La Sapienza
Università degli Studi di Roma
Dipartimento di Informatica e Sistemistica

Computer Networks II

RIP - **R**outing **I**nformation **P**rotocol

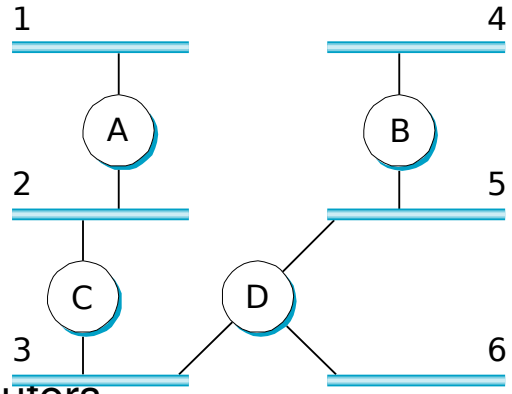
Luca Becchetti

Luca.Becchetti@dis.uniroma1.it

A.A. 2009/2010

RIP

- RIP is a Distance Vector Routing Protocol
  - Metric typically depends on links' states (on/off) and/or their capacities
- Every node exchanges information only with adjacent nodes
  - Two nodes are adjacent if they are directly connected in the same subnet
- Used in small or medium-sized networks
- Widespread
  - In Berkeley Unix distribution (BSD, 1982)
  - Standardized in RFC 1058 (1988)
  - RIP-v2, RFC 2453 (1998)
- Very close to theoretical version
  - Main difference: networks rathen than single routers
- Very simple, modest overhead but:
  - Relatively slow convergence
  - Equilibrium might be a sub-optimal state



RIP

- Notation:
  - M : # networks to which node x is directly connected
  - w(x,i) : cost of arc leaving x to network i ($1 \le i \le M$)
  - N : overall number of networks
  - L(x,j) : estimate of shortest path length from x to network j
  - R(x,j) : next router in shortest path from x to network j
- Router x maintains following vectors:

$$W_x = \begin{bmatrix} w(x,1) \\ \vdots \\ w(x,M) \end{bmatrix}$$

Cost vector for links leaving x

$$L_x = \begin{bmatrix} L(x,1) \\ \vdots \\ L(x,N) \end{bmatrix}$$

Shortest path vector to other networks

$$R_x = \begin{bmatrix} R(x,1) \\ \vdots \\ R(x,N) \end{bmatrix}$$

Next-hop vector

RIP

- Uses distributed Bellman-Ford algorithm
- Uses hop-count as metric
- Adjacent routers exchange vectors L periodically (about 30 sec)
  - A router therefore knows the latest version of the L vectors sent by its neighbours
- Node x updates its vectors as follows:

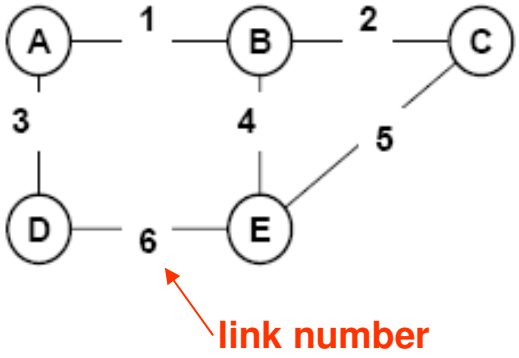
$$L(x,n) = \min_{y \in M_x} \left[w(x,N_{xy}) + L(y,n) \right]$$

$$R(x,n) = y$$

Node achieving minimum

  - **M_x : set of neighbours of node x**
  - **N_{xy} : subnet interconnecting nodes x and y**

RIP – Example – initialization 1/8



Initial state

- Empty routing tables

Metric

- Distance (no. hops)

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	?	?	?	?
	Link	local	?	?	?	?

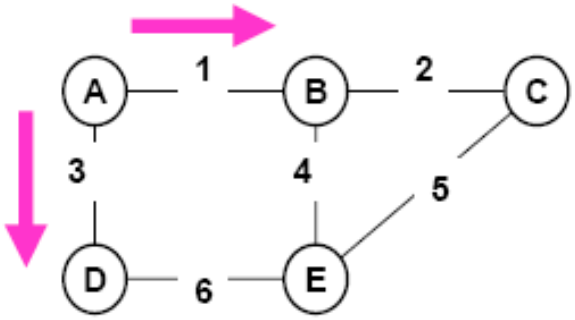
B	Destinazione	A	B	C	D	E
	Distanza	?	0	?	?	?
	Link	?	local	?	?	?

C	Destinazione	A	B	C	D	E
	Distanza	?	?	0	?	?
	Link	?	?	local	?	?

D	Destinazione	A	B	C	D	E
	Distanza	?	?	?	0	?
	Link	?	?	?	local	?

