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 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!)

"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

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 directly 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

- **all routing table‘s net-IDs are periodically refreshed by routing-update messages**
- **if a routing update tells a <u>better</u> 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 <u>any</u> source ("trusted news")

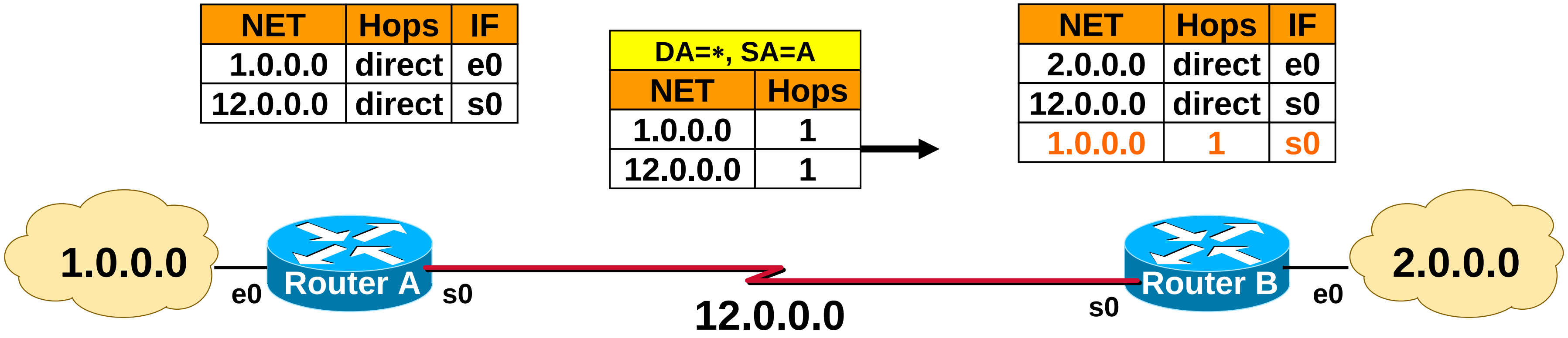
RIP Principles 5

- **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

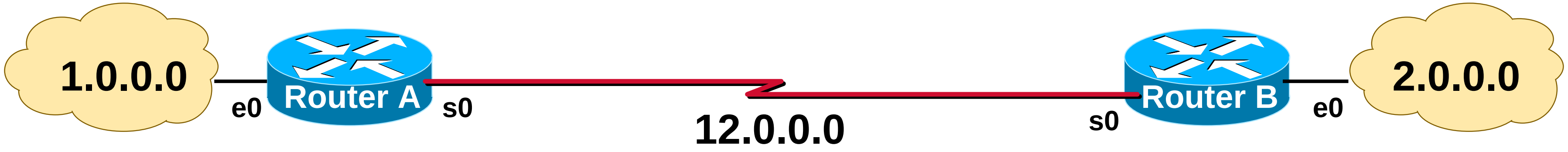
NET	Hops	IF
2.0.0.0	direct	e0
12.0.0.0	direct	s0
1.0.0.0	1	s0

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).

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

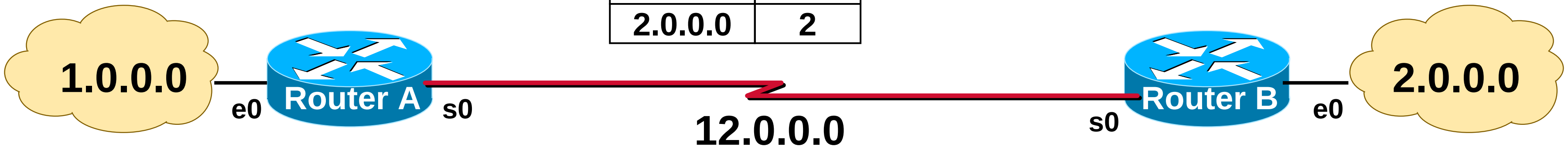
Router B sends first routing update containing directly connected networks and already learned network 1.0.0.0;
Router A learns new network 2.0.0.0 (good news);
Router B ignores network 12.0.0.0 and 1.0.0.0 in the received update (bad news).

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

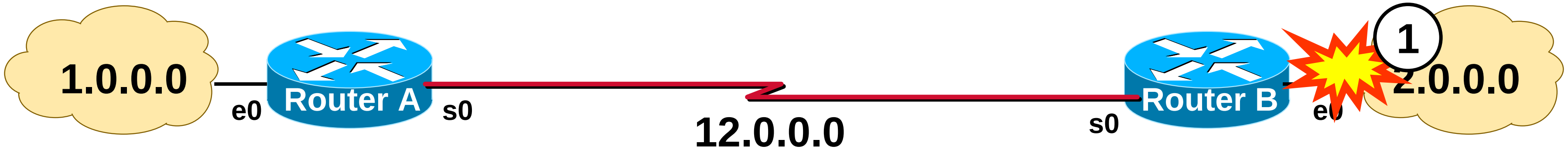
Now the peridiodical updates refreshes the learned networks

Without Split Horizon (4)

