Intra-Domain Routing

Richard T. B. Ma
School of Computing
National University of Singapore

CS 3103: Compute Networks and Protocols

What is an Autonomous System?

- Usually a network of routers redundantly interconnected
- Controlled by a single administrative domain (one company could have several ASNs but a given ASN is typically controlled by a specific group)
- Common routing protocol and policy
- Identified by a globally unique AS Number (ASN)

Example: Singtel

Navigation
Home Page
Logout

Your Records
Peering Record
User Account

Search Records
Networks
Exchange Points
Facilities
Common Points

Suggestions
Comments
New Exchange
New Facility

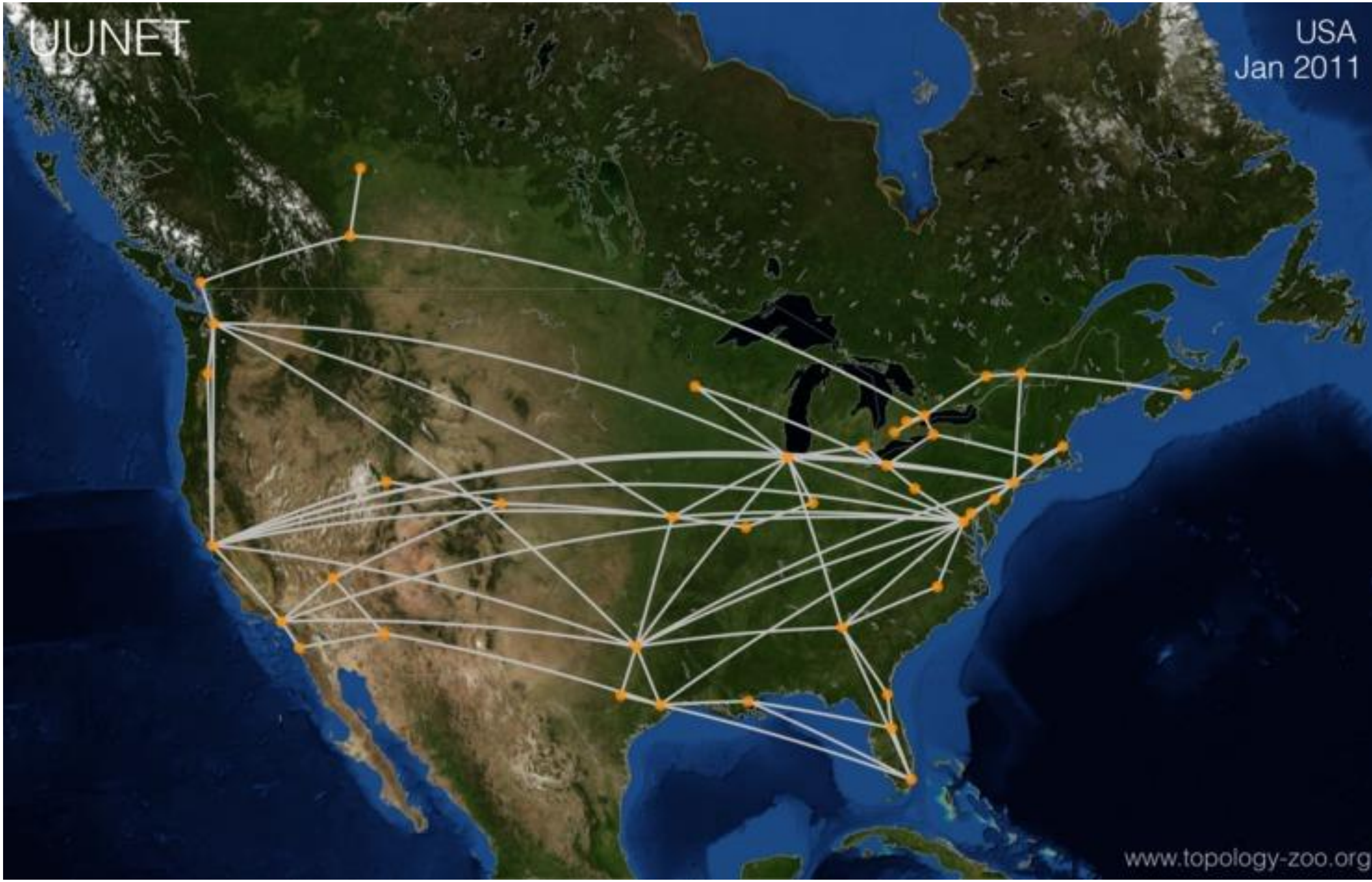
Search Peering Networks

Company Name	Sing
Network Type	Select Value
Traffic Levels	Select Value
Traffic Ratio	Select Value

Peering Networks Search Results

Company Name	ASN	General Policy
HousingCenter	28707	Open
Institute of Molecular and Cell Biology, Singapore	38181	Open
Level3 formerly Global Crossing	3549	Restrictive
NTT Communications - NTT Singapore	17645	Selective
RisingNet.com	33211	Open
Singapore Internet Exchange (SGIX)	55518	Open
Singapore Telecommunications Limited	7473	Selective
Single Host Internet Brasil	52919	Open

AS Topology: UUNET (AS701)



AS Topology: Viatel, Renater

http://www.viatel.com/

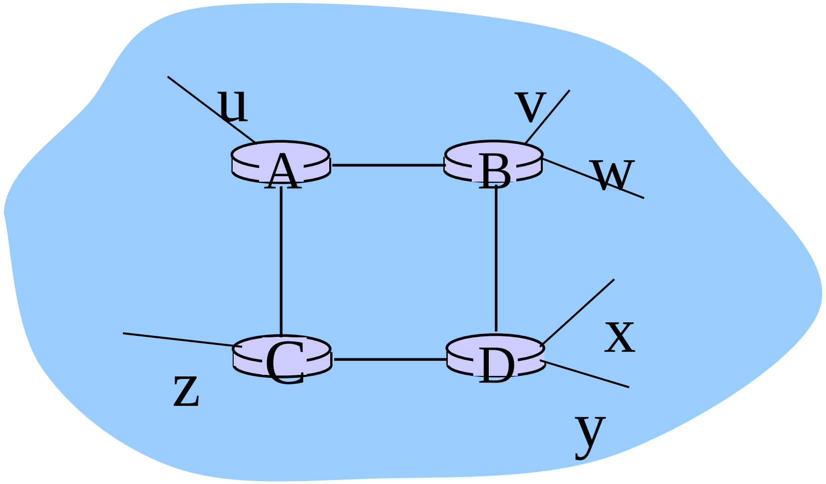
https://www.renater.fr/

Intra-AS Routing

- also known as Interior Gateway Protocols (IGP)
- most common Intra-AS routing protocols:
  - RIP: Routing Information Protocol (based on Bellman-Ford algorithm)
  - OSPF: Open Shortest Path First (based on Dijkstra algorithm)
  - IGRP: Interior Gateway Routing Protocol (Cisco proprietary)

RIP (Routing Information Protocol)

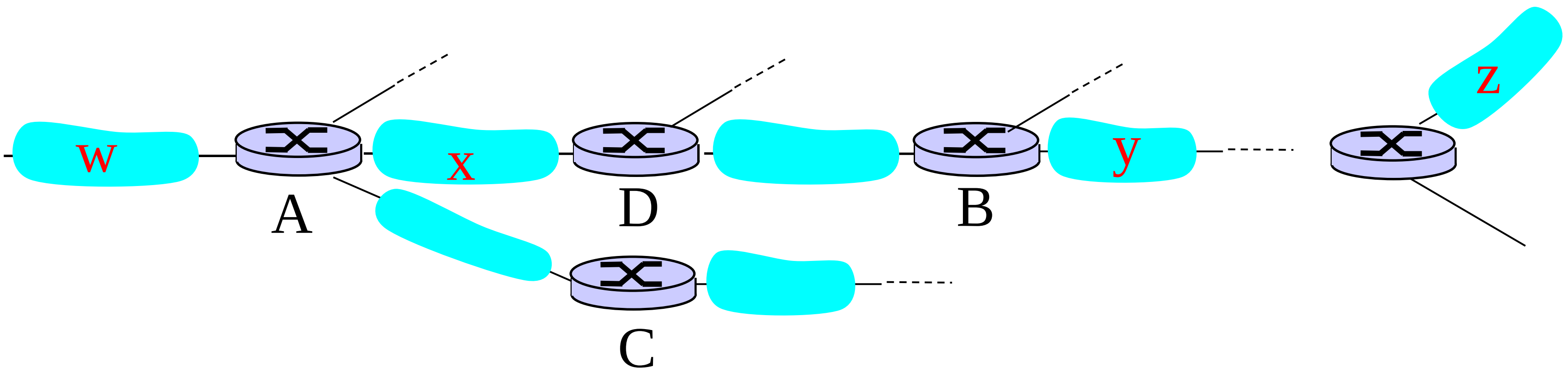
- included in BSD-UNIX distribution in 1982, RFC 2453
- distance vector algorithm
  - distance metric: # hops, each link has cost 1
  - DVs exchanged with neighbors every 30 sec in response message (aka advertisement)
  - each advertisement: list of up to 25 destination *subnets* *(in IP addressing sense)*



from router A to destination subnets:

subnet	hops
u	1
v	2
w	2
x	3
y	3
z	2

RIP: High-level idea



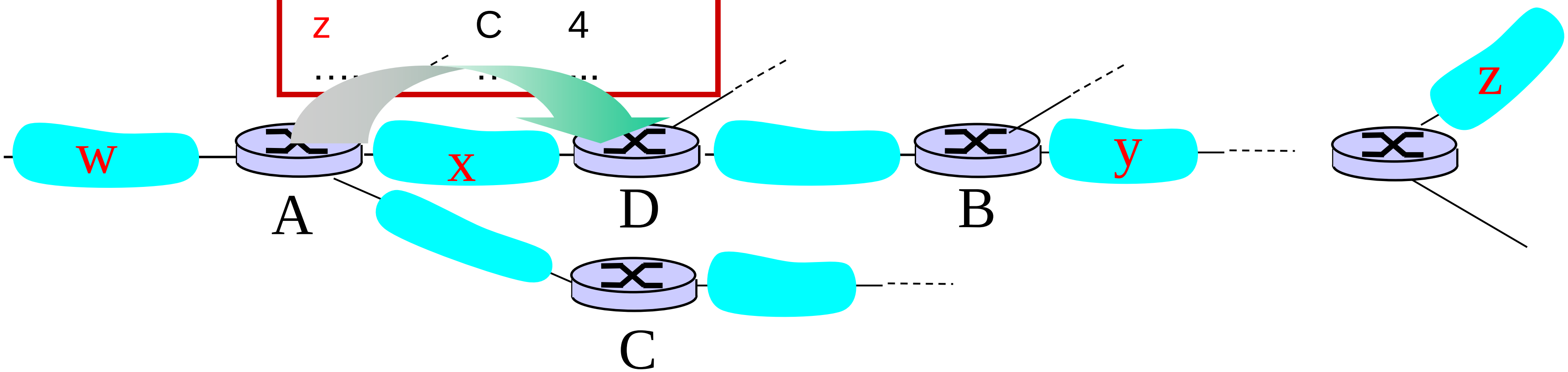
routing table in router D

destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	B	7
x	--	1
....	....	....

RIP: High-level idea

A-to-D advertisement

dest	next	hops
w	-	1
x	-	1
z	C	4
....	...	....