E	Destinazione	A	B	C	D	E
	Distanza	?	?	?	?	0
	Link	?	?	?	?	local

RIP – Example – Initialization 2/8



Step 2:
A sends update to B and D

A	Address	A	---	---	---	---
	Metric	0	---	---	---	---

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	?	?	?	?
	Link	local	?	?	?	?

B	Destinazione	A	B	C	D	E
	Distanza	1	0	?	?	?
	Link	1	local	?	?	?

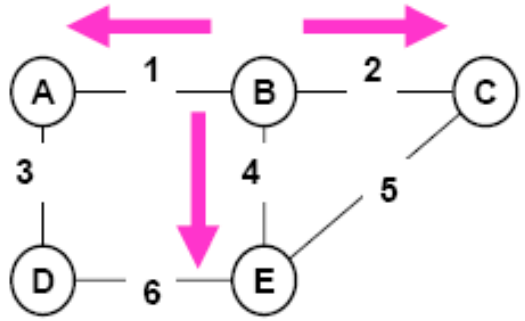
C	Destinazione	A	B	C	D	E
	Distanza	?	?	0	?	?
	Link	?	?	local	?	?

D	Destinazione	A	B	C	D	E
	Distanza	1	?	?	0	?
	Link	3	?	?	local	?

E	Destinazione	A	B	C	D	E
	Distanza	?	?	?	?	0
	Link	?	?	?	?	local

Note how routers receive information about network topology from senders of RIP messages with null value of metric

RIP – Example – Initialization 3/8



Step 3:
B sends update to B, C and A

B	Address	A	B	---	---	---
	Metric	1	0	---	---	---

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	1	?	?	?
	Link	local	1	?	?	?

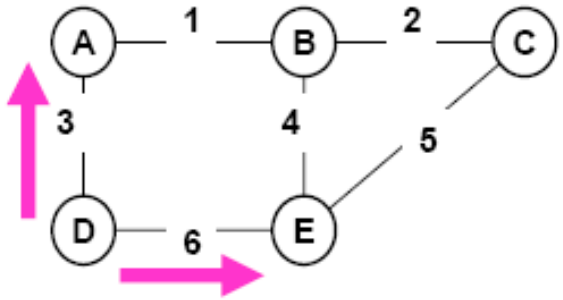
B	Destinazione	A	B	C	D	E
	Distanza	1	0	?	?	?
	Link	1	local	?	?	?

C	Destinazione	A	B	C	D	E
	Distanza	2	1	0	?	?
	Link	2	2	local	?	?

D	Destinazione	A	B	C	D	E
	Distanza	1	?	?	0	?
	Link	3	?	?	local	?

E	Destinazione	A	B	C	D	E
	Distanza	2	1	?	?	0
	Link	4	4	?	?	local

RIP – Example – Initialization 4/8



Step 4:
D sends update messages to A and E

D	Address	A	---	---	D	---
	Metric	1	---	---	0	---

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	1	?	1	?
	Link	local	1	?	3	?

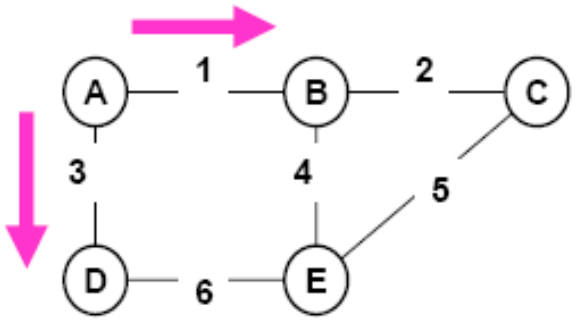
B	Destinazione	A	B	C	D	E
	Distanza	1	0	?	?	?
	Link	1	local	?	?	?

C	Destinazione	A	B	C	D	E
	Distanza	2	1	0	?	?
	Link	2	2	local	?	?

D	Destinazione	A	B	C	D	E
	Distanza	1	?	?	0	?
	Link	3	?	?	local	?

E	Destinazione	A	B	C	D	E
	Distanza	2	1	?	1	0
	Link	4	4	?	6	local

RIP – Example – Initialization 5/8



Step 5:
A sends update messages to B and D

A	Address	A	B	---	D	---
	Metric	0	1	---	1	---

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	1	?	1	?
	Link	local	1	?	3	?

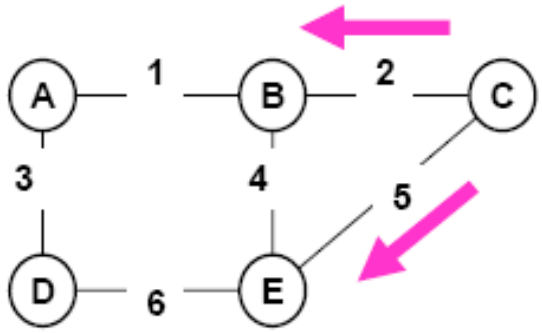
B	Destinazione	A	B	C	D	E
	Distanza	1	0	?	2	?
	Link	1	local	?	1	?

