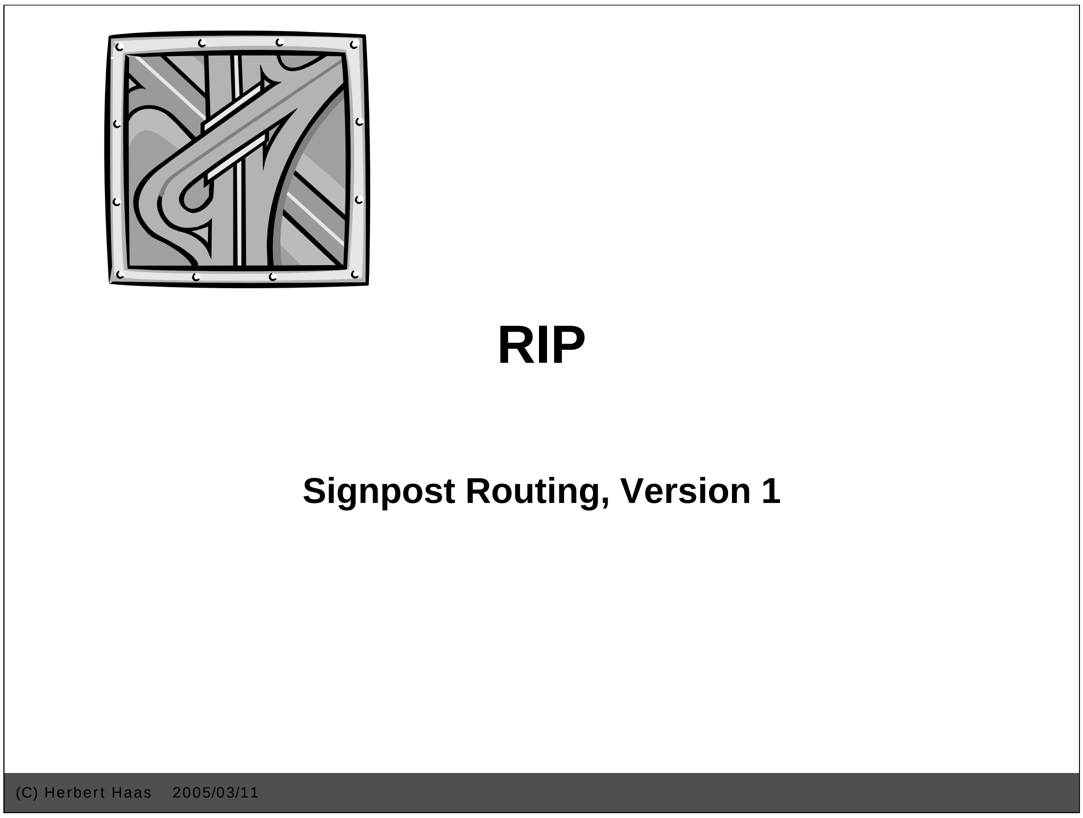
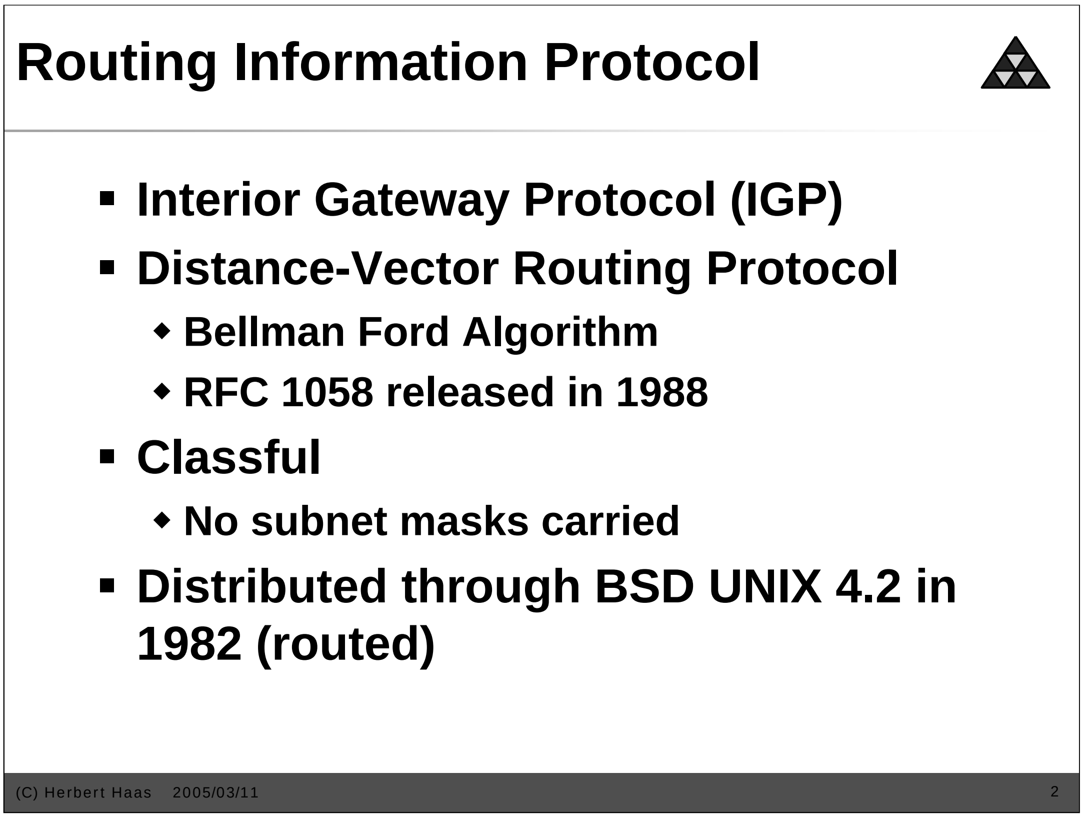


RIP

Signpost Routing, Version 1



Routing Information Protocol

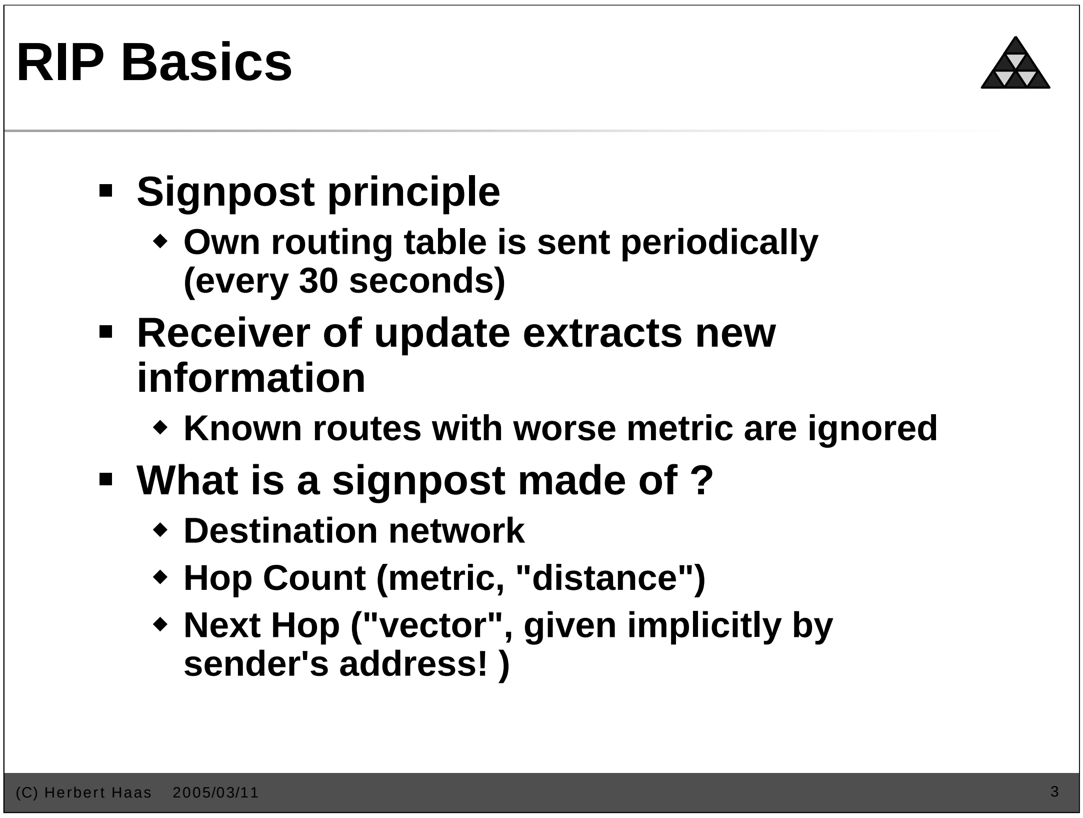
- Interior Gateway Protocol (IGP)
- Distance-Vector Routing Protocol
  - Bellman Ford Algorithm
  - RFC 1058 released in 1988
- Classful
  - No subnet masks carried
- Distributed through BSD UNIX 4.2 in 1982 (routed)

RIP is a so-called distance vector routing protocol – its routing updates are like "signposts" pointing to the shortest-hop path to known destination networks. The algorithm has been developed by R. E. Bellman, L. R. Ford, and D. R. Fulkerson and has first been implemented in the ARPANET in 1969. In the mid-1970s, Xerox created the "Gateway Information Protocol" (GWINFO) to route the Palo Alto Research Center (PARC) Universal Protocol, also known as "PUP". PUP became the Xerox Network Systems (XNS) protocol suite and GWINFO became XNS RIP. And XNS-RIP was the basis for Novell's IPX RIP, Appletalk's Routing Table Maintenance Protocol (RTMP), and IP RIP. We will only discuss IP RIP here.

RIP is an Interior Gateway Protocol (IGP), that is, RIP is only used inside an Autonomous System. Further explainations are given in the BGP modules.

RIP is an classful routing protocol, because RIP (version 1) does not bind subnet-masks to the routes. So RIP (version 1) assumes classful addressing. Subnet masks can be used as long as discontiguos subnetting is avoided.

Typically, every UNIX variant includes routed [route-dee for routing demon] as part of the operating system, so UNIX-workstations can be configured to determine each RIP router in the network and hence a default-route entry would not be necessary.

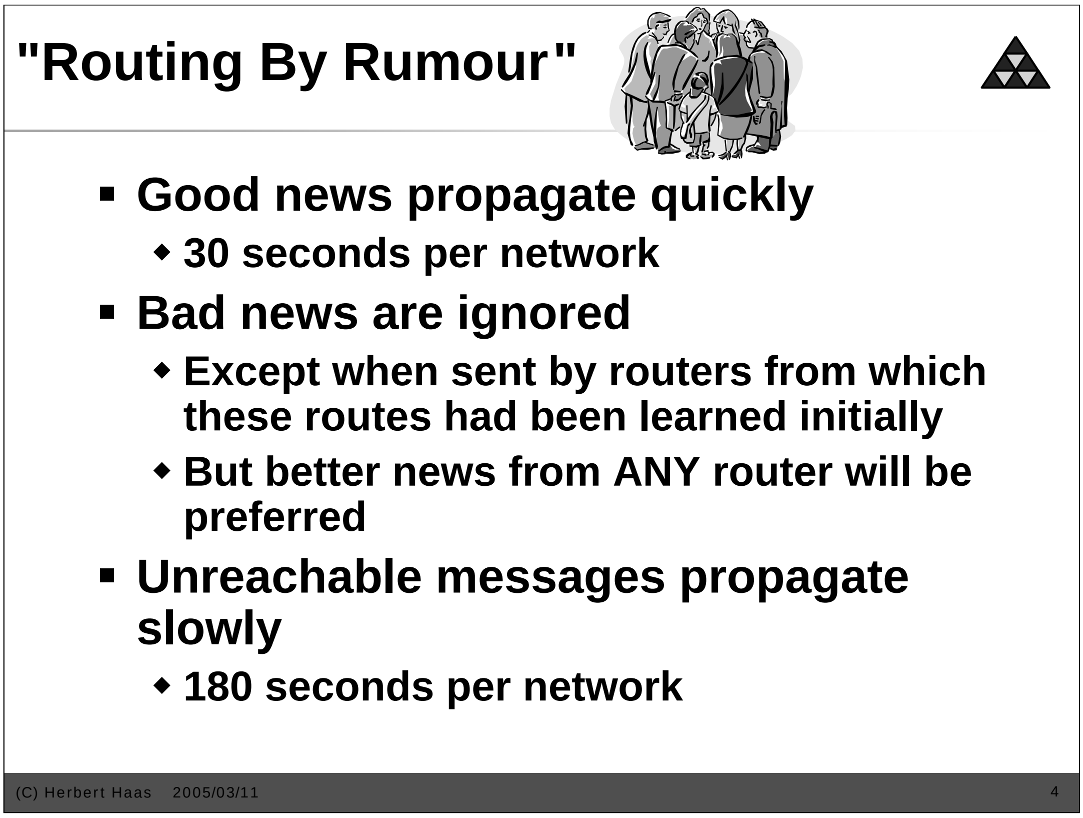


RIP Basics

- Signpost principle
  - Own routing table is sent periodically (every 30 seconds)
- Receiver of update extracts new information
  - Known routes with worse metric are ignored
- What is a signpost made of ?
  - Destination network
  - Hop Count (metric, "distance")
  - Next Hop ("vector", given implicitly by sender's address!)

The whole distance vector philosophy is based upon the signpost principle – each router sends periodically a copy of his own routing table to each neighbor. Upon receiving such routing update, a router extracts unknown routes or routes that improved in metrics. For RIP the update period is 30 seconds.

Using this principle, each router learns how to reach destinations only via signpost – the routing details *along* the path are unknown. The routing update (signpost) basically consists of a list of destination networks and hop counts ("distances") associated to it. For all these destinations there is only one next hop: the sending router's address.



"Routing By Rumour"

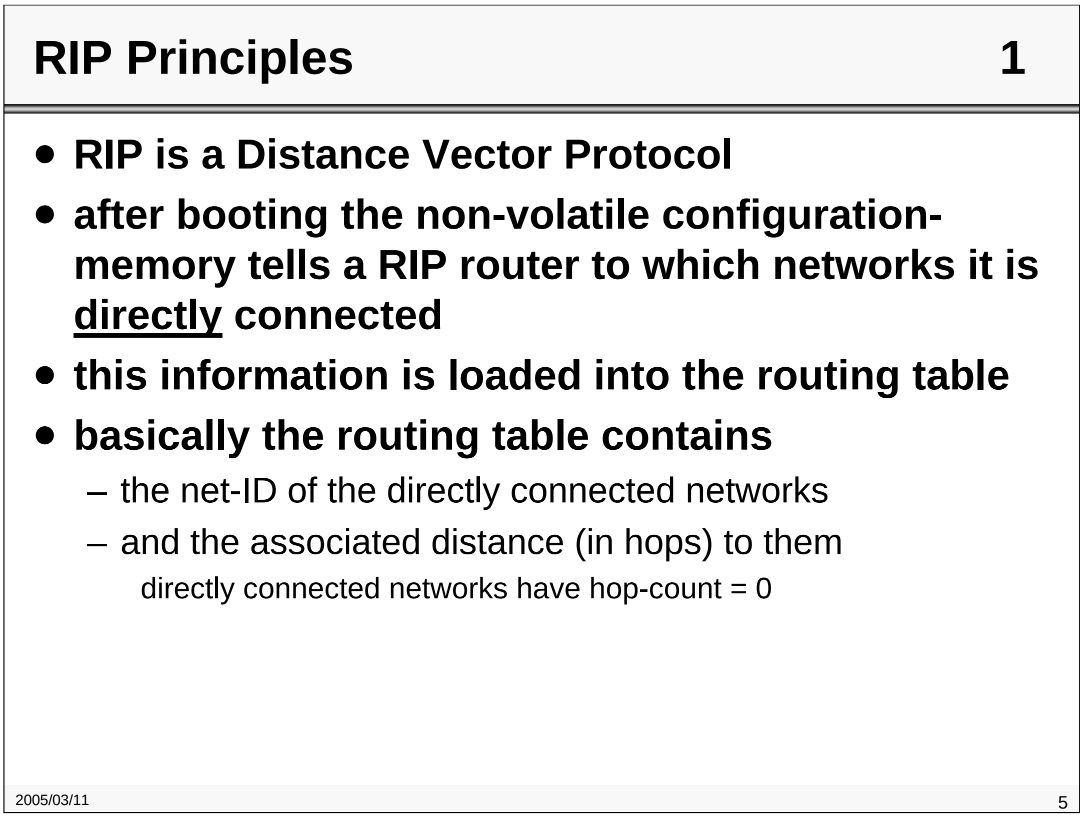
- Good news propagate quickly
  - 30 seconds per network
- Bad news are ignored
  - Except when sent by routers from which these routes had been learned initially
  - But better news from ANY router will be preferred
- Unreachable messages propagate slowly
  - 180 seconds per network

Bad news (= network reachabilities with worse metric) are only accepted if this message has been sent by that router from which we previously learned about that route.

Since RIP should discover the best routes to each destination, any routing update is accepted that contains a better route than previously learned.