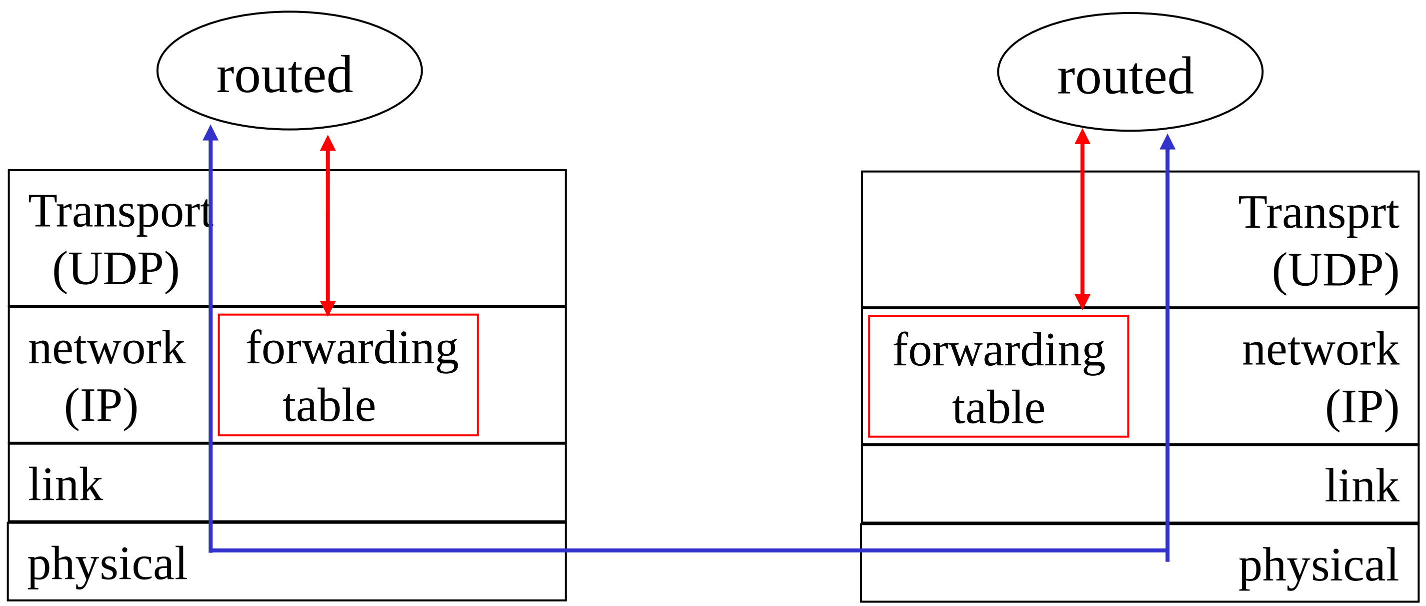


routing table in router D

destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	~~B~~ A	~~7~~ 5
x	--	1
....	....	....

RIP Table processing

- RIP routing tables managed by **application-level** process called route-d (daemon)
- advertisements sent in UDP packets, periodically repeated



RIP Messages

- Communication between RIP software elements is done with *RIP messages*.
- These messages are sent using the UDP, with reserved port 520
  - even though RIP is considered part of layer 3 like other routing protocols, it behaves more like an application in terms of how it sends messages.

Message Types

- ***RIP Request***:
  - A message sent by a router to another router asking it to send back all or part of its routing table.
- ***RIP Response***:
  - A message sent by a router containing all or part of its routing table.
  - This message is ***not*** sent just in response to an *RIP Request* message. So it's not really a good name… so, it is also called ***RIP advertisement***

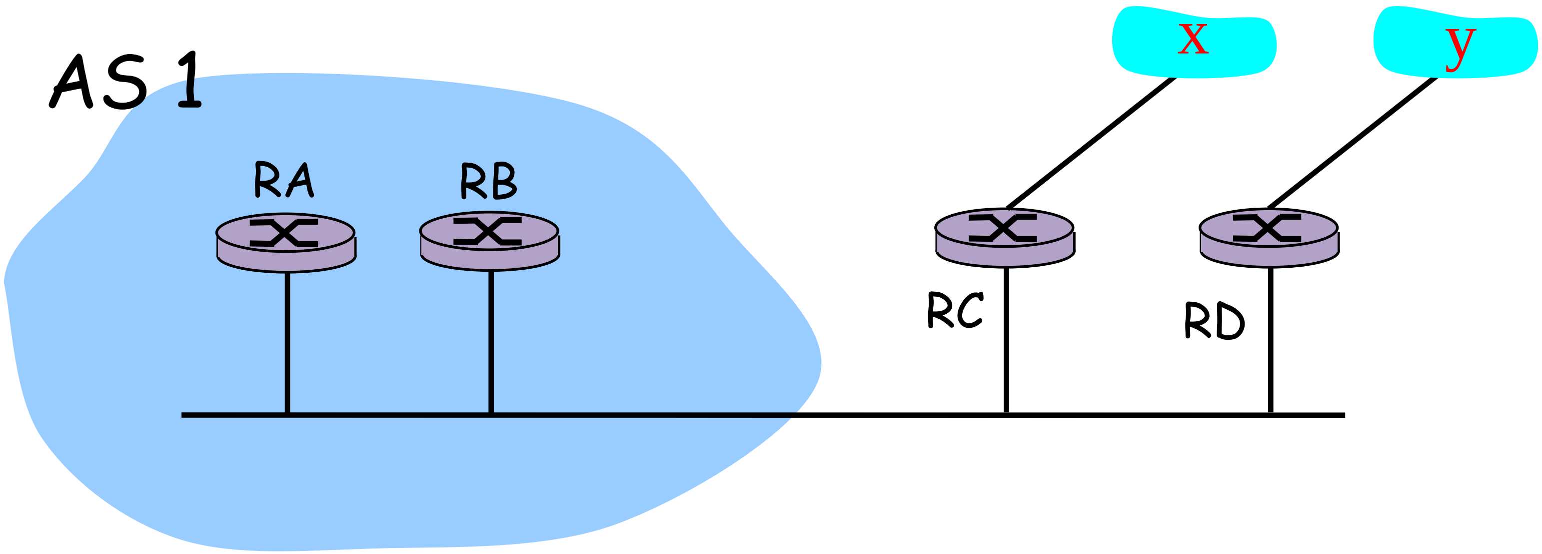
RIP-2 Message Format

<table>
<tr><td>Command (1)</td><td>Version (1)</td><td>Unused and must be zero (2)</td></tr>
<tr><td colspan="2">Address family identifier (2)</td><td>Route tag (2)</td></tr>
<tr><td colspan="3">IP address (4)</td></tr>
<tr><td colspan="3">RIP Entry #1 Subnet mask (4)</td></tr>
<tr><td colspan="3">Next hop (4)</td></tr>
<tr><td colspan="3">Metric (4)</td></tr>
<tr><td colspan="3">RIP Entry #2</td></tr>
<tr><td colspan="3">⋮</td></tr>
<tr><td colspan="3">RIP Entry #N</td></tr>
</table>

Note: Field sizes are given in octets/bytes

- *Route tag:*
  - *Carries information like the autonomous system number*
  - *identify internal/external RIP routes*
- *Next hop: immediate next-hop IP*

The use of "next hop"



- RD does not run RIP, AS 1 could exchange route with external AS via RC
- RC could put RD's IP address in the "next hop" when advertising subnet y
  - Use 0.0.0.0 if next-hop is the originator (default route)
  - Next-hop should be directly reachable on the logical subnet over which the advertisement is made

Request Messages

<table>
<tr><td>Command: 1</td><td>Version: 2</td><td colspan="2">All 0s</td></tr>
<tr><td colspan="2">Address family identifier</td><td colspan="2">All 0s</td></tr>
<tr><td colspan="4">IP address (4)</td></tr>
<tr><td>RIP Entry #1</td><td colspan="3">Subnet mask (4)</td></tr>
<tr><td colspan="4">All 0s</td></tr>
<tr><td colspan="4">All 0s</td></tr>
<tr><td colspan="4">RIP Entry #2</td></tr>
<tr><td colspan="4">⋮</td></tr>
<tr><td colspan="4">RIP Entry #N</td></tr>
</table>

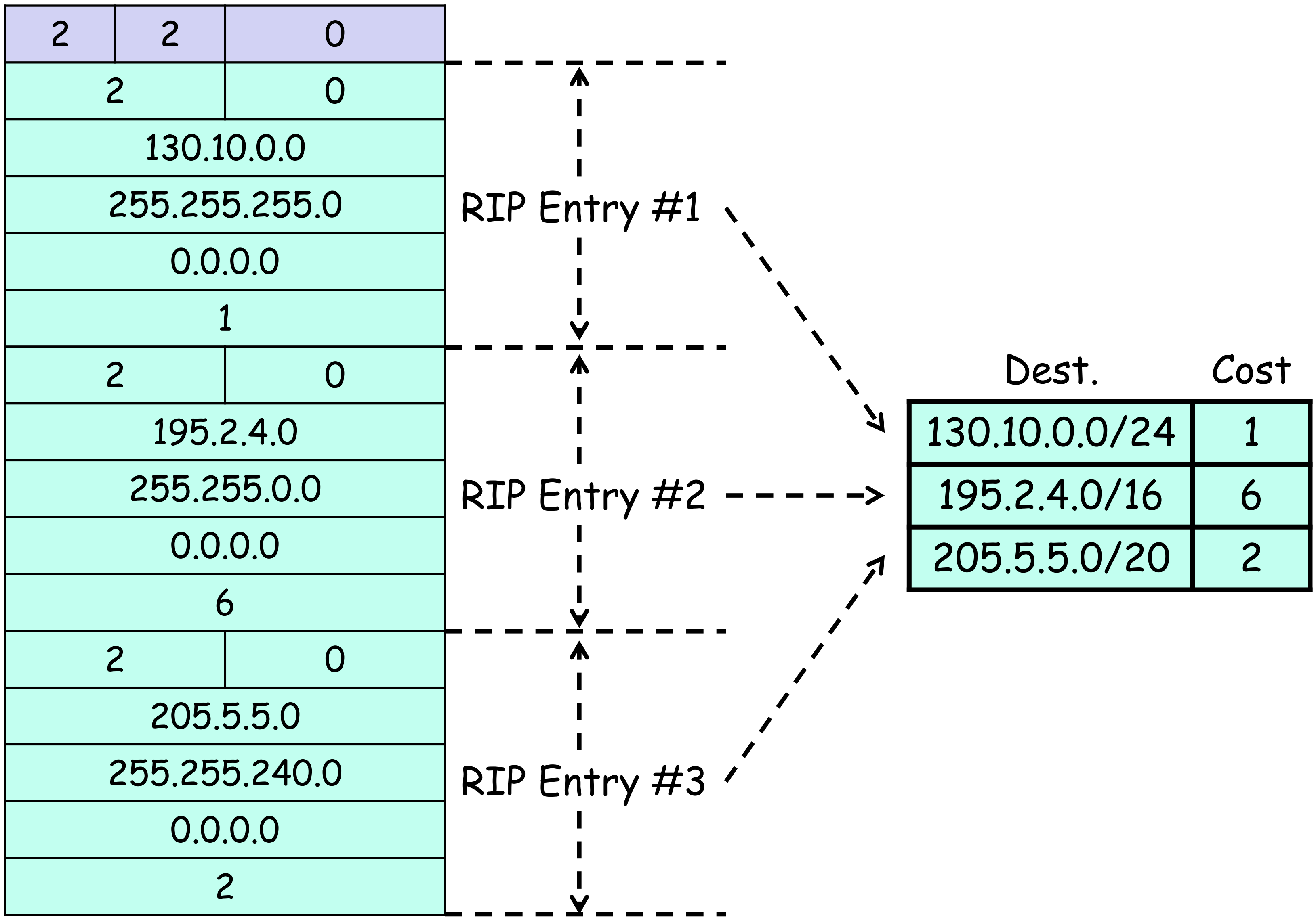
- Unicast/broadcast by routers that need particular route information
- Processed entry by entry: match entries and fill in metric and next-hop in response.