C	Destinazione	A	B	C	D	E
	Distanza	2	1	0	?	?
	Link	2	2	local	?	?

D	Destinazione	A	B	C	D	E
	Distanza	1	2	?	0	?
	Link	3	3	?	local	?

E	Destinazione	A	B	C	D	E
	Distanza	2	1	?	1	0
	Link	4	4	?	6	local

RIP – Example – Inizializzazione 6/8



Step 6: C sends updates to B and E

C	Address	A	B	C	---	---
	Metric	2	1	0	---	---

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	1	?	1	?
	Link	local	1	?	3	?

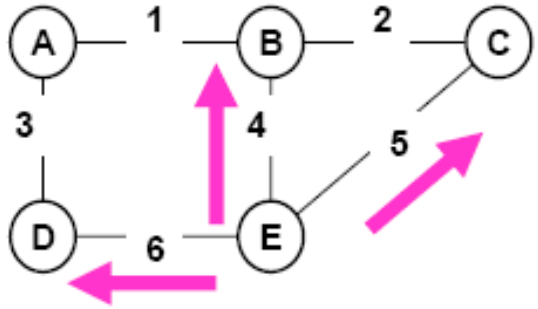
B	Destinazione	A	B	C	D	E
	Distanza	1	0	1	2	?
	Link	1	local	2	1	?

C	Destinazione	A	B	C	D	E
	Distanza	2	1	0	?	?
	Link	2	2	local	?	?

D	Destinazione	A	B	C	D	E
	Distanza	1	2	?	0	?
	Link	3	3	?	local	?

E	Destinazione	A	B	C	D	E
	Distanza	2	1	1	1	0
	Link	4	4	5	6	local

RIP – Example – Initialization 7/8



Step 7: E sends updates to B, C and D

E	Address	A	B	C	D	E
	Metric	2	1	1	1	0

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	1	?	1	?
	Link	local	1	?	3	?

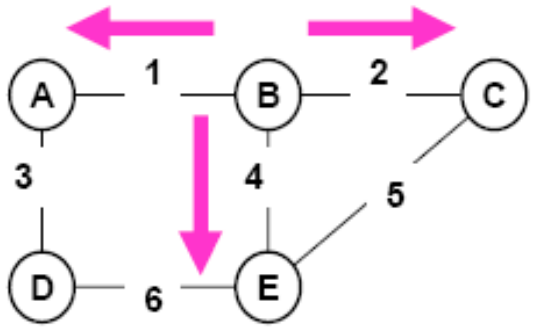
B	Destinazione	A	B	C	D	E
	Distanza	1	0	1	2	1
	Link	1	local	2	1	4

C	Destinazione	A	B	C	D	E
	Distanza	2	1	0	2	1
	Link	2	2	local	5	5

D	Destinazione	A	B	C	D	E
	Distanza	1	2	2	0	1
	Link	3	3	6	local	6

E	Destinazione	A	B	C	D	E
	Distanza	2	1	1	1	0
	Link	4	4	5	6	local

RIP – Example – Initialization 8/8



Step 8: B sends updates to A, C and E

B	Address	A	B	C	D	E
	Metric	1	0	1	2	1

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	1	2	1	2
	Link	local	1	1	3	1

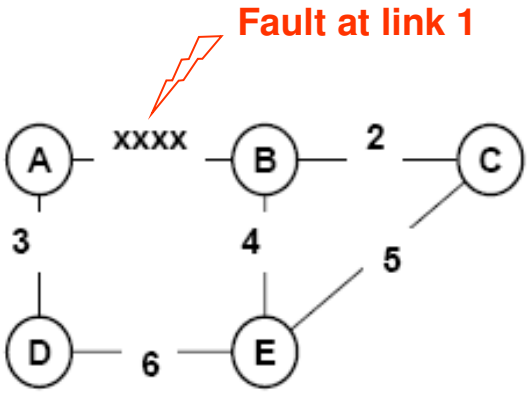
B	Destinazione	A	B	C	D	E
	Distanza	1	0	1	2	1
	Link	1	local	2	1	4

C	Destinazione	A	B	C	D	E
	Distanza	2	1	0	2	1
	Link	2	2	local	5	5

D	Destinazione	A	B	C	D	E
	Distanza	1	2	2	0	1
	Link	3	3	6	local	6

E	Destinazione	A	B	C	D	E
	Distanza	2	1	1	1	0
	Link	4	4	5	6	local

RIP – Example – link fault 1/4



Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	inf	inf	1	inf
	Link	local	1	1	3	1

