Dynamic Routing Protocols I
RIP

Relates to Lab 4.

The first module on dynamic routing protocols. This module provides an overview of routing, introduces terminology (interdomain, intradomain, autonomous system),

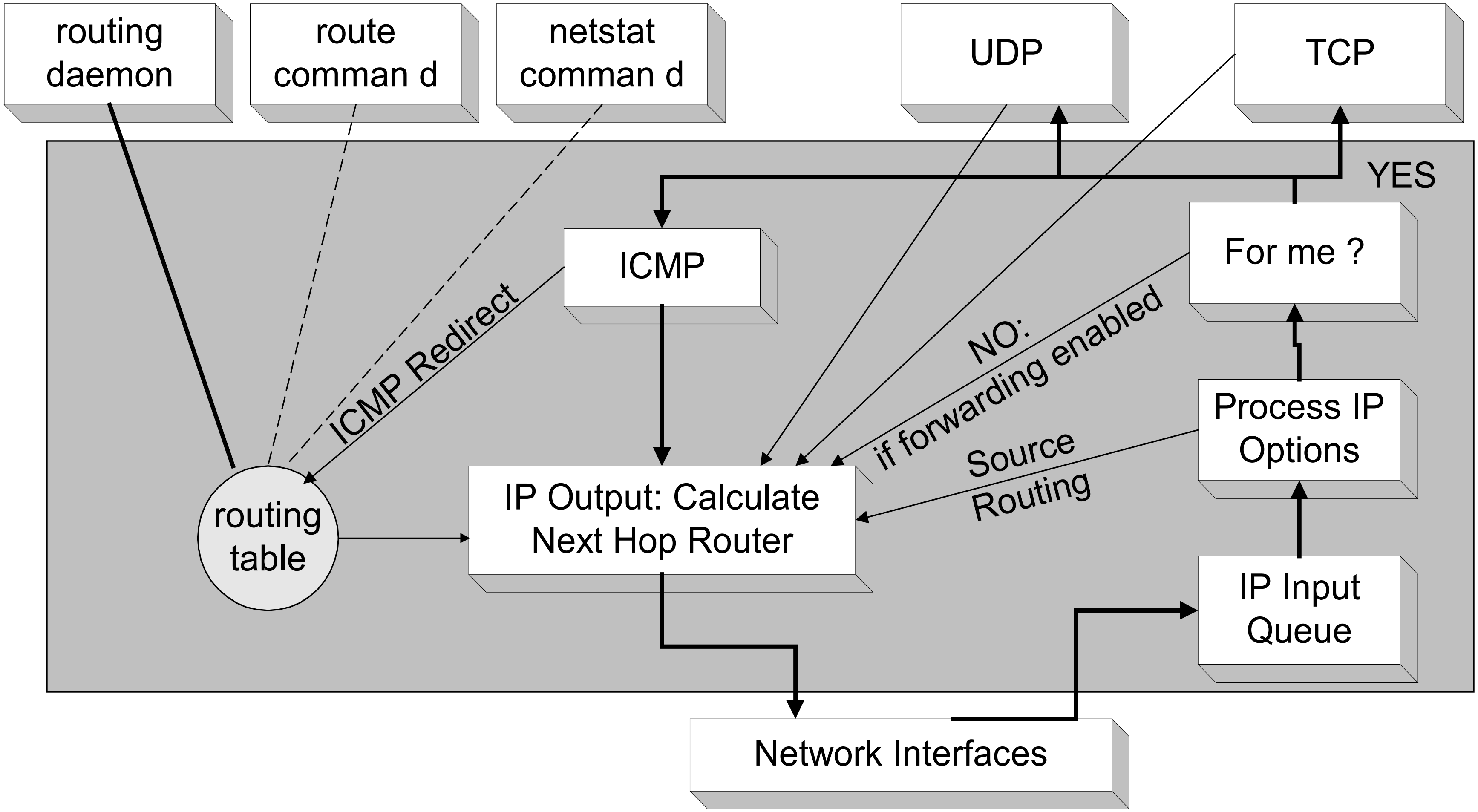
Routing

- **Recall:** There are two parts to routing IP packets:
  1. How to pass a packet from an input interface to the output interface of a router (packet forwarding) ?
  2. How to find and setup a route ?

- We already discussed the packet forwarding part

- There are two approaches for calculating the routing tables:
  - Static Routing
  - Dynamic Routing: Routes are calculated by a routing protocol

IP Routing



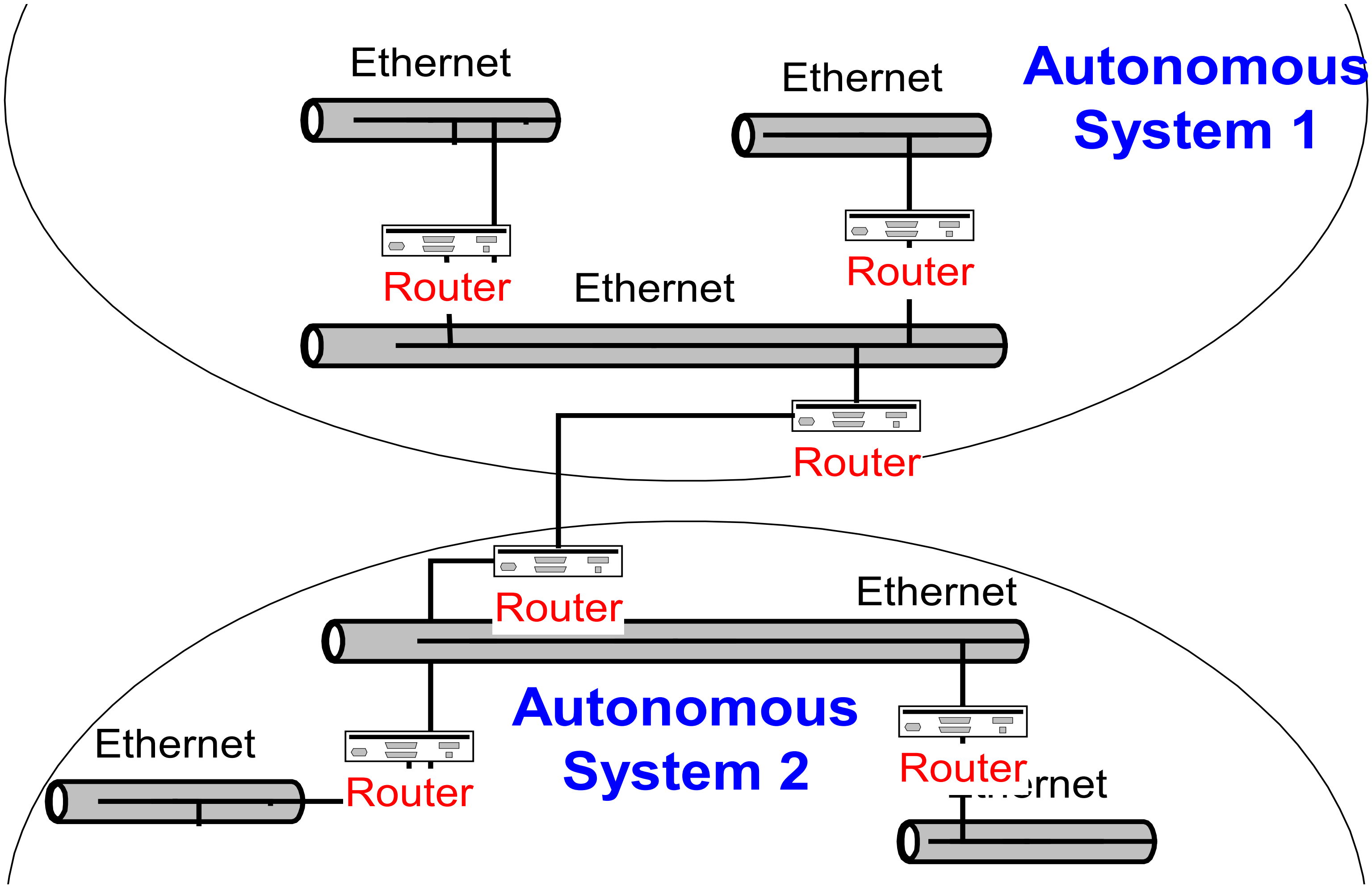
routing daemon
route comman d
netstat comman d
UDP
TCP
YES
ICMP
For me ?
ICMP Redirect
NO: if forwarding enabled
Process IP Options
Source Routing
routing table
IP Output: Calculate Next Hop Router
IP Input Queue
Network Interfaces