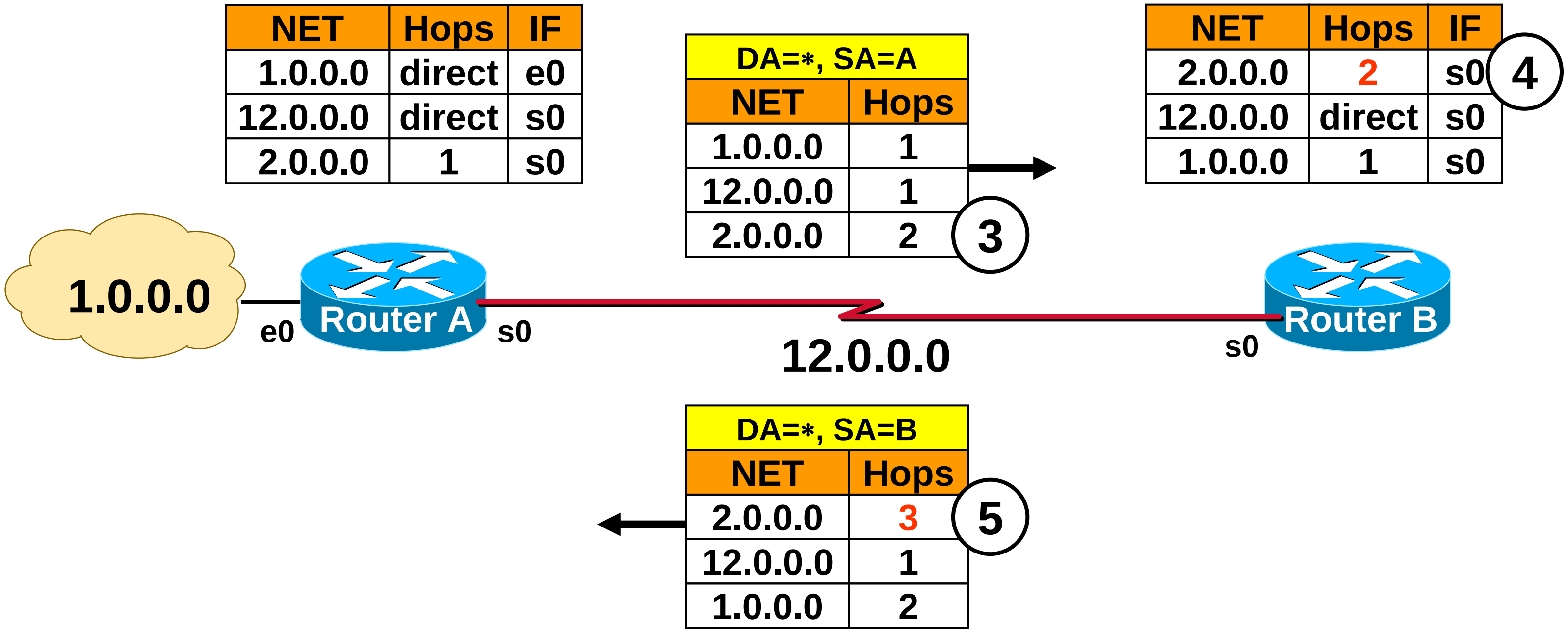
NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
2.0.0.0	1	s0

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

2



Without Split Horizon (5)



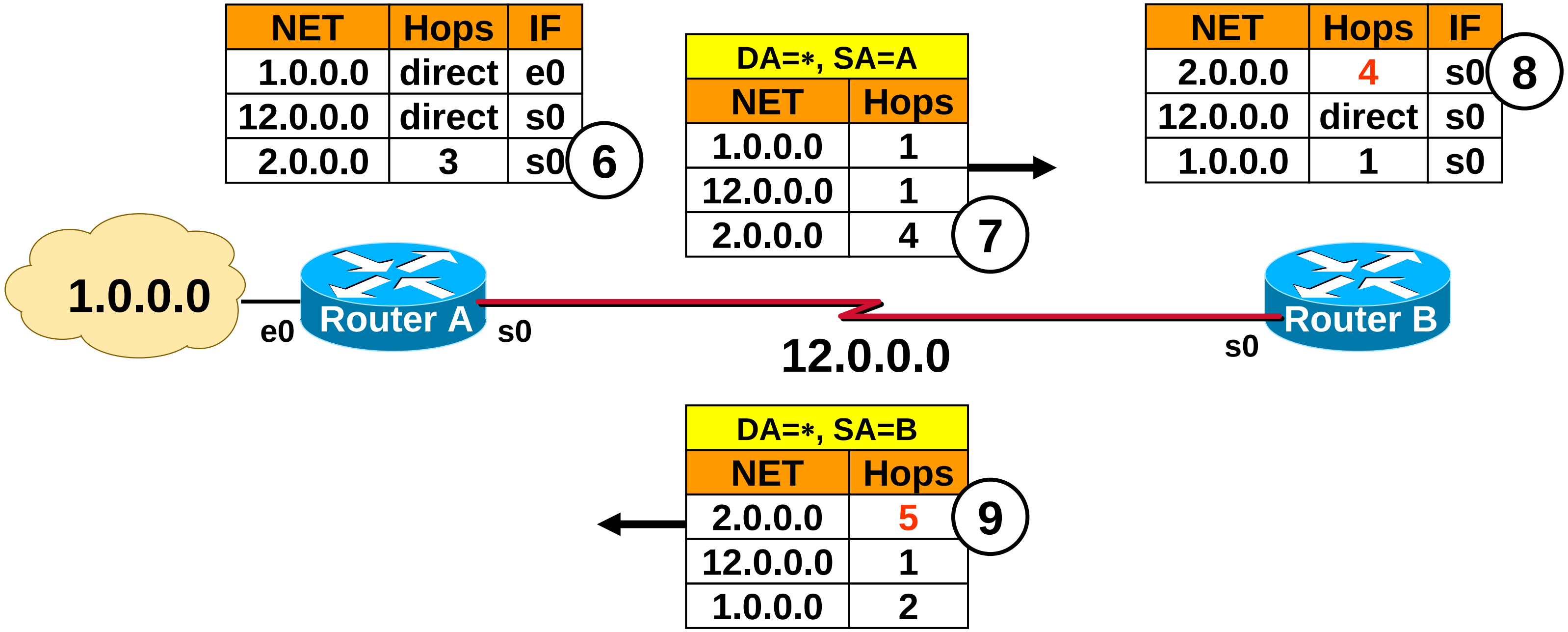
NET	Hops	IF
1.0.0.0	direct	e0
12.0.0.0	direct	s0
2.0.0.0	1	s0