B	Destinazione	A	B	C	D	E
	Distanza	inf	0	1	inf	1
	Link	1	local	2	1	4

C	Destinazione	A	B	C	D	E
	Distanza	2	1	0	2	1
	Link	2	2	local	5	5

D	Destinazione	A	B	C	D	E
	Distanza	1	2	2	0	1
	Link	3	3	6	local	6

E	Destinazione	A	B	C	D	E
	Distanza	2	1	1	1	0
	Link	4	4	5	6	local

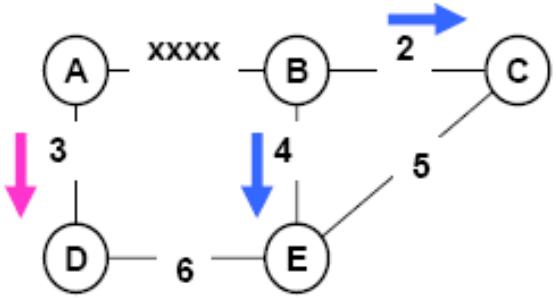
Initial state

- Network at equilibrium
- Link AB faults

Metric

- Distance (hop number)

RIP – Example – link fault 2/4



Step 1

Update from A to D

A	Address	A	B	C	D	E
	Metric	0	inf	inf	1	inf

Update from B to C and E

B	Address	A	B	C	D	E
	Metric	inf	0	1	inf	1

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	inf	inf	1	inf
	Link	local	1	1	3	1

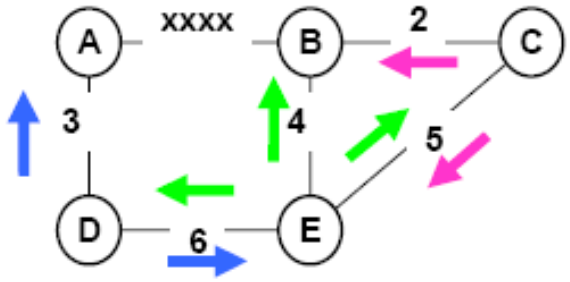
B	Destinazione	A	B	C	D	E
	Distanza	inf	0	1	inf	1
	Link	1	local	2	1	4

C	Destinazione	A	B	C	D	E
	Distanza	inf	1	0	2	1
	Link	2	2	local	5	5

D	Destinazione	A	B	C	D	E
	Distanza	1	inf	2	0	1
	Link	3	3	6	local	6

E	Destinazione	A	B	C	D	E
	Distanza	inf	1	1	1	0
	Link	4	4	5	6	local

RIP – Example – link fault 3/4



Step 2

Message from C to B and E

C	Address	A	B	C	D	E
	Metric	inf	1	0	2	1

Message from D to A and E

D	Address	A	B	C	D	E
	Metric	1	inf	2	0	1

Message from E to B, C and D

E	Address	A	B	C	D	E
	Metric	inf	1	1	1	0

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	inf	3	1	2
	Link	local	1	3	3	3

