EP2120 Internetworking

IK2218 Protocols and Principles of the Internet

Routing
Lecture 6

György Dán

KTH/EE/LCN

Literature:

Forouzan, TCP/IP Protocol Suite

(3ed Ch 14)(4ed Ch 11)

Slides courtesy of Olof Hagsand

Detailed reading instructions

Forouzan: TCP/IP Protocol Suite (4ed):

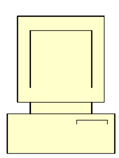
- Chapter 11: Unicast routing protocols
- You need to complement with slides, especially if you do not make the routing lab
- 11.6 OSPF: Skip detailed packet descriptions
- 11.8 BGP: Skip detailed packet descriptions

EP2120: Lab4 : Introduction to routing

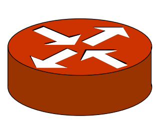
Routers

3

What is a router?



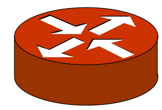
- Host (end-system)
  - One or many network interfaces
  - Can *not* forward packets between interfaces



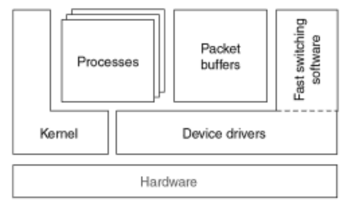
- Router
  - Two or more interfaces
  - Can forward packets between interfaces
  - Forwards on Layer 3

4

What does a router do?

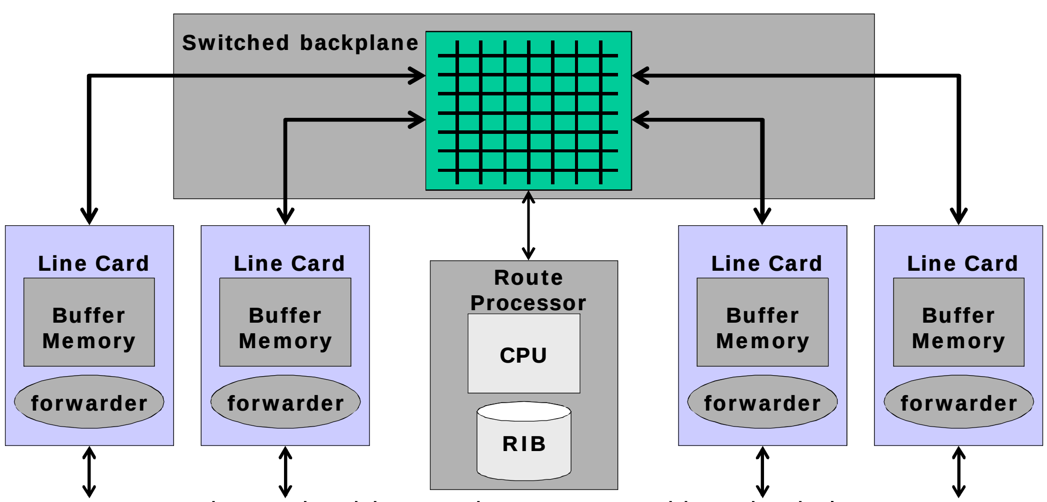


- Packet forwarding – "real time" at line speed
  - Not only IPv4
  - IPv6, MPLS, Tunneling,...
  - (But never naming,..)
- Filter traffic
  - Access lists based on src/dst, etc.
- Metering/Shaping/Policing
  - Measuring, forming and dropping traffic
- Computing routes - Routing
  - Build forwarding table in the "background"



Bollapragada, et. al. "Inside Cisco IOS Software Architecture," CCIE, 2000

Inside a router, example

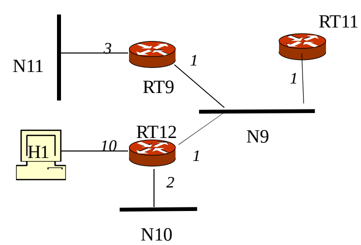


- Linecards with ports interconnected by a backplane
  - forwarding (data plane) – often in hardware
- Route processor (RP) runs routing protocols and management
  - *control-plane* processing

Routing Algorithms

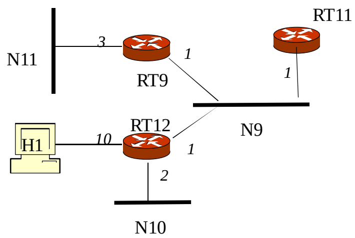
Routing algorithms

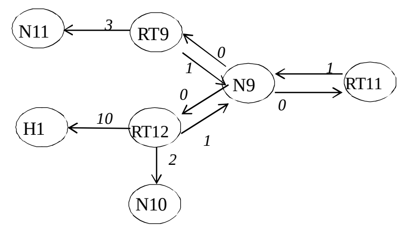
- Problem
  - –Find best path from RT11 to H1
- Typically based on shortest path algorithms (from graph theory)
  - Bellman-Ford algorithm
    - Used by Distance-Vector protocols (RIP, IGRP, BGP)
  - Dijkstra's algorithm
    - –Used by Link-State protocols (OSPF, IS-IS)
- Other algorithms used in
  - Multicast routing
  - Ad-hoc routing
  - Sensor networks
  - Delay-tolerant networks
  - Software defined networks



Networks vs. Graphs

- Network – product of engineering
  - hosts, interfaces, broadcast/unicast links
  - addresses, hierarchical layering, etc.
- Protocols have to work on networks

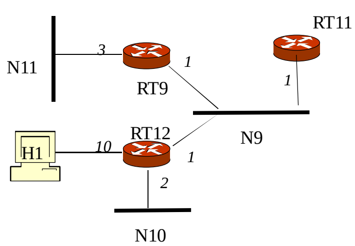


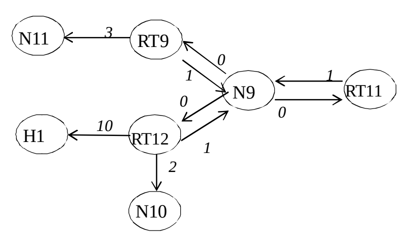


9

Networks vs. Graphs

- Graph *G(V,E)* – mathematical abstraction
  - (un)ordered pair of nodes *V* and edges *E*
  - weighted graph: $W: E \rightarrow R$
  - (s,d) Path: sequence of edges from *s* to *d*
  - Path cost: sum of edge costs
- Algorithms usually defined on graphs
- Note the modeling of the broadcast link N9



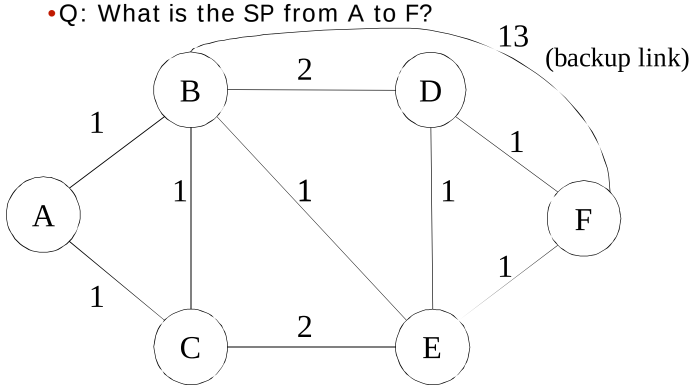


10

Shortest Path (SP) Problem

- Given weighted graph *G(V,E)* and nodes (*s*,*d*)
  - *Weight denotes cost*
- Find an (*s*,*d*) path with minimal path cost

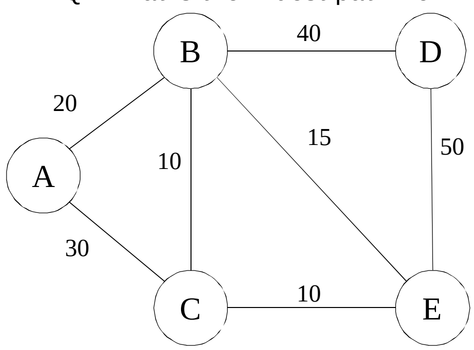
- Equal cost multipath (ECMP)
  - Set of paths with the minimal cost
- Q: What is the SP from A to F?