Autonomous Systems

- An **autonomous system** is a region of the Internet that is administered by a single entity.
- Examples of autonomous regions are:
    - UVA’s campus network
    - MCI’s backbone network
    - Regional Internet Service Provider

- Routing is done differently within an autonomous system (**intradomain routing**) and between autonomous system **(interdomain routing**).

Autonomous Systems (AS)



Interdomain and Intradomain Routing

Intradomain Routing

- Routing within an AS
- Ignores the Internet outside the AS
- Protocols for Intradomain routing are also called **Interior Gateway Protocols** or **IGP's**.
- Popular protocols are
  - RIP (simple, old)
  - OSPF (better)

Interdomain Routing

- Routing between AS's
- Assumes that the Internet consists of a collection of interconnected AS's
- Normally, there is one dedicated router in each AS that handles interdomain traffic.
- Protocols for interdomain routing are also called **Exterior Gateway Protocols** or **EGP's.**
- Routing protocols:
  - EGP
  - BGP (more recent)

Components of a Routing Algorithm

- A procedure for sending and receiving reachability information about network to other routers

- A procedure for calculating optimal routes
  - Routes are calculated using a shortest path algorithm:
    - **Goal**: Given a network were each link is assigned a cost. Find the path with the least cost between two networks with minimum cost.

- A procedures for reacting to and advertising topology changes

Approaches to Shortest Path Routing

- There are two basic routing algorithms found on the Internet.

1. Distance Vector Routing

- Each node knows the distance (=cost) to its directly connected neighbors
- A node sends periodically a list of routing updates to its neighbors.
- If all nodes update their distances, the routing tables eventually converge
- New nodes advertise themselves to their neighbors

2. Link State Routing

- Each node knows the distance to its neighbors
- The distance information (=link state) is broadcast to all nodes in the network
- Each node calculates the routing tables independently

Routing Algorithms in the Internet

Distance Vector	Link State
• **Routing Information Protocol (RIP)** • Gateway-to-Gateway Protocol (GGP) • Exterior Gateway Protocol (EGP) • Interior Gateway Routing Protocol (IGRP)	• Intermediate System - Intermediate System (IS-IS) • **Open Shortest Path First (OSPF)**

RIP

- RIP is **dynamic routing protocol** used in local and wide area network
- Uses **Distance Vector Routing** algorithm
- Defined in **RFC 1058 (1988)**
- **RIPng** (RIP Next Generation) – IPv6

History

- Bellman-Ford algorithm – 1967 – ARPANET
- Earliest version – RIP – Gateway Information Protocol
- Later version – RIP was part of Xerox Network System (XNS)
- **Hop count** – routing metric

- **Prevents routing loops** ***by implementing a limit on the number of hops allowed in path*** from source to destination

- **Maximum hop count is 15**

- A hop count 16 is considered an infinite distance

- Implements **Split Horizon, Route Poisoning**, and **holddown** mechanism to prevent incorrect routing

Convergence

- Process of getting consistent routing information to all other nodes

Dynamic IP Routing Protocols

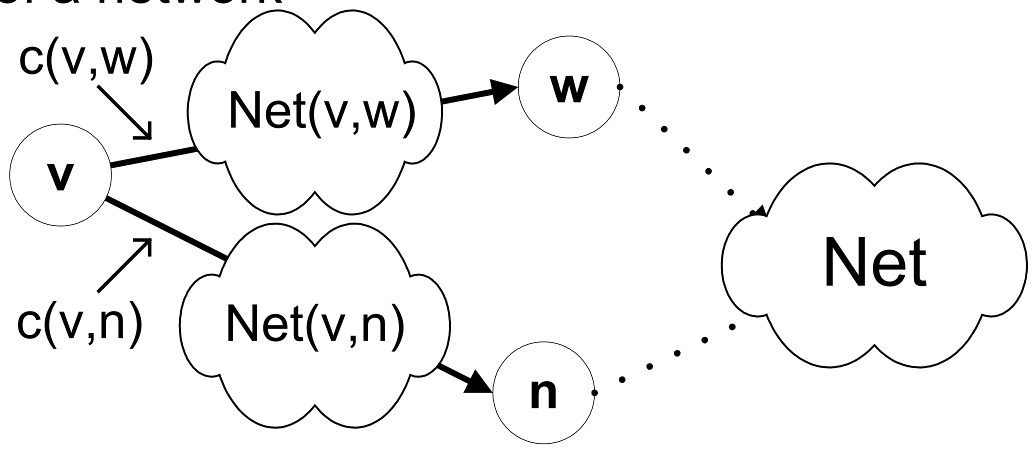
- In Unix systems, the dynamic setting of routing tables is done by the **routed** or **gated** daemons
- The routing daemons execute the following intradomain and interdomain routing protocols

	intradomain			*interdomain*	
Daemon	***Hello***	***RIP***	***OSPF***	***EGP***	***BGP***
`routed`		V1			
`Gated` **`(Version 3)`**	Yes	V1 V2	V2	Yes	V2, V3

A network as a graph

- **In the following, networks are represented as a network graph**:
  - nodes are connected by networks
    - network can be a link or a LAN
  - network interface has cost
  - networks are destinations
  - Net(v,w) is an IP address of a network

- For ease of notation, we often replace the clouds between nodes by simple links.