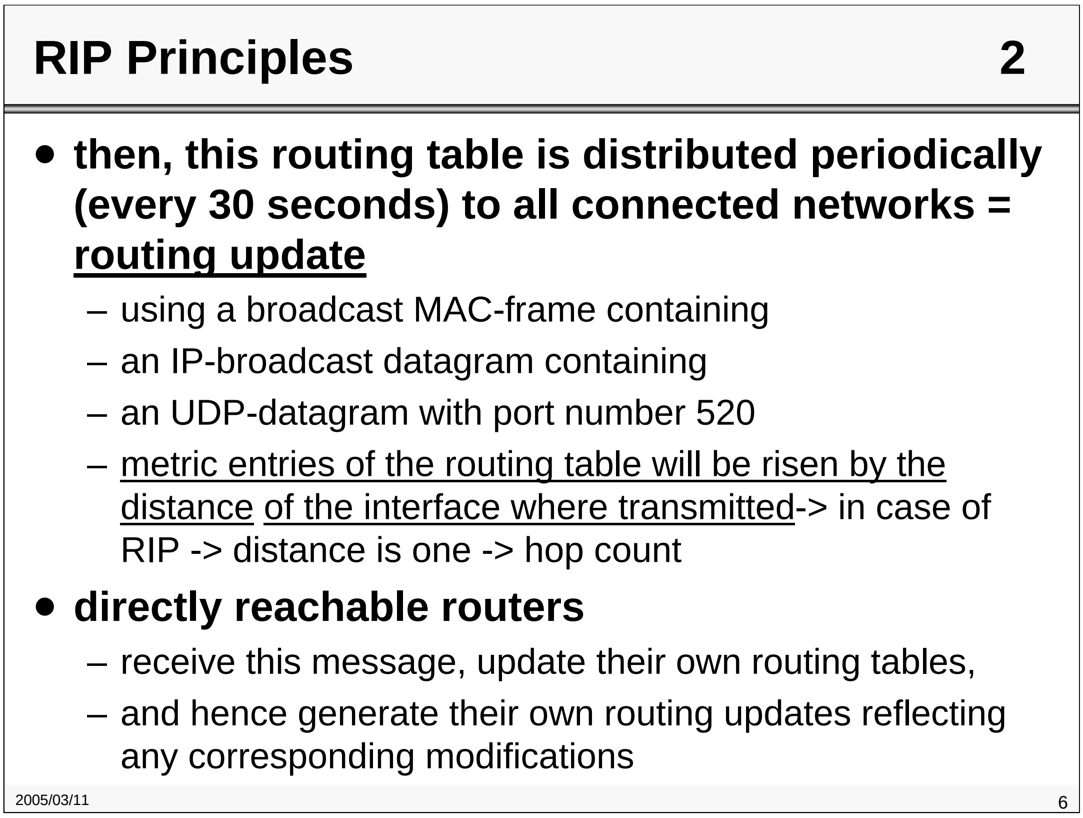
A route is declared unreachable without being refreshed by routing updates during 180 seconds.

In the worst case "bad news" propagate very slowly through the network. Special unreachable-messages have been introduced in order to improve the convergence time. Unreachable messages are normal routing updates but with metric set to "infinity".



RIP Principles 1

- **RIP is a Distance Vector Protocol**
- **after booting the non-volatile configuration-memory tells a RIP router to which networks it is <u>directly</u> connected**
- **this information is loaded into the routing table**
- **basically the routing table contains**
  - the net-ID of the directly connected networks
  - and the associated distance (in hops) to them
    - directly connected networks have hop-count = 0



RIP Principles 2

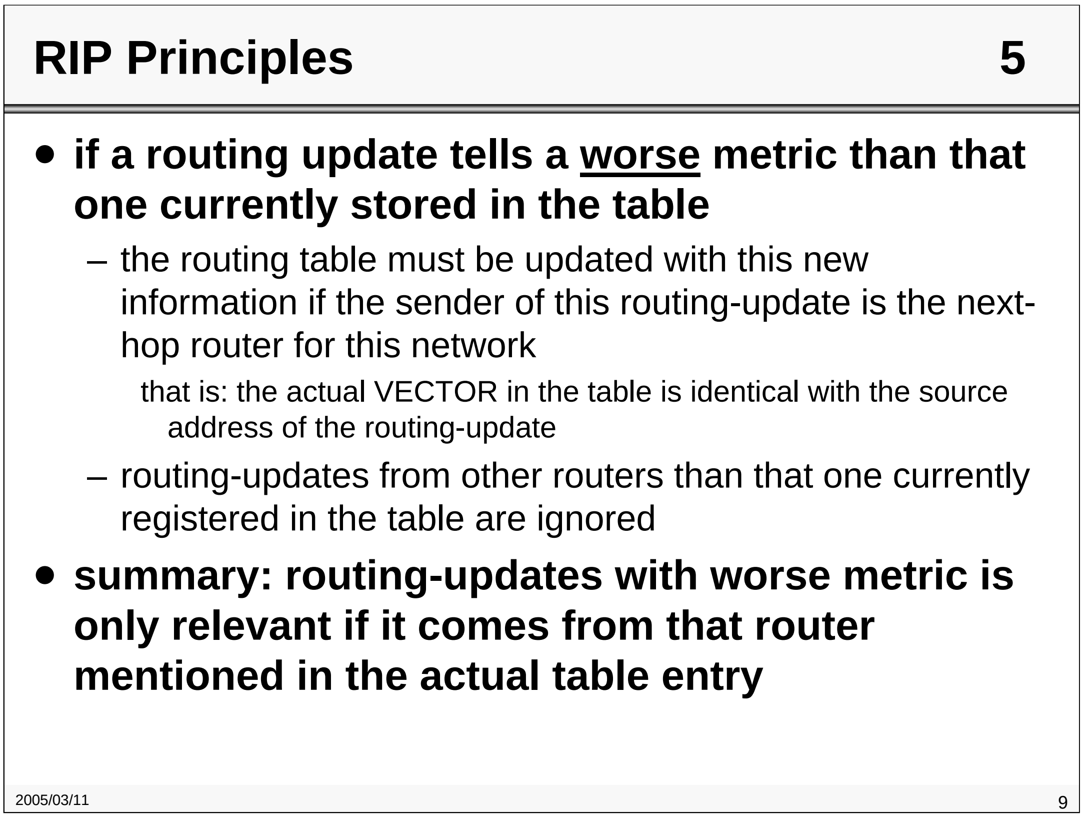
- **then, this routing table is distributed periodically (every 30 seconds) to all connected networks = routing update**
  - – using a broadcast MAC-frame containing
  - – an IP-broadcast datagram containing
  - – an UDP-datagram with port number 520
  - – metric entries of the routing table will be risen by the distance of the interface where transmitted-> in case of RIP -> distance is one -> hop count
- **directly reachable routers**
  - – receive this message, update their own routing tables,
  - – and hence generate their own routing updates reflecting any corresponding modifications

RIP Principles 3

- **after a specific time**
  - all routers know about all network addresses of the whole network
- **if different routing updates (from different routers) contain the same net-ID**
  - then there are redundant paths to this network
  - only the path with the lowest hop-count is stored in the routing-table
  - on receiving equal hop counts, the net-ID of the earlier one will be selected (and all other associated data)
- **hence, between each two networks exists exactly one active path**

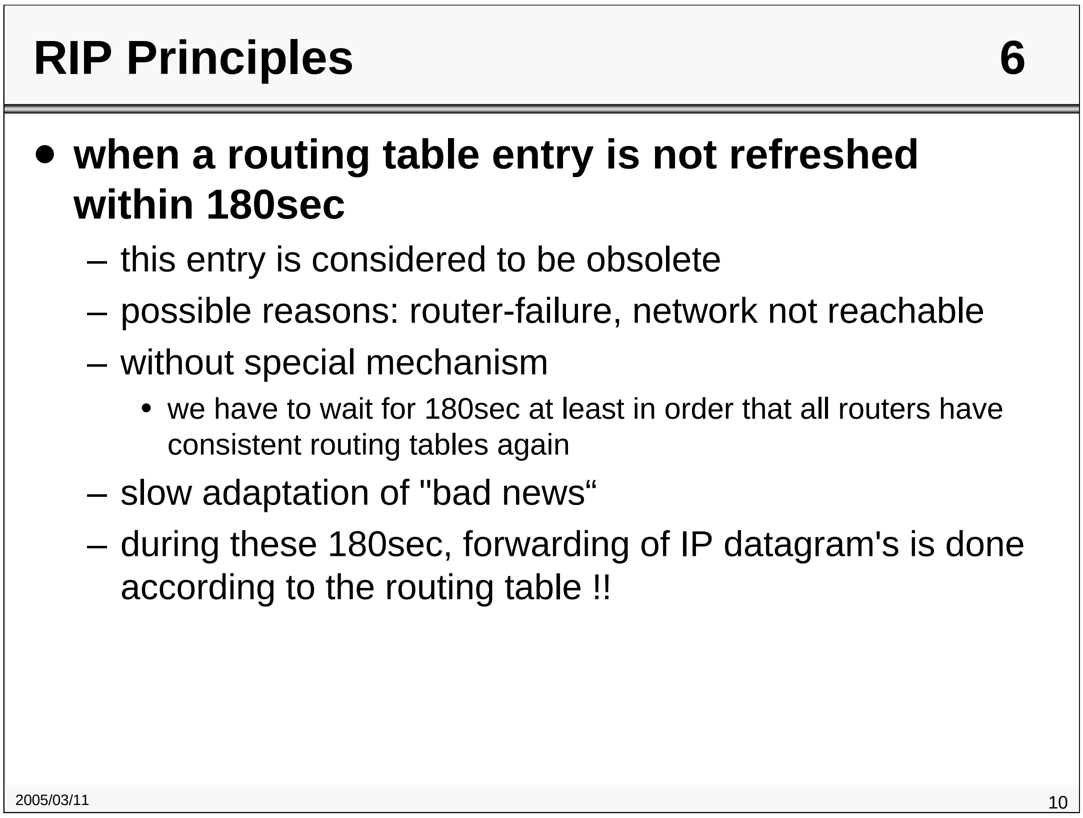
RIP Principles 4

- **all routing table‘s net-IDs are periodically refreshed by routing-update messages**
- **if a routing update tells a better metric than that one currently stored in the table**
  - the routing table must be updated with this new information
  - this update does not take care about if the sender of this routing-update is also the router which is currently selected as next hop
  - "good news“ are quickly adapted
  - RIP trusts good news from any source ("trusted news")



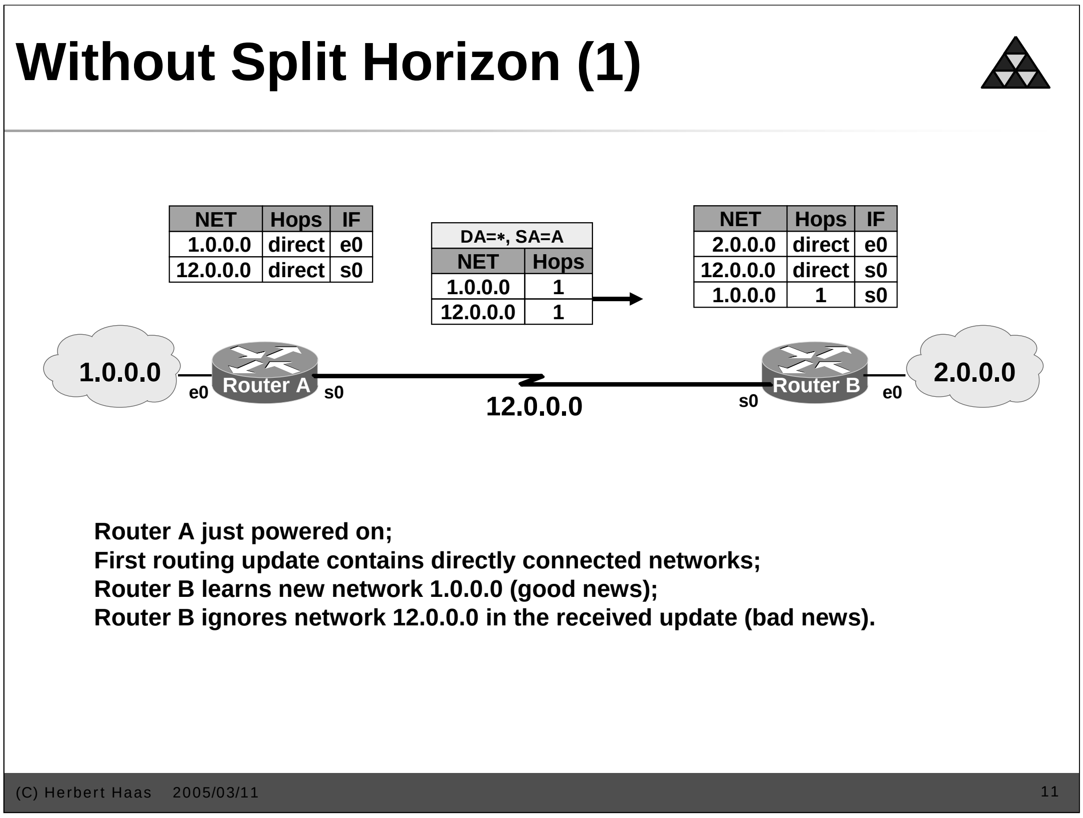
RIP Principles 5

- **if a routing update tells a <u>worse</u> metric than that one currently stored in the table**
  - the routing table must be updated with this new information if the sender of this routing-update is the next-hop router for this network
    - that is: the actual VECTOR in the table is identical with the source address of the routing-update
  - routing-updates from other routers than that one currently registered in the table are ignored
- **summary: routing-updates with worse metric is only relevant if it comes from that router mentioned in the actual table entry**



RIP Principles 6

- **when a routing table entry is not refreshed within 180sec**
  - – this entry is considered to be obsolete
  - – possible reasons: router-failure, network not reachable
  - – without special mechanism
    - we have to wait for 180sec at least in order that all routers have consistent routing tables again
  - – slow adaptation of "bad news“
  - – during these 180sec, forwarding of IP datagram's is done according to the routing table !!



Without Split Horizon (1)

NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0

DA=∗, SA=A

NET	Hops
1.0.0.0	1
12.0.0.0	1

NET	Hops	IF
2.0.0.0	direct	e0
12.0.0.0	direct	s0
1.0.0.0	1	s0

1.0.0.0 e0 Router A s0

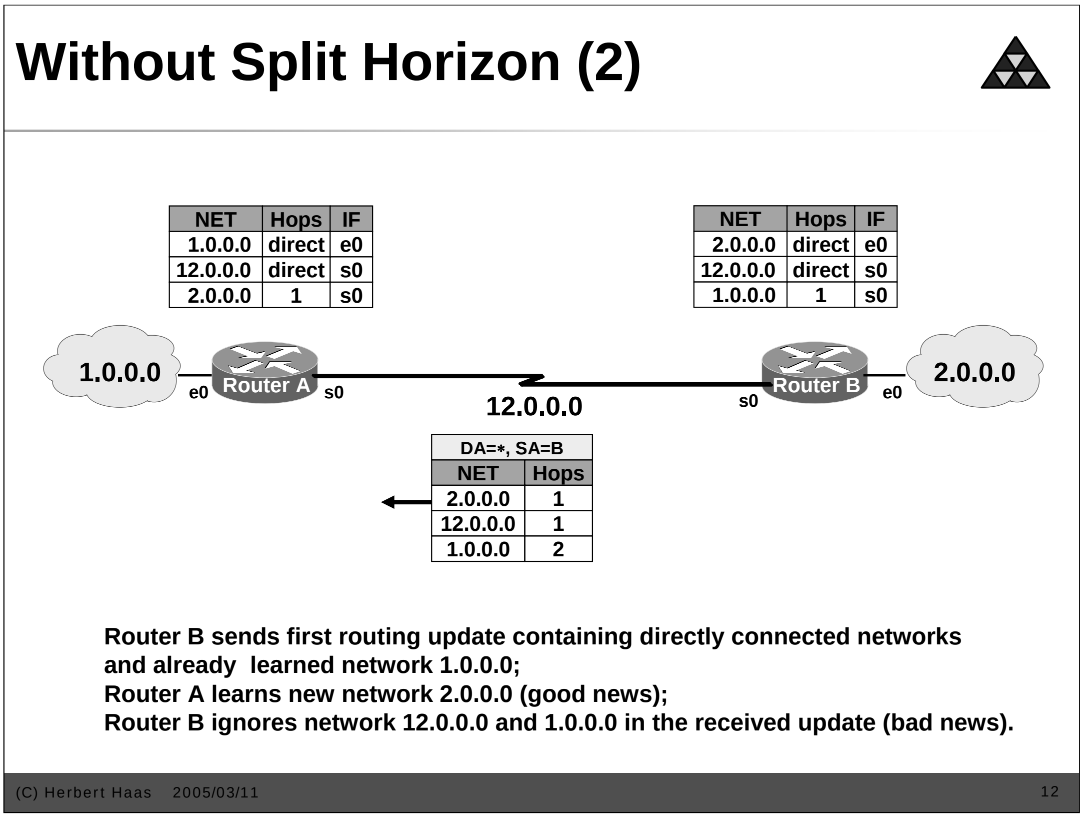
12.0.0.0

s0 Router B e0 2.0.0.0

Router A just powered on;
First routing update contains directly connected networks;
Router B learns new network 1.0.0.0 (good news);
Router B ignores network 12.0.0.0 in the received update (bad news).