Alternative: Widest Path Problem

- Numbers denote *width*
  - *e.g., available* bandwidth
- Find path with maximum width
  - Also called ”Unsplittable maximum flow” problem
- SP algorithms can be modified to solve widest path problem

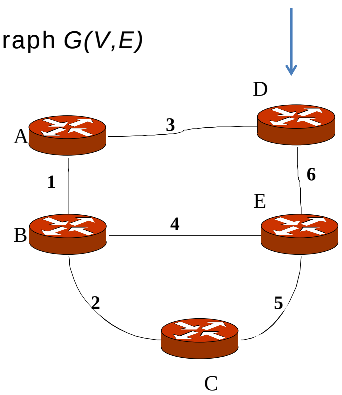
- Q: What is the widest path from A -> E?



Bellman-Ford Algorithm

Find shortest path from s to all nodes in digraph $G(V,E)$

1) Initialization:
  $d[s]:=0;$
  $\forall v \in V \setminus \{s\}\ d[v]:=\infty;$
  $pred[v]:=null;$
2) Iterative approximation:
  for $i=1$ to $|V|-1$ do
    for each $(u,v) \in E$
      if $d[u] + w_{(u,v)} < d[v]$ do
        $d[v] := d[u] + w_{(u,v)};$
        $pred[v] := u;$



Algorithm complexity (w/o negative cycle) $O(|E||V|)$

Bang-Jensen, Gutin, "Digraphs: Theory, Algorithms and Applications," Springer, 2007

13

Bellman-Ford algorithm and Distance-Vector protocols

- Distributed Bellman-Ford algorithm
  - Every router r
    - send a list of distance-vectors d(r,v) (route with cost) to each neighbor $n \in N(r)$ periodically
    - select the route with smallest metric (positive integer)
      - if d(r,v)>d(r,n)+d(n,v) then d(r,v)=d(r,n)+d(n,v) and nexthop(v)=n
      - ...?
- Protocols that use Bellman-Ford are called *Distance-vector* protocols

14

Distributed Bellman-Ford Algorithm and DV protocols

Dest	Cost	NextHop
...	...	...

- Data structure at node *r*: *"distance vector" table*
  - –One entry for every destination *d* in the network
  - –For every *d* stores the metric *M(r,d)* (distance) and the next-hop $n \in N(r)$
- Periodic message exchange
  - –Send the table (distance vector) to all neighbors
- For each update that comes in from a neighbor $n' \in N(r)$ (with a metric *M(n',d)* to *d*)
  1. Compute *m=M(r,n')+M(n',d)*
  2. if *(m<M(r,d)) then n=n', M(r,d)=m*
  3. *elseif (n=n') then M(r,d)=m   %new value from same*

- In protocols M is bounded, typically to 16
  - – The upper bound is defined as unreachable (infinity)

15

Example: Distance-vector

A's initial state: (directly connected networks)

Dest	Cost	NextHop
B	1	-
D	3	-

A distributes this DV to its neighbours (B and D)

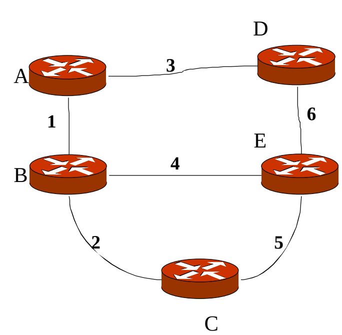
A receives B's (initial) distance vector

Dest	Cost
A	1
C	2
E	4

A's state after merging B's DV:

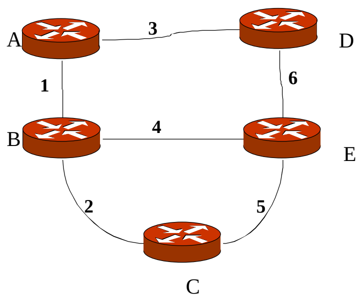
Dest	Cost	NextHop
B	1	-
C	3	B
D	3	-
E	5	B

A distributes this DV to its neighbours (B and D)



16

Example: Complete and final state



	A	B	C	D	E
A		1		3	
B	1		2		4
C		2			5
D	3				6
E		4	5	6	

Link metric matrix

A's Distance-Vector

	A	B	C	D	E
A	0				
B		0			
C			0		
D				0	
E					0

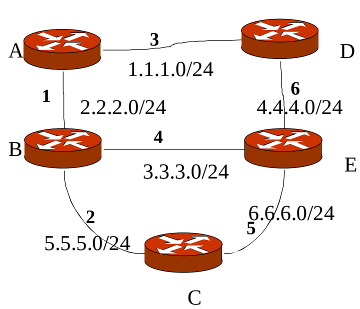
Initial state

	A	B	C	D	E
A	0	1	3	3	5
B	1	0	2	4	4
C	3	2	0	6	5
D	3	4	6	0	6
E	5	4	5	6	0

Final state

Going to real networks

- IP networks require destinations and nexthops (not just nodes)
  - –Destinations are networks (e.g., 192.16.32.0/24)
  - –Next-hops are IP addresses (e.g., 192.16.32.1)
- Suppose the topology changes, e.g., routers, links crash?
  - –Use timers (counters) and age the entries
  - –If you do not hear from a router in (e.g.) 180s, mark it as invalid
  - –Send updates every (e.g.) 30s



Dest	Cost	NextHop
1.1.1.0/24	3	-
2.2.2.0/24	1	-
3.3.3.0/24	5	2.2.2.2
4.4.4.0/24	9	1.1.1.2
5.5.5.0/24	3	2.2.2.2
6.6.6.0/24	8	2.2.2.2

Converged routing state of A

Distance Vector Problem: Count to Infinity (Two-node instability)

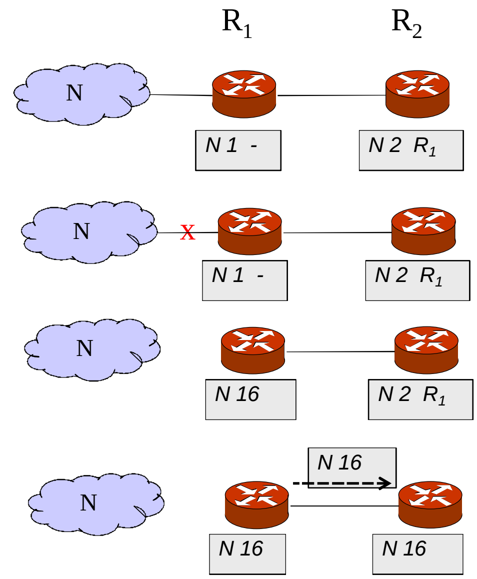
Initially, R_1 and R_2 both have a route to *N* with metric 1 and 2, respectively.

The link between R_1 and *N* fails.

R_1 removes its route to *N*, by setting its metric to 16 (infinity).

One of two things can happen:

1) R_1 reports its route to R_2. Everything is fine.