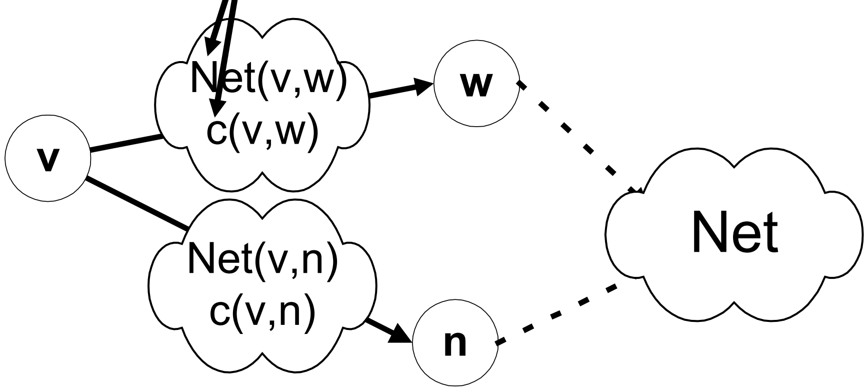


Distance Vector Algorithm: Routing Table

c(v,w): cost to transmit on the interface to network Net(v,w)

Net(v,w): Network address of the network between v and w.

The network can be a link, but could also be a LAN



RoutingTable of node v

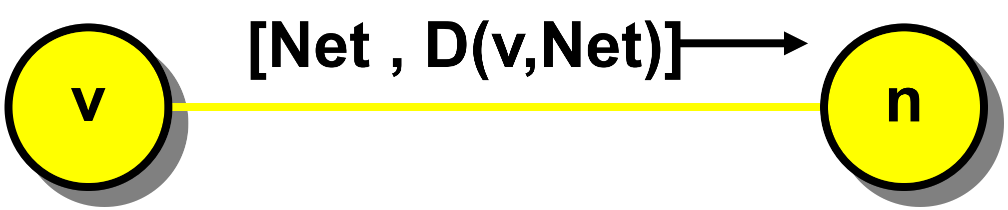
Dest	via (next hop)	cost
Net	n	D(v,Net)

Distance Vector Algorithm: Messages

RoutingTable of node v

Dest	via (next hop)	cost
Net	n	D(v,Net)

- Nodes send messages to their neighbors which contain routing table entries



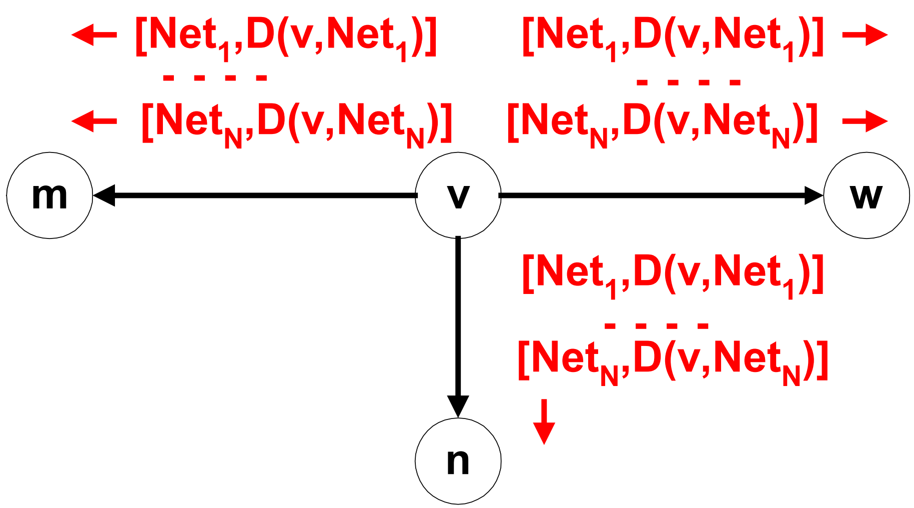
- A message has the format: **[Net , D(v,Net)]** means ***"My cost to go to Net is D (v,Net)"***

Distance Vector Algorithm: Sending Updates

RoutingTable of node v

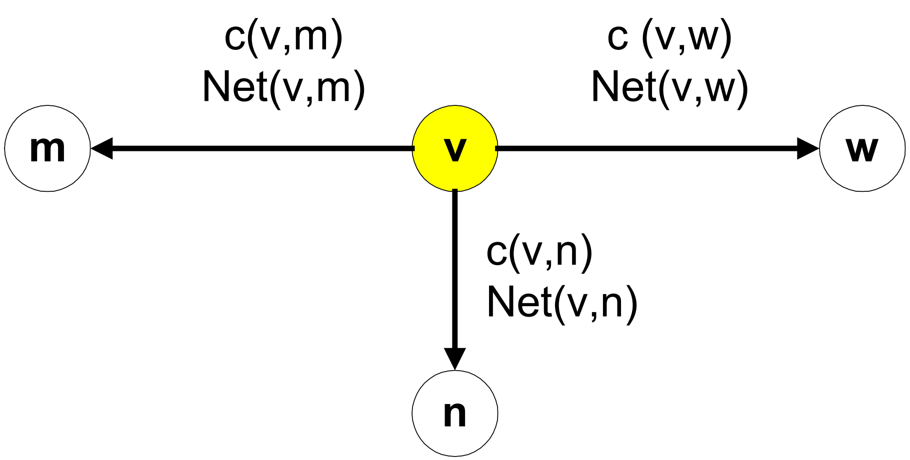
Dest	via (next hop)	cost
Net_1	m	$D(v,Net_1)$
Net_2	n	$D(v,Net_2)$
- - -	- - -	- - -
Net_N	w	$D(v,Net_N)$

Periodically, each node v sends the content of its routing table to its neighbors:



Initiating Routing Table I

- Suppose a new node v becomes active.
- The cost to access directly connected networks is zero:
  - D (v, Net(v,m)) = 0
  - D (v, Net(v,w)) = 0
  - D (v, Net(v,n)) = 0