Request Messages

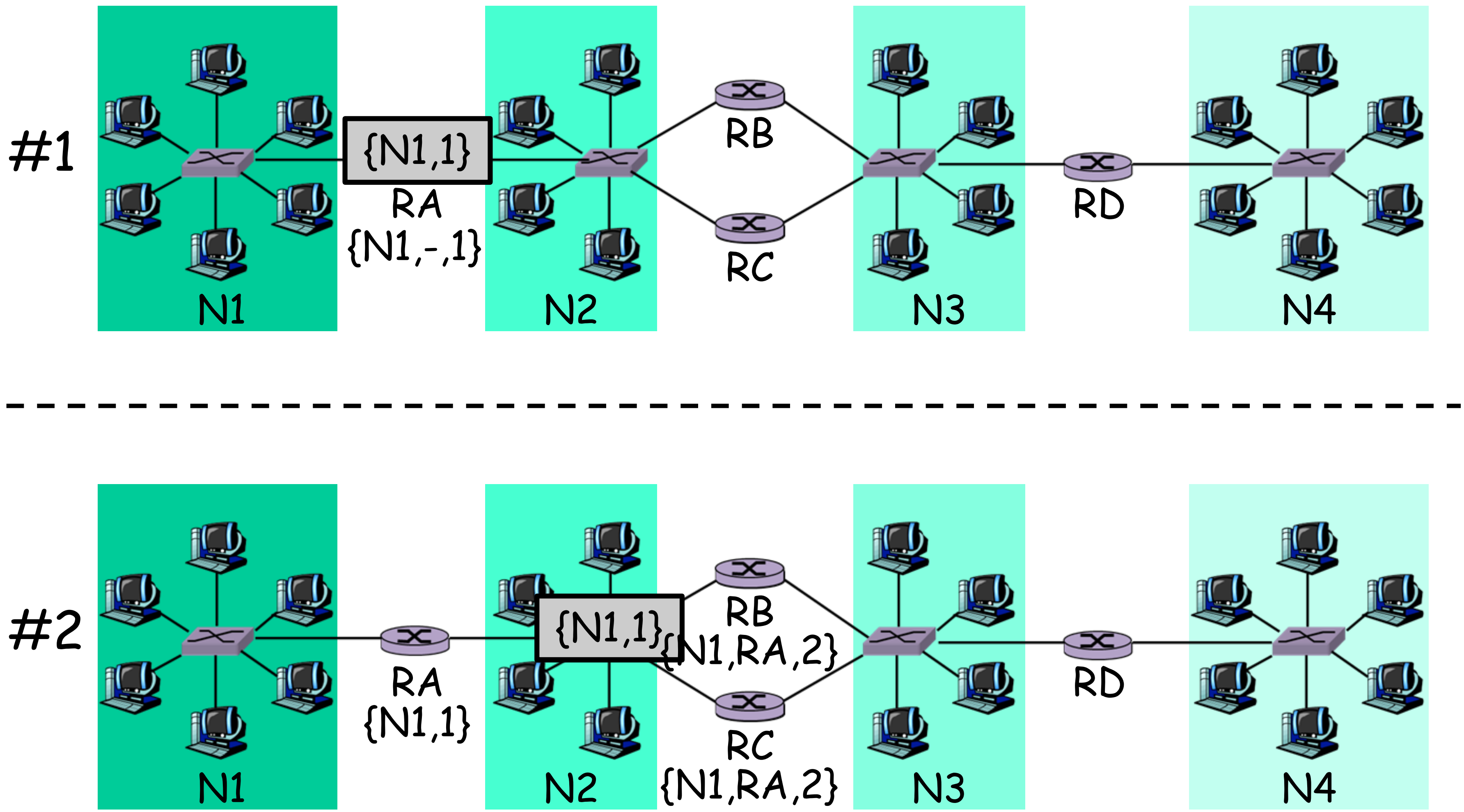
	Command: 1	Version: 2	All 0s
	Address family identifier: 0		All 0s
	All 0s		
RIP Entry #1	All 0s		
	All 0s		
	16		

- Special request for the entire routing table
- Only 1 RIP entry in the request with
  - AFI set to 0 and metric set to 16
- Broadcast by routers which have just come up