This is the basic principal of RIP (Without Split Horizon). Every 30 seconds a router sends his whole routing table to every neighbor router and increases the Hop-Count by 1.

The router who receives this data add the new information in his routing table. If a router already knows about a better path – for example a direct connection to a net -- he will ignore this information.



Without Split Horizon (2)

NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
2.0.0.0	1	s0

NET	Hops	IF
2.0.0.0	direct	e0
12.0.0.0	direct	s0
1.0.0.0	1	s0

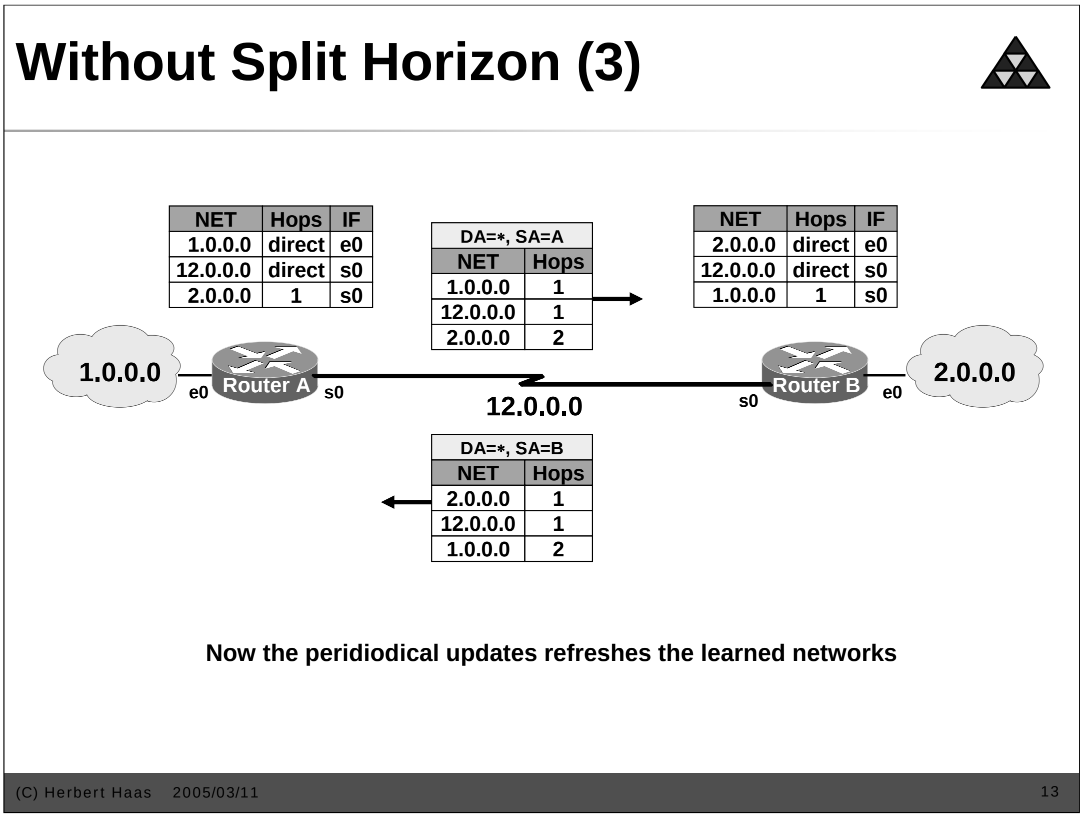
DA=*, SA=B

NET	Hops
2.0.0.0	1
12.0.0.0	1
1.0.0.0	2

Router B sends first routing update containing directly connected networks and already learned network 1.0.0.0;
Router A learns new network 2.0.0.0 (good news);
Router B ignores network 12.0.0.0 and 1.0.0.0 in the received update (bad news).

This is the basic principal of RIP (Without Split Horizon). Every 30 seconds a router sends his whole routing table to every neighbor router and increases the Hop-Count by 1.

The router who receives this data add the new information in his routing table. If a router already knows about a better path – for example a direct connection to a net -- he will ignore this information.



Without Split Horizon (3)

NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
2.0.0.0	1	s0

DA=∗, SA=A

NET	Hops
1.0.0.0	1
12.0.0.0	1
2.0.0.0	2

NET	Hops	IF
2.0.0.0	direct	e0
12.0.0.0	direct	s0
1.0.0.0	1	s0

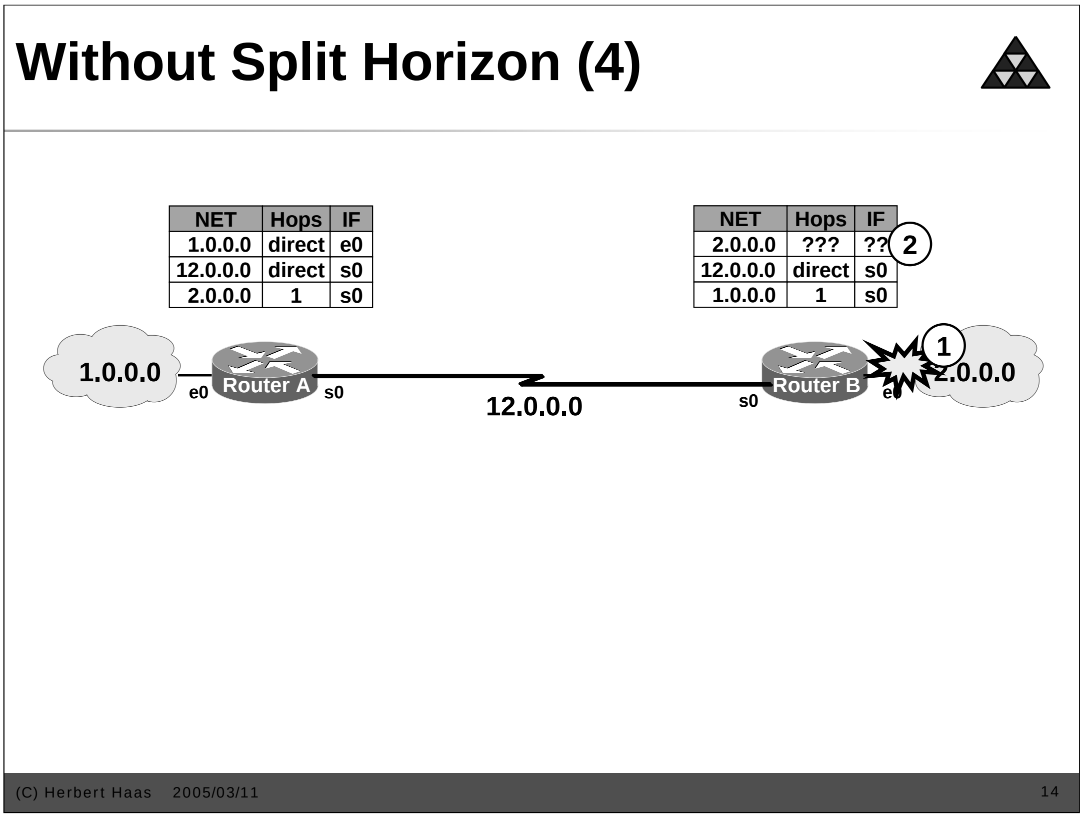
DA=∗, SA=B

NET	Hops
2.0.0.0	1
12.0.0.0	1
1.0.0.0	2

Now the peridiodical updates refreshes the learned networks

This is the basic principal of RIP (Without Split Horizon). Every 30 seconds a router sends his whole routing table to every neighbor router and increases the Hop-Count by 1.

The router who receives this data add the new information in his routing table. If a router already knows about a better path – for example a direct connection to a net -- he will ignore this information.

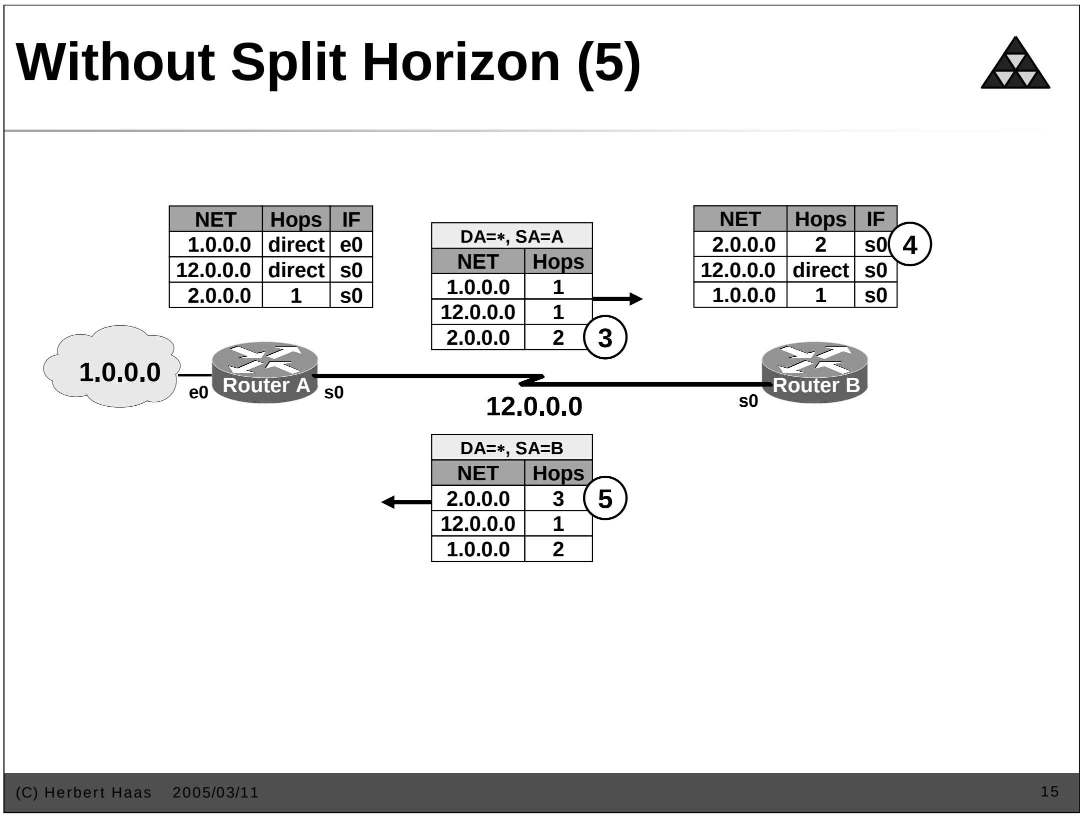


Without Split Horizon (4)

NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
2.0.0.0	1	s0

NET	Hops	IF
2.0.0.0	???	??
12.0.0.0	direct	s0
1.0.0.0	1	s0

In this example we see what would happen if network 2 crashes. Immediately, router B has no more information about this net. What would happen if router A sends a routing update now?



Without Split Horizon (5)

NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
2.0.0.0	1	s0

DA=∗, SA=A

NET	Hops
1.0.0.0	1
12.0.0.0	1
2.0.0.0	2

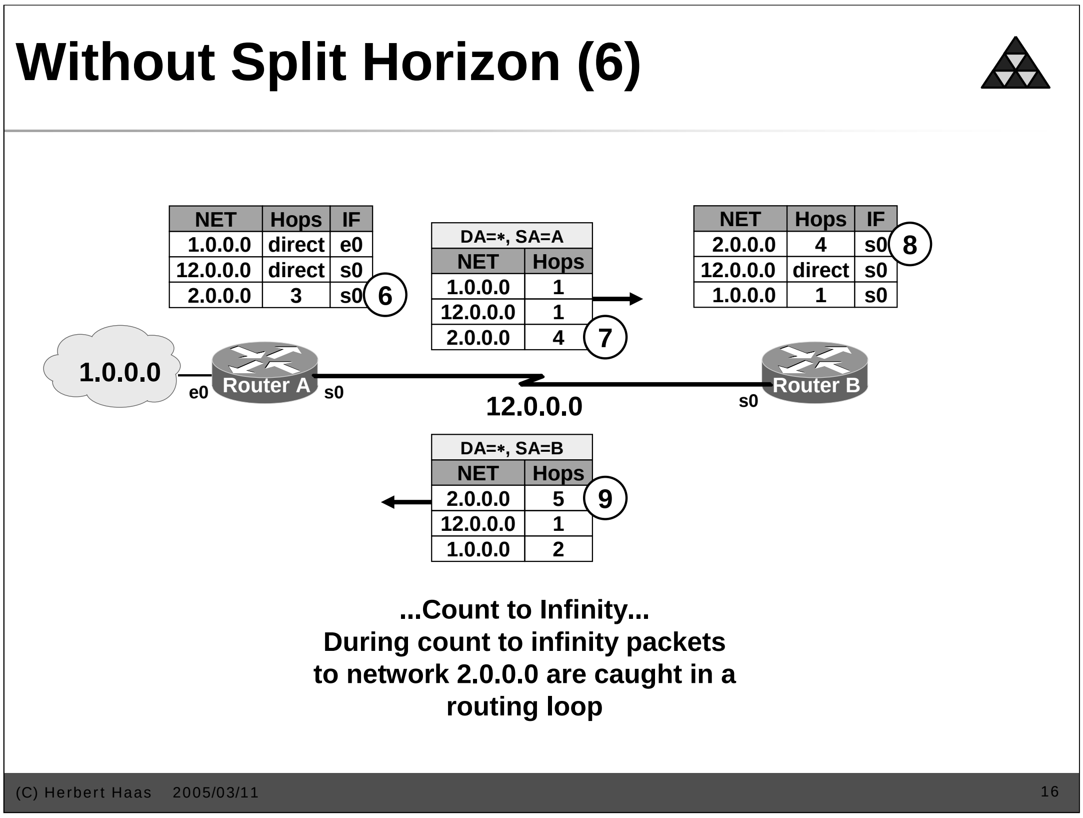
NET	Hops	IF
2.0.0.0	2	s0
12.0.0.0	direct	s0
1.0.0.0	1	s0

DA=∗, SA=B

NET	Hops
2.0.0.0	3
12.0.0.0	1
1.0.0.0	2

(C) Herbert Haas 2005/03/11 15

Now router B receives a routing update from router A including reach ability information about network 2. Because router B has no information about network 2 he adds this information in his routing table and continuous sending his normal routing updates to router A, hereby increasing the hop count by 1.



Without Split Horizon (6)

NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
2.0.0.0	3	s0

6

DA=*, SA=A	
NET	Hops
1.0.0.0	1
12.0.0.0	1
2.0.0.0	4

7

NET	Hops	IF
2.0.0.0	4	s0
12.0.0.0	direct	s0
1.0.0.0	1	s0

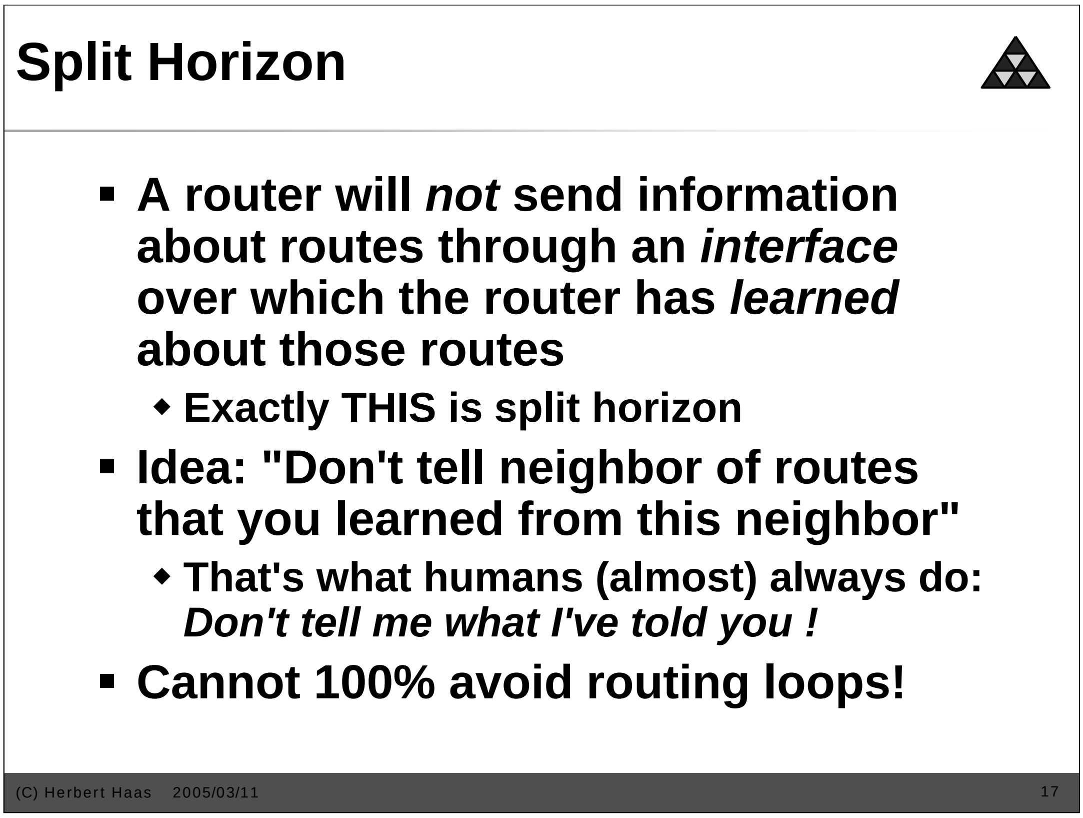
8

DA=*, SA=B	
NET	Hops
2.0.0.0	5
12.0.0.0	1
1.0.0.0	2

9

...Count to Infinity...
During count to infinity packets to network 2.0.0.0 are caught in a routing loop