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

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

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

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

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

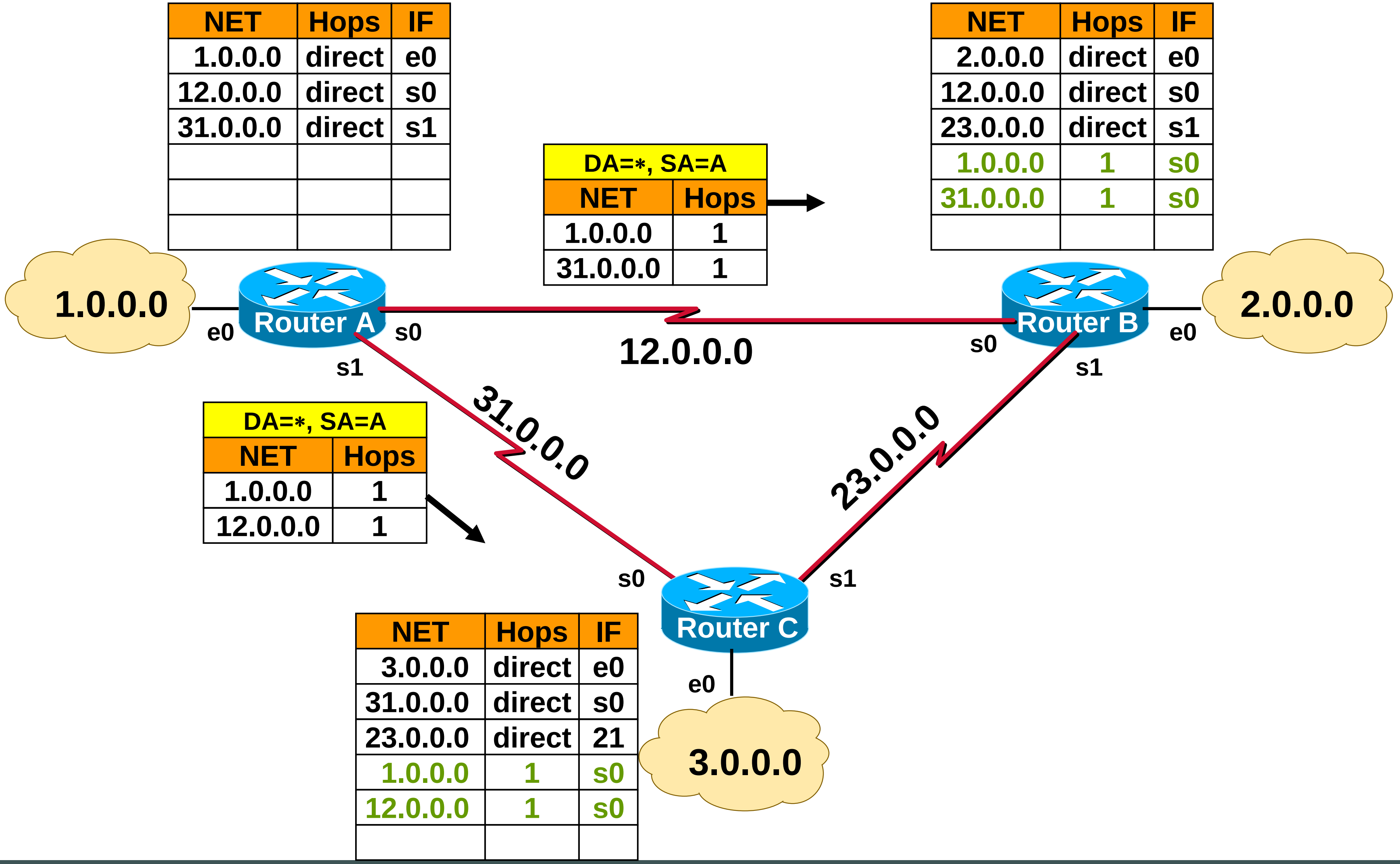
...Count to Infinity...

During count to infinity packets to network 2.0.0.0 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!

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

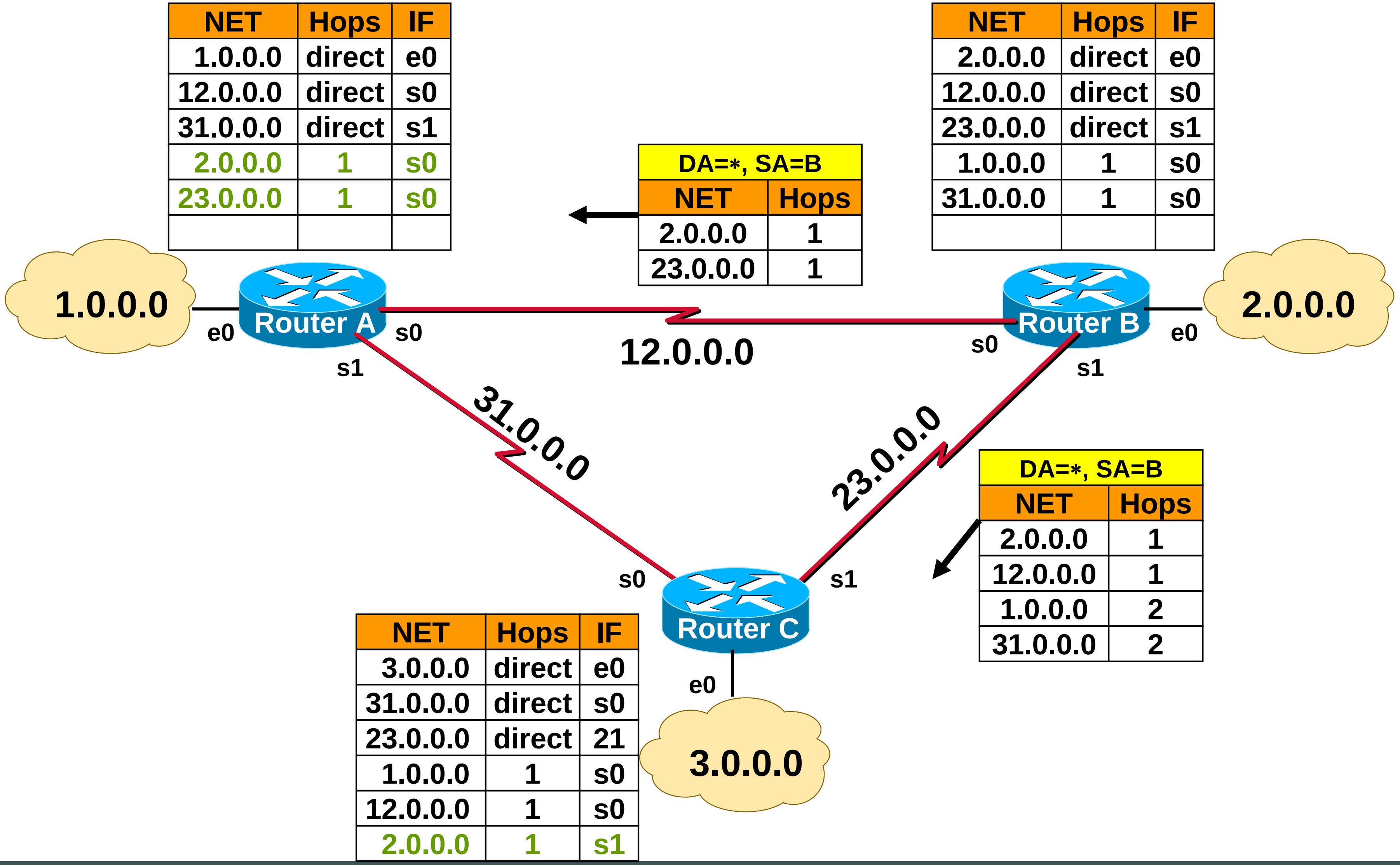
DA=*, SA=A	
NET	Hops
1.0.0.0	1
31.0.0.0	1