Response Messages

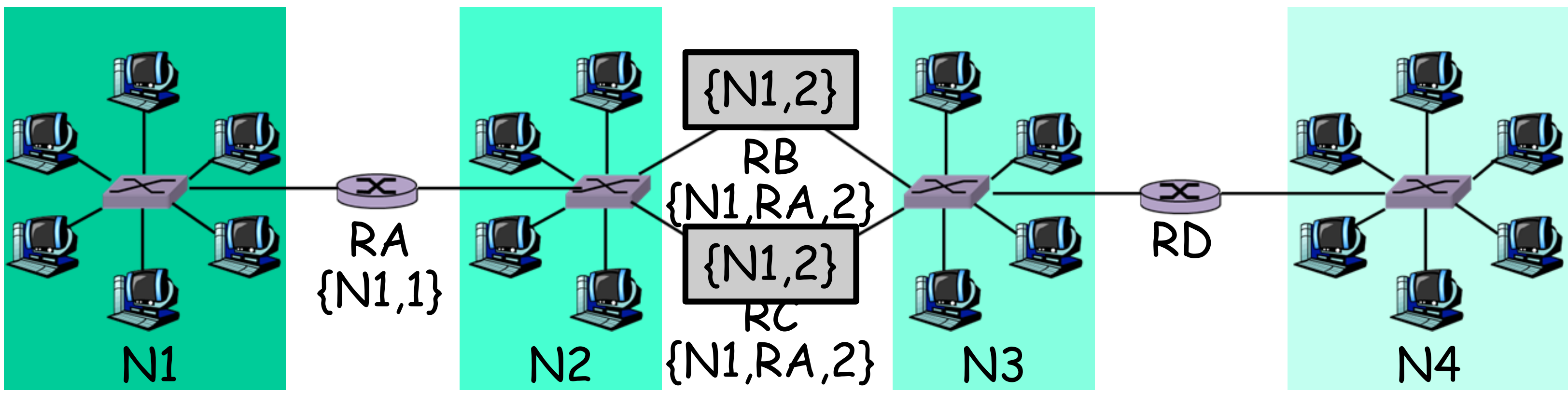


RIP: Illustrations

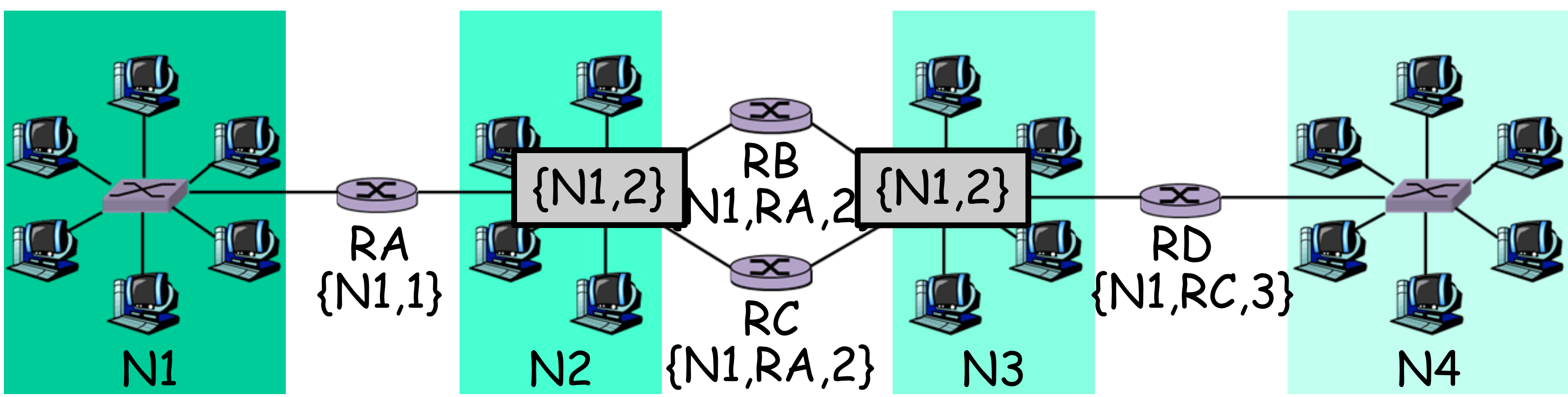


RIP: Illustrations

#3

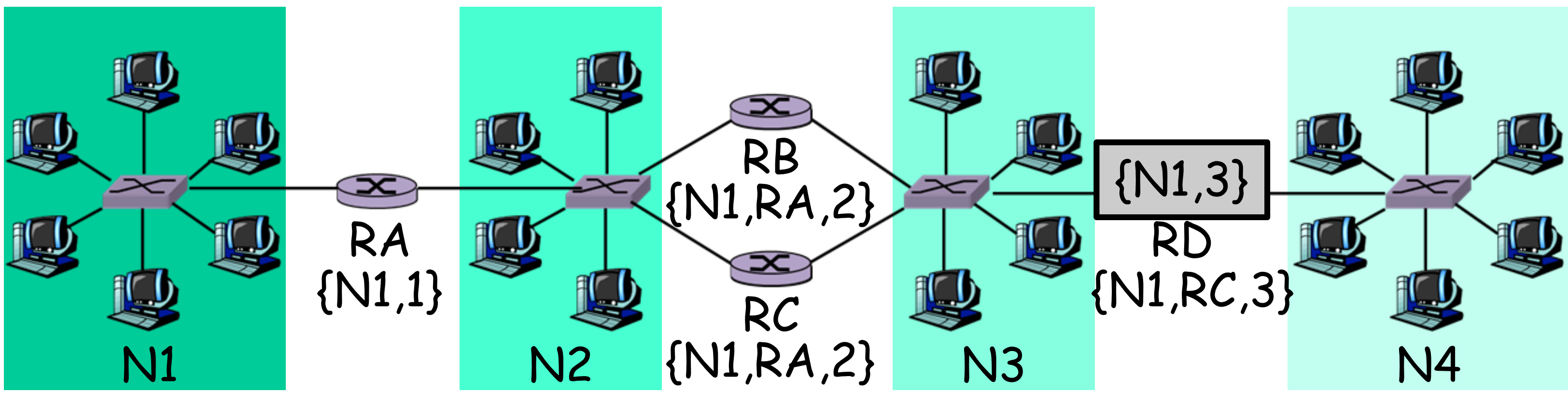


#4

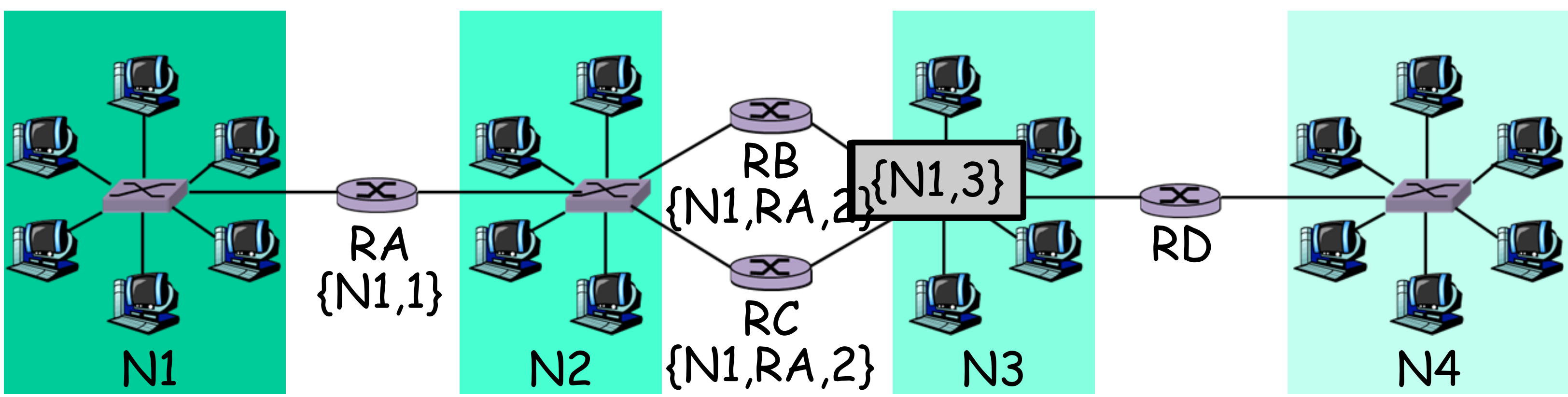


RIP: Illustrations

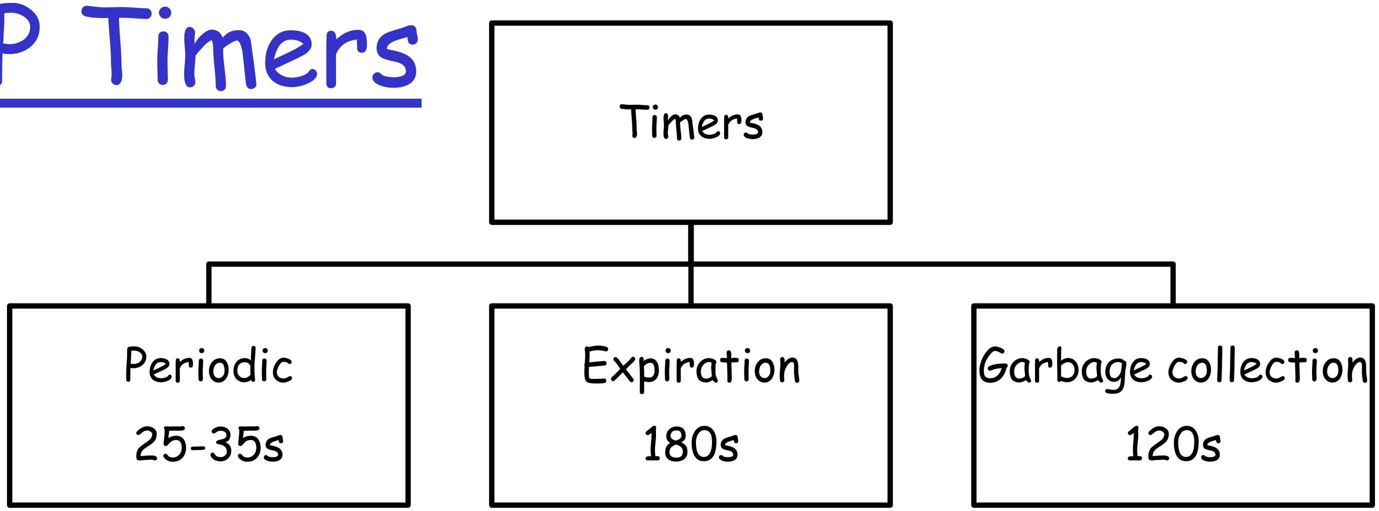
#5



#6



RIP Timers



- Periodic timer controls the sending of messages.
- Expiration timer governs the validity of a route.
- Garbage collection timer advertises the failure of a route

Periodic Timer

- A periodic timer is associated with each router, i.e., a node in the routing graph.
- Periodic timer controls the advertising of regular update messages
  - Even nothing has been changed
- Update every 30s in theory
- Randomize between 25s-35s to avoid synchronization

Expiration Timer

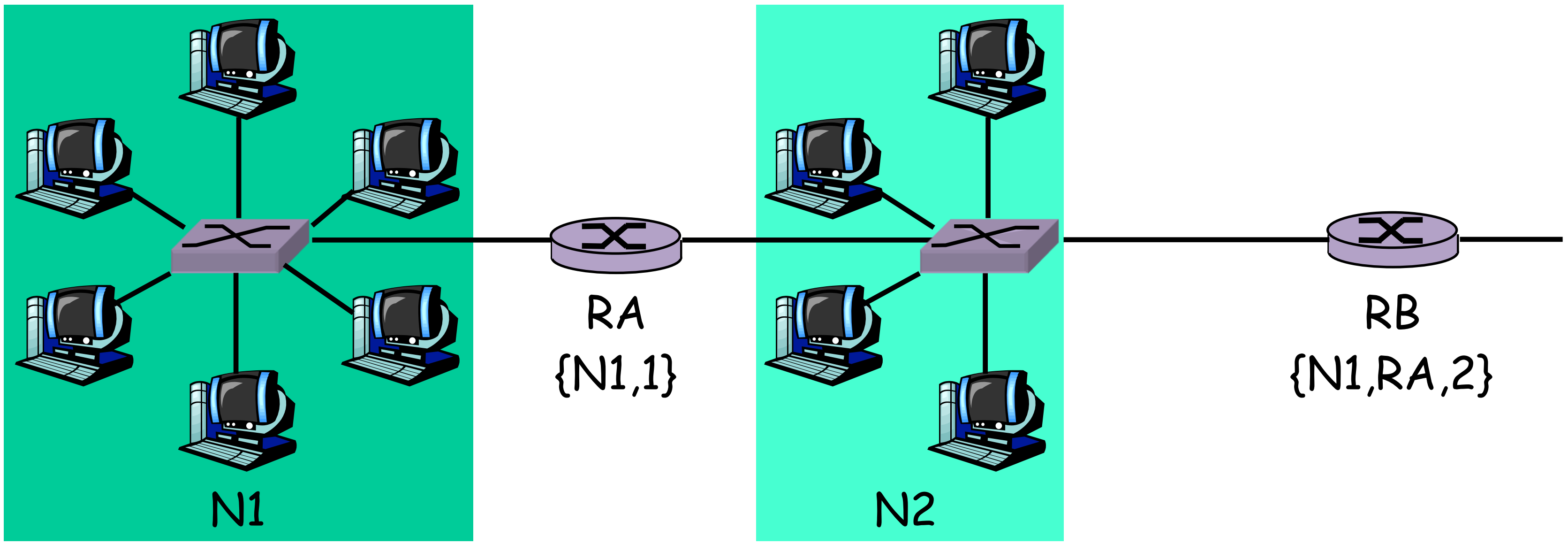
- An expiration timer is associated with each active route, i.e., an entry in routing table.
- An expiration timer is started whenever a route entry is installed in the routing table.
- When RIP Response with information about a route arrives, the route is "refreshed" and expiration timer is reset.
- A route is marked for deletion and its metric is set to be "infinity" upon timeout.

Garbage collection Timer

- A garbage collection timer is associated with each ***expired*** route.
- When a route is marked for deletion, a *Garbage-Collection* timer is also started.
- When this Timer timeout, the marked expired route entry will be deleted.
- Provide a time window (120s) for notifying invalid routes to neighbors.