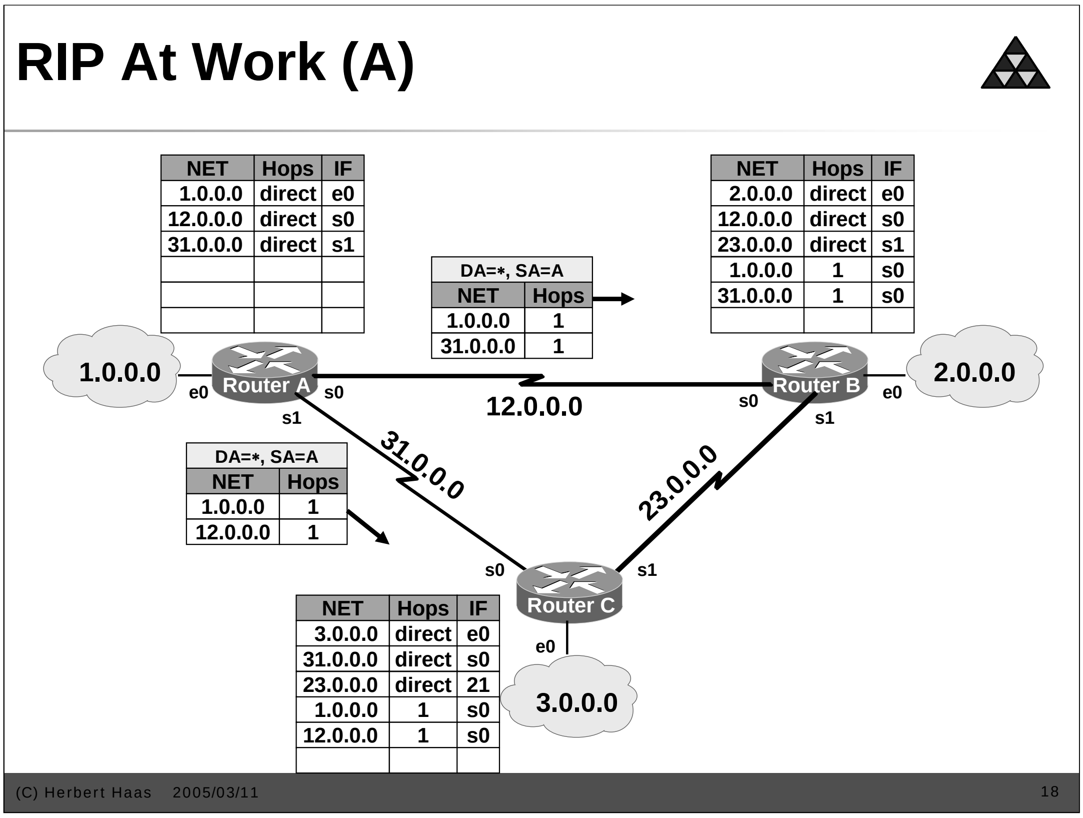
Either router A or router B has information about the Network 2, both router will increase the hop count by 1 every routing update. Count to infinity accurse. Now Update packets are caught in a routing loop.



Split Horizon

- **A router will *not* send information about routes through an *interface* over which the router has *learned* about those routes**
  - **Exactly THIS is split horizon**
- **Idea: "Don't tell neighbor of routes that you learned from this neighbor"**
  - **That's what humans (almost) always do: *Don't tell me what I've told you !***
- **Cannot 100% avoid routing loops!**

Nowadays all routers work with Split Horizon, there is now RIP-Network without it. The principle of Spilt Horizon is simple: “Don’t tell neighbor of routes that you learned from him”.



RIP At Work (A)

NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
31.0.0.0	direct	s1

DA=∗, SA=A

NET	Hops
1.0.0.0	1
31.0.0.0	1

NET	Hops	IF
2.0.0.0	direct	e0
12.0.0.0	direct	s0
23.0.0.0	direct	s1
1.0.0.0	1	s0
31.0.0.0	1	s0

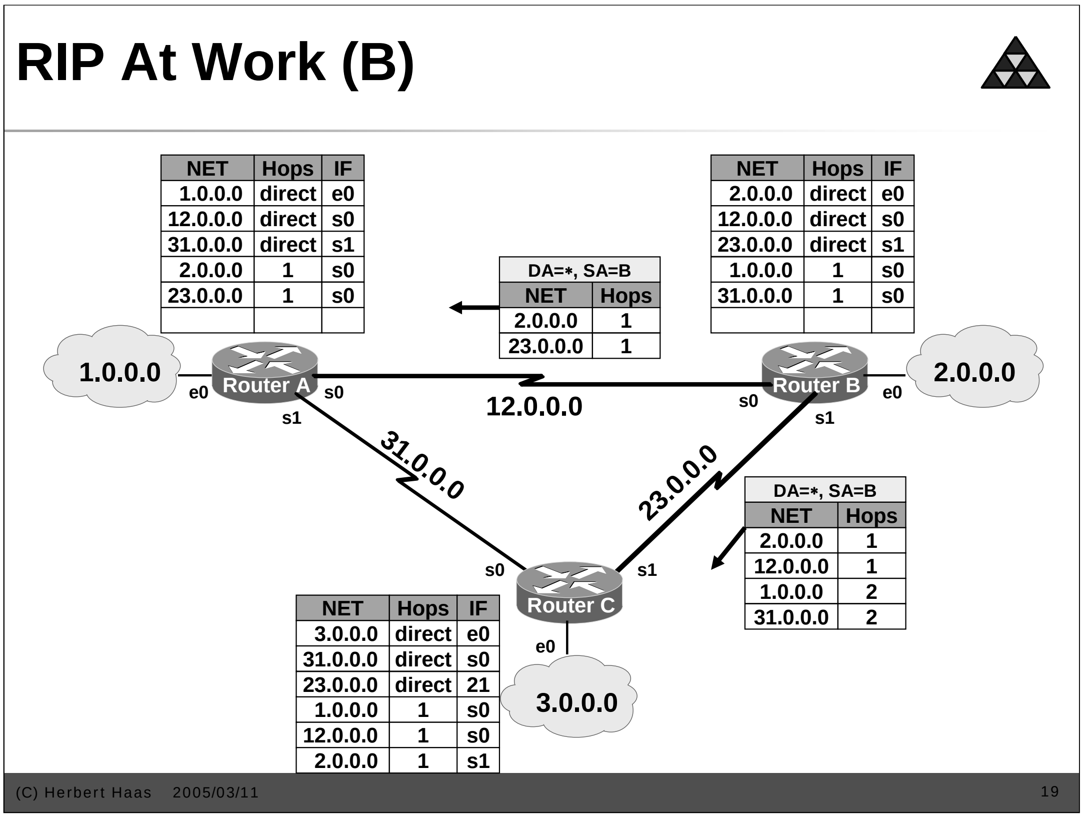
DA=∗, SA=A

NET	Hops
1.0.0.0	1
12.0.0.0	1

NET	Hops	IF
3.0.0.0	direct	e0
31.0.0.0	direct	s0
23.0.0.0	direct	21
1.0.0.0	1	s0
12.0.0.0	1	s0

(C) Herbert Haas 2005/03/11 18

Split Horizon at work: Router A didn’t tell router B about the network 12 and router A didn’t tell router C about the network 31, because the router knows that router B must have a direct connection to network 12 and that router C must have a direct connection to network 31.



RIP At Work (B)

NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
31.0.0.0	direct	s1
2.0.0.0	1	s0
23.0.0.0	1	s0

NET	Hops	IF
2.0.0.0	direct	e0
12.0.0.0	direct	s0
23.0.0.0	direct	s1
1.0.0.0	1	s0
31.0.0.0	1	s0

DA=*, SA=B

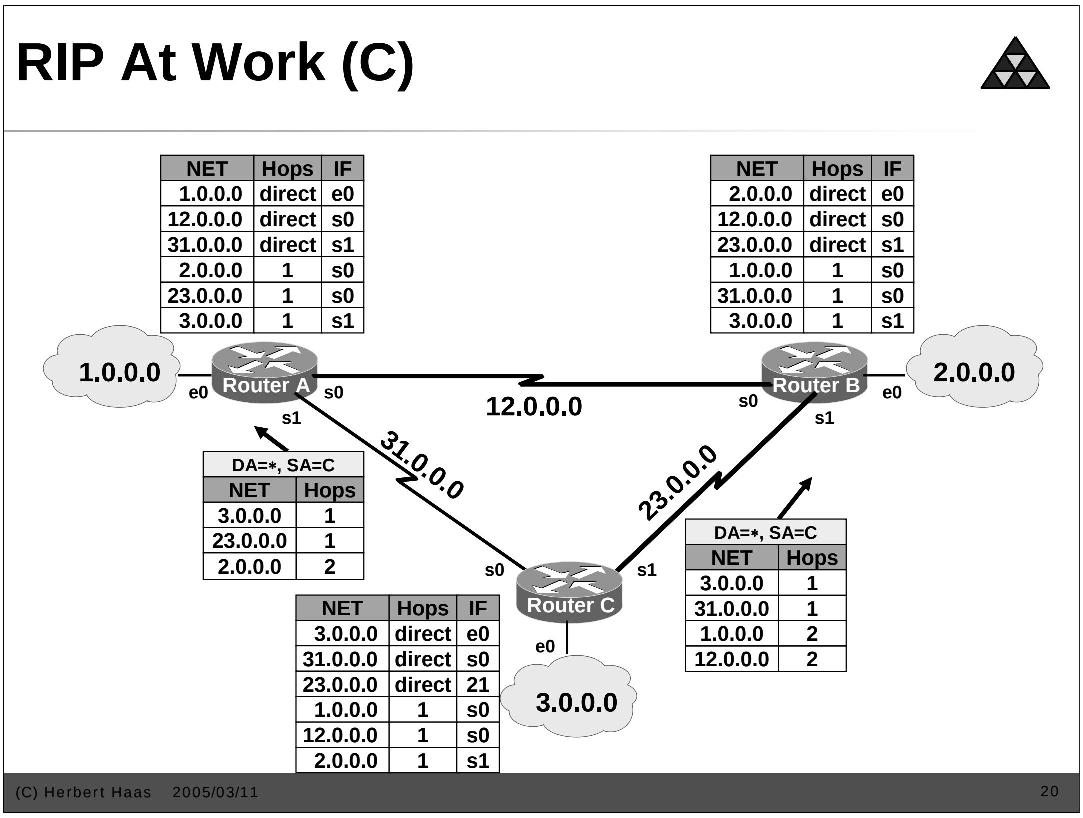
NET	Hops
2.0.0.0	1
23.0.0.0	1

DA=*, SA=B

NET	Hops
2.0.0.0	1
12.0.0.0	1
1.0.0.0	2
31.0.0.0	2

NET	Hops	IF
3.0.0.0	direct	e0
31.0.0.0	direct	s0
23.0.0.0	direct	21
1.0.0.0	1	s0
12.0.0.0	1	s0
2.0.0.0	1	s1

And so router B tells router A only about network 2 and 23 and router C only about network 2 and 12.



RIP At Work (C)

NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
31.0.0.0	direct	s1
2.0.0.0	1	s0
23.0.0.0	1	s0
3.0.0.0	1	s1

NET	Hops	IF
2.0.0.0	direct	e0
12.0.0.0	direct	s0
23.0.0.0	direct	s1
1.0.0.0	1	s0
31.0.0.0	1	s0
3.0.0.0	1	s1

DA=*, SA=C

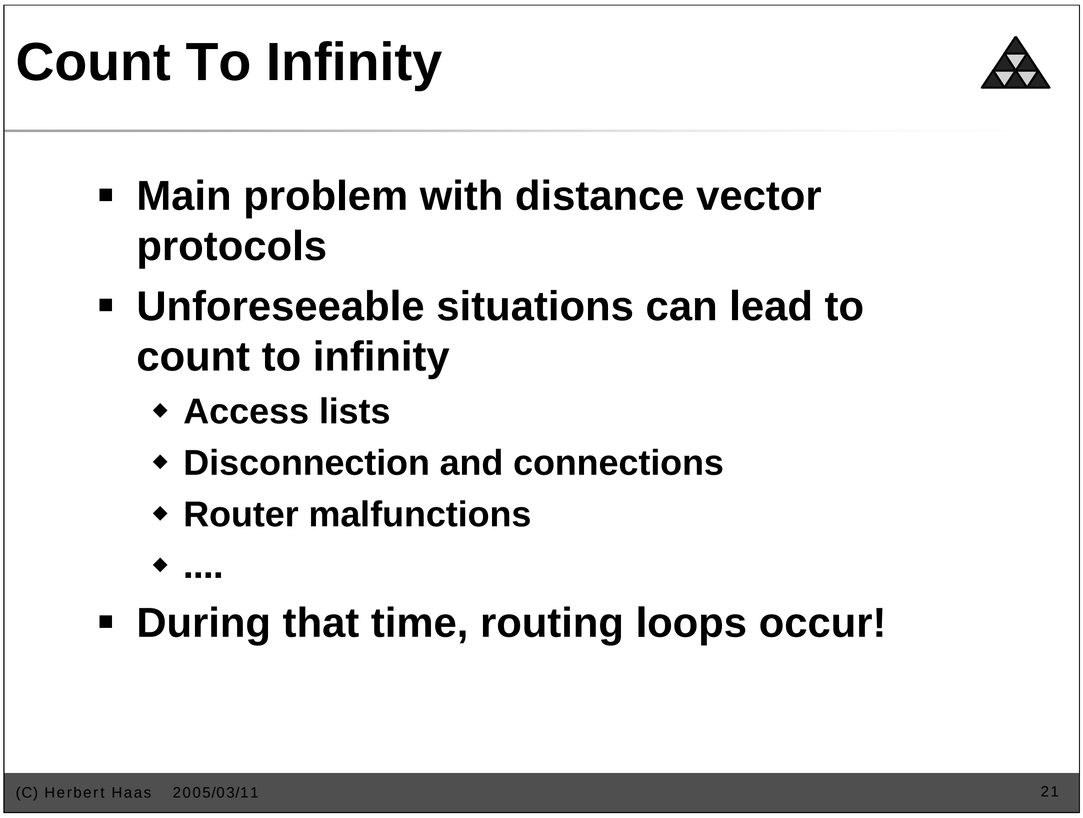
NET	Hops
3.0.0.0	1
23.0.0.0	1
2.0.0.0	2

DA=*, SA=C

NET	Hops
3.0.0.0	1
31.0.0.0	1
1.0.0.0	2
12.0.0.0	2

NET	Hops	IF
3.0.0.0	direct	e0
31.0.0.0	direct	s0
23.0.0.0	direct	21
1.0.0.0	1	s0
12.0.0.0	1	s0
2.0.0.0	1	s1

Router C do the same. At the end every router knows the route to every network.

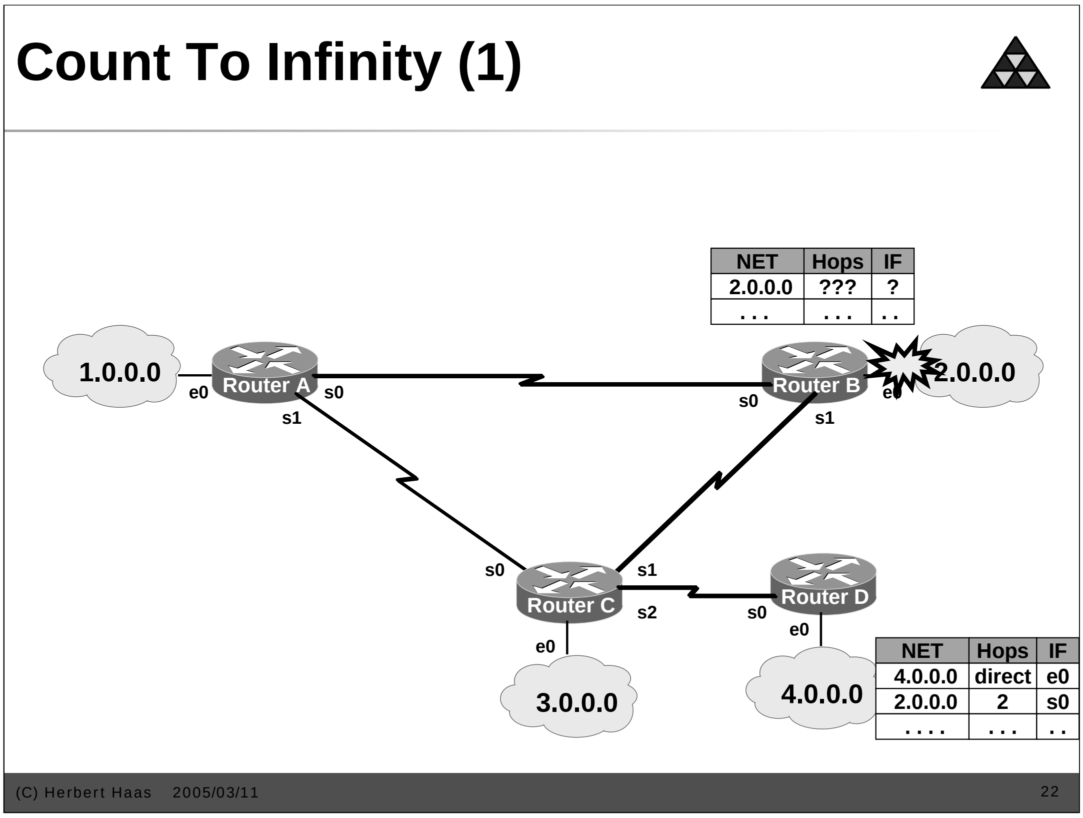


Count To Infinity

- Main problem with distance vector protocols
- Unforeseeable situations can lead to count to infinity
  - Access lists
  - Disconnection and connections
  - Router malfunctions
  - ....
- During that time, routing loops occur!