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

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

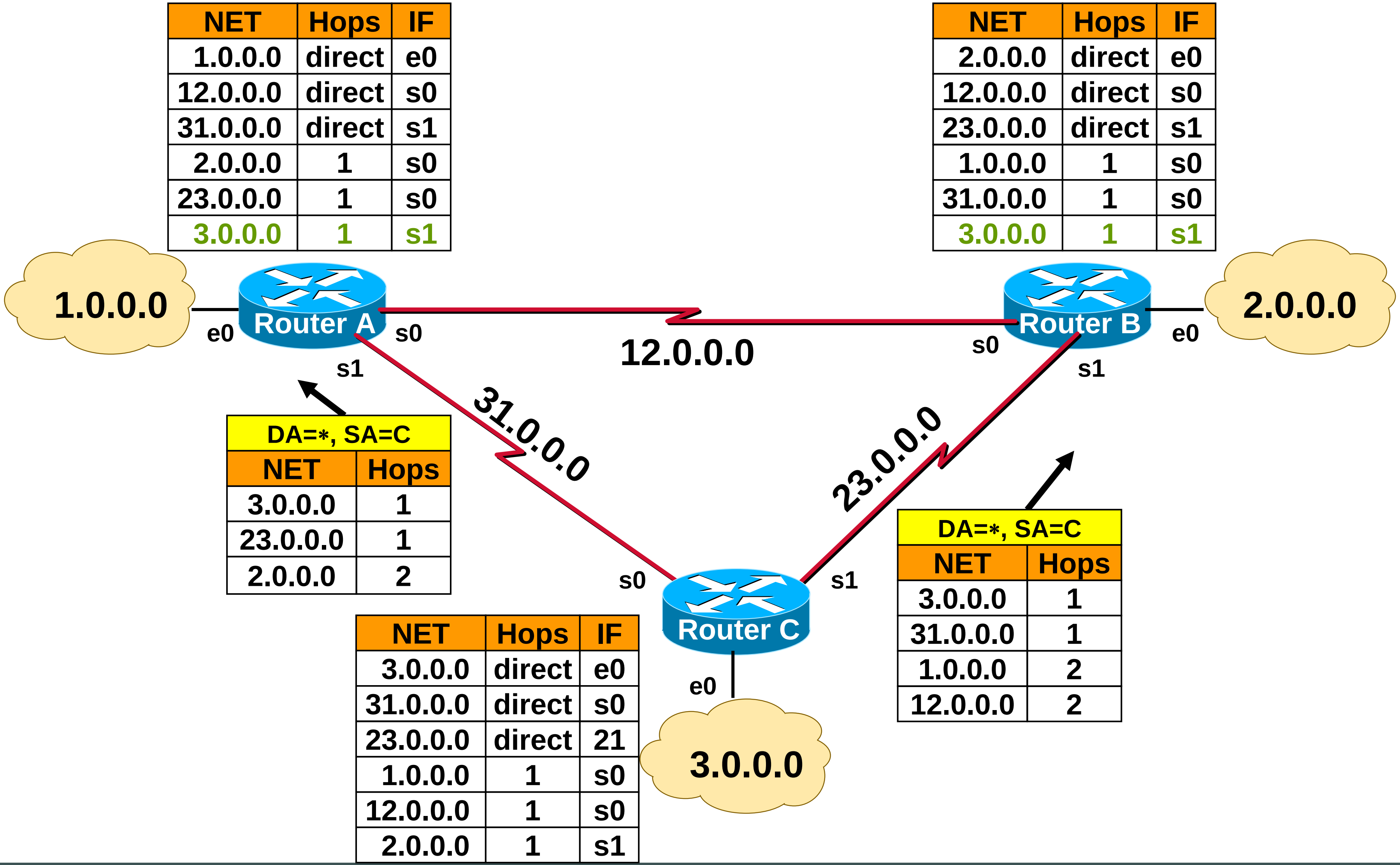
DA=∗, SA=B	
NET	Hops
2.0.0.0	1
23.0.0.0	1

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=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