Distance Vector Problem: Count to Infinity (Two-node instability)

2) R_2, which still has a route to N, advertises it to R_1.

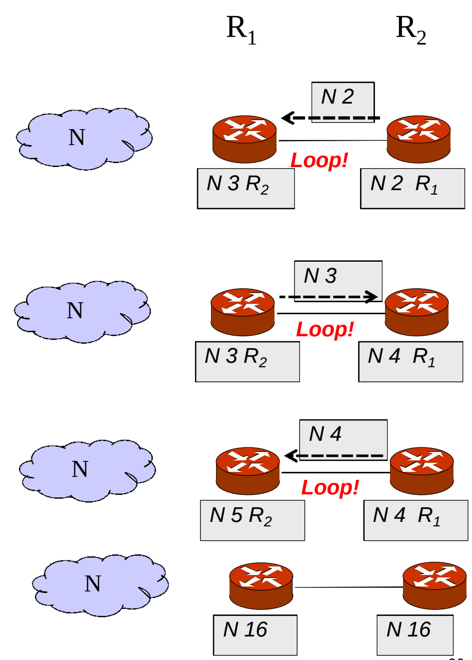
Things start to go wrong: packets to N loop until their TTL expires!

Eventually (~10-20s), R_1 sends an update to R_2. The cost to N increases, but the loop remains.

Yet some time later, R_2 sends an update to R_1.

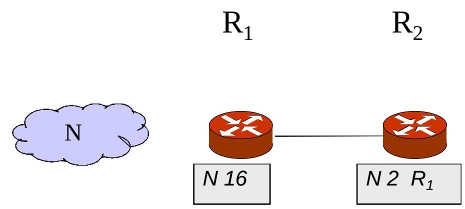
…

Finally, the cost reaches infinity at 16, and N is unreachable. The loop is broken!

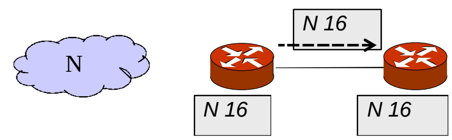


Solution: Split Horizon

- *Do not send routes back over the same interface from where the route 'arrived'.*
  - –Helps to avoid “mutual deception”: two routers tell each other they can reach a destination via each other.



R_2, does not announce the route to N to R_1 since that is where it was learnt from.

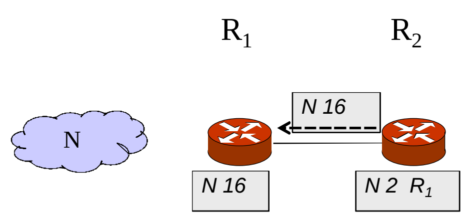


Eventually, R_1 reports its route to R_2 and everything is fine.

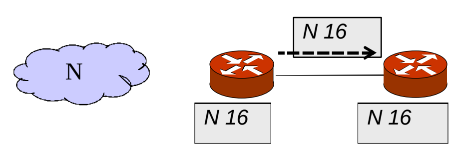
21

Solution: Split Horizon + Poison Reverse

- *Advertise reverse routes with a metric of 16 (i.e., unreachable)*
  - Does not add information but breaks loops faster
  - Adds protocol overhead



R_2 always announces an unreachable route for N to R_1.

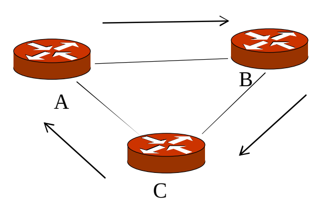


Eventually, R_1 reports its route to R_2 and everything is fine.

22

Remaining problems

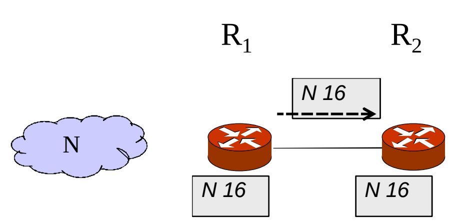
- More than two routers involved in mutual deception
  - A may believe it has a route through B, B through C, and C through A
- Split horizon with poison reverse does not help ☹



Solution: Triggered Update

- *Send out update immediately when metrics change*
  - Only the changed route, not the complete table
  - This may lead to a cascade of updates
    - Apply the rule above recursively!
    - Therefore, triggered updates are not allowed more often than, for example 1-5 seconds.
- *Must* use triggered update when deleting routes (M=16)
- May use triggered update when changing routes (M changes)

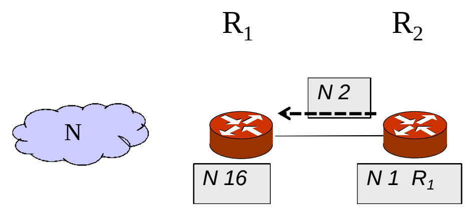
R_1 announces the broken link *immediately*



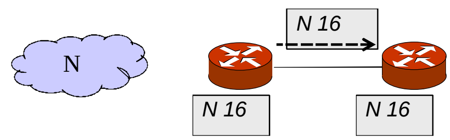
Solution: Hold Down

- *When a route is removed, no update of this route is accepted* for some period of time (hold-down time)
  - gives everyone a chance to remove the route.

R_1 ignores updates to N from R_2 for some period of time.



Eventually, R_1 sends the update to R_2.



25

Distributed Bellman-Ford and Path vector protocols

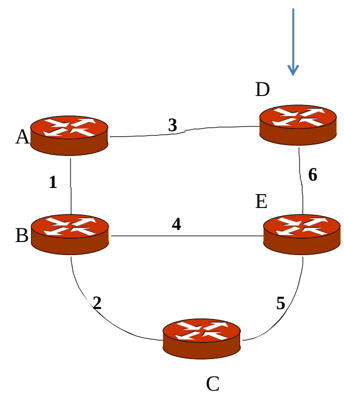
- Distance-vector = (destination, metric, next-hop)
  - Example:
    - –<dst: 10.1.10/24, metric: 5, nexthop: 10.2.3.4>
  - Convergence problems
    - –Example: count-to-infinity

- Path-vector = (destination, path, next-hop)
  - extends the information with a *path* to the destination
  - –Enables loop detection ⇒ avoid count-to-infinity
- Example:
  - –<dst: 10.1.10/24, path: r1,r2,r3, nexthop: 10.2.3.4>

27

Dijkstra's shortest path algorithm

- Data structure
  - Link-state database (the weighted graph $G(V,E)$)
  - Permanent set S
  - Tentative set Q



- Compute a shortest path delivery tree rooted in $r \in V$
  1. Initialization
     $d[r]=0$, $S=\{r\}$, $p=r$, $\forall v \in V \setminus \{r\}$ $d[v]=\infty$
  2. Expansion of permanent set
     $\forall v \in N(p)$ $d[v]=\min(d[v], d[p]+w_{(v,p)})$, $Q=Q \cup \{v\}$
  3. Find $p \in Q$ s.t. $d[p]=\min(d[v], v \in Q)$
     $S=S \cup \{p\}$, $Q=Q \setminus \{p\}$
  4. Go to (2)

- Computational complexity
  - $O(|V|^2)$->$O(|V|log|V|+|E|)$ using Fibonacci heap

Bang-Jensen, Gutin, "Digraphs: Theory, Algorithms and Applications," Springer, 2007
Fredman, Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms," J.ACM 34 (3): 596–615