RoutingTable

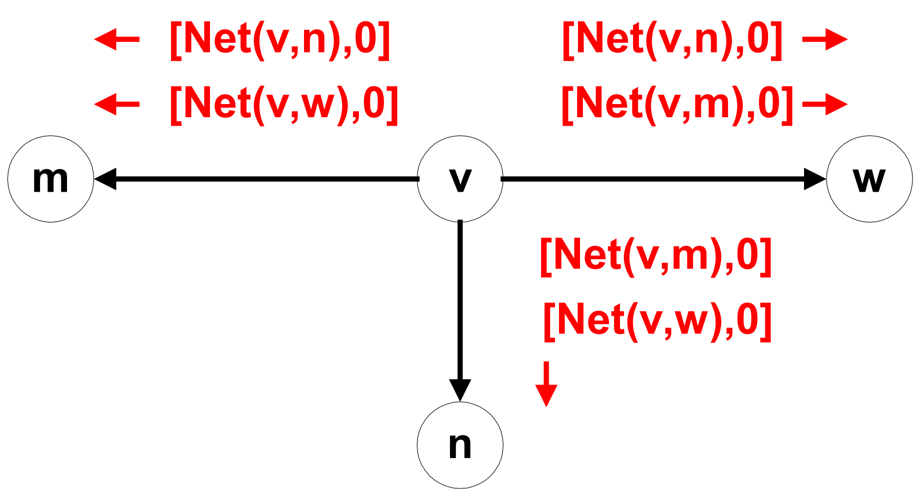
Dest	via (next hop)	cost
Net(v,m)	m	0
Net(v,w)	w	0
Net(v,n)	n	0

Initiating Routing Table II

RoutingTable

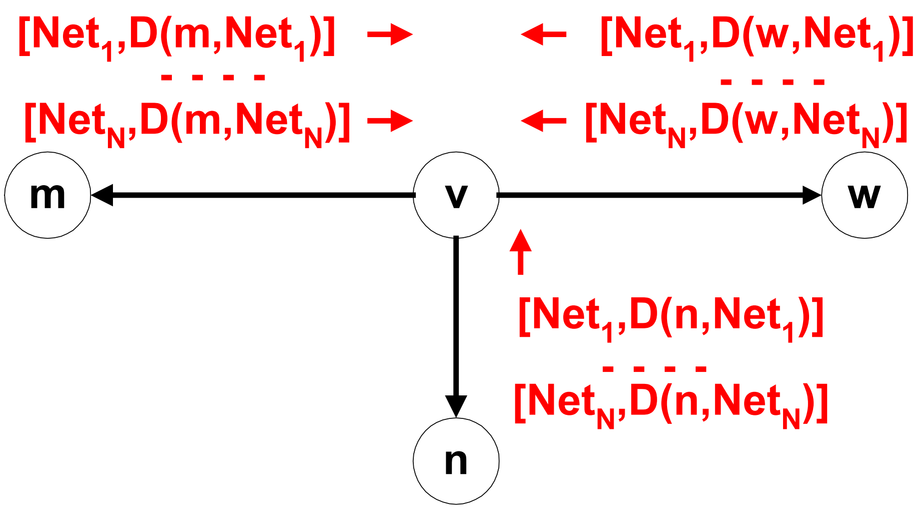
Dest	via (next hop)	cost
Net(v,m)	m	0
Net(v,w)	w	0
Net(v,n)	n	0

- New node v sends the routing table entry to all its neighbors:



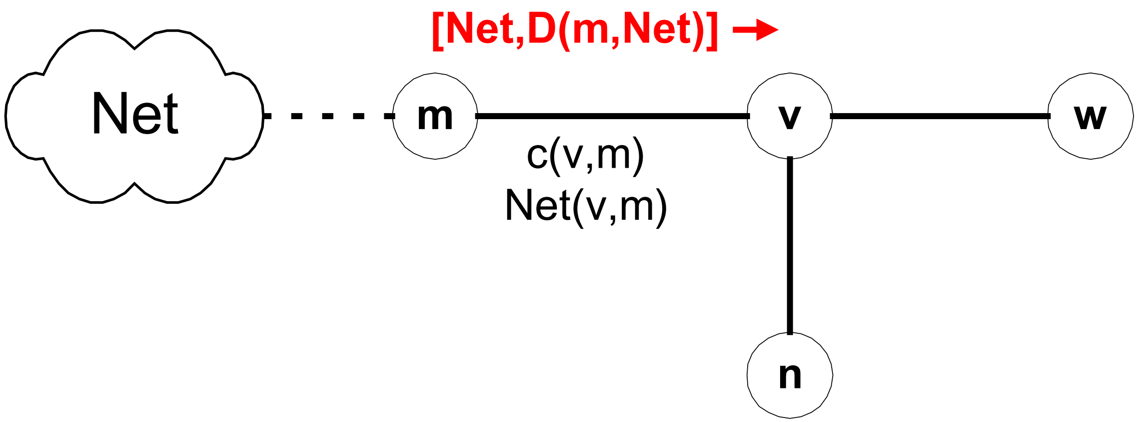
Initiating Routing Table III

- Node v receives the routing tables from other nodes and builds up its routing table



Updating Routing Tables I

- Suppose node v receives a message from node m: **[Net,D(m,Net)]**

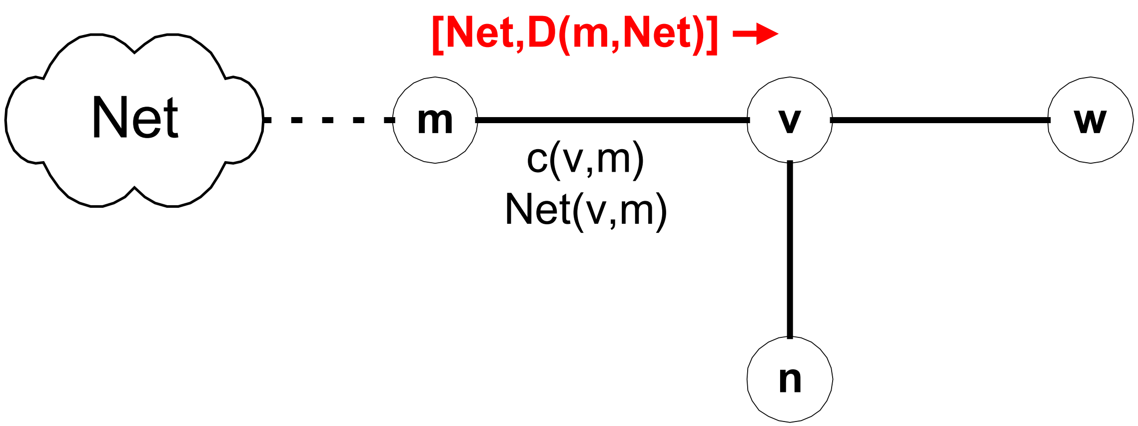


Node v updates its routing table and sends out further messages if the message reduces the cost of a route:

```
if  ( D(m,Net) + c (v,m) < D (v,Net) ) {
      D^new (v,Net) := D (m,Net) + c (v,m);
      Update routing table;
      send message [Net, D^new (v,Net)] to all neighbors
}
```