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

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

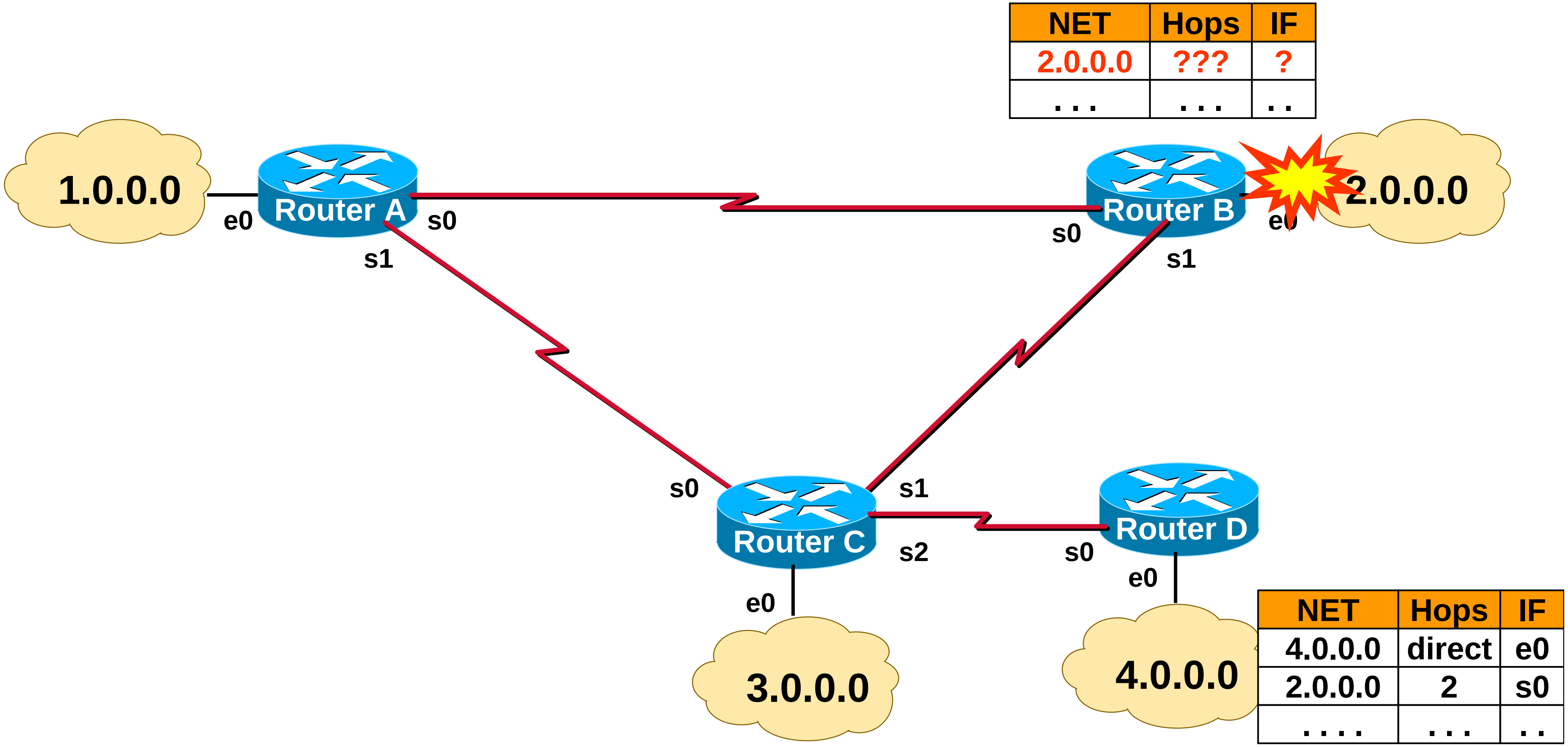
DA=∗, SA=C

NET	Hops
3.0.0.0	1
31.0.0.0	1
1.0.0.0	2
12.0.0.0	2

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!**

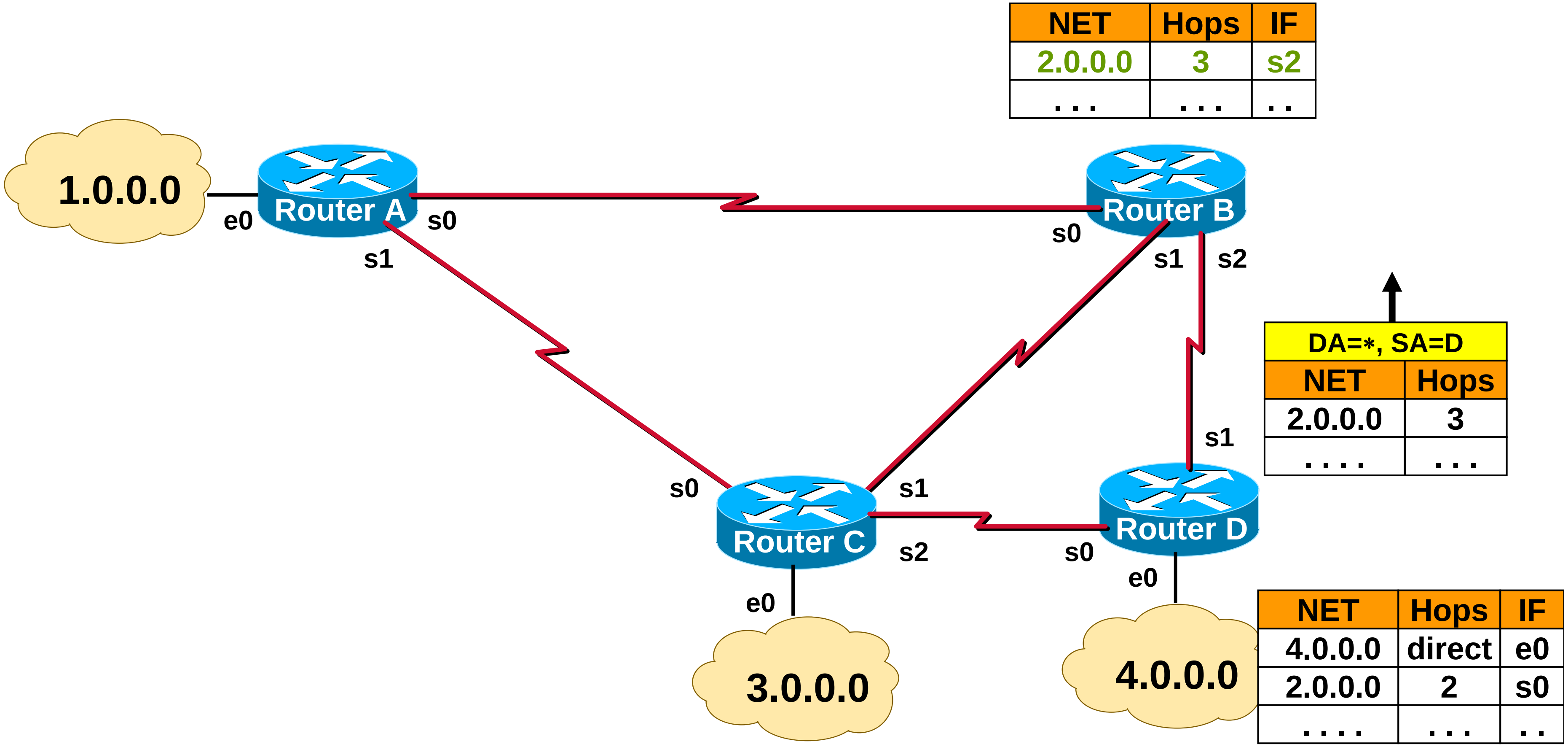
Count To Infinity (1)



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

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

Count To Infinity (2)