28

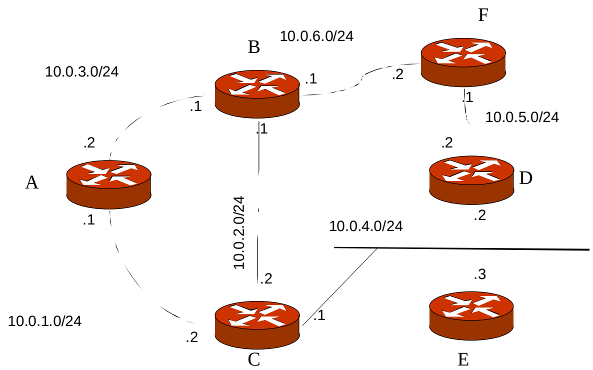
Dijsktra's Algorithm and Link-state Routing

- Every router spreads information about its links to its neighbors
- The information is flooded to every router in the routing domain
  - Every router has knowledge of the entire network topology
- Every router computes the SP to each prefix in the network
  - Dijkstra's algorithm

- Two well-known link-state routing protocols
  - OSPF - popular among organizations (KTH uses OSPF)
  - IS-IS - popular among operators (SUNET uses IS-IS)

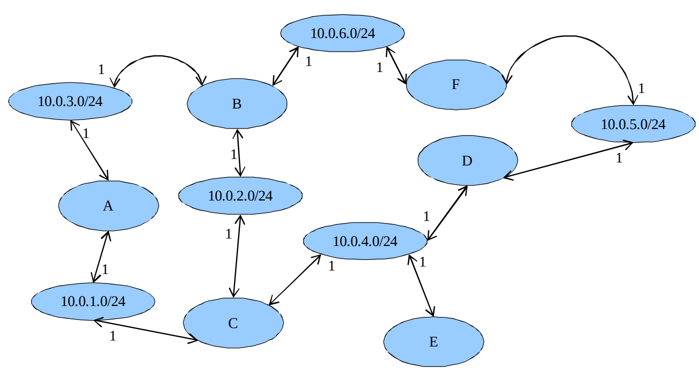
29

Example network



30

Example graph



31

Exercise: Dijkstra from A

Permanent set	Tentative set
A 0 -	10.0.3.0/24 1 - 10.0.1.0/24 1 -

Exercise: Dijkstra from A

Permanent set	Tentative set
A 0 - 10.0.3.0/24 1 -	~~10.0.3.0/24 1 -~~ 10.0.1.0/24 1 - B 1 -

Exercise: Dijkstra from A

Permanent set	Tentative set
A 0 - 10.0.3.0/24 1 - B 1 -	10.0.1.0/24 1 - ~~B 1 -~~ 10.0.2.0/24 2 B 10.0.6.0/24 2 B

Exercise: Dijkstra from A

Permanent set	Tentative set
A 0 - 10.0.3.0/24 1 - B 1 - 10.0.1.0/24 1 -	~~10.0.1.0/24 1 -~~ 10.0.2.0/24 2 B 10.0.6.0/24 2 B C 1 -

Exercise: Dijkstra from A

Permanent set	Tentative set
A 0 - 10.0.3.0/24 1 - B 1 - 10.0.1.0/24 1 - C 1 -	10.0.2.0/24 2 B 10.0.6.0/24 2 B ~~C 1 -~~ 10.0.2.0/24 2 C 10.0.4.0/24 2 C

36

Exercise: Dijkstra from A

Permanent set	Tentative set
A 0 - 10.0.3.0/24 1 - B 1 - 10.0.1.0/24 1 - C 1 - 10.0.2.0/24 2 B 10.0.2.0/24 2 C	~~10.0.2.0/24 2 B~~ 10.0.6.0/24 2 B ~~10.0.2.0/24 2 C~~ 10.0.4.0/24 2 C

Note: ECMP

ECMP: Equal Cost MultiPath. More than one path.

37

Exercise: Dijkstra from A

Permanent set	Tentative set
A 0 -	
10.0.3.0/24 1 -	
B 1 -	
10.0.1.0/24 1 -	
C 1 -	10.0.6.0/24 2 B
10.0.2.0/24 2 B	
10.0.2.0/24 2 C	
10.0.4.0/24 2 C	~~10.0.4.0/24 2 C~~
	D 2 C
	E 2 C

38

Exercise: Dijkstra from A

Permanent set	Tentative set
A 0 -	
10.0.3.0/24 1 -	
B 1 -	
10.0.1.0/24 1 -	
C 1 -	10.0.6.0/24 2 B
10.0.2.0/24 2 B	
10.0.2.0/24 2 C	
10.0.4.0/24 2 C	
E 2 C	~~D 2 C~~
D 2 C	~~E 2 C~~
	10.0.5.0/24 3 C

39

Exercise: Dijkstra from A

Permanent set	Tentative set
A 0 - 10.0.3.0/24 1 - B 1 - 10.0.1.0/24 1 - C 1 - 10.0.2.0/24 2 B 10.0.2.0/24 2 C 10.0.4.0/24 2 C E 2 C D 2 C 10.0.6.0/24 2 B	~~10.0.6.0/24 2 B~~ 10.0.5.0/24 3 C F 2 B

Exercise: Dijkstra from A

Permanent set	Tentative set
A 0 - 10.0.3.0/24 1 - B 1 - 10.0.1.0/24 1 - C 1 - 10.0.2.0/24 2 B 10.0.2.0/24 2 C 10.0.4.0/24 2 C E 2 C D 2 C 10.0.6.0/24 2 B F 2 B	10.0.5.0/24 3 C ~~F 2 B~~ 10.0.5.0/24 3 B

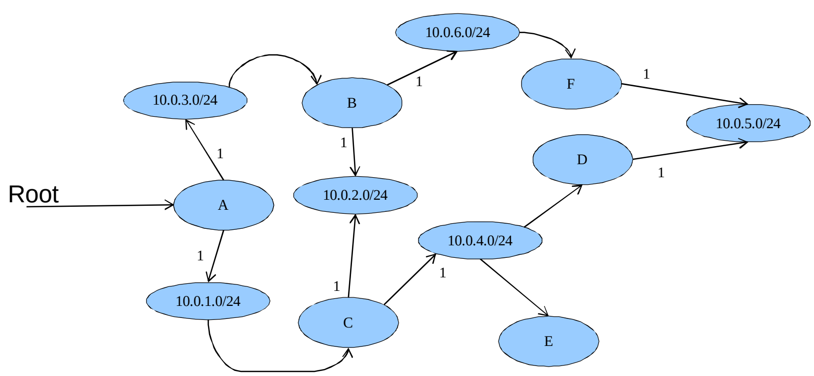
Exercise: Dijkstra (complete)

Permanent set	Tentative set
A 0 -	
10.0.3.0/24 1 -	
B 1 -	
10.0.1.0/24 1 -	
C 1 -	
10.0.2.0/24 2 B	
10.0.2.0/24 2 C	
10.0.4.0/24 2 C	
E 2 C	
D 2 C	
10.0.6.0/24 2 B	~~10.0.5.0/24 3 C~~
F 2 B	
10.0.5.0/24 3 B	~~10.0.5.0/24 3 B~~
10.0.5.0/24 3 C	

Note: ECMP

Exercise: Dijkstra tree graph view

- Compare with table view in the previous slide
- Note the ECMP routes to 10.0.2.0/24 and 10.0.5.0/24