Updating Routing Tables II

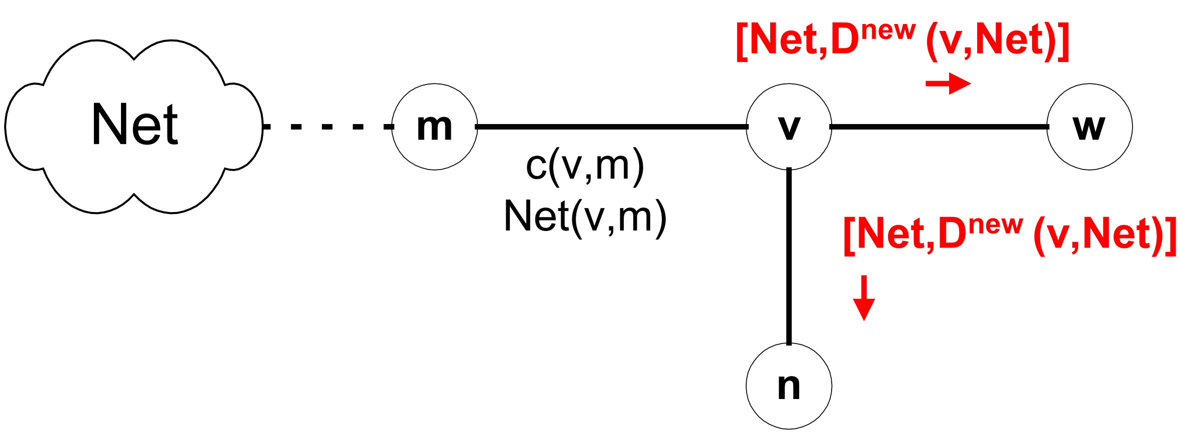
- Before receiving the message:



RoutingTable

Dest	via (next hop)	cost
Net	??	D(v,Net)

- Suppose D (m,Net) + c (v,m) < D (v,Net):

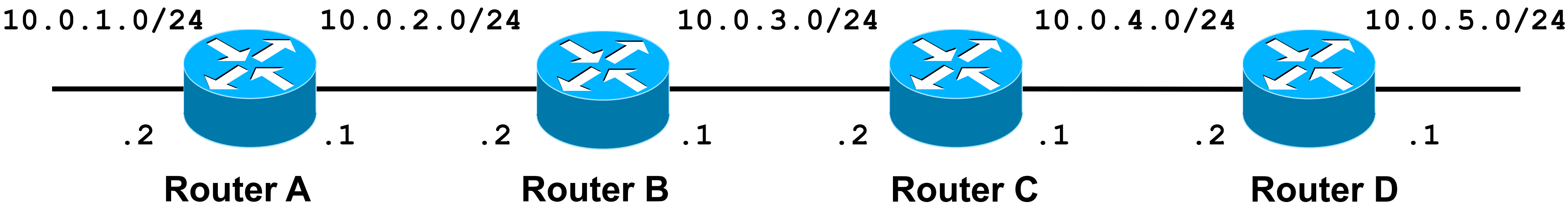


RoutingTable

Dest	via (next hop)	cost
Net	m	D^{new}(v,Net)

Example

Assume: - link cost is 1, i.e., c(v,w) = 1
- all updates, updates occur simultaneously
- Initially, each router only knows the cost of connected interfaces



Router A

Net	via	cost
t=0:		
10.0.1.0	-	0
10.0.2.0	-	0
t=1:		
10.0.1.0	-	0
10.0.2.0	-	0
10.0.3.0	10.0.2.2	1
t=2:		
10.0.1.0	-	0
10.0.2.0	-	0
10.0.3.0	10.0.2.2	1
10.0.4.0	10.0.2.2	2

Router B

Net	via	cost
t=0:		
10.0.2.0	-	0
10.0.3.0	-	0
t=1:		
10.0.1.0	10.0.2.1	1
10.0.2.0	-	0
10.0.3.0	-	0
10.0.4.0	10.0.3.2	1
t=2:		
10.0.1.0	10.0.2.1	1
10.0.2.0	-	0
10.0.3.0	-	0
10.0.4.0	10.0.3.2	1
10.0.5.0	10.0.3.2	2

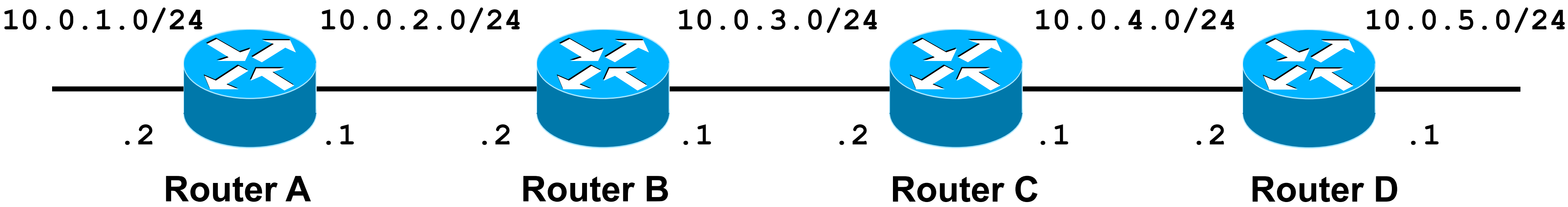
Router C

Net	via	cost
t=0:		
10.0.3.0	-	0
10.0.4.0	-	0
t=1:		
10.0.2.0	10.0.3.1	1
10.0.3.0	-	0
10.0.4.0	-	0
10.0.5.0	10.0.4.2	1
t=2:		
10.0.1.0	10.0.3.1	2
10.0.2.0	10.0.3.1	1
10.0.3.0	-	0
10.0.4.0	-	0
10.0.5.0	10.0.4.2	1

Router D

Net	via	cost
t=0:		
10.0.4.0	-	0
10.0.5.0	-	0
t=1:		
10.0.3.0	10.0.4.1	1
10.0.4.0	-	0
10.0.5.0	-	0
t=2:		
10.0.2.0	10.0.4.1	2
10.0.3.0	10.0.4.1	1
10.0.4.0	-	0
10.0.5.0	-	0