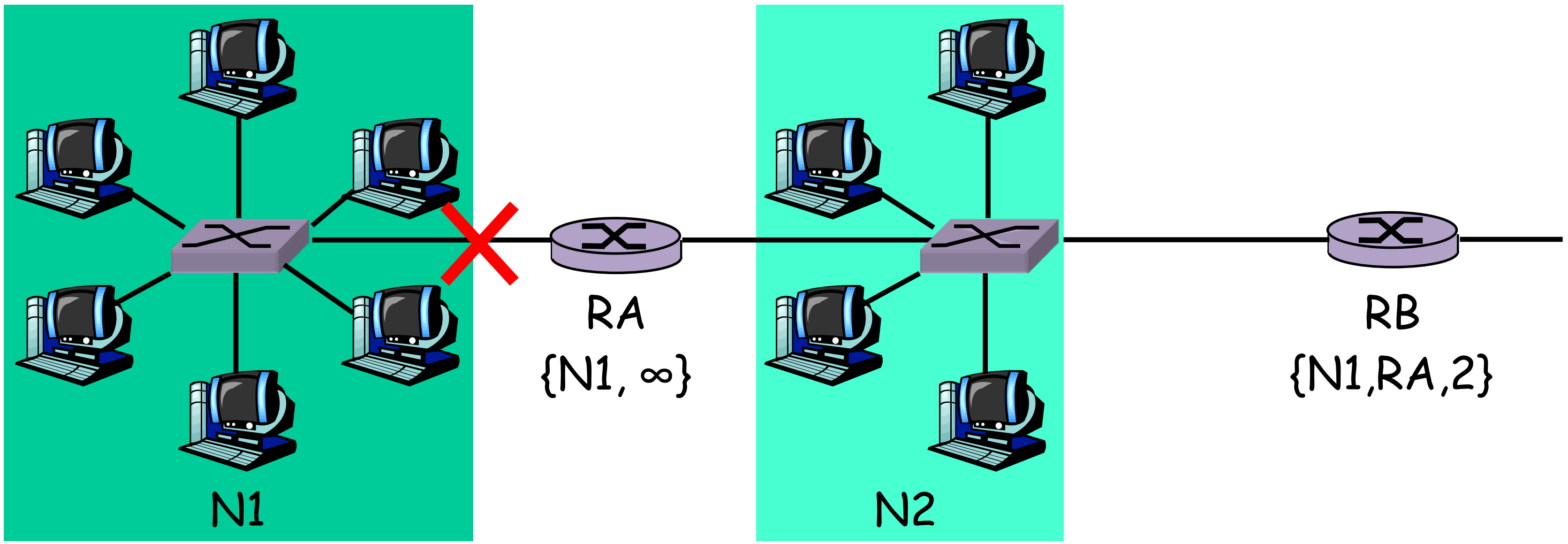
Count-To-Infinity Problem

At first everything works fine



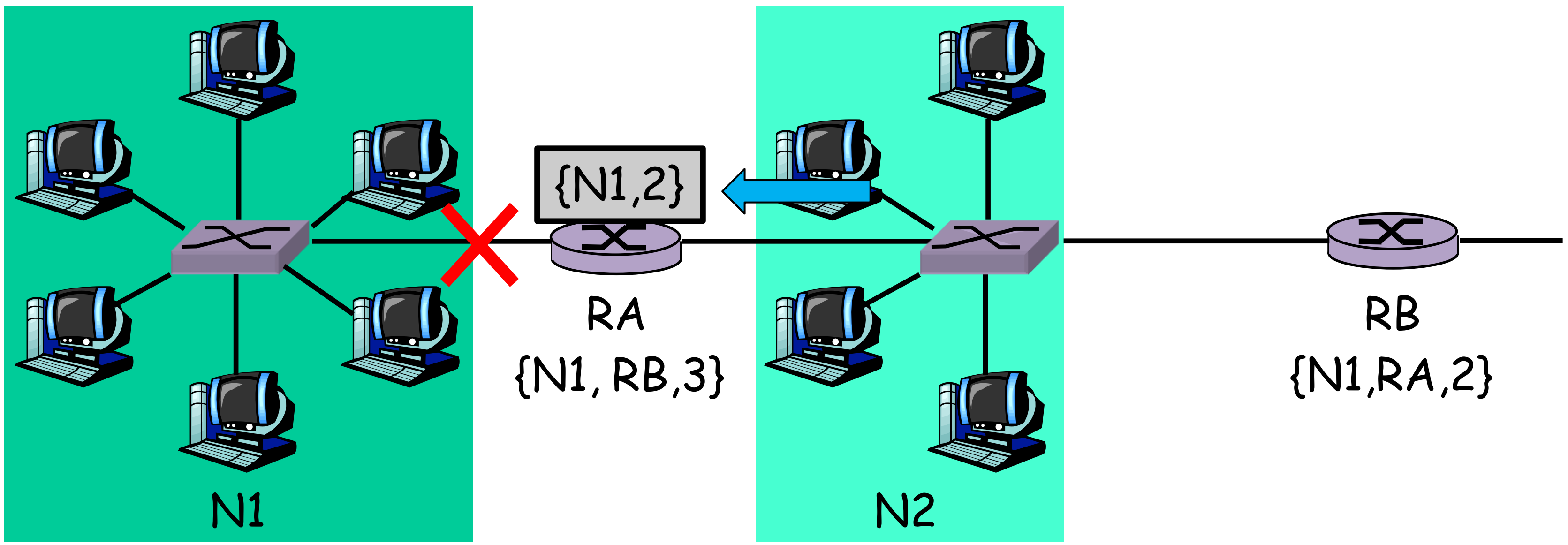
Count-To-Infinity Problem

Later link between RA and N1 fails



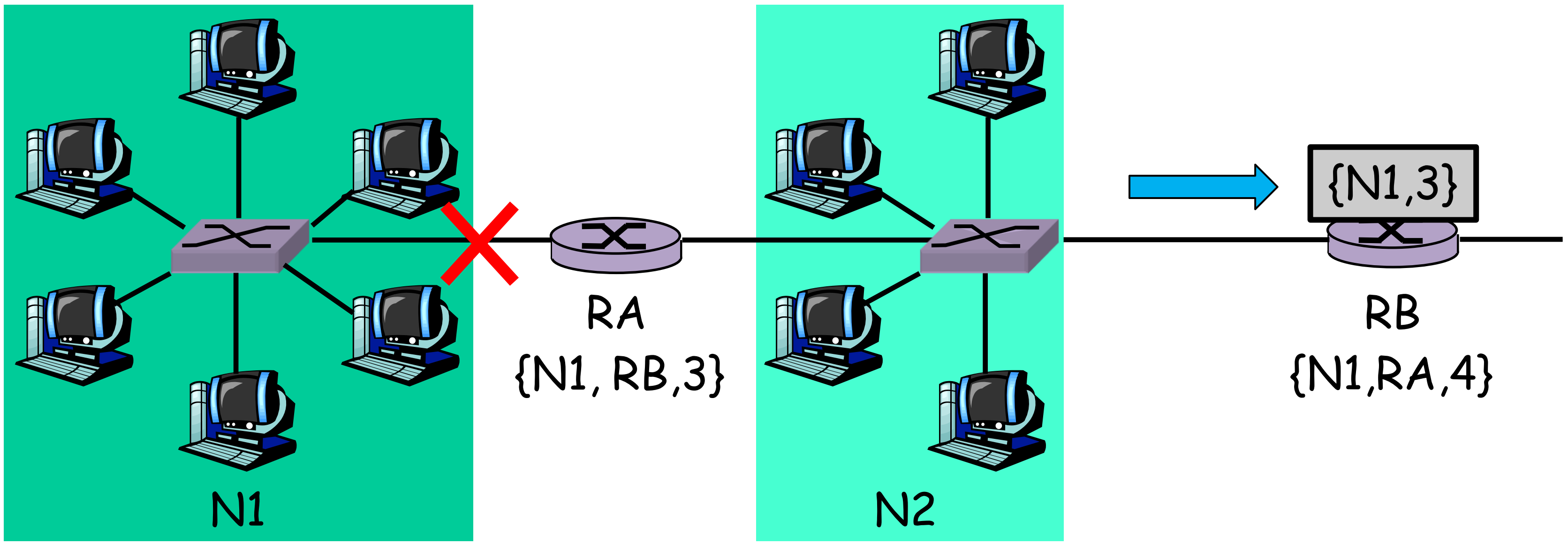
Count-To-Infinity Problem

Before RA sends new information out,
RA receives update from RB



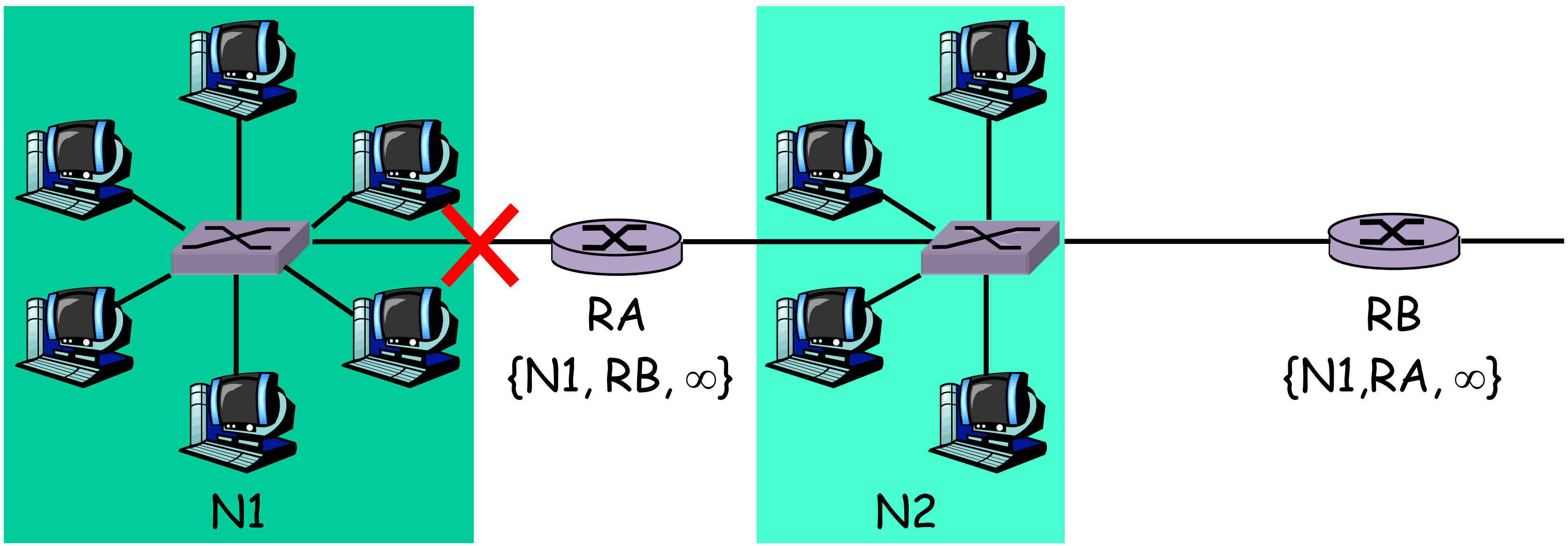
Count-To-Infinity Problem

Next, RB receives update from RA



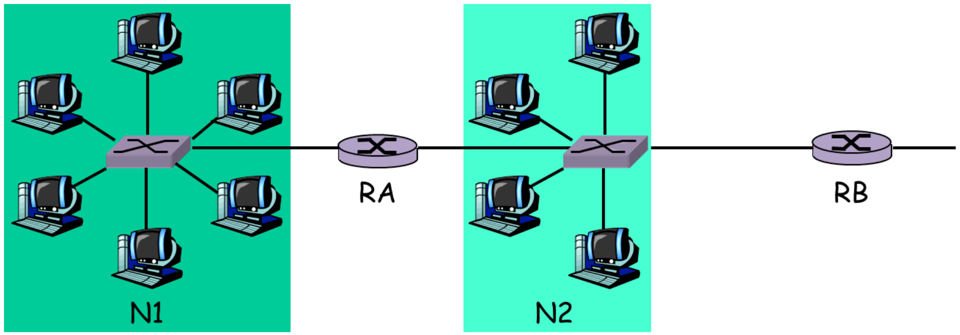
Count-To-Infinity Problem

Finally, they count to infinity



Practical consequences: Inconsistent views of the network creating routing loops!

Split Horizon



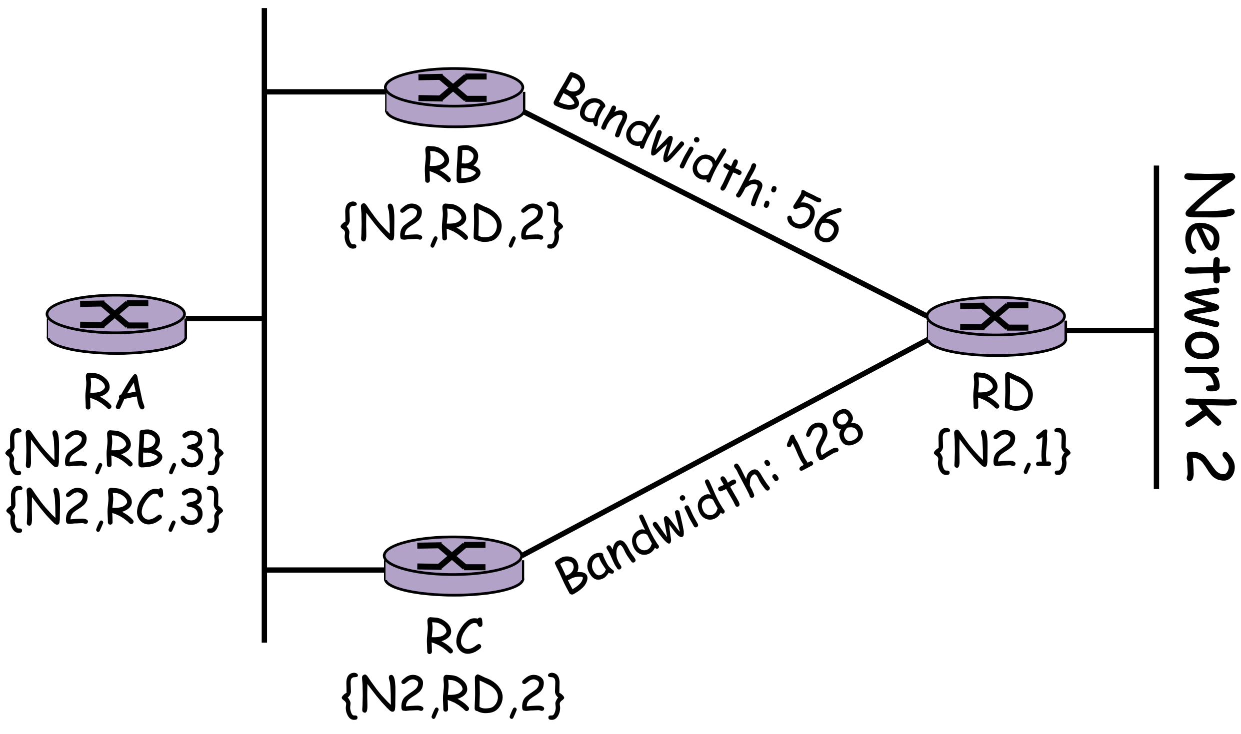
- A simple solution:
  - have Router *B* not mention the route to N1 in any *RIP Response* messages it sends to Router *A*.
- New rule: Split Horizon
  - when a router sends out an *RIP Response* on any of the networks to which it is connected, it omits any route information that was originally learned from that network.
  - the router effectively splits its view of the network, sending different information on certain interfaces than on others.