B	Destinazione	A	B	C	D	E
	Distanza	inf	0	1	2	1
	Link	1	local	2	4	4

C	Destinazione	A	B	C	D	E
	Distanza	inf	1	0	2	1
	Link	2	2	local	5	5

D	Destinazione	A	B	C	D	E
	Distanza	1	2	2	0	1
	Link	3	6	6	local	6

E	Destinazione	A	B	C	D	E
	Distanza	2	1	1	1	0
	Link	6	4	5	6	local

RIP – Example – Link fault 4/4

Step 3

Message from A to D

A	Address	A	B	C	D	E
	Metric	0	inf	3	1	2

Message from B to C and E

B	Address	A	B	C	D	E
	Metric	inf	0	1	2	1

Message from D to A and E

D	Address	A	B	C	D	E
	Metric	1	2	2	0	1

Message from E to B, C and D

E	Address	A	B	C	D	E
	Metric	2	1	1	1	0

Routing Table

A	Destinazione	A	B	C	D	E
	Distanza	0	3	3	1	2
	Link	local	3	3	3	3

B	Destinazione	A	B	C	D	E
	Distanza	3	0	1	2	1
	Link	4	local	2	4	4

C	Destinazione	A	B	C	D	E
	Distanza	3	1	0	2	1
	Link	5	2	local	5	5

D	Destinazione	A	B	C	D	E
	Distanza	1	2	2	0	1
	Link	3	6	6	local	6

E	Destinazione	A	B	C	D	E
	Distanza	2	1	1	1	0
	Link	6	4	5	6	local

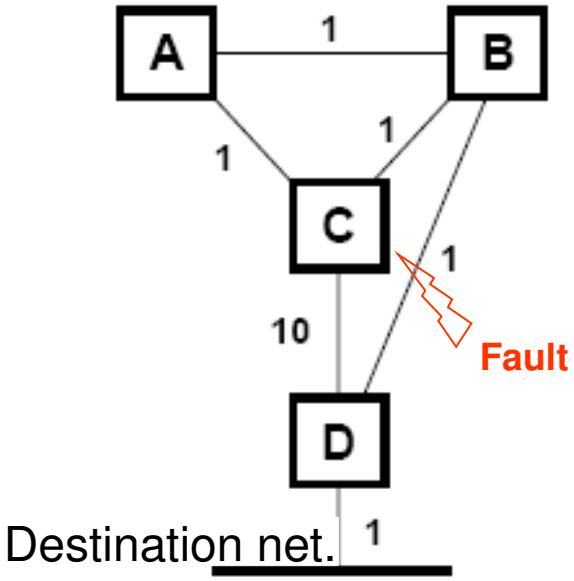
RIP

- Convergence time depends on number of routers that must be crossed by update messages
- RIP - Disadvantages:
  - Instability due to old routings that persist for relatively long
  - Convergence speed
    - In the order of minutes for large networks
  - Overhead due to periodic transmission of distance vectors even in the absence of changes in networks state
  - Metric: only hop number
  - Max 15 hop
  - “Flat” networks
  - Impossible to have variable-length network prefixes
    - Serious problem
  - New version (RIP v2) in 1994 only addressed some of these problems

RIP: count to infinity problem

- Main problem which occurs when
  - A network becomes unreachable [e.g., due to a link fault]
  - Old routings persist because of RIP's slow convergnece
  - **Metrics are not unitary**
- Possible fixes:
  - Avoid use of metrics with wide ranges, so as to avoid long permanence in sub-optimal, non-equilibrium states [RIP uses range of 16 values]
  - Split Horizon

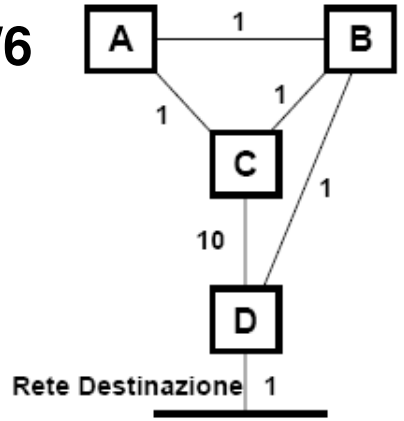
RIP: count to infinity – Example 1/6



Metrics for destination network

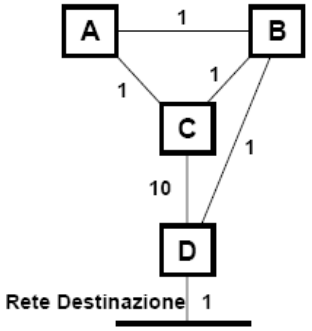
- Node A → Metric = 3 Router B
- Node B → Metric = 2 Router D
- Node C → Metric = 3 Router B
- Node D → Metric = 1 Directly connected

RIP: Counting to infinity – Example 2/6



- If link BD faults (metric becomes infinite):
  - All paths will croos node C
  - Very slow convergence
  - **Convergence time depends on value of metric at link CD**
- Process is the following:
  - Node B soon senses fault (L(B,D) = infinity)
  - Nodes A and C send RIP update messages for route to D indicating new paths, respectively through C and A
  - New paths cross B
  - B erroneously believes that there is a route to D traversing A or C
  - Routing converges only after hop count reaches value of the weight on link CD

RIP: Counting to infinity – Example 3/6



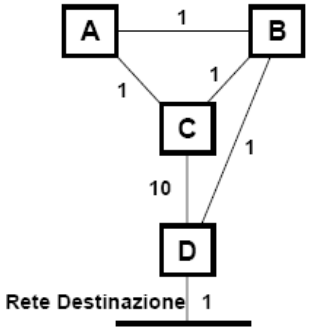
- Evolution of path towards dest. network starting from routers A, B, C and D

Step 1

Nodo	Next hop	Metr.
A	B	3
B	----	∞
C	B	3
D	Diretto	1

- **B updates A and C with new vector L**
- **Meanwhile, B receives from A and C the information (incorrect) that from A and C it is possible to reach destination network with metric cost 3**

RIP: Counting to infinity – Example 4/6



- Evolution of path towards dest. network starting from routers A, B, C and D

Step 1

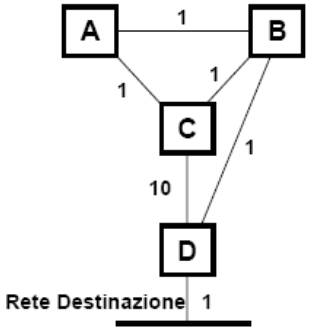
Nodo	Next hop	Metr.
A	B	3
B	----	∞
C	B	3
D	Diretto	1