Because of the simple principle of RIP (Distance Vector protocol), we cannot prevent Count to Infinity. Access Lists, Disconnection and connections, Router malfunction, etc can always lead to it, there is no 100% solution.

We need a more general approach to avoid that → **Maximum Hop Count,** that's the only failsafe solution.



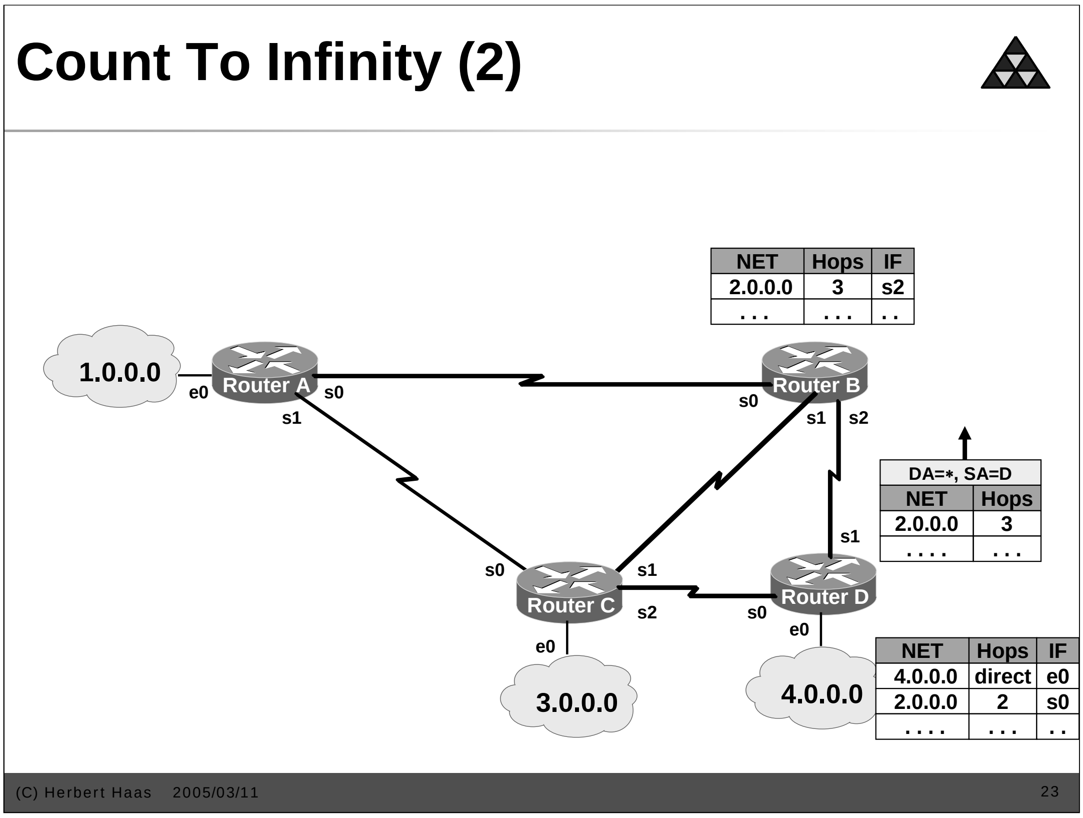
Count To Infinity (1)

NET	Hops	IF
2.0.0.0	???	?
. . .	. . .	. .

NET	Hops	IF
4.0.0.0	direct	e0
2.0.0.0	2	s0
. . . .	. . .	. .

Lets us look to another example where Count to Infinity is approaching. Although Split Horizon is implemented !

We have a network with 4 routers, suddenly net 2 crash.



Count To Infinity (2)

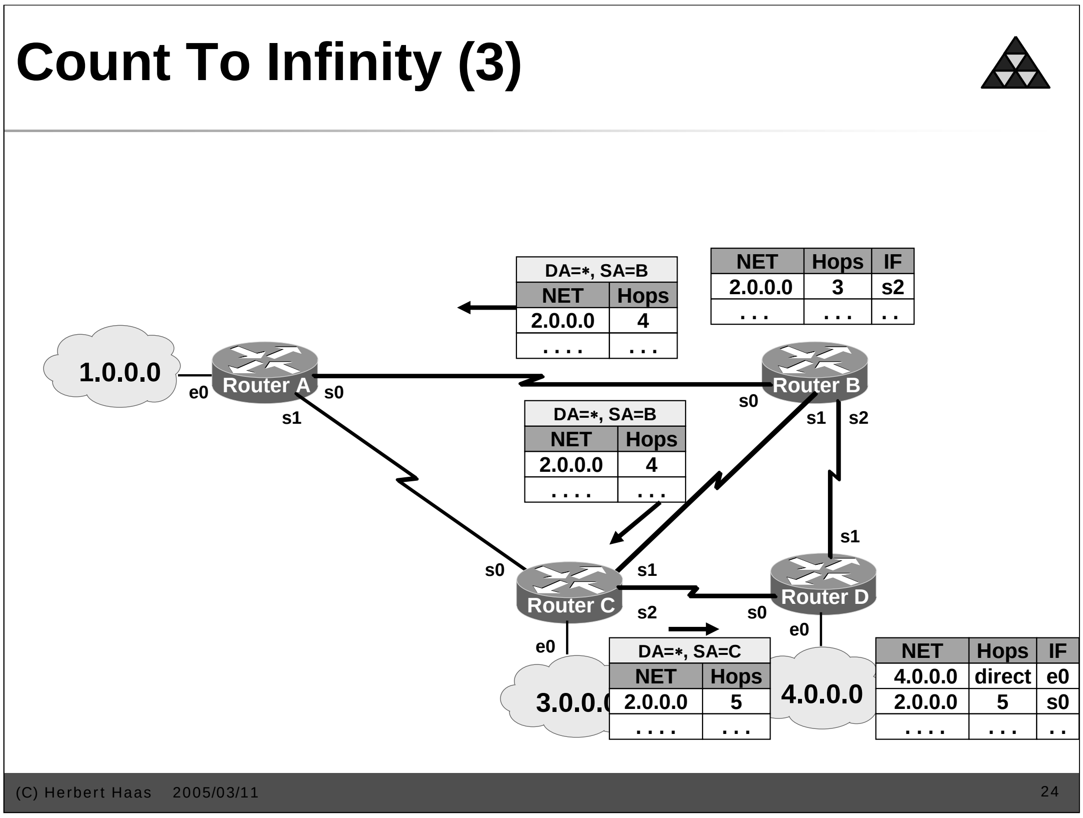
NET	Hops	IF
2.0.0.0	3	s2
. . .	. . .	. .

DA=*, SA=D

NET	Hops
2.0.0.0	3
. . . .	. . .

NET	Hops	IF
4.0.0.0	direct	e0
2.0.0.0	2	s0
. . . .	. . .	. .

And a new connection established between router B and router D. Now, a normal routing update is send from router D to router B (with information about net 2, of course).



Count To Infinity (3)

NET	Hops	IF
2.0.0.0	3	s2
. . .	. . .	. .

DA=∗, SA=B

NET	Hops
2.0.0.0	4
. . . .	. . .

DA=∗, SA=B

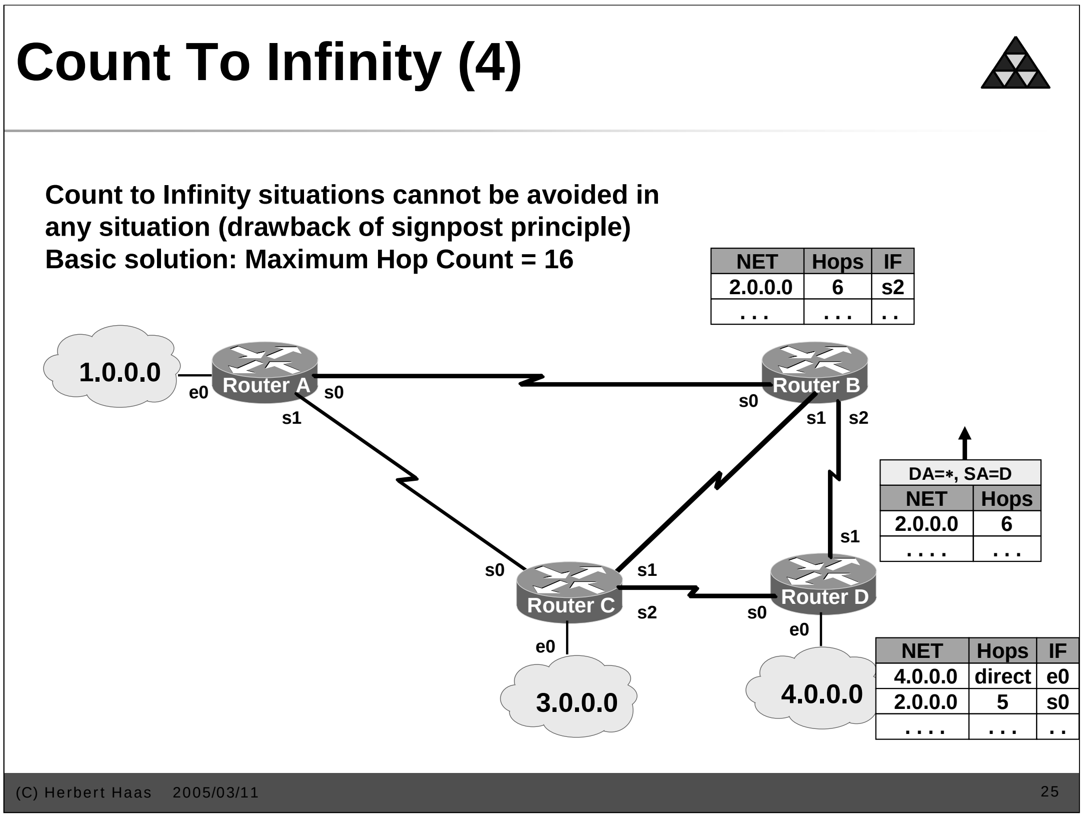
NET	Hops
2.0.0.0	4
. . . .	. . .

DA=∗, SA=C

NET	Hops
2.0.0.0	5
. . . .	. . .

NET	Hops	IF
4.0.0.0	direct	e0
2.0.0.0	5	s0
. . . .	. . .	. .

Router B doesn’t know where network 2 is gone. So he sends information about network 2 (increasing hop count by 1) to every neighbor router.



Count To Infinity (4)

Count to Infinity situations cannot be avoided in any situation (drawback of signpost principle)

Basic solution: Maximum Hop Count = 16

NET	Hops	IF
2.0.0.0	6	s2
. . .	. . .	. .

DA=*, SA=D

NET	Hops
2.0.0.0	6
. . . .	. . .

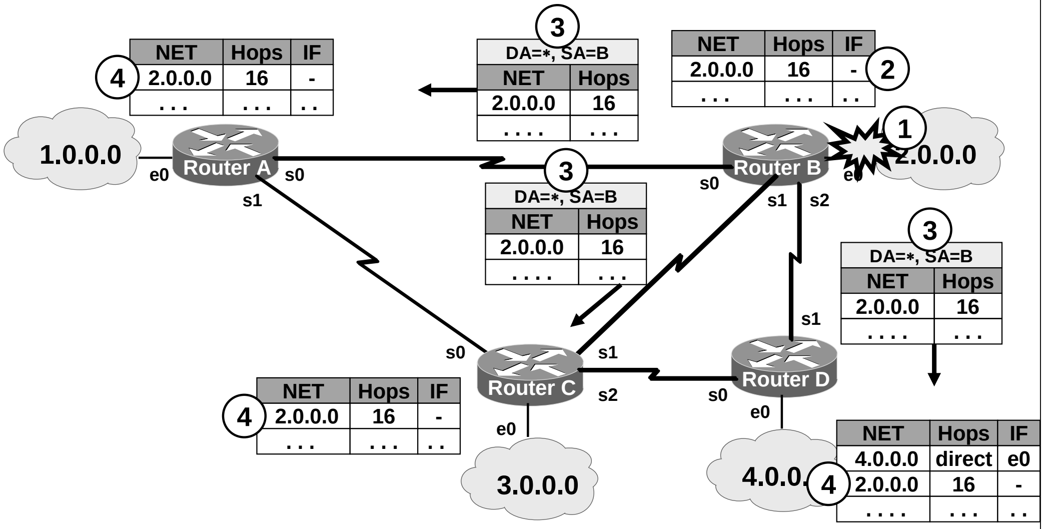
NET	Hops	IF
4.0.0.0	direct	e0
2.0.0.0	5	s0
. . . .	. . .	. .

(C) Herbert Haas 2005/03/11 25

Count to infinity accurse. Only the maximum Hop Count, the basic solution, can stop this problem.

Maximum Hop Count = 16

Upon network failure, the route is marked as INVALID (hop count 16) and propagated.



NET	Hops	IF
2.0.0.0	16	-
. . .	. . .	. .

DA=*, SA=B	
NET	Hops
2.0.0.0	16
. . . .	. . .

NET	Hops	IF
2.0.0.0	16	-
. . .	. . .	. .

DA=*, SA=B	
NET	Hops
2.0.0.0	16
. . . .	. . .

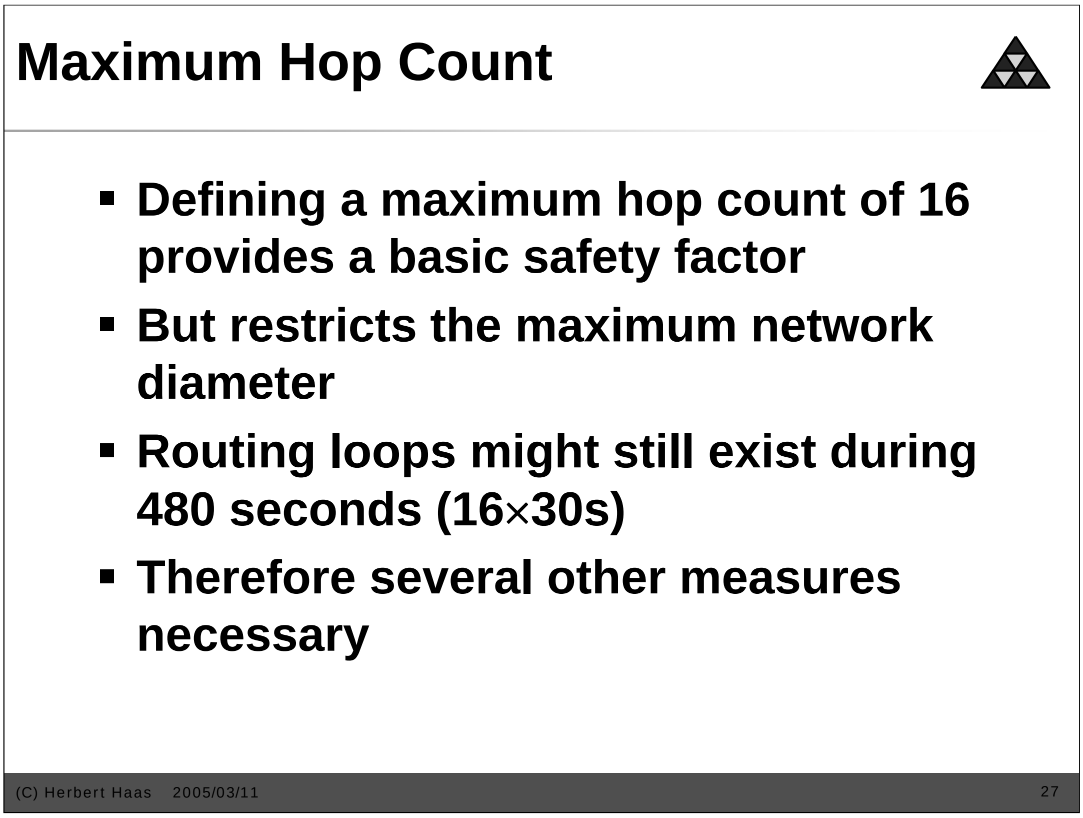
DA=*, SA=B	
NET	Hops
2.0.0.0	16
. . . .	. . .