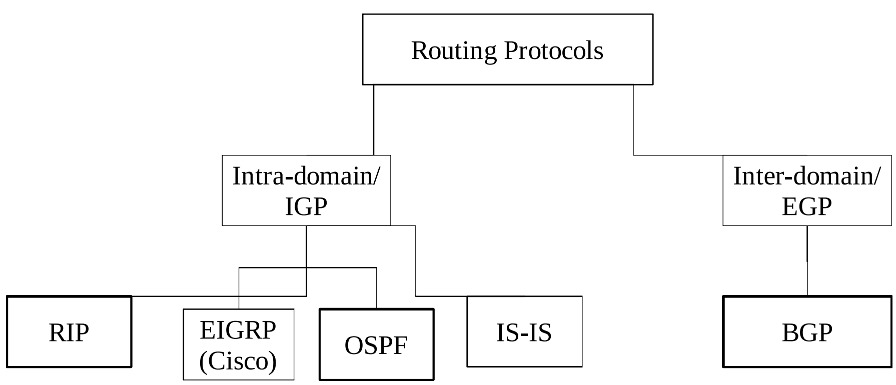


Link-state vs. Distance-vector

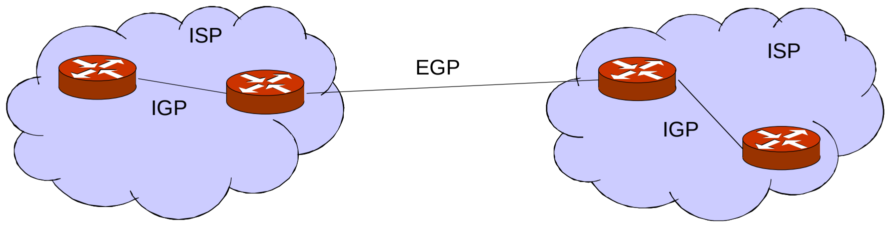
- Distributed database model
- Advantages
  - –More functionality due to distribution of original data
  - –No dependency on intermediate routers
  - –Easier to troubleshoot
  - –Fast convergence: when the network changes, new routes are computed quickly
  - –Less bandwidth consuming

- Distributed processing model
- Advantages
  - –Less complex – easier to implement and to administer
  - –Needs less memory

Popular Unicast Routing Protocols



IGP/EGP



IGP

- –Interior Gateway Protocol.
- –Runs within a network/domain (intra-domain)
- –Handles *internal* routes within a domain
- –Examples: RIP, OSPF, IS-IS

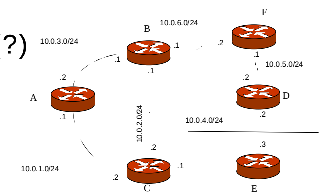
EGP

- –Exterior Gateway Protocol.
- –Primarily exchanges routes between networks/domains (inter-domain)
- –Handles *external* routes
- –Examples: BGP, static routing
- –Note: an EGP can handle *external* routes *within* a domain (IBGP)

Intra-domain routing protocols

Routing Information Protocol – RIP2

- Distance vector routing protocol
  - RIP-1 (RFC 1058), RIP-2 (RFC 2453), RIPng (RFC 2080)
- Metric: hop count
  - 1: directly connected
  - 16: infinity
- Supports networks with diameter ≤ 15
- Timeout timer
  - Purge routes that are not refreshed
- Authentication...
- Messages carried in UDP datagrams (?)
  - –Broadcast (RIP-1)
  - –IP Multicast (RIP-2): 224.0.0.9
  - –IPv6 Multicast (RIPng): FF02::9



Disadvantages with RIP

- Slow convergence
  - Changes propagate slowly
  - Each node only speaks ~every 30 seconds; information propagation time over several hops is long
- Instability
  - –After a router or link failure RIP takes minutes to stabilize
- Hop count may not be the best indicator for the best route
- Network diameter ≤ 15
  - The maximum useful metric value is 15
- Uses much bandwidth
  - –Sends the whole distance vector in updates (not when triggered)

Why would anyone use RIP?

–Easy to implement

–Generally available

–Implementations have been rigorously tested

–Simple to configure

–Little overhead (for small networks)

Quiz

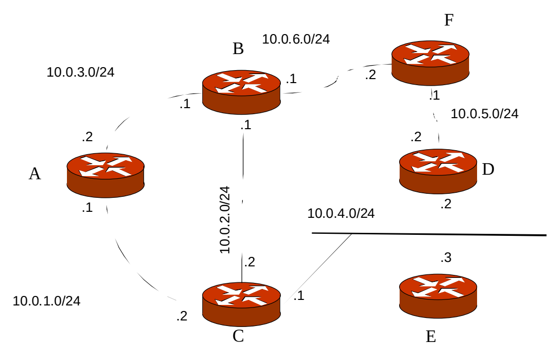
Consider the network shown below and assume that RIPv2 is used for routing. Initially, at *t=0*, all distance vectors are based on local information only. Assume that there is only one router speaking at a time, and each router speaks once every 30 seconds. What is the shortest amount of time need for RIPv2 to converge?

a) 1 s

b) 30 s

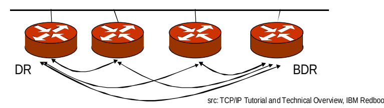
c) 60 s

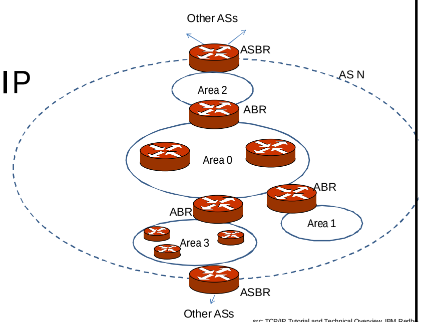
d) 2 hours



Open Shortest Path First protocol (OSPF)

- Link-state routing protocol
  - OSPFv2 (RFC2328), OSPFv3 (RFC5340)
- Metric: arbitrary
  - Often related to link speed (inverse proportional)
- Scaling achieved through hierarchy
  - Every network segment has 1 designated router (+1 backup) – DR, BDR
  - AS split into areas – use Dijkstra for an area
- Authentication...
- Messages carried directly on top of IP
  - IP Multicast: 224.0.0.5
  - IPv6 Multicast: FF02::5





OSPF: Link-state and the Protocol

- Router describes its environment
  - Networks (links) that it is connected to ("link state")
  - Link-states are the elements of the distributed database
- OSPF protocol components

1) *Hello* protocol
   - Detection of neighboring routers
   - Election of *designated router* (and backup) → adjacency

2) *Exchange* protocol
   - Exchange link-state between adjacent routers

3) Reliable *flooding*
   - When links change/age: send update to adjacent routers and flood *recursively*

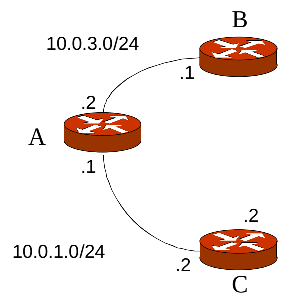
4) Shortest path calculation
   - Compute shortest path tree to all destinations using Dijkstra's algorithm

53

Example: OSPF link state

- Every router creates the link-state of its connected links (router LSA)
- Every DR creates the link-state of each of its networks (network LSA)

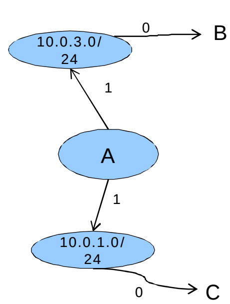
- Assume A is the designated router (DR) of the two segments
- Translate the network below to link states (from A's point of view)