Step 2

Nodo	Next hop	Metr.
A	C	4
B	C	4
C	A	4
D	Diretto	1

- **A and C receive B's L vector and change their paths toward destination**

RIP: Counting to infinity – Example 5/6



- Evolution of path towards dest. network starting from routers A, B, C and D

Step 1

Nodo	Next hop	Metr.
A	B	3
B	----	∞
C	B	3
D	Diretto	1

Step 2

Nodo	Next hop	Metr.
A	C	4
B	C	4
C	A	4
D	Diretto	1

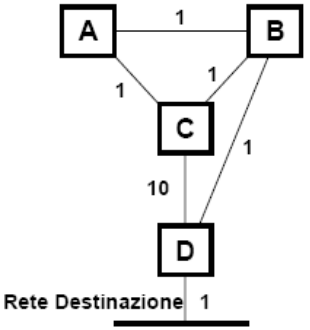
Step 3

Nodo	Next hop	Metr.
A	C	5
B	C	5
C	A	5
D	Diretto	1

At this point:

- **B receives L vectors from A and C**
- **A receives L vector from B and C**
- **C receives L vector from A and B**

RIP: Counting to infinity – Example 6/6



- Evolution of path towards dest. network starting from routers A, B, C and D

Step 1

Nodo	Next hop	Metr.
A	B	3
B	----	∞
C	B	3
D	Diretto	1

Step 2

Nodo	Next hop	Metr.
A	C	4
B	C	4
C	A	4
D	Diretto	1

Step 3

Nodo	Next hop	Metr.
A	C	5
B	C	5
C	A	5
D	Diretto	1

Step 4

Nodo	Next hop	Metr.
A	C	6
B	C	6
C	A	6
D	Diretto	1

Loop terminates when metric of temporary path becomes larger than that of real, alternative path

Step 10

Nodo	Next hop	Metr.
A	C	12
B	C	12
C	D	11
D	Diretto	1

RIP: Split Horizon

- Goal: avoid count to infinity [routing loop]

- In the following situation it makes no sense that
  - A tries to reach X over B
  - A announces to B its own distance to X

- A node sends different update messages (so, different versions of L vector) to neighbours, according to the following rules:
  - Omit, in vector L sent over a link, metrics of paths computed from updates received from that link (simple split horizon)
  - In an L vector transmitted over a link, set to ∞ metrics for paths recomputed from updates received from that link (split horizon with poisonous reverse)

RIP: Split Horizon – Example

- Evolution of shortest path to destination network starting at routers A, B, C and D

Step 1

Nodo	Next hop	Metr.
A	B	3
B	----	∞
C	B	3
D	Diretto	1

Step 2

Nodo	Next hop	Metr.
A	C	4
B	---	∞
C	A	4
D	Diretto	1

Step 3

Nodo	Next hop	Metr.
A	---	∞
B	---	∞
C	D	11
D	Diretto	1

Step 4

Nodo	Next hop	Metr.
A	C	12
B	C	12
C	D	11
D	Diretto	1

RIP: Triggered Updates

- Split horizon with poisonous reverse technique interrupts loops onvolving two routers
  - Loops involving three or more nodes possible
    - A believes having a path through B
    - B believes having a path through C
    - C believes having a path through A
  - Remedy:
    - Every time a routing table is updated, node sends update messages to adjacent nodes
    - Convergence time reduces, since it eliminates timeout waiting time

RIP v1

- Format compatible with Berkeley Software Distribution (BSD) UNIX software
- A RIP message (**advertisement**) is trasmitted over UDP (port 520)
- **Two message types**
  - **Request**
    - Sent by a router to a neighbour to request neighbour's routing table or part thereof
  - **Response**
    - **Used by a router to send whol or part of router's distance vector table**
    - Sent:
      - Every 30 seconds [**refresh**]
      - In response to a Request packet
      - When routing table changes (Triggered updates)

