policy**: Inter is concerned with policies (which provider we must select/avoid, etc). Intra is contained in a single organization, so, no policy decisions necessary
- **Scale**: Inter provides an extra level of routing table size and routing update traffic reduction above the Intra layer
- **Performance**: Intra is focused on performance metrics; needs to keep costs low. In Inter it is difficult to propagate performance metrics efficiently (latency, privacy etc). Besides, policy related information is more meaningful.

We need **BOTH**!