Example



Router A

Net	via	cost
t=2:		
10.0.1.0	-	0
10.0.2.0	-	0
10.0.3.0	10.0.2.2	1
10.0.4.0	10.0.2.2	2
t=3:		
10.0.1.0	-	0
10.0.2.0	-	0
10.0.3.0	10.0.2.2	1
10.0.4.0	10.0.2.2	2
10.0.5.0	10.0.2.2	3

Router B

Net	via	cost
t=2:		
10.0.1.0	10.0.2.1	1
10.0.2.0	-	0
10.0.3.0	-	0
10.0.4.0	10.0.3.2	1
10.0.5.0	10.0.3.2	2
t=3:		
10.0.1.0	10.0.2.1	1
10.0.2.0	-	0
10.0.3.0	-	0
10.0.4.0	10.0.3.2	1
10.0.5.0	10.0.3.2	2

Router C

Net	via	cost
t=2:		
10.0.1.0	10.0.3.1	2
10.0.2.0	10.0.3.1	1
10.0.3.0	-	0
10.0.4.0	-	0
10.0.5.0	10.0.4.2	1
t=3:		
10.0.1.0	10.0.3.1	2
10.0.2.0	10.0.3.1	1
10.0.3.0	-	0
10.0.4.0	-	0
10.0.5.0	10.0.4.2	1

Router D

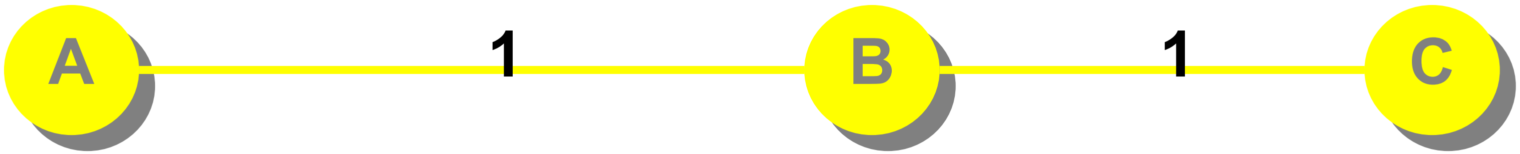
Net	via	cost
t=2:		
10.0.2.0	10.0.4.1	2
10.0.3.0	10.0.4.1	1
10.0.4.0	-	0
10.0.5.0	-	0
t=3:		
10.0.1.0	10.0.4.1	3
10.0.2.0	10.0.4.1	2
10.0.3.0	10.0.4.1	1
10.0.4.0	-	0
10.0.5.0	-	0

Now, routing tables have converged !

Characteristics of Distance Vector Routing

- **Periodic Updates:** Updates to the routing tables are sent at the end of a certain time period. A typical value is 90 seconds.
- **Triggered Updates:** If a metric changes on a link, a router immediately sends out an update without waiting for the end of the update period.
- **Full Routing Table Update**: Most distance vector routing protocol send their neighbors the entire routing table (not only entries which change).
- **Route invalidation timers:** Routing table entries are invalid if they are not refreshed. A typical value is to invalidate an entry if no update is received after 3-6 update periods.

The Count-to-Infinity Problem



A's Routing Table

to	via (next hop)	cost
C	B	2

B's Routing Table

to	via (next hop)	cost
C	C	1

now link B-C goes down

C	B	2					C	-	oo
			C	2 →	← C	oo			
C	-	oo					C	A	3
			C	oo →	← C	3			
C	B	4					C	-	oo
			C	4 →	← C	oo			

Count-to-Infinity

- The reason for the count-to-infinity problem is that each node only has a “next-hop-view”
- For example, in the first step, A did not realize that its route (with cost 2) to C went through node B

- How can the Count-to-Infinity problem be solved?

Count-to-Infinity

- The reason for the count-to-infinity problem is that each node only has a “next-hop-view”
- For example, in the first step, A did not realize that its route (with cost 2) to C went through node B

- How can the Count-to-Infinity problem be solved?
- **Solution 1:** Always advertise the entire path in an update message (**Path vectors**)
  - If routing tables are large, the routing messages require substantial bandwidth
  - BGP uses this solution

Count-to-Infinity

- The reason for the count-to-infinity problem is that each node only has a "next-hop-view"
- For example, in the first step, A did not realize that its route (with cost 2) to C went through node B

- How can the Count-to-Infinity problem be solved?
- **Solution 2:** Never advertise the cost to a neighbor if this neighbor is the next hop on the current path **(Split Horizon)**
  - Example: A would not send the first routing update to B, since B is the next hop on A's current route to C
  - Split Horizon does not solve count-to-infinity in all cases!