NET	Hops	IF
2.0.0.0	16	-
. . .	. . .	. .

NET	Hops	IF
4.0.0.0	direct	e0
2.0.0.0	16	-
. . . .	. . .	. .

After 16 Hops the Net 2 is now marked as **invalid**.

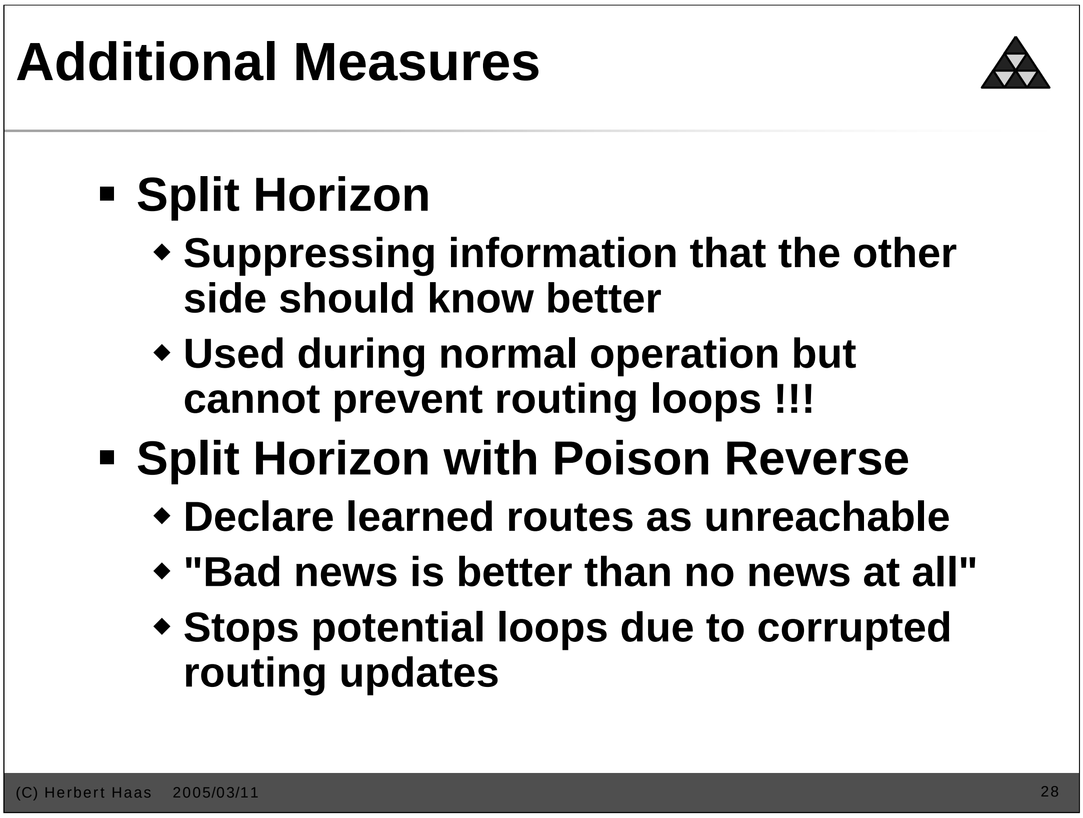
Of course, this unreachabilty-information would be propagated deeper into the network if there are additional routers.



Maximum Hop Count

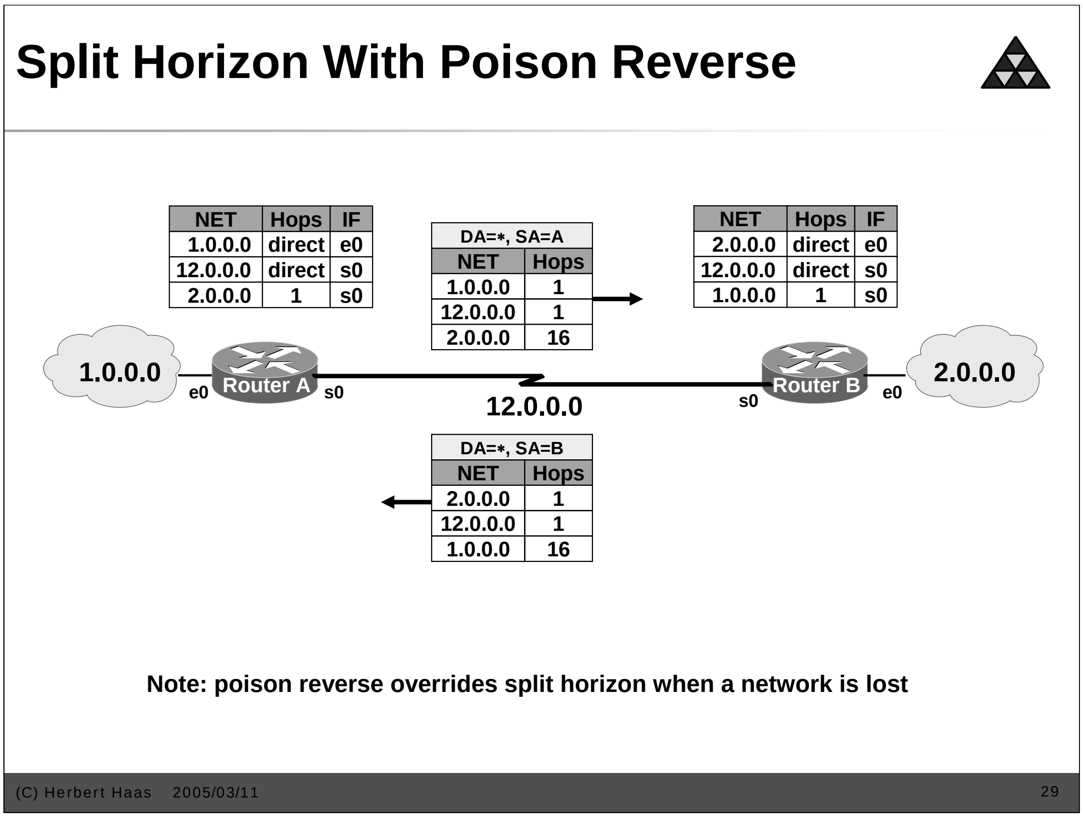
- Defining a maximum hop count of 16 provides a basic safety factor
- But restricts the maximum network diameter
- Routing loops might still exist during 480 seconds (16×30s)
- Therefore several other measures necessary

The maximum hop count is a basic safety factor, but it is also the main drawback of RIP. It restrict the maximum network diameter, and the rooting loops exist for 480 seconds. During Count to Infinity there is a bad routing and the network must deal with unnecessary traffic. So we need other measures like Poison Reverse.



Additional Measures

- Split Horizon
  - Suppressing information that the other side should know better
  - Used during normal operation but cannot prevent routing loops !!!
- Split Horizon with Poison Reverse
  - Declare learned routes as unreachable
  - "Bad news is better than no news at all"
  - Stops potential loops due to corrupted routing updates



Split Horizon With Poison Reverse

NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
2.0.0.0	1	s0

DA=∗, SA=A	
NET	Hops
1.0.0.0	1
12.0.0.0	1
2.0.0.0	16

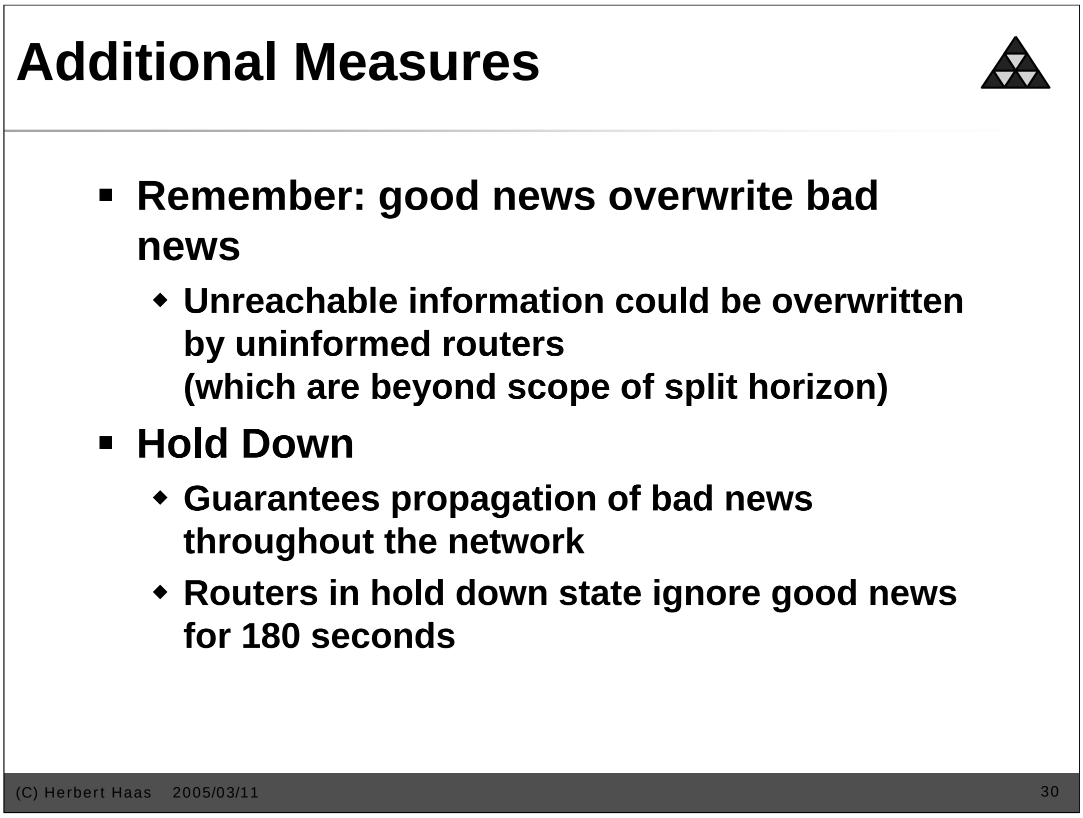
NET	Hops	IF
2.0.0.0	direct	e0
12.0.0.0	direct	s0
1.0.0.0	1	s0

DA=∗, SA=B	
NET	Hops
2.0.0.0	1
12.0.0.0	1
1.0.0.0	16

Note: poison reverse overrides split horizon when a network is lost

Split horizon with poisoned reverse includes also reverse routes in updates, but sets their metrics to infinity. This is safer than simple split horizon: If two gateways have routes pointing at each other, advertising reverse routes with a metric of 16 will break the loop immediately.

Note: Split Horizon with Poison Reverse is not used by Cisco Routers (however poison updates are indeed used when e. g. an interface goes down).

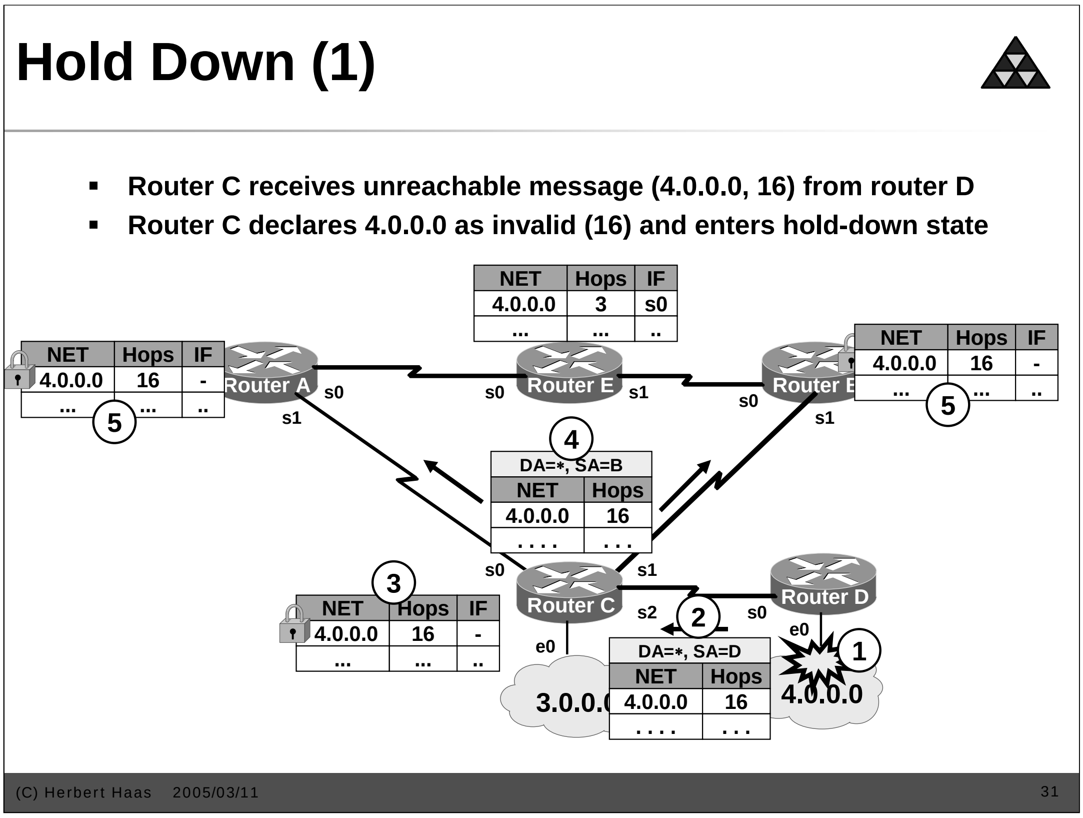


Additional Measures

- Remember: good news overwrite bad news
  - Unreachable information could be overwritten by uninformed routers
    (which are beyond scope of split horizon)
- Hold Down
  - Guarantees propagation of bad news throughout the network
  - Routers in hold down state ignore good news for 180 seconds

RIP needs long time to send bad news over the whole network (remember the 480 seconds). To guarantee that the bad news send throughout the network, the **hold down** measure is implemented. After a router receives “bad news” he will ignore all “good news” about the same route for 180 seconds.

Note: Hold-down timers are not explicitly required by RFC 1058. However most vendors (also Cisco) implemented it.



Hold Down (1)

- Router C receives unreachable message (4.0.0.0, 16) from router D
- Router C declares 4.0.0.0 as invalid (16) and enters hold-down state

NET	Hops	IF
4.0.0.0	3	s0
...	...	..

NET	Hops	IF
4.0.0.0	16	-
...	...	..

NET	Hops	IF
4.0.0.0	16	-
...	...	..

DA=∗, SA=B

NET	Hops
4.0.0.0	16
. . . .	. . .

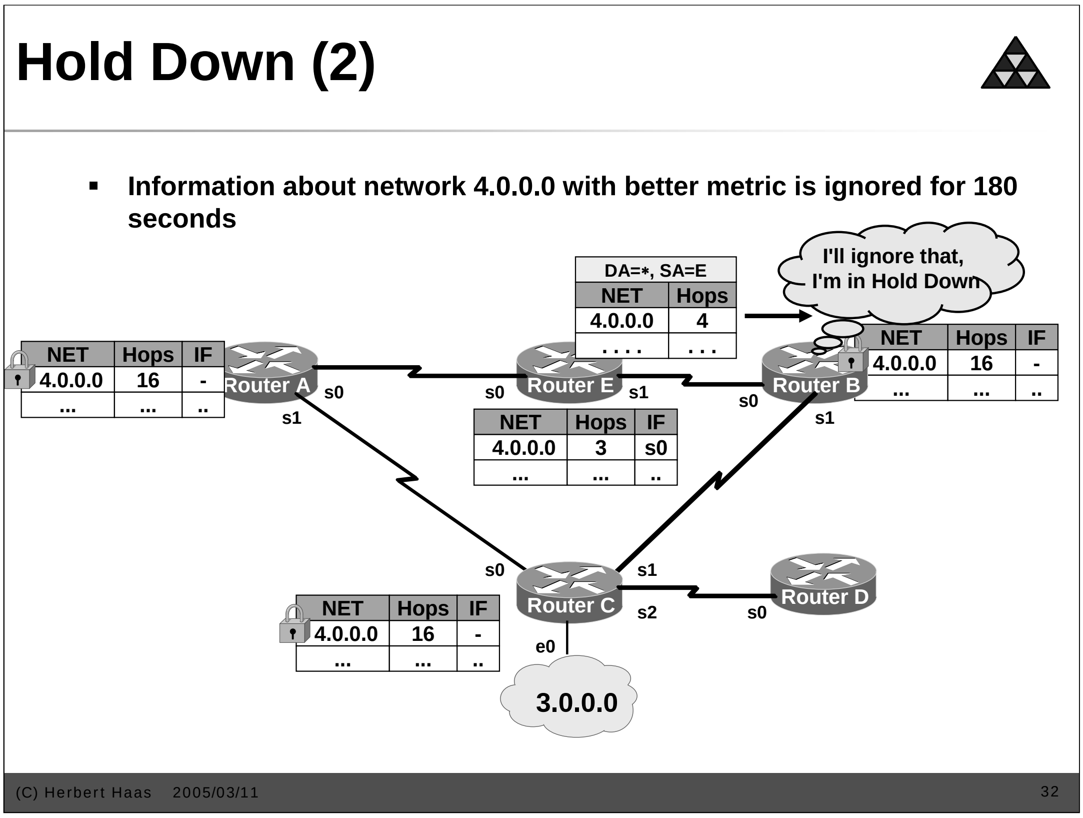
NET	Hops	IF
4.0.0.0	16	-
...	...	..