Example: OSPF link state

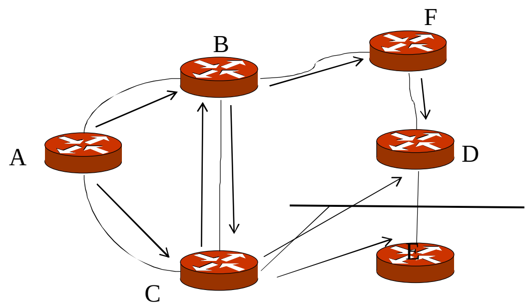
- Every router creates the link-state of its connected links (router LSA)
- Every DR creates the link-state of each of its networks (network LSA)

- Example:
  - *A* is connected to two 'transit' links, connecting to B and C, respectively
  - *A* is the *designated router* of these sub-networks
    - The transit links in this case 'belong' to *A*
  - *A* distributes three link-states
    - –One for A itself (it is connected to two transit networks) – router LSA
    - –One for each transit link (routers connected to the link) – network LSA



OSPF Link-state Flooding

- Every router distributes its link-state to all other routers
  - Initially
  - After link/router changes
  - Periodically (every ~30 mins)
- Reliable flooding to all routers
  - –OSPF implements error control (flooding is reliable)
  - –The most complex part of OSPF (not Dijkstra!)
- Example: 'A' floods its link-state by sending it to its neighbors, who in turn distribute it to their neighbors, etc



56

Quiz

Consider the the network shown in the figure below. What is a valid set of designated routers?

a) A, B, C, D

b) A, F, C, E

c) A, F, C

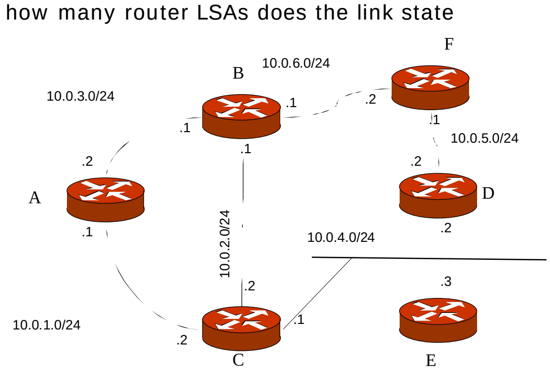
e) All of the above

How many network LSAs and how many router LSAs does the link state database consist of?

a) 6 and 6

b) 3 and 8

c) it depends on the DRs



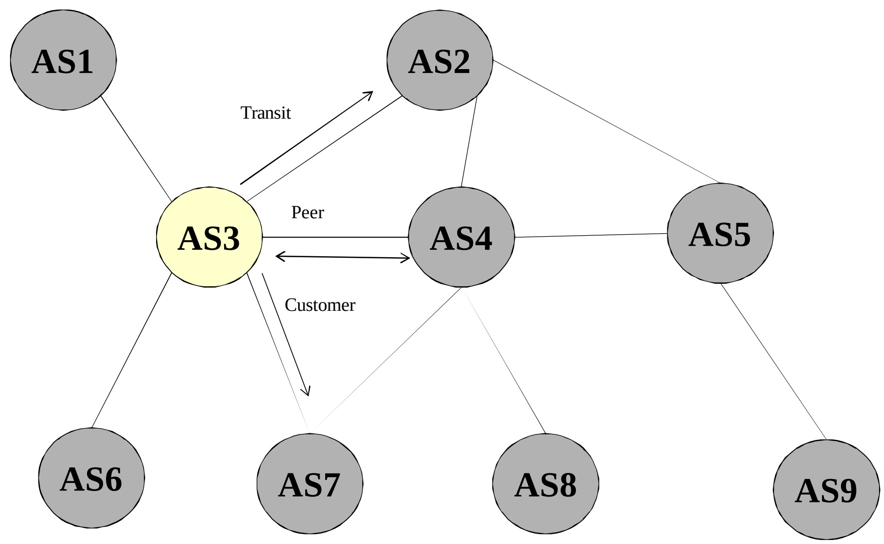
Inter-domain routing

Autonomous Systems (AS)

- A set of routers that
  - have a single routing policy
  - run under a single technical administration
- AS may be
  - A single network or group of networks
  - University, business, organization, operator
- All interior policies, protocols, etc are hidden within the AS
  - Abstracted by outside world as an Autonomous System

- Every AS has an Autonomous System Number (ASN)
  - Assigned by RIR from IANA
  - Two bytes long: 0-65535
    - Example: ASN 1653 for SUNET
  - Transitioning to four-byte ASNs
    - RFC 4893: BGP Support for Four-octet AS Number Space

AS peering and transit relations



60

Whois example

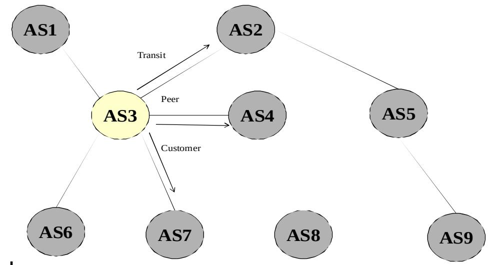
```
gelimer.kthnoc.net> whois -h whois.ripe.net AS1653
aut-num:        AS1653
as-name:        SUNET
descr:          SUNET Swedish University Network
import:         from AS42 accept AS42
export:         to AS42 announce AS-SUNET
import:         from AS702 accept AS702:RS-EURO AS702:RS-CUSTOMER
export:         to AS702 announce AS-SUNET
import:         from AS2603 accept any                    %NORDUnet
export:         to AS2603 announce AS-SUNET
import:         from AS2831 accept AS2831 AS2832
export:         to AS2831 announce any
import:         from AS2833 accept AS2833
export:         to AS2833 announce any
import:         from AS2834 accept AS2834
export:         to AS2834 announce any

gelimer.kthnoc.net> whois -h whois.ripe.net AS-SUNET
as-set:         AS-SUNET
descr:          SUNET AS Macro
descr:          ASes served by SUNET
members:        AS1653, AS2831, AS2832, AS2833, AS2834, AS2835, AS2837
members:        AS2838, AS2839, AS2840, AS2841, AS2842, AS2843, AS2844
members:        AS2845, AS2846, AS3224, AS5601, AS8748, AS8973, AS9088
members:        AS12384, AS15980, AS16251, AS20513, AS25072, AS28726
members:        AS-NETNOD
```

(Edited example)

61

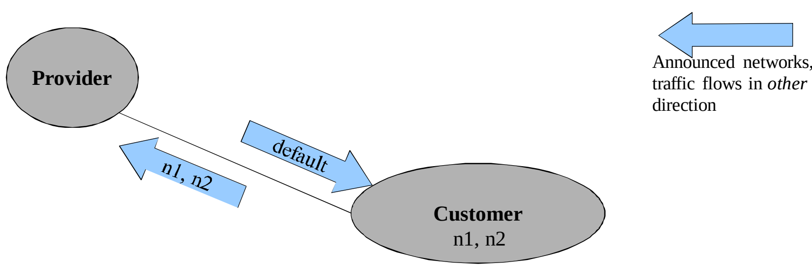
Inter-AS relations



- Definition through prefix sets
  - Customer prefix set
  - Peering prefix set
  - Transit prefix set
- Example rules
  - Customer prefixes: announce to transit and peers
  - Peer and transit prefixes: announce to customers (not to peers)
  - Prefer prefixes from peers over prefixes from transit
  - Do not accept illegal (e.g., RFC 1918) or unknown prefixes from customers
  - Load balance over several transit providers
  - Filter traffic (e.g., src addresses) according to the prefixes announced