Drawback of Split Horizon

- Split Horizon does solve count-to-infinity problem in the previous scenario
- However, in some cases, Split Horizon increases convergence time in response to a link/router failure

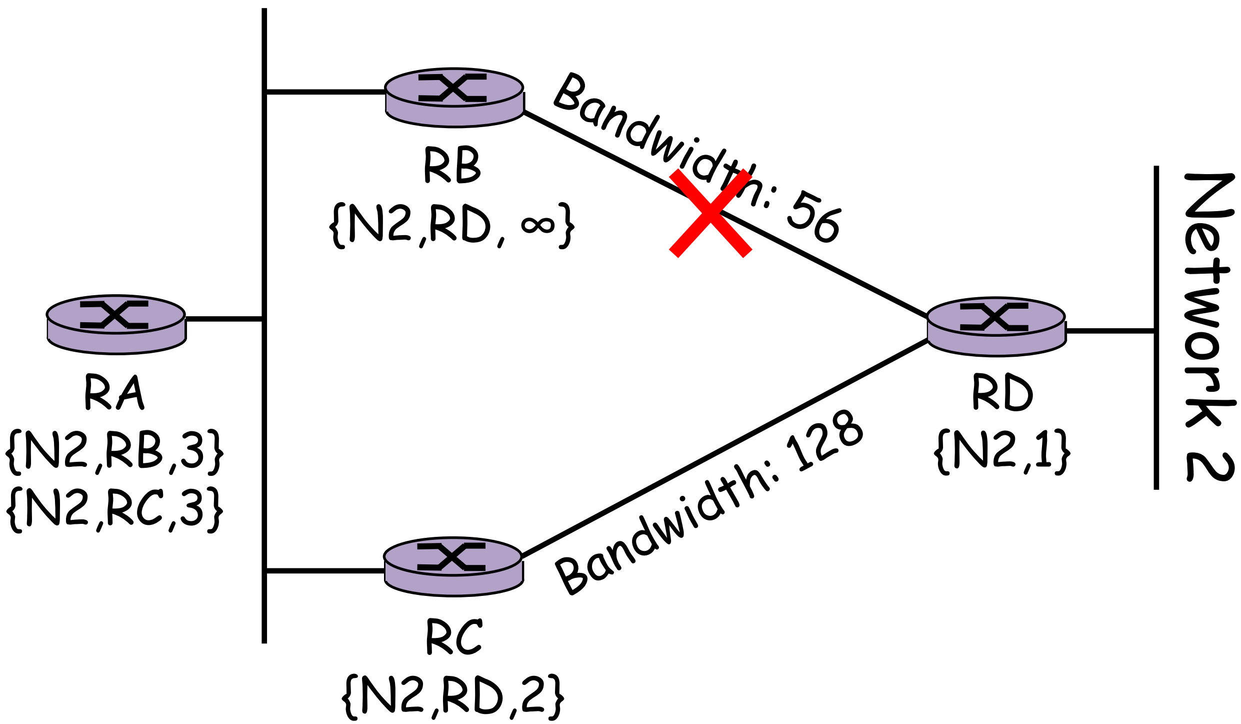
Drawback of Split Horizon

At first, everything works.



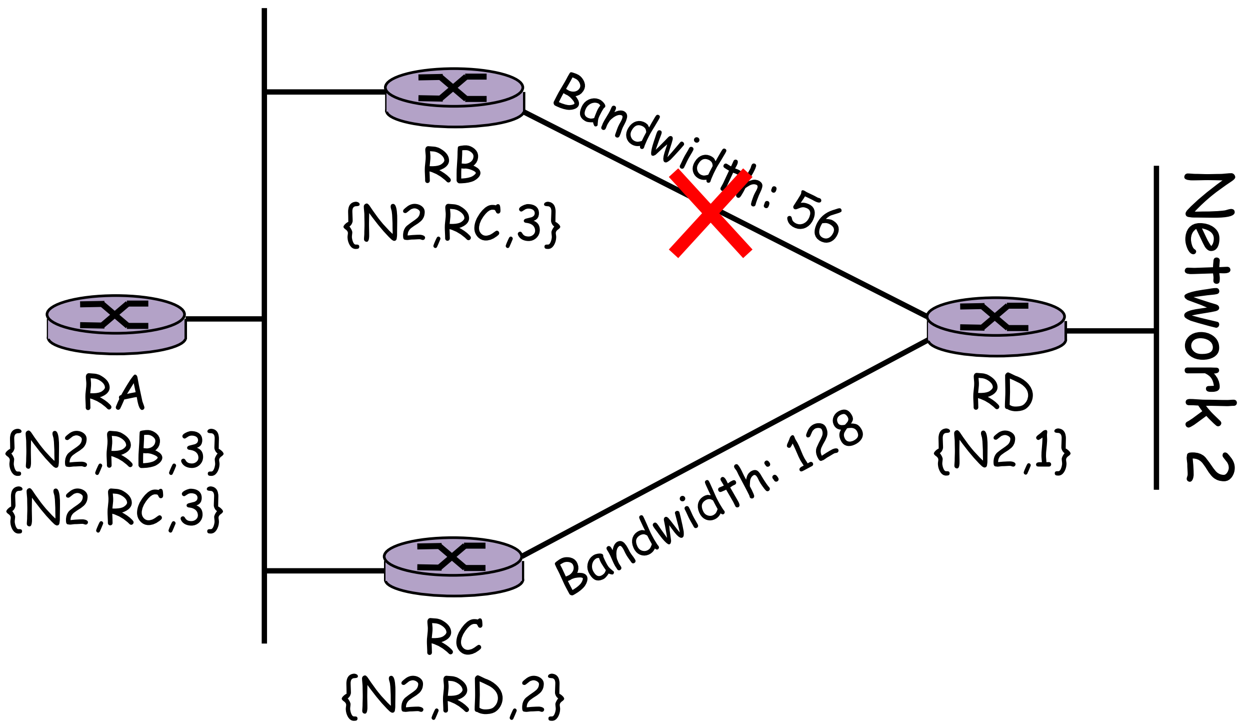
Drawback of Split Horizon

Later, link between RB and RD fails



Drawback of Split Horizon

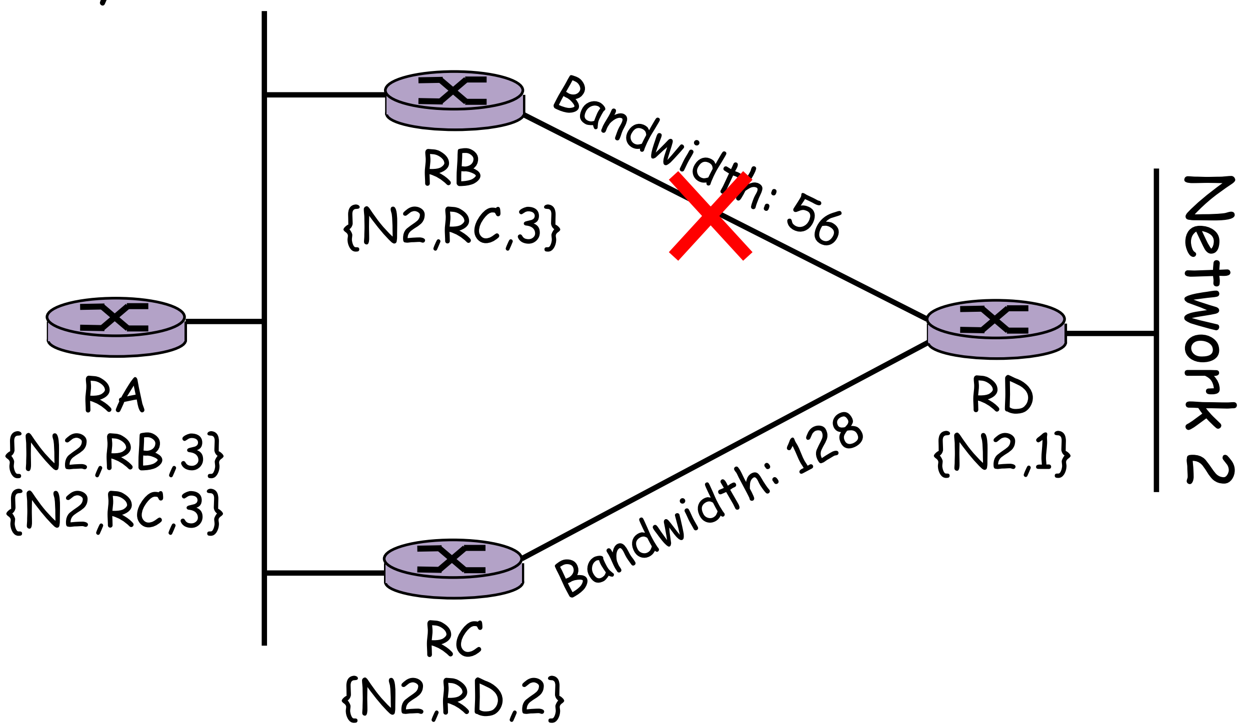
Before RB informs neighbors, RB receives update from RC



Drawback of Split Horizon

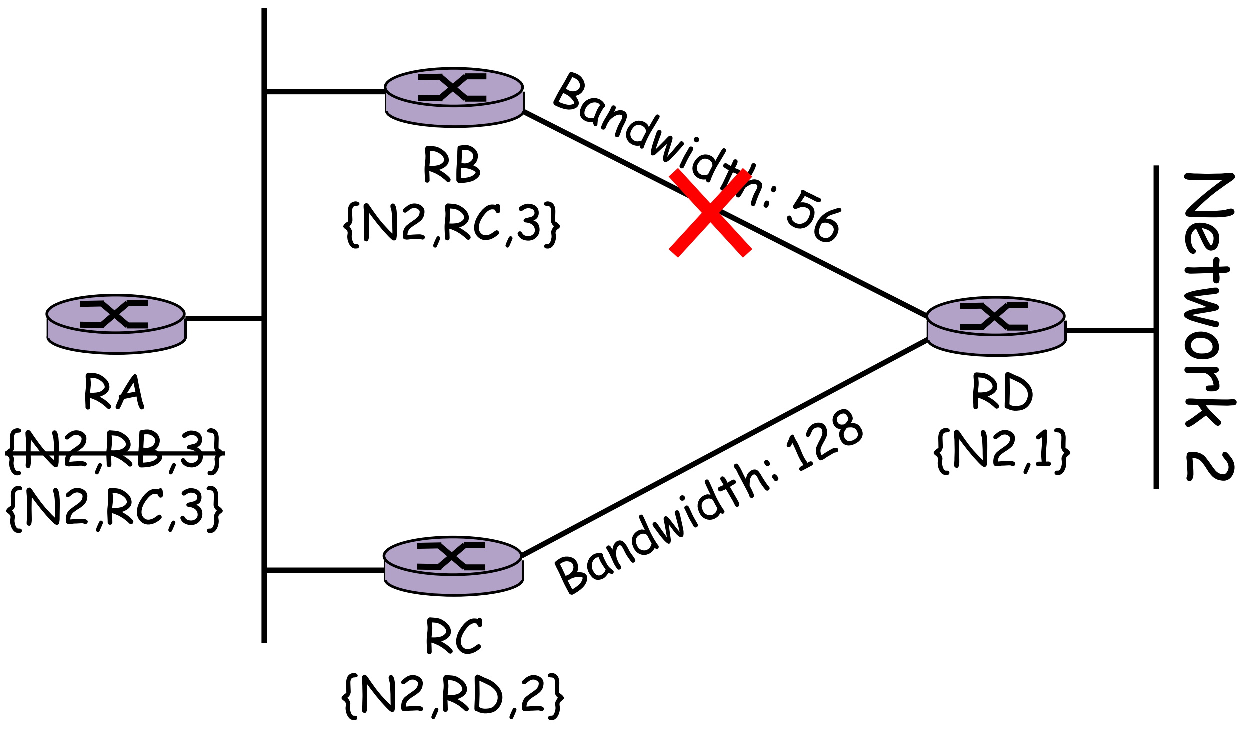
By split horizon, RB will not advertise route to N2 to RA, because it uses RC to access N2