RIP v1

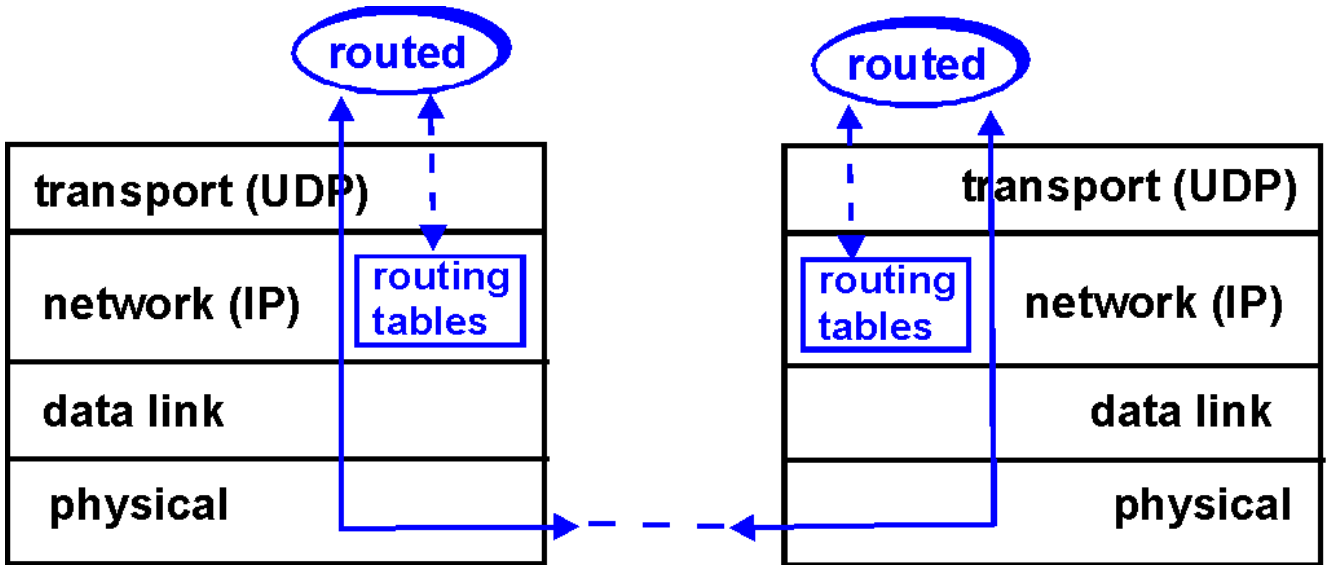
- Maximum size of RIP messages is 512 bytes
  - At most 25 blocks per message
  - If more than 25 destinations neew to be refreshed → multiple RIP messages
- Does not allow transmission of subnet mask
  - A router must know the structure of a subnet address in order to distinguish network prefix and host address
- Metric is an integer between 1 and 16
  - 16 means infinity
- A destination address is removed from routing table if it has not been “refreshed” for 180 s
- In the presence of frequent and consecutive changes in link state, RIP message are sent with frequency of 1 to 5 s

RIP v1

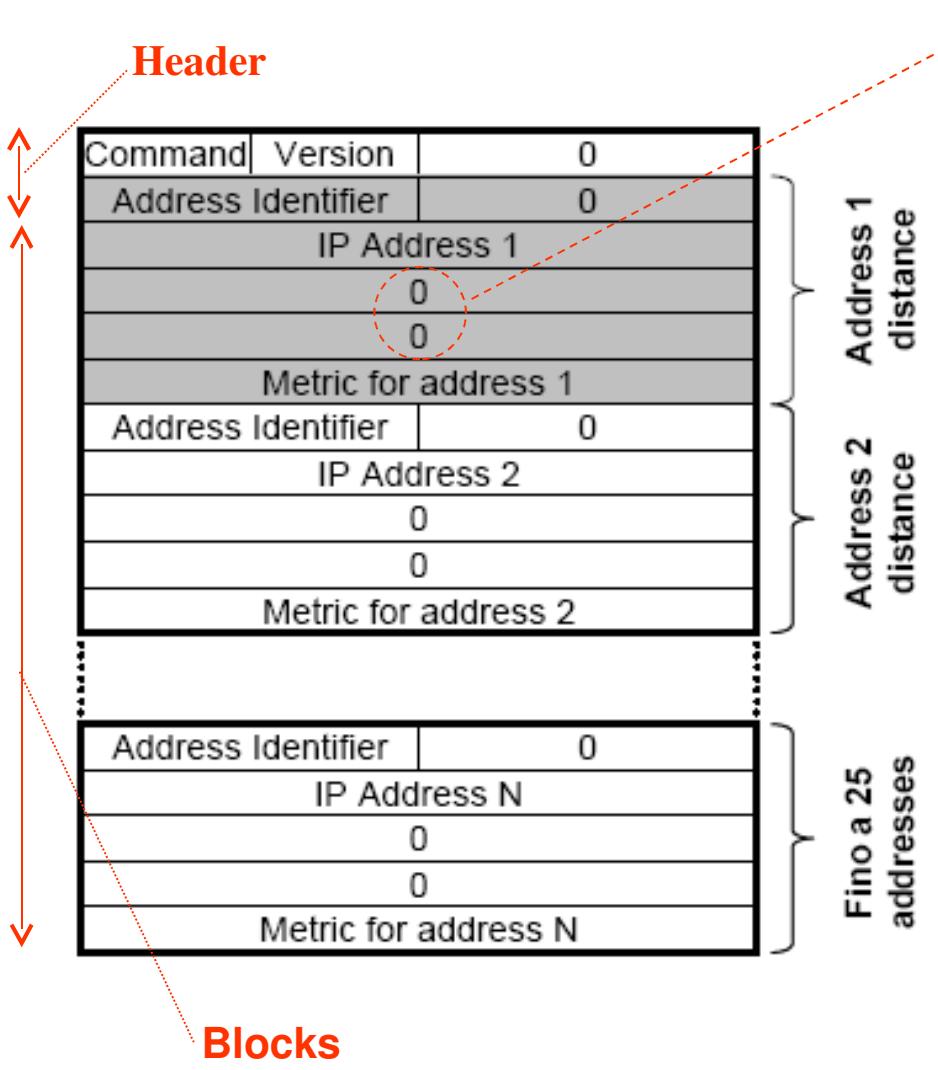
- Fault recovery:
  - If no advertisement received for 180 sec. [↔ network address not refreshed], assume link is “down”
    - Remove paths using that link; notify (send advertisement) neighbours
    - Neighbours do the same if respective routing tables change (see DVR scheme)
    - Information propagates over the networks