62

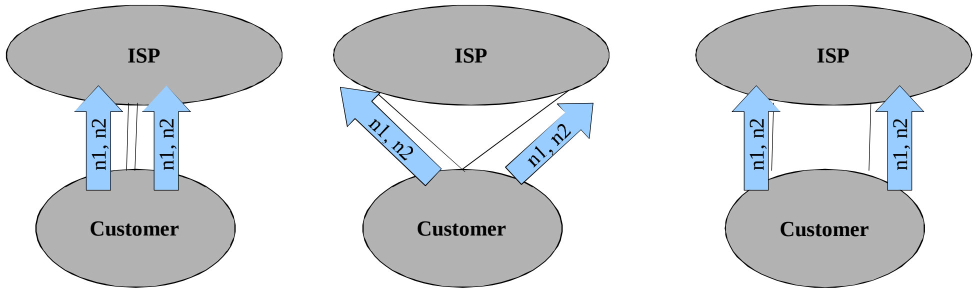
ISP Relations: Customer Stub AS



- Typical customer/provider topology
- Customer
  - Can use address block of provider
  - Does not need to be a separate AS
  - Can use default route to reach the Provider and Internet
- Routing
  - Typically static routing
  - Can be dynamic (BGP)

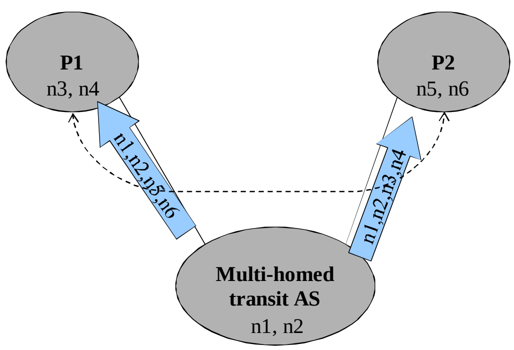
63

ISP Relations: Multi-homed customer



- Multi-homing
  - Load sharing or geographical traffic distribution
  - Reliability and performance
- Multi-homed non-transit AS
  - Non-transit AS does not allow external traffic to pass through
- What to think about
  - How to announce the prefixes
  - Default routes
  - Symmetrical routing
  - Packet filtering
  - Address aggregation, etc

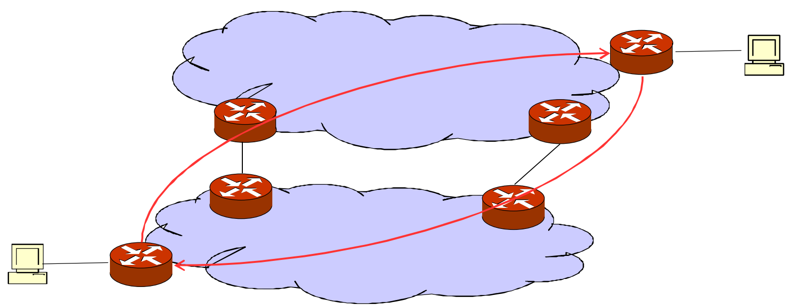
ISP Relations: Provider Multi-homed Transit AS



- Transits traffic through own network
- Most general configuration - Internet provider

Policy example: Asymmetric Routing

- A rule rather than an exception:
  - To- traffic and from- traffic take different paths
- Hot-potato routing
  - Send traffic out of your AS as soon as possible
- Cold-potato
  - Try to keep your traffic as long as possible.



66

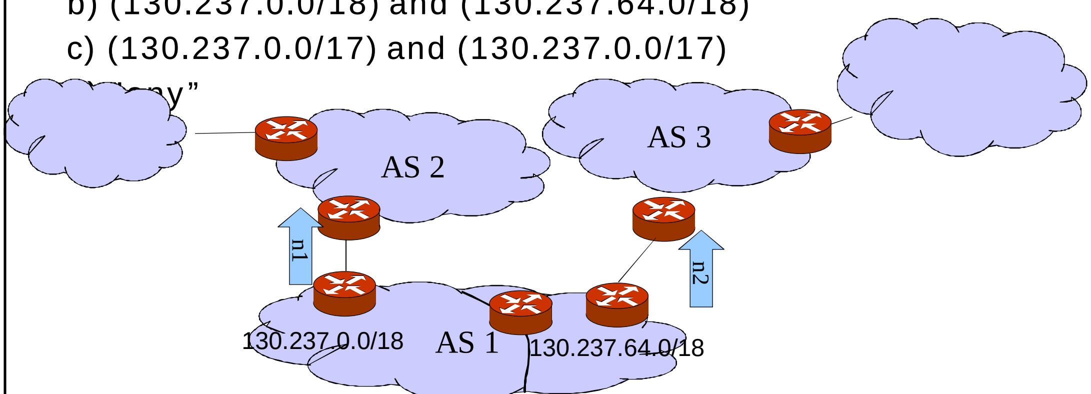
Quiz

AS1 would like traffic to .0.0/8 to be delivered via AS2 and traffic to .64.0/18 via AS3. What network(s) does AS1 announce to AS2 and AS3?

a) (130.237.0.0/18, 130.237.0.0/17) and (130.237.64.0/18, 130.237.0.0/17)

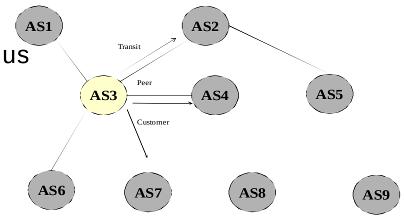
b) (130.237.0.0/18) and (130.237.64.0/18)

c) (130.237.0.0/17) and (130.237.0.0/17)



67

Inter-domain routing

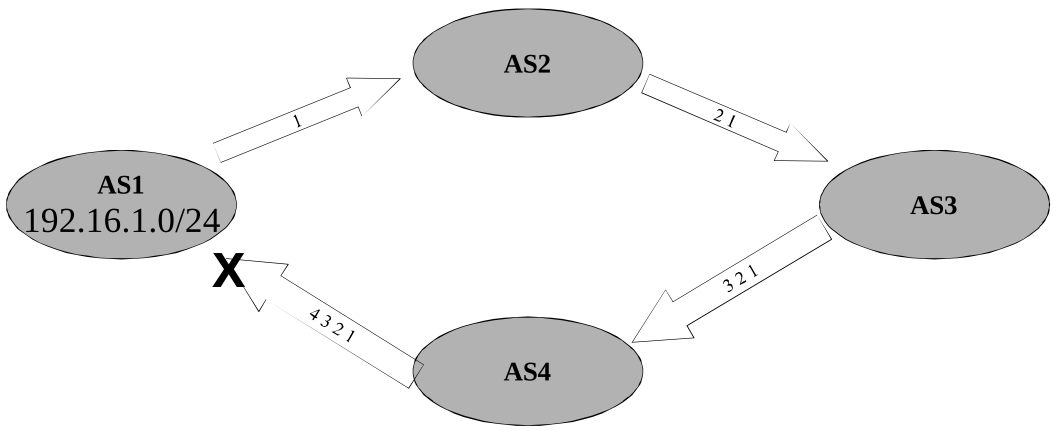


- Objective
  - Bind together tens of thousands of autonomous IP networks that constitute the Internet
- Requirements
  - Scalability, efficiency
  - Express relations
  - Support policy decisions and
- Perspective of a network
  - Spread routing information to the outside world
    - Originate and aggregate address prefixes
    - Announce prefixes to other domains
    - Tag prefixes with routing information
  - Receive information from the outside world
    - Receive and choose (filter) between prefixes from other domains
  - Transfer information through your routing domain
    - Received information from one domain may be transferred (and possibly modified) to other domains