So, RA cannot know about the failure of link RB-RD



Drawback of Split Horizon

Route entry {N2,RB,3} will be valid till it expires (180s)

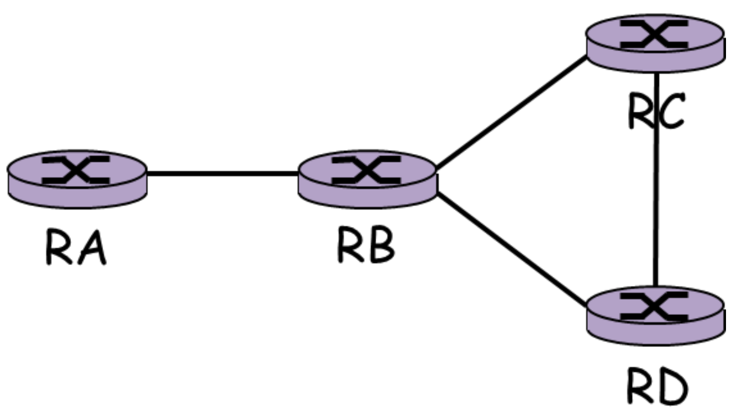


Split Horizon With Poisoned Reverse

- Enhancement: Poisoned reverse
  - include those routes in the message but set their metric to infinity
  - *Poisoning* the routes that we want to make sure routers on that interface don't use.
  - in the previous case, RB advertises a metric infinity to RC (and RA).
- Rationale
  - Bad news is better than no news
  - Provide more insurance
- A tradeoff between convergence and message overhead

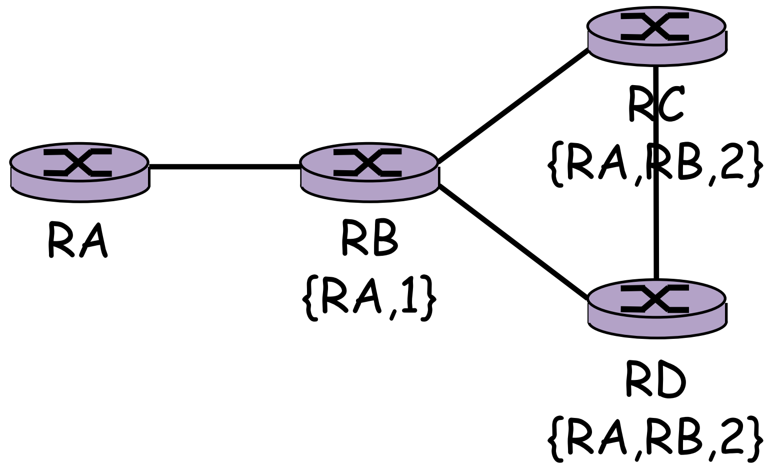
Split Horizon Limitation

- Note, however, that split horizon may not ***always*** solve the "counting to infinity" problem, especially in the case where multiple routers are connected indirectly.
- Example: 3 routers configured in a triangle:



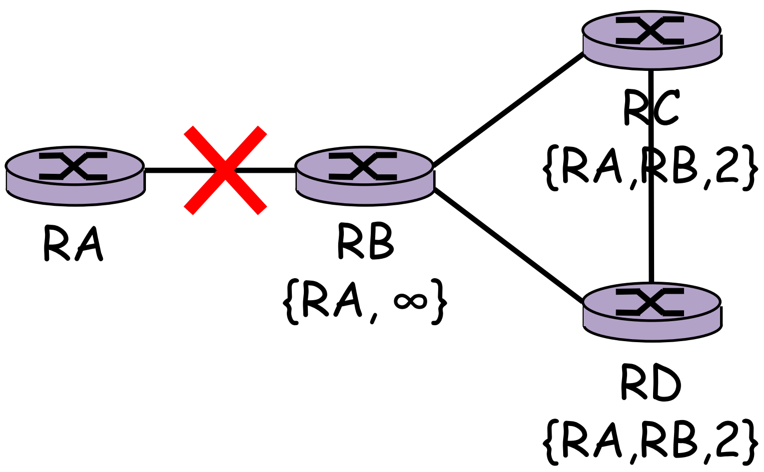
Count-To-Infinity Problem (cont.)

At first, everything works fine



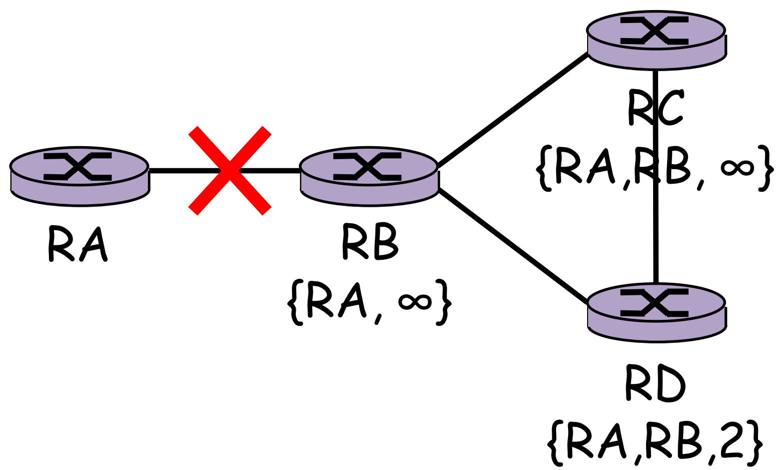
Count-To-Infinity Problem (cont.)

Then, link between RA and RB is down



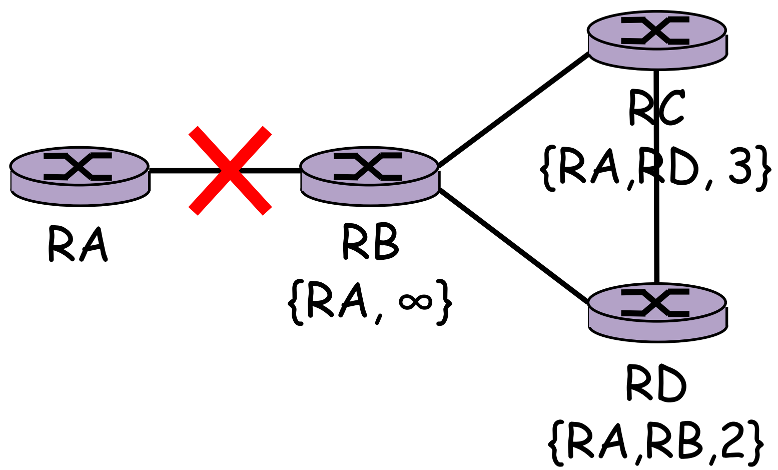
Count-To-Infinity Problem (cont.)

Next, RB sends route to RC and RD.
Unfortunately, the packet to RD is lost



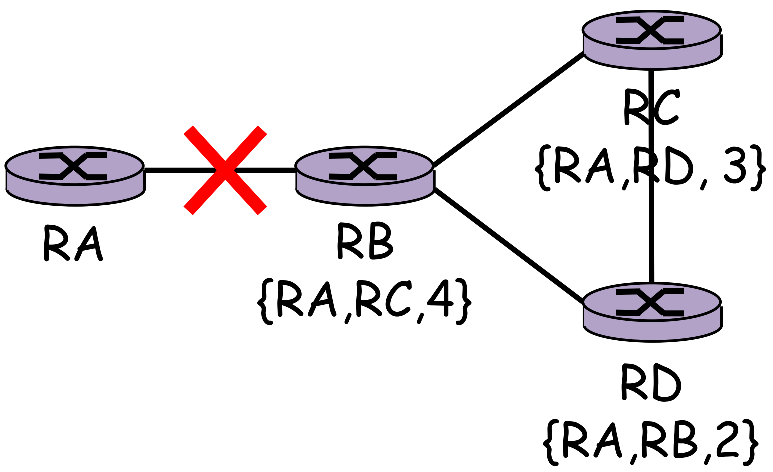
Count-To-Infinity Problem (cont.)

Now RC receives update from RD



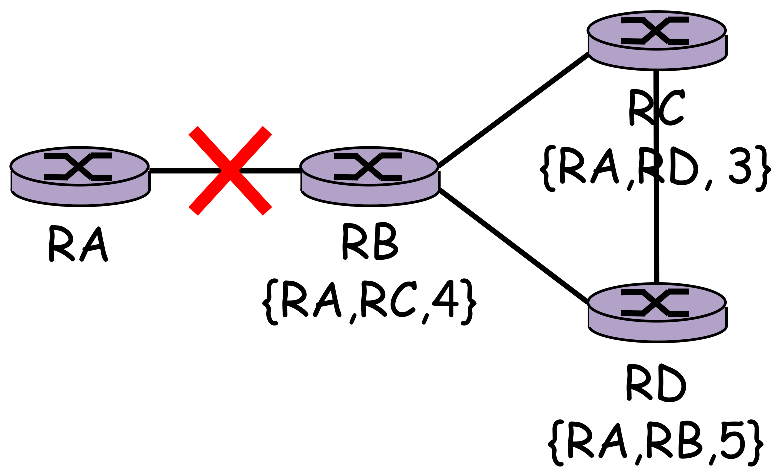
Count-To-Infinity Problem (cont.)

RB receives update from RC



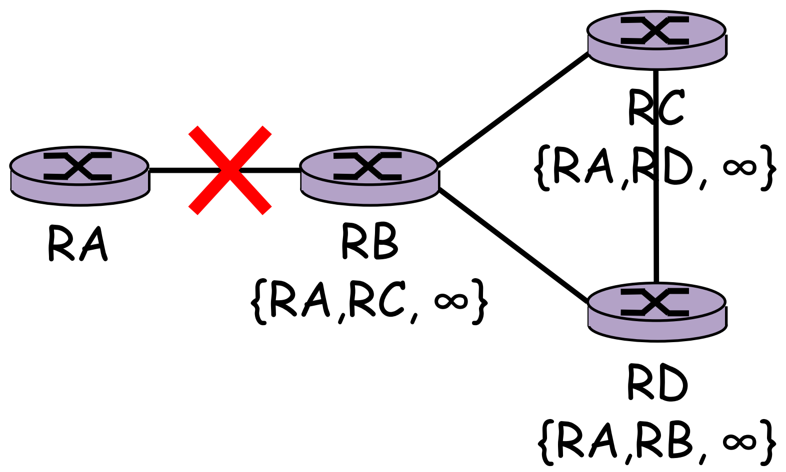
Count-To-Infinity Problem (cont.)