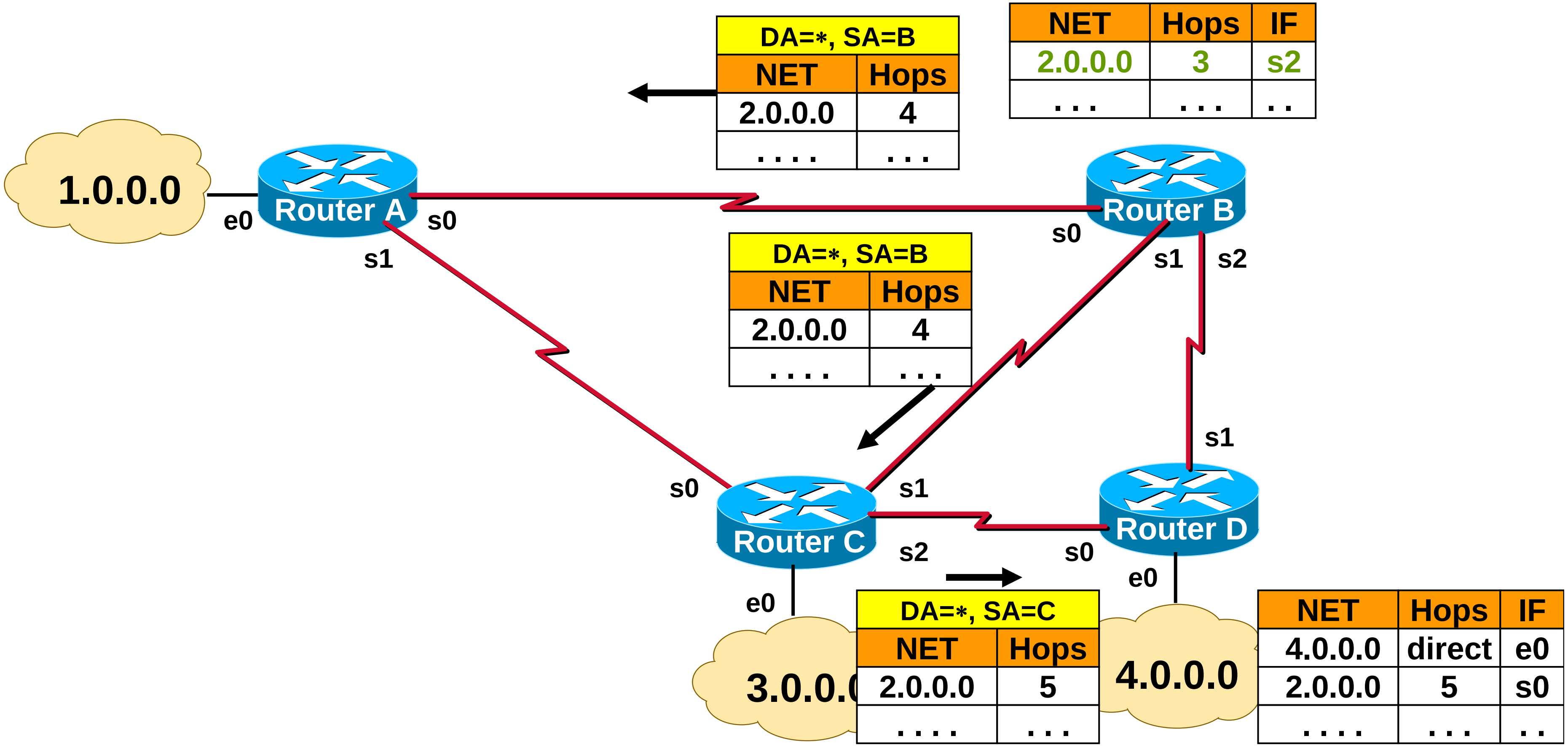


NET	Hops	IF
2.0.0.0	3	s2
. . .	. . .	. .

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
. . . .	. . .	. .

Count To Infinity (3)



DA=*, SA=B	
NET	Hops
2.0.0.0	4
. . . .	. . .

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

DA=*, SA=B	
NET	Hops
2.0.0.0	4
. . . .	. . .

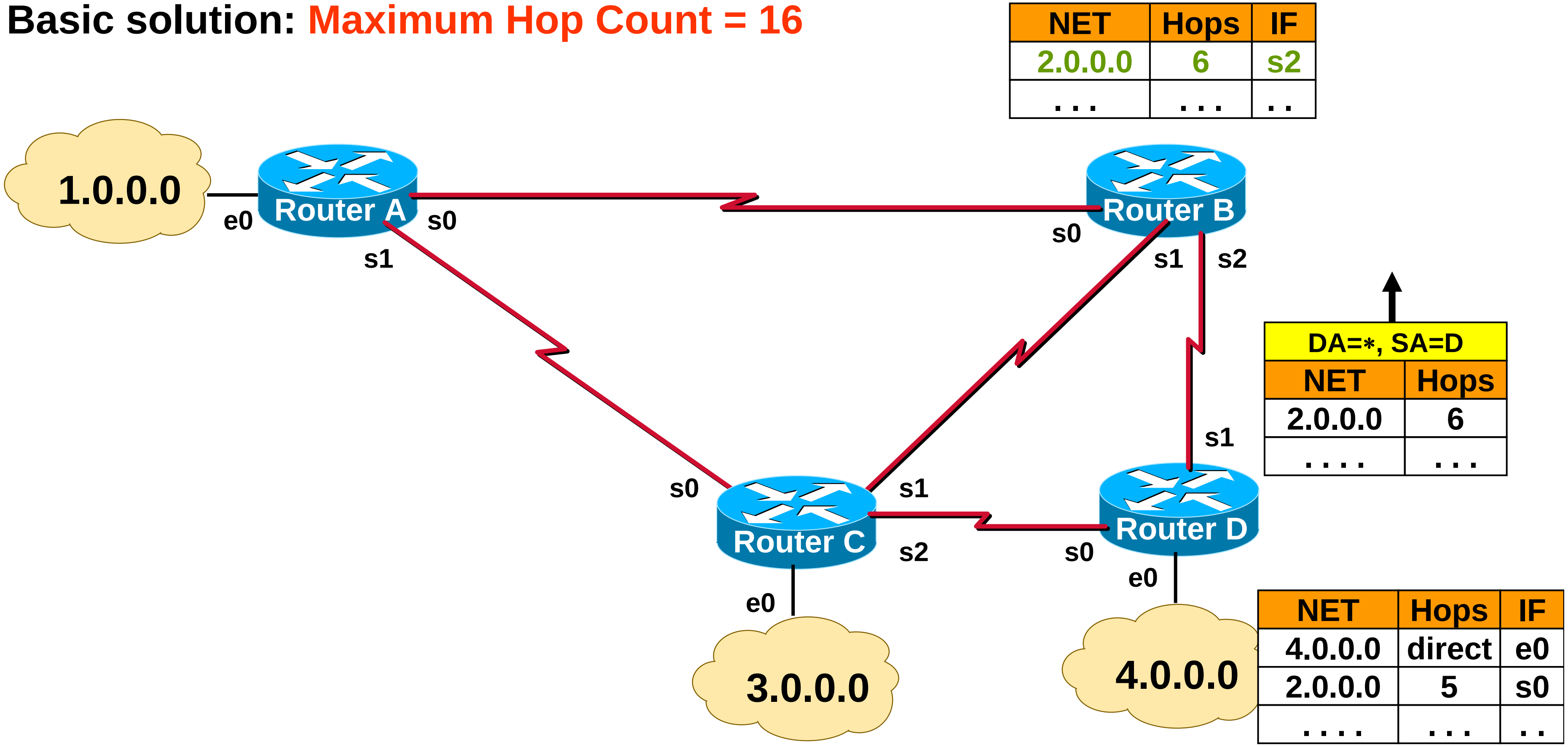
DA=*, SA=C	
NET	Hops
2.0.0.0	5
. . . .	. . .