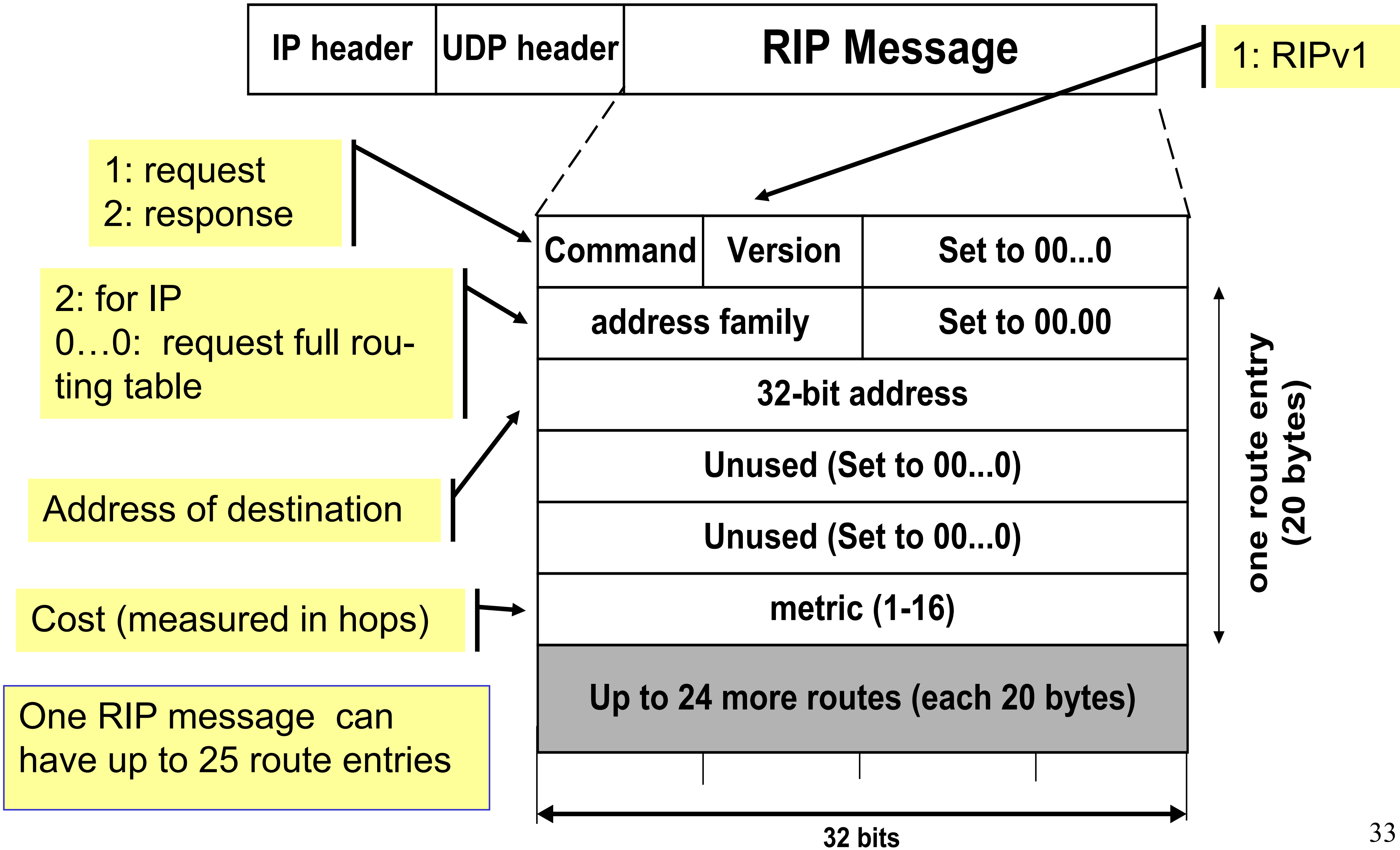
RIP - Routing Information Protocol

- A simple intradomain protocol
- Straightforward implementation of Distance Vector Routing
- ***Each router advertises its distance vector every 30 seconds (or whenever its routing table changes) to all of its neighbors***
- ***RIP always uses 1 as link metric***
- ***Maximum hop count is 15, with "16" equal to "∞"***
- ***Routes are timeout (set to 16) after 3 minutes if they are not updated***

RIP - History

- Late 1960s : Distance Vector protocols were used in the ARPANET
- Mid-1970s: XNS (Xerox Network system) routing protocol is the precursor of RIP in IP (and Novell's IPX RIP and Apple's routing protocol)
- 1982 Release of **routed** for BSD Unix
- 1988 RIPv1 (RFC 1058)
  - classful routing
- 1993 RIPv2 (RFC 1388)
  - adds subnet masks with each route entry
  - allows classless routing
- 1998 Current version of RIPv2 (RFC 2453)

RIPv1 Packet Format

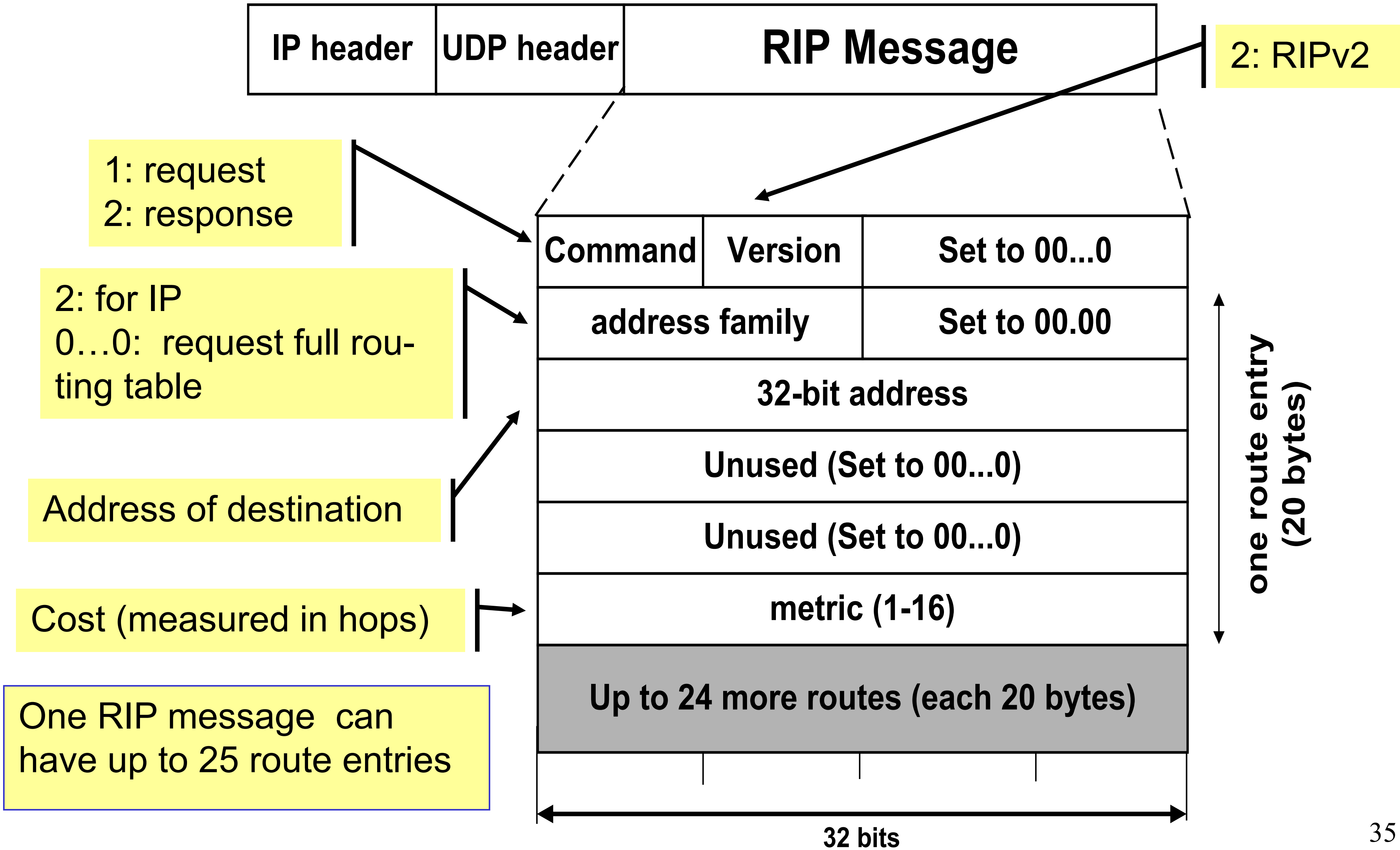


RIPv2

- RIPv2 is an extends RIPv1:
  - Subnet masks are carried in the route information
  - Authentication of routing messages
  - Route information carries next-hop address
  - Exploites IP multicasting