DA=∗, SA=D

NET	Hops
4.0.0.0	16
. . . .	. . .

(C) Herbert Haas 2005/03/11 31

In this example we see the functionary of Hold Down. After Net 4 crashes, router D send this information to Router C. Router C added this information and activate “hold down”. After this he sends this information to his neighbor routers, which do the same after they receive the information about net 4.



Hold Down (2)

- **Information about network 4.0.0.0 with better metric is ignored for 180 seconds**

DA=∗, SA=E

NET	Hops
4.0.0.0	4
. . . .	. . .

NET	Hops	IF
4.0.0.0	16	-
...	...	..

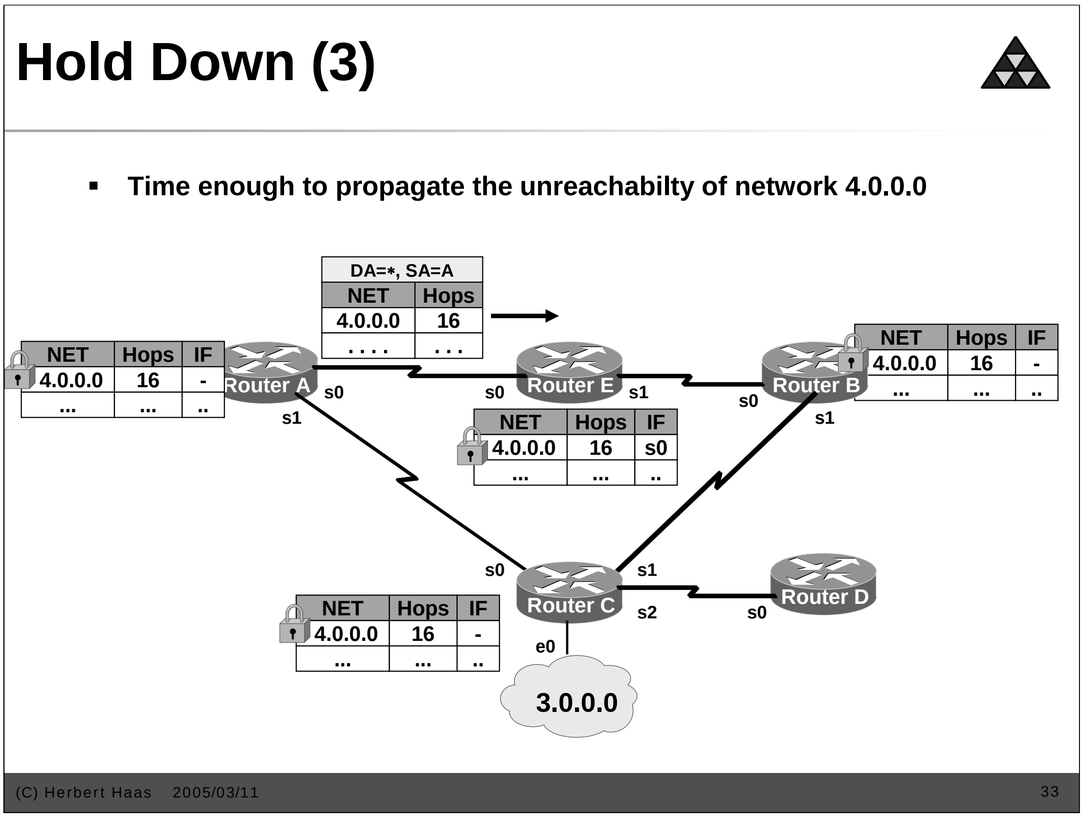
NET	Hops	IF
4.0.0.0	3	s0
...	...	..

NET	Hops	IF
4.0.0.0	16	-
...	...	..

NET	Hops	IF
4.0.0.0	16	-
...	...	..

(C) Herbert Haas 2005/03/11 32

Router E didn’t get information that net 4 crashes yet, so he normally sends his routing update. But the information’s from router E couldn’t overwrite routing informations of router B or router A. Because these router are in the “hold down” status, and ignore these update messages.



Hold Down (3)

- Time enough to propagate the unreachabilty of network 4.0.0.0

DA=∗, SA=A

NET	Hops
4.0.0.0	16
. . . .	. . .

Router A

NET	Hops	IF
4.0.0.0	16	-
...	...	..

Router E

NET	Hops	IF
4.0.0.0	16	s0
...	...	..

Router B

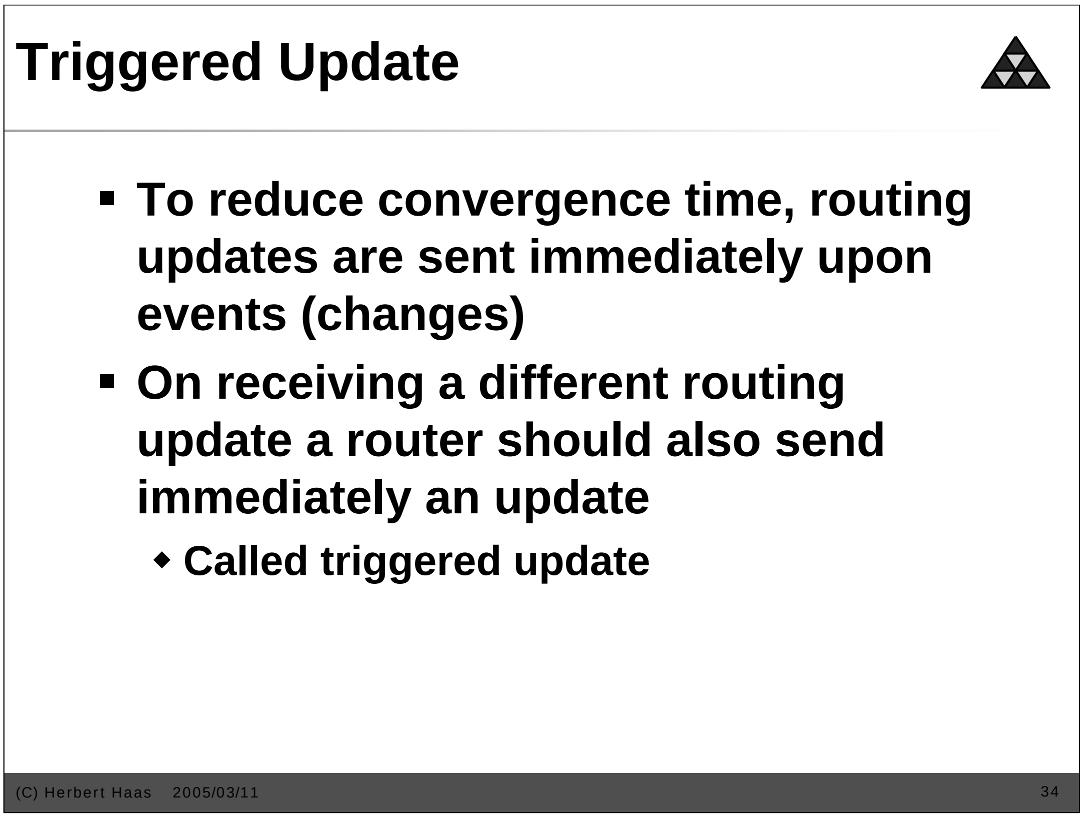
NET	Hops	IF
4.0.0.0	16	-
...	...	..

Router C

NET	Hops	IF
4.0.0.0	16	-
...	...	..

(C) Herbert Haas 2005/03/11 33

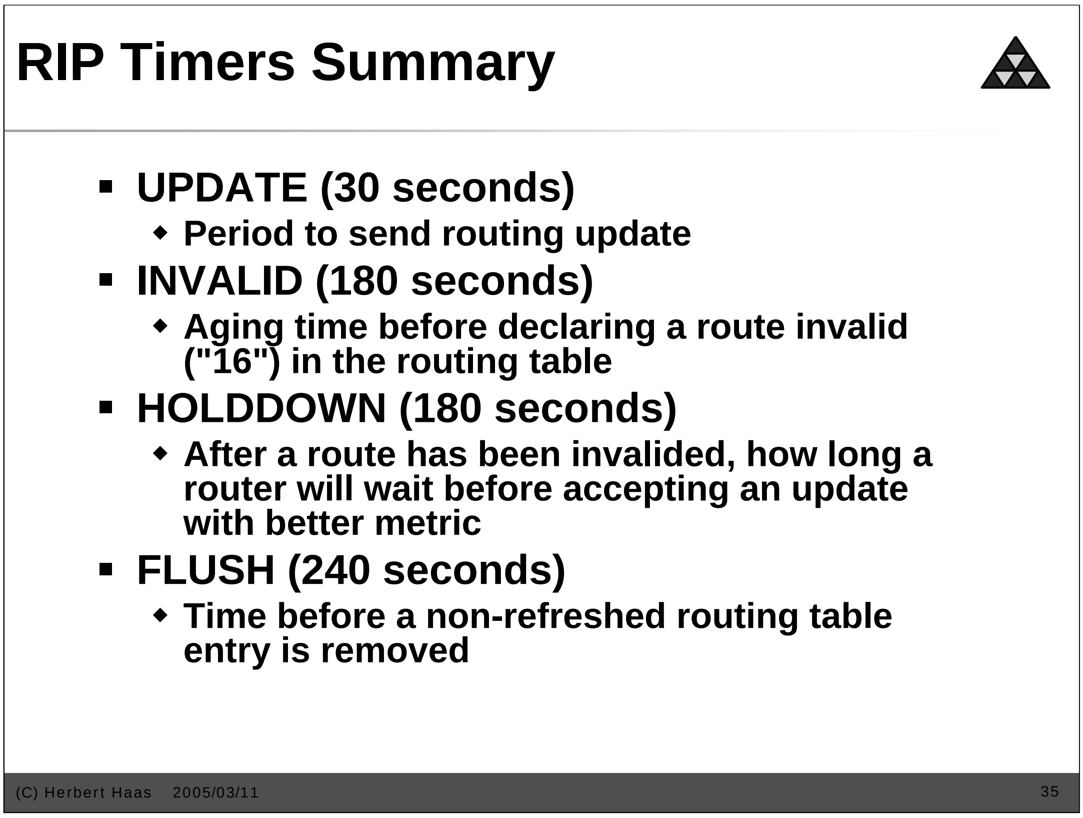
Soon every router knows that network 4 is unreachable.



Triggered Update

- To reduce convergence time, routing updates are sent immediately upon events (changes)
- On receiving a different routing update a router should also send immediately an update
  - Called triggered update

To speed up the convergence time, “**triggered update**” has been introduced. After a router notice a network failure, he immediately sends a routing update to indicate this failure. So the router didn’t wait for the expiration of the 30 seconds. Triggered update can used with all events (e.g. a new link established).

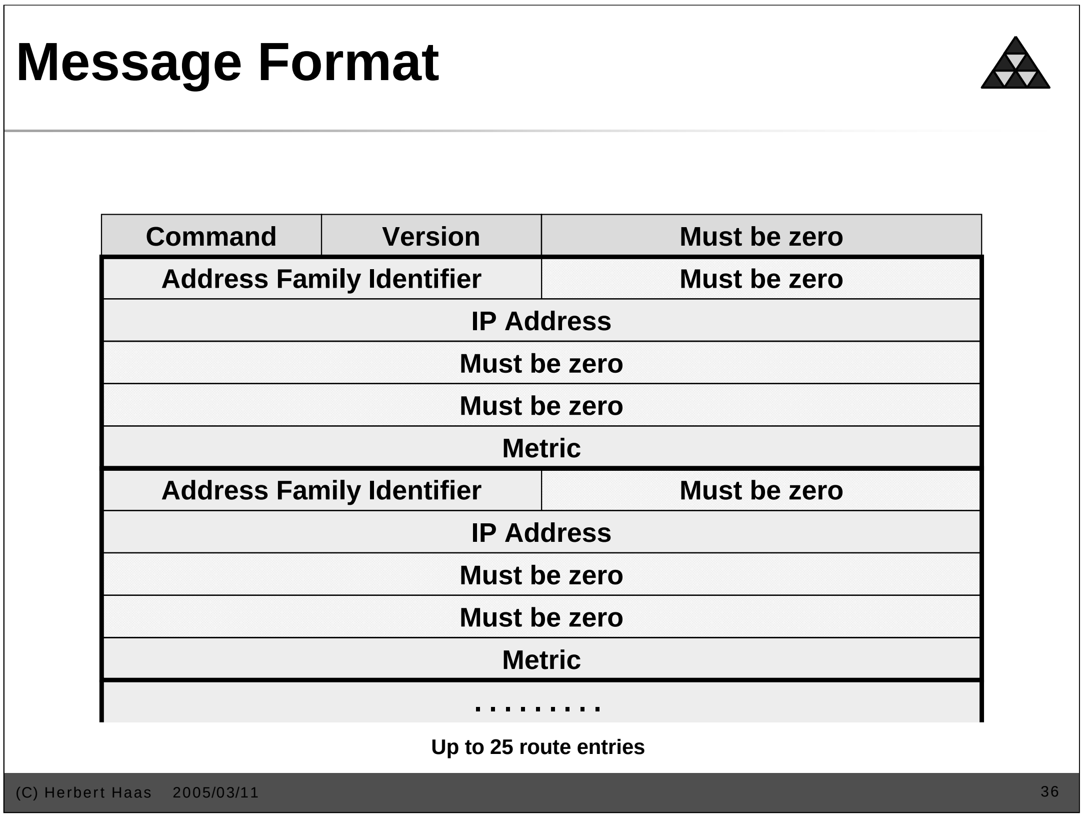


RIP Timers Summary

- **UPDATE (30 seconds)**
  - **Period to send routing update**
- **INVALID (180 seconds)**
  - **Aging time before declaring a route invalid ("16") in the routing table**
- **HOLDDOWN (180 seconds)**
  - **After a route has been invalided, how long a router will wait before accepting an update with better metric**
- **FLUSH (240 seconds)**
  - **Time before a non-refreshed routing table entry is removed**

The FLUSH timer is also known as "Garbage Collection Timer" and RFC 1058 suggests additional 120 seconds after expiring of the INVALID timer.

HOLDDOWN timers are not explicitely required by RFC 1058, however they are supported by most implementations today, e. g. by Cisco IOS. Note that the FLUSH timer expires before the HOLDDOWN timer.

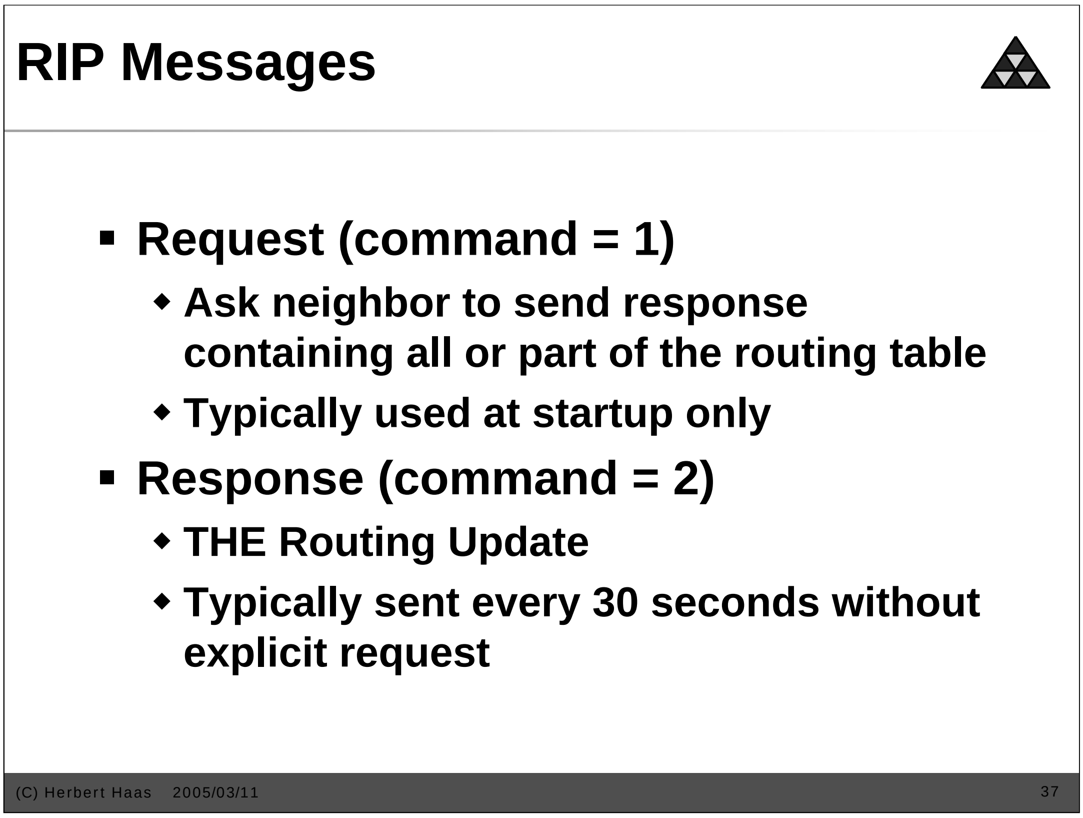


Message Format

Command	Version	Must be zero
Address Family Identifier		Must be zero
IP Address		
Must be zero		
Must be zero		
Metric		
Address Family Identifier		Must be zero
IP Address		
Must be zero		
Must be zero		
Metric		
. . . .		