Again, RD receives update from RB



Count-To-Infinity Problem (cont.)

Finally, those three routers count to infinity

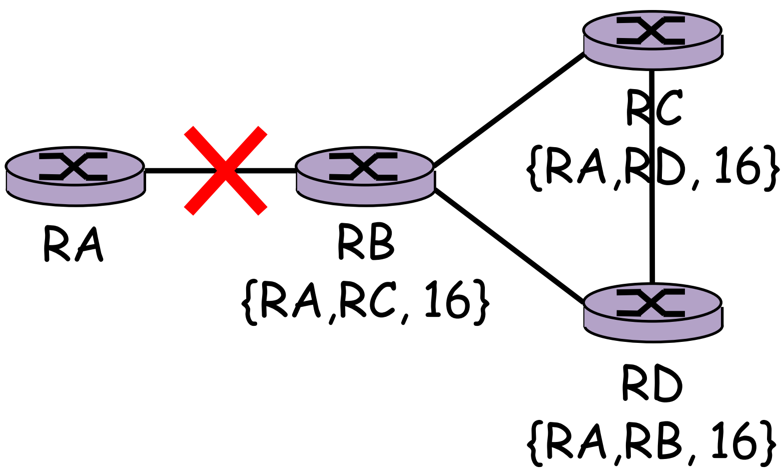


Solution: "RIP Infinity"

- On the one hand, if a network becomes completely inaccessible, counting to infinity needs to be stopped as soon as possible
- On the other hand, infinity must be large enough that no real route is that big
- RIP Infinity is 16 because the designers of RIP believed that RIP was unlikely to be practical for networks with a diameter larger than 15 → works only for small networks.

Count-To-Infinity Problem (cont.)

Now, routers will stop when distance reaches 16

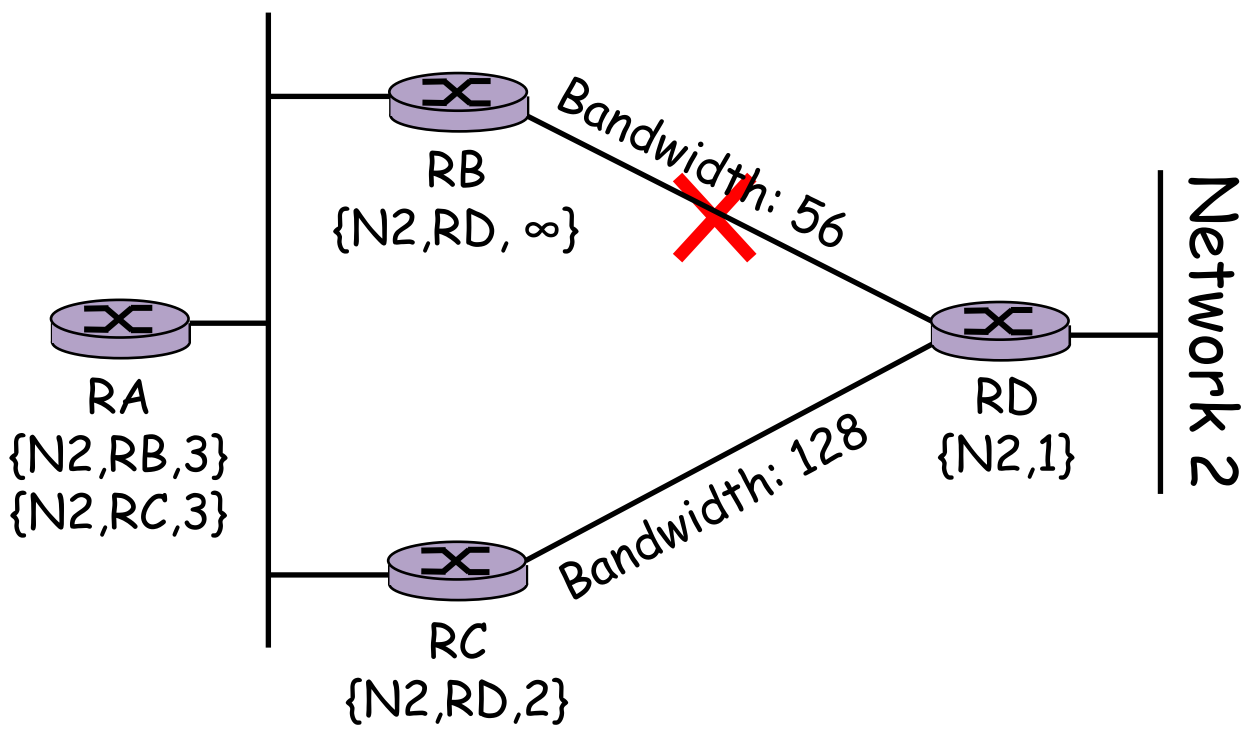


Triggered Updates

- Problem: Slow convergence
- Solution: Triggered Updates
  - Whenever a router changes the metric for a route it is required to (almost) immediately send out an *RIP Response* to tell its immediate neighbor routers about the change.
- Why almost :
  - waits a random amount of time, from 1 to 5 seconds
  - Reduce the load on network

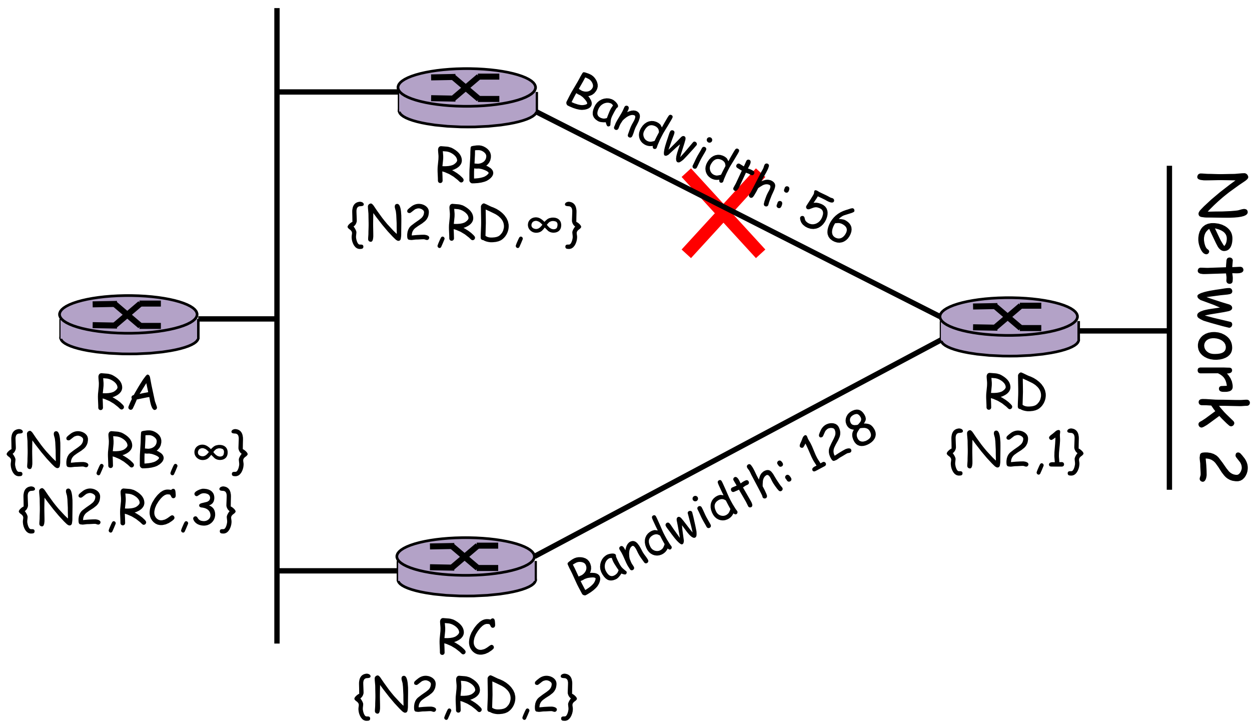
Use Case of Triggered Updates

Link between RB and RD fails



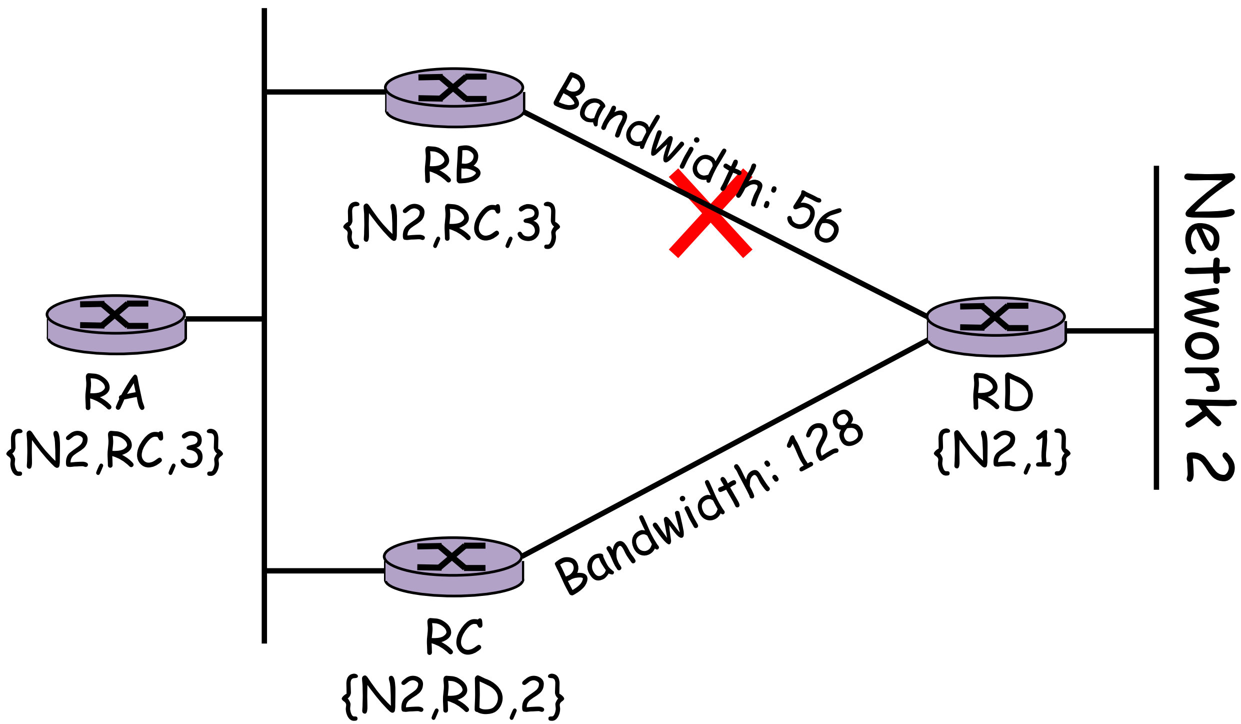
Use Case of Triggered Updates

RA receives the update from RB right after the route changes



Use Case of Triggered Updates

RB receives update from RC and the network works fine now



Features and Limitations of RIP

- Features:
  - Split horizon
  - Poisoned reverse
  - Triggered updates
  - Garbage Collection timer
  - RIP infinity

- Limitations:
  - Small RIP Infinity value, i.e., 16
  - Static metric
    - Hop Count as a distance metric
    - Lack of support for dynamic (real-time) metrics