RIP v1

- Routing table management
  - RIP routing tables are managed by an application process called route-d (daemon in Unix terminology)
  - Advertisement messages encapsulated in UDP packets (no need for reliable transfer since updates sent periodically)



RIP v1



Header

- **Command**
  - request
  - response
- **Version**

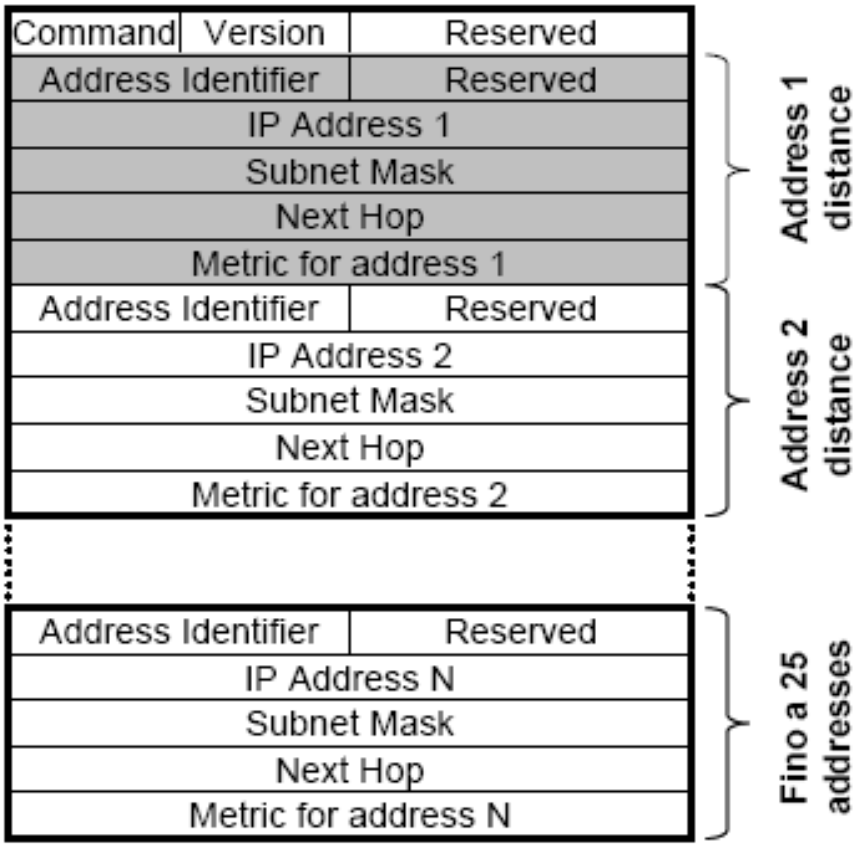
Block

- **IP address**
  - rete, sottorete o host
- **Metric**
  - Distance from network indicated by IP address

RIP v2

- Extends RIP v1 functionalities
  - Can be adopted in medium-sized networks
  - Supports variable subnetting or supernetting (CIDR)
  - Backward-compatible with RIP v1
  - Routers of different type may coexist on the same subnet
  - Supports authentication
- Metric same as RIP v1
  - Integer from 1 to 16
  - Value 16 means •
- RIP v2 uses “reserved” bytes in RIP v1 packet format

RIP v2



- **IP address**
  - Net, subnet or host
- **Next Hop**
  - Next hop router on path to destination address in the block. Used for routers not running RIP
- **Metric**
  - Distance from destination

EIGRP

- **Enhanced Interior Gateway Routing Protocol** (EIGRP) è un altro protocollo della famiglia Distance Vector
- È un protocollo di routing proprietario [Cisco] utilizzato dai router per scambiarsi informazioni di routing all'interno di un Autonomous System
- Il protocollo è progettato per minimizzare
  - l'instabilità dei percorsi dopo un cambiamento della topologia della rete
  - l'overhead associato al traffico addizionale
  - l'utilizzo del processore dei router.