Up to 25 route entries

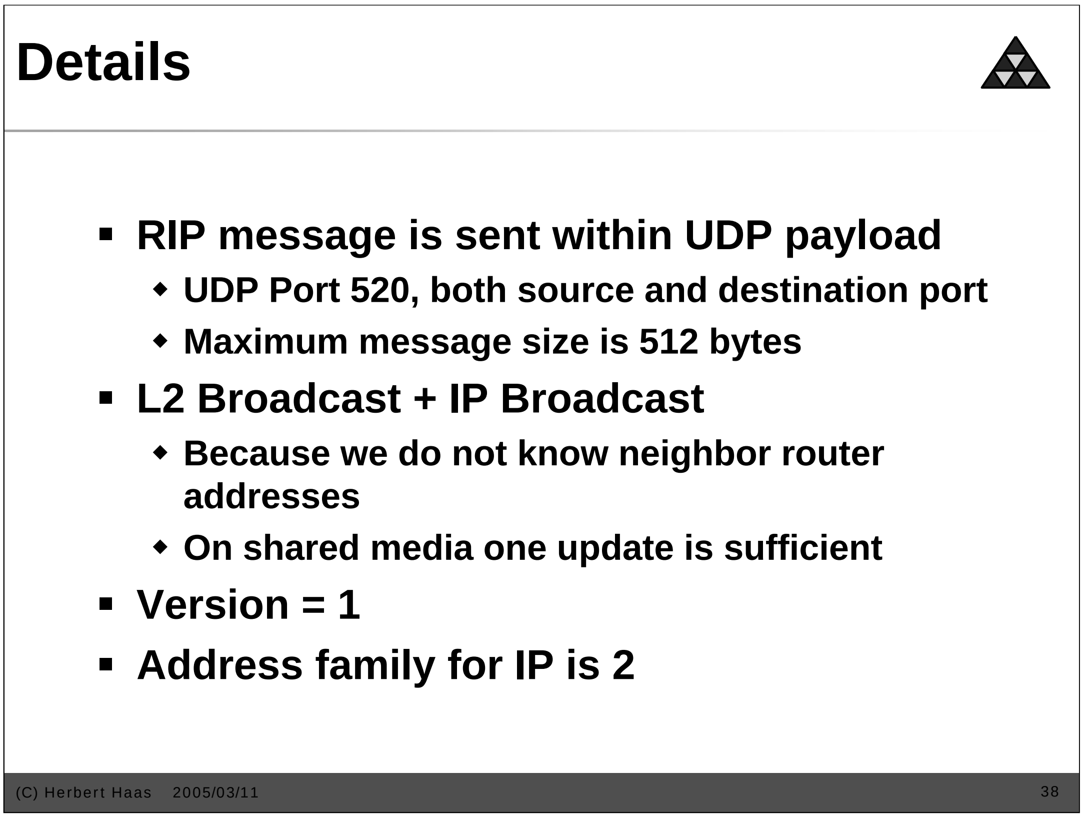
The RIP version 1 Message format simply consists of a header, indicating command-type and version, and a number of sections reflecting a routing table entry. Up to 25 route entries per packet are allowed. Note that each route does not include a "next-hop" address! The next-hop address is simply the source address of this packet, that is, the originator declares himself as next-hop for all listed routes. Also note that there are several fields reserved as "Must be zero" to leave space for future improvements. We will see, that RIPv2 uses these fields.



RIP Messages

- Request (command = 1)
  - Ask neighbor to send response containing all or part of the routing table
  - Typically used at startup only
- Response (command = 2)
  - THE Routing Update
  - Typically sent every 30 seconds without explicit request

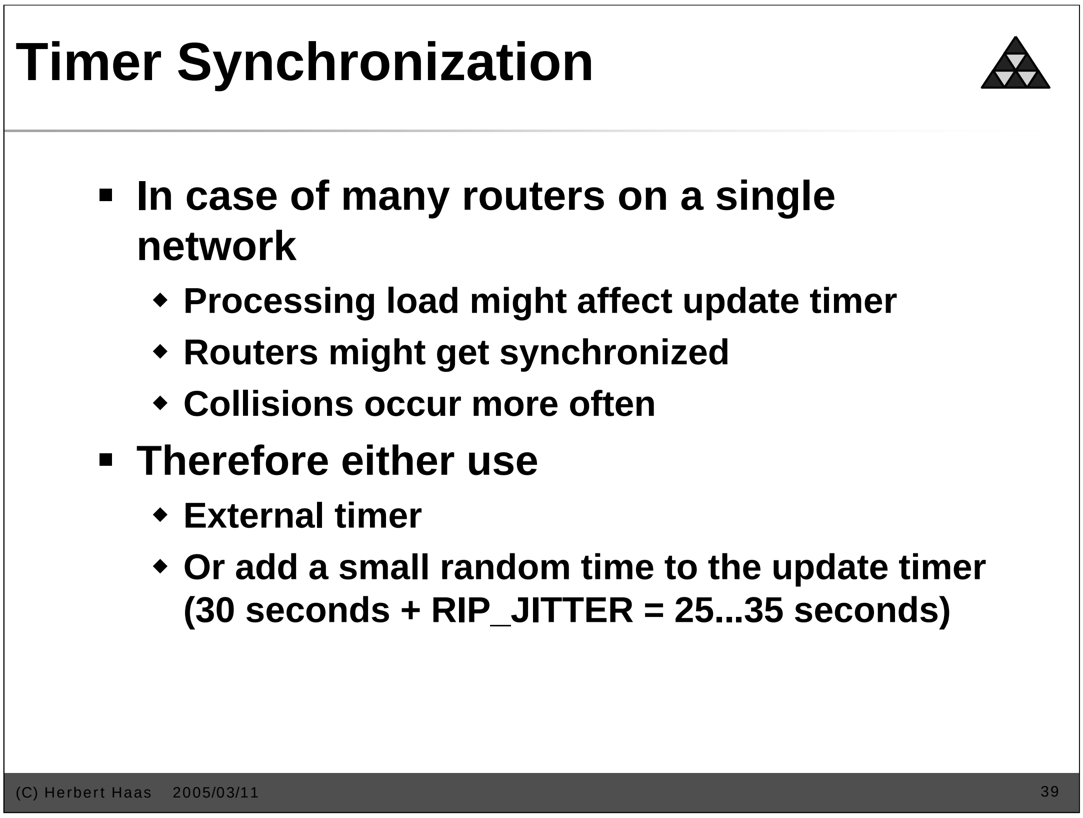
Note that a request is for specific entries (i. e. not for the whole table), the requested information is returned in any case, that is no split horizon is performed and even subnets are returned if requested. If there is exactly one entry in the request, with an address family identifier of zero and a metric of infinity (16), this is a request to send the entire routing table.



Details

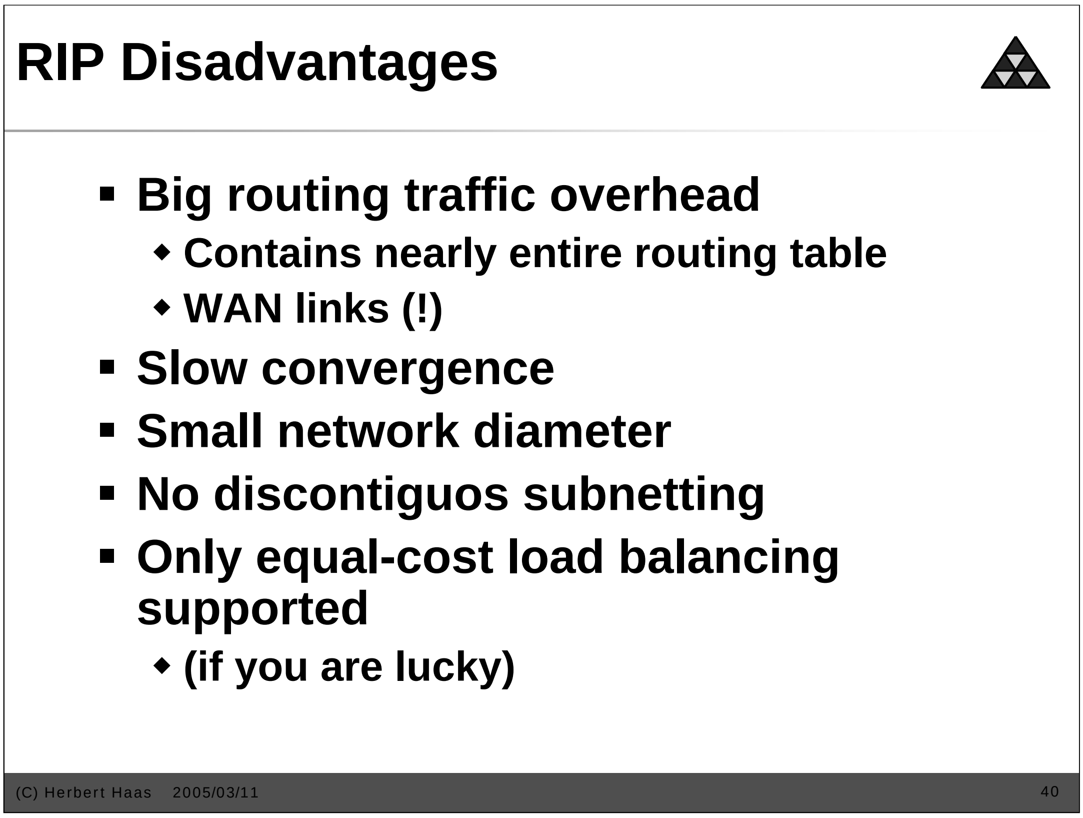
- RIP message is sent within UDP payload
  - UDP Port 520, both source and destination port
  - Maximum message size is 512 bytes
- L2 Broadcast + IP Broadcast
  - Because we do not know neighbor router addresses
  - On shared media one update is sufficient
- Version = 1
- Address family for IP is 2

If RIP messages are generated from any other port than 520 even "silent" processes must response. This is an old RFC requirement, don't expect everything works that way...



Timer Synchronization

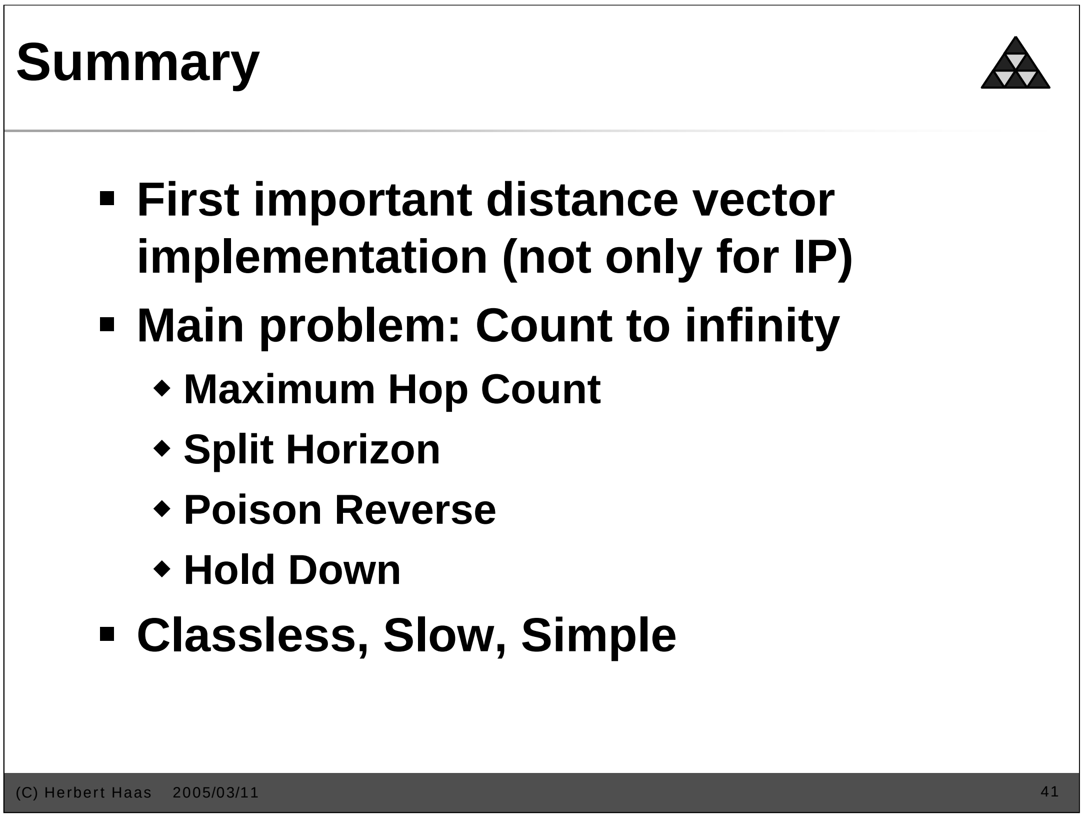
- In case of many routers on a single network
  - Processing load might affect update timer
  - Routers might get synchronized
  - Collisions occur more often
- Therefore either use
  - External timer
  - Or add a small random time to the update timer (30 seconds + RIP_JITTER = 25...35 seconds)



RIP Disadvantages

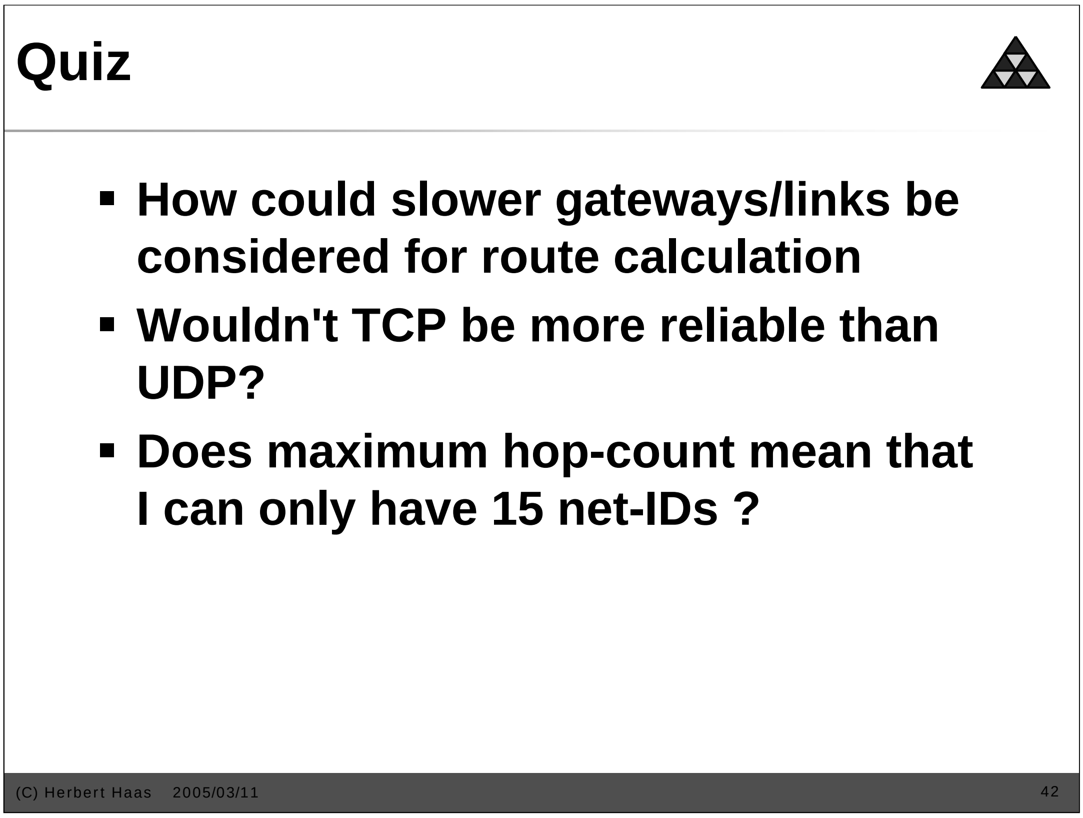
- Big routing traffic overhead
  - Contains nearly entire routing table
  - WAN links (!)
- Slow convergence
- Small network diameter
- No discontiguos subnetting
- Only equal-cost load balancing supported
  - (if you are lucky)

RIP is an old protocol and only used in small networks.



Summary

- First important distance vector implementation (not only for IP)
- Main problem: Count to infinity
  - Maximum Hop Count
  - Split Horizon
  - Poison Reverse
  - Hold Down
- Classless, Slow, Simple



Quiz

- How could slower gateways/links be considered for route calculation
- Wouldn't TCP be more reliable than UDP?
- Does maximum hop-count mean that I can only have 15 net-IDs ?