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

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
. . .	. . .	. .

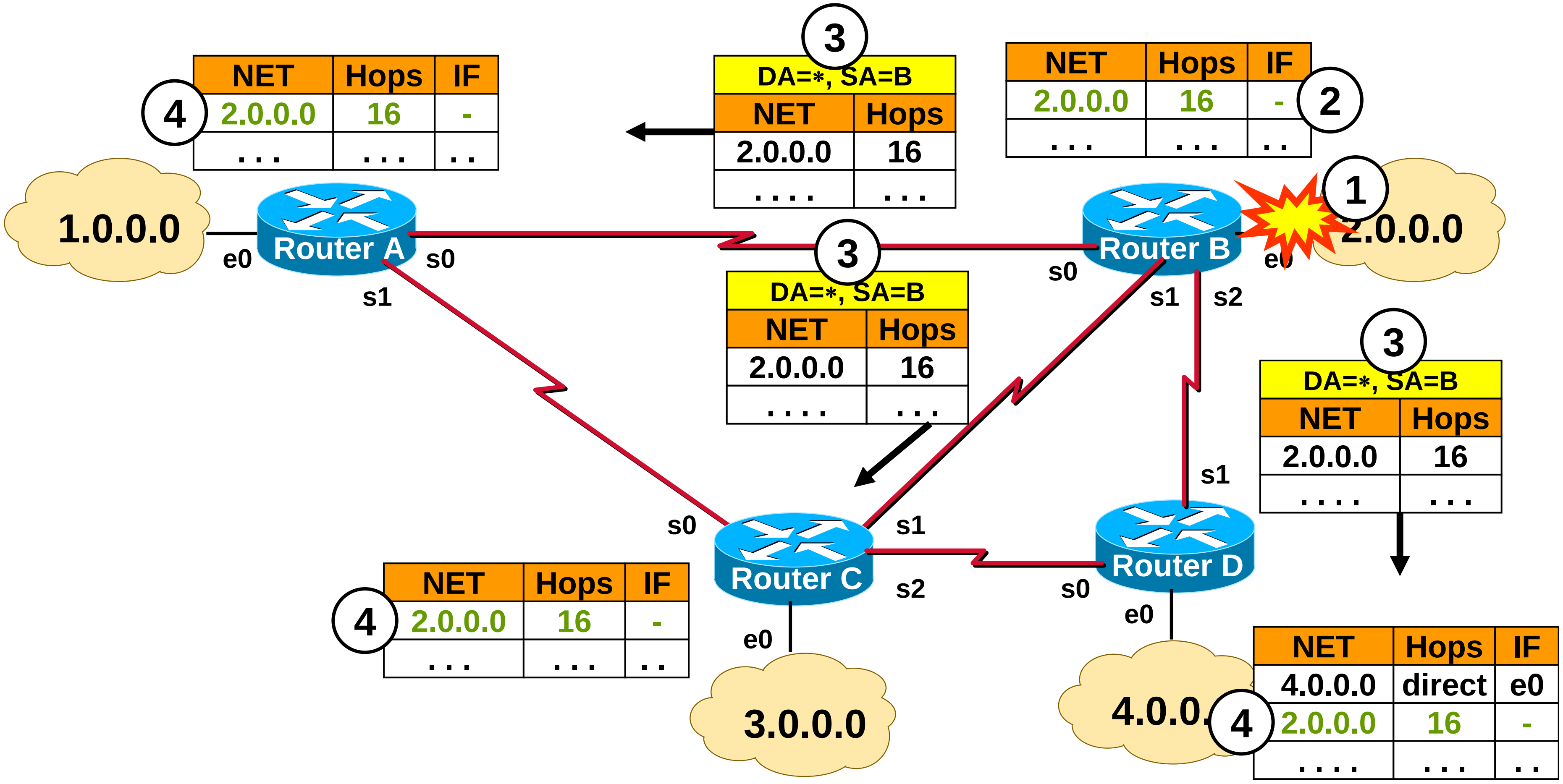
DA=∗, SA=D

NET	Hops
2.0.0.0	6
. . . .	. . .

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

Maximum Hop Count = 16

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



④

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

③

DA=*, SA=B

NET	Hops
2.0.0.0	16
. . . .	. . .

②

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

③

DA=*, SA=B

NET	Hops
2.0.0.0	16
. . . .	. . .

③

DA=*, SA=B

NET	Hops
2.0.0.0	16
. . . .	. . .