Border Gateway Protocol (BGP) v4

- Path-vector routing protocol
  - Border Gateway Protocol version 4 (RFC4271)
  - Path vector consists of AS:s, not IP addresses
    - Hides internal structure in the domains
    - Loop detection only on AS-numbers!
    - Example: <dst: 10.1.10/24, path: AS1:AS3:AS5, nexthop: 10.2.3.4>
- Used between domains (AS:s)
  - Views the Internet as a collection of AS:s
- Supports the *destination-based* forwarding paradigm
  - Other relations are not expressed: sources, tos, link load
- Uses TCP for data transmission between BGP peers
- Maintains a database (RIBs) of network layer reachability information
- Tags destinations with *path attributes*
  - Describe different properties of the destination (e.g., preferences)
  - Can express and enforce policy decisions at AS level

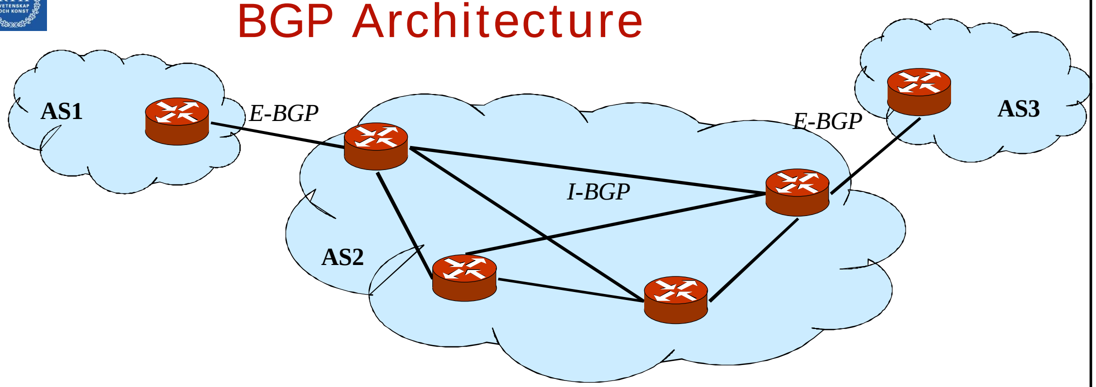
AS-PATH attribute



- AS-PATH used to break loops (between AS:s)
- AS1 announces 192.16.1.0/24 to AS2 and detects its own ASN when received from AS4
- AS-PATH is the most well-known path-attribute, there are several others

70

BGP Architecture

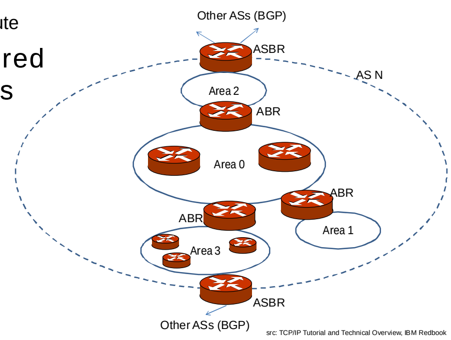


- BGP has two uses/variants
  - –E-BGP: exchanges external routes between border routers *between* AS:s
  - –I-BGP : synchronises *external* routes *within* an AS (IGP takes care of internal routes)
- BGP interacts with Internal routing (OSPF/IS-IS/RIP/...)
  - –*Redistributes* internal / external routes between the two protocols

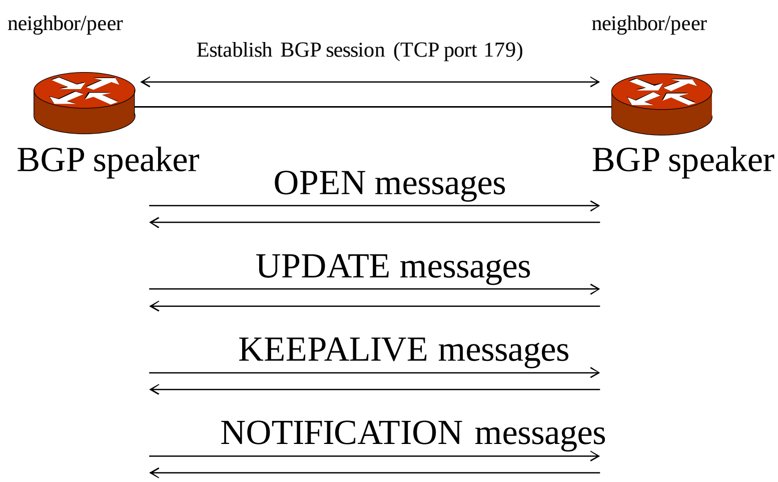
71

Redistribution of routing information

- If several protocols are running on the same router
  - –E.g., an OSPF as interior and BGP as exterior
- The router can distribute routes from one protocol to another
  - –Interior routes need to be advertized to the Internet
    - Typically these routes are aggregated
  - –Exterior routes (or a default) may need to be injected into the interior network
    - But only a subset – the backbone tables are very large
    - Necessary for domain carrying *transit* traffic
    - Not necessary for a domain using only a default route
- Typically, redistributed routes are filtered in different ways due to routing policies

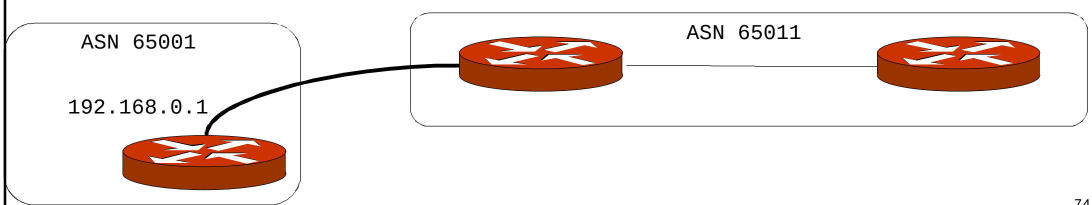


BGP Operation



Example: JunOS BGP configuration

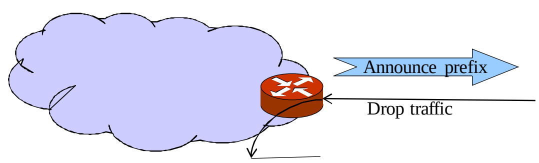
```
routing-options {
  autonomous-system 65011;
}
protocols {
  bgp {
     group EXTERN {
       type external;
       peer-as 65001;
       export MYNETWORK;
       neighbour 192.168.0.1;
       }
     }
  }
}
```



74

Black-holing

- Black-holing
  - Announce prefix, but traffic to the prefix is dropped (not delivered)
- Loops: circular announcements causing packet loops
  - –TTL is decremented until packet drops -> same effect as black-holing
- Reasons:
  - –Transient errors due to long convergence (see count-to-infinity in distance-vector)
  - –Misconfigurations
  - –Attacks (DOS, man-in-the-middle)
  - –Response to attacks: create a black-hole for attacked prefixes which removes DOS traffic



77

Routing Summary

- Routing = computation of “best” paths
  - for use in forwarding table
- Algorithms for shortest path computation
  - Bellman-Ford
  - Dijkstra
- Intra-domain routing protocols
  - Distance-vector (RIP, ...)
  - Link-state (OSPF, ...)
- Inter-domain routing protocol
  - BGP (Path-vector)

78