- Extensions of RIPv2 are carried in unused fields of RIPv1 messages

RIPv2 Packet Format



RIPv2 Packet Format

IP header	UDP header	RIPv2 Message

2: RIPv2

Command	Version	Set to 00.00
address family		route tag
IP address		
Subnet Mask		
Next-Hop IP address		
metric (1-16)		
Up to 24 more routes (each 20 bytes)		

32 bits

one route entry (20 bytes)

Used to carry information from other routing protocols (e.g., autonomous system number)

Subnet mask for IP address

Identifies a better next-hop address on the same subnet than the advertising router, if one exists (otherwise 0….0)

RIP Messages

- This is the operation of RIP in `routed`. Dedicated port for RIP is UDP port 520.

- Two types of messages:
  - **Request messages**
    - used to ask neighboring nodes for an update
  - **Response messages**
    - contains an update

Routing with RIP

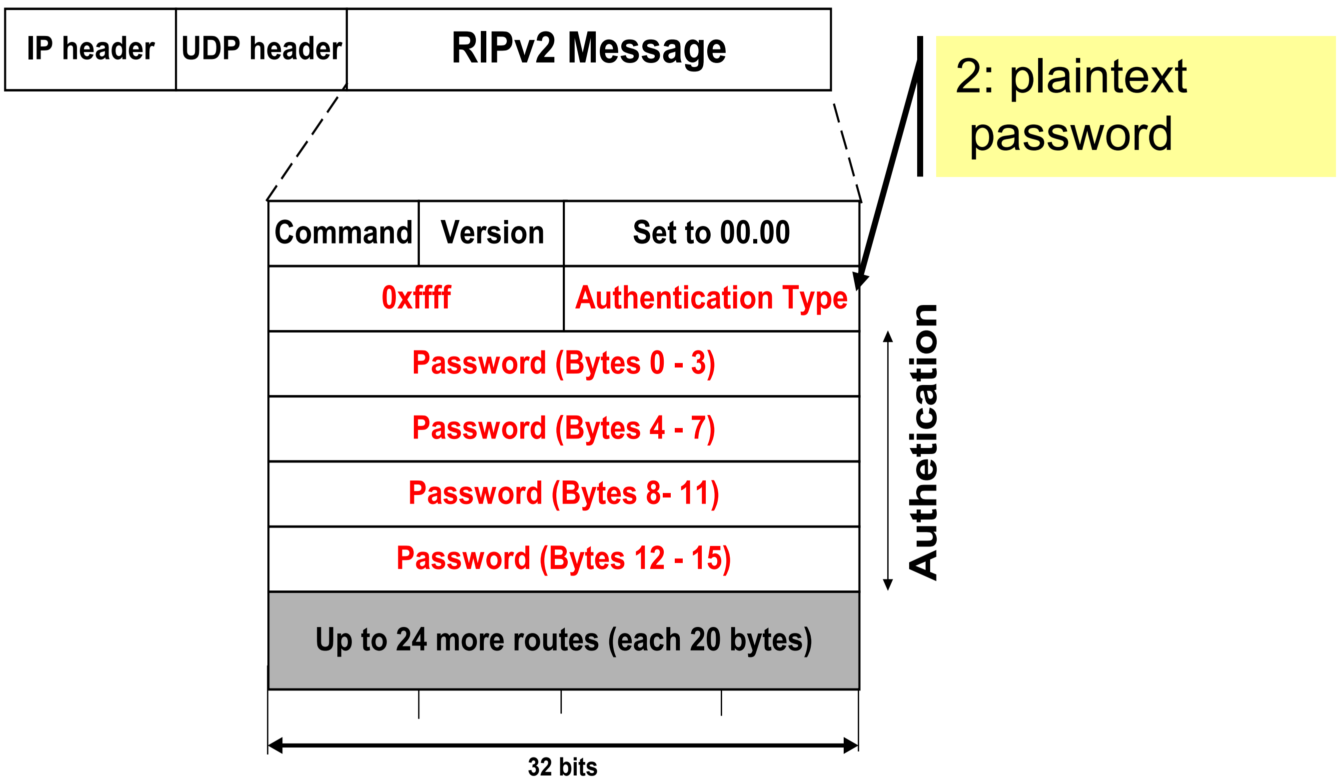
- **Initialization:** Send a **request packet** (command = 1, address family=0..0) on all interfaces:
    - RIPv1 uses broadcast if possible,
    - RIPv2 uses multicast address 224.0.0.9, if possible

  requesting routing tables from neighboring routers
- **Request received**: Routers that receive above request send their entire routing table
- **Response received**: Update the routing table

- **Regular routing updates**: Every 30 seconds, send all or part of the routing tables to every neighbor in an response message
- **Triggered Updates:** Whenever the metric for a route change, send entire routing table.

RIP Security

- Issue: Sending bogus routing updates to a router
- RIPv1: No protection
- RIPv2: Simple authentication scheme



RIP Problems

- RIP takes a long time to stabilize
  - Even for a small network, it takes several minutes until the routing tables have settled after a change
- RIP has all the problems of distance vector algorithms, e.g., count-to-Infinity
    - » RIP uses split horizon to avoid count-to-infinity
- The maximum path in RIP is 15 hops

Difference between Distance Vector & Link State

Distance Vector	Link State
• Dynamic routing algorithm	• Dynamic routing algorithm
• Each router computes a distance between itself and each possible destination i.e. its immediate neighbors	• Each router shares knowledge of its neighbors with every other router in the network
• Router **shares its knowledge about the whole network to its neighbors** and accordingly updates the table based on its neighbors.	• Router ***sends its information about its neighbors only to all the routers through flooding.***

Difference between Distance Vector & Link State

Distance Vector	Link State
• The sharing of information with the neighbors ***takes place at regular intervals.***	• Information sharing ***takes place only whenever there is a change***
• It makes ***use of Bellman-Ford Algorithm for making routing tables.***	• It makes ***use of Dijkstra's Algorithm for making routing tables.***
• ***Problems – Count to infinity problem which can be solved by splitting horizon.***	• ***Problems – Heavy traffic due to flooding of packets.***
	• Flooding can result in infinite looping which can be solved by using the Time to live (TTL) field