④

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

④

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

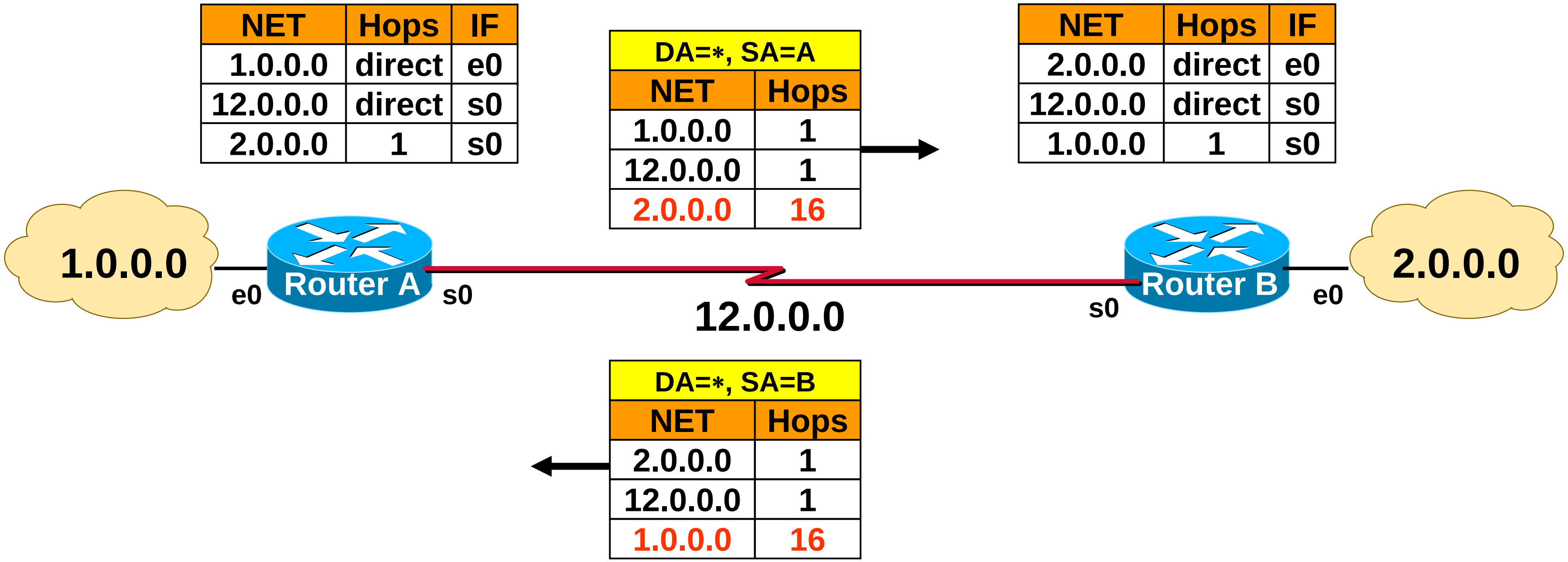
Maximum Hop Count

- 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

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

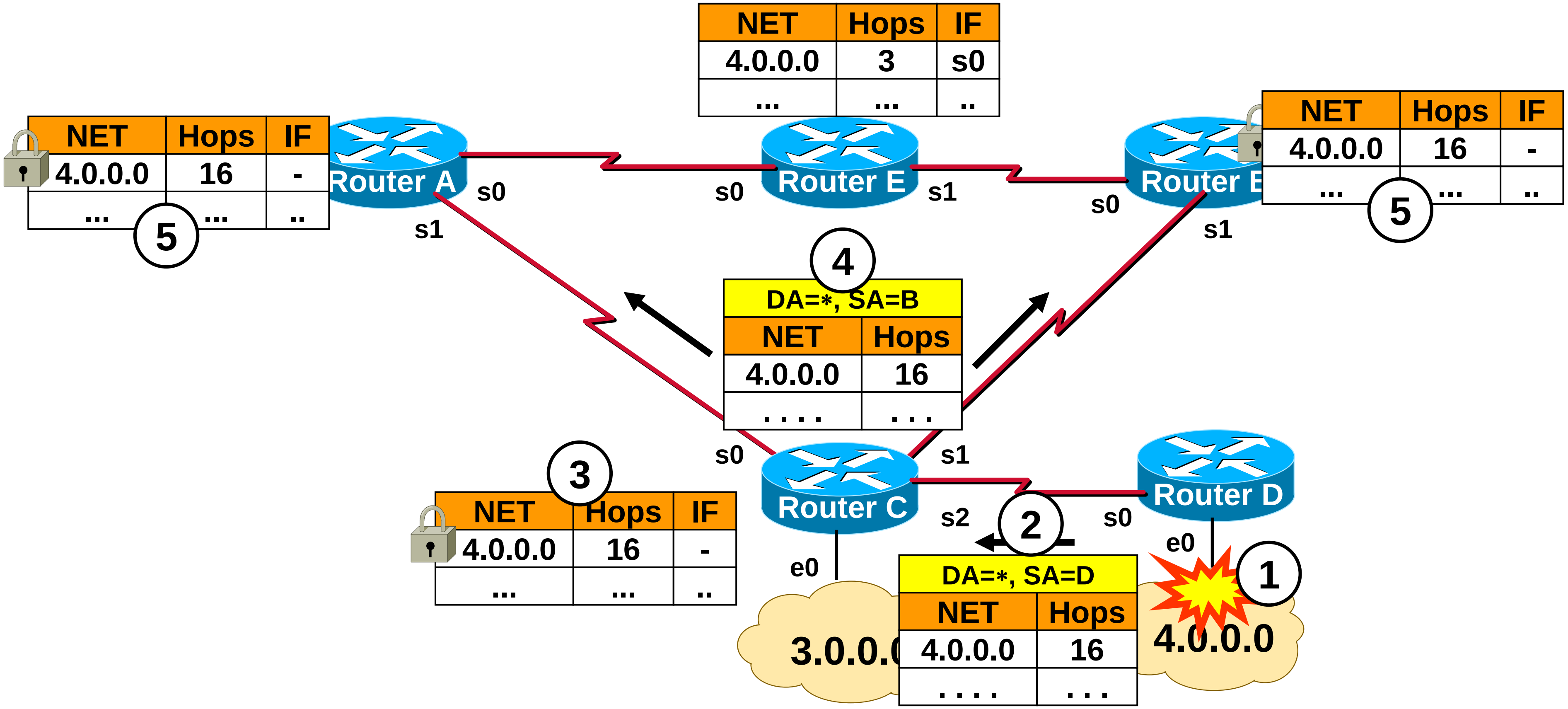
Note: poison reverse overrides split horizon when a network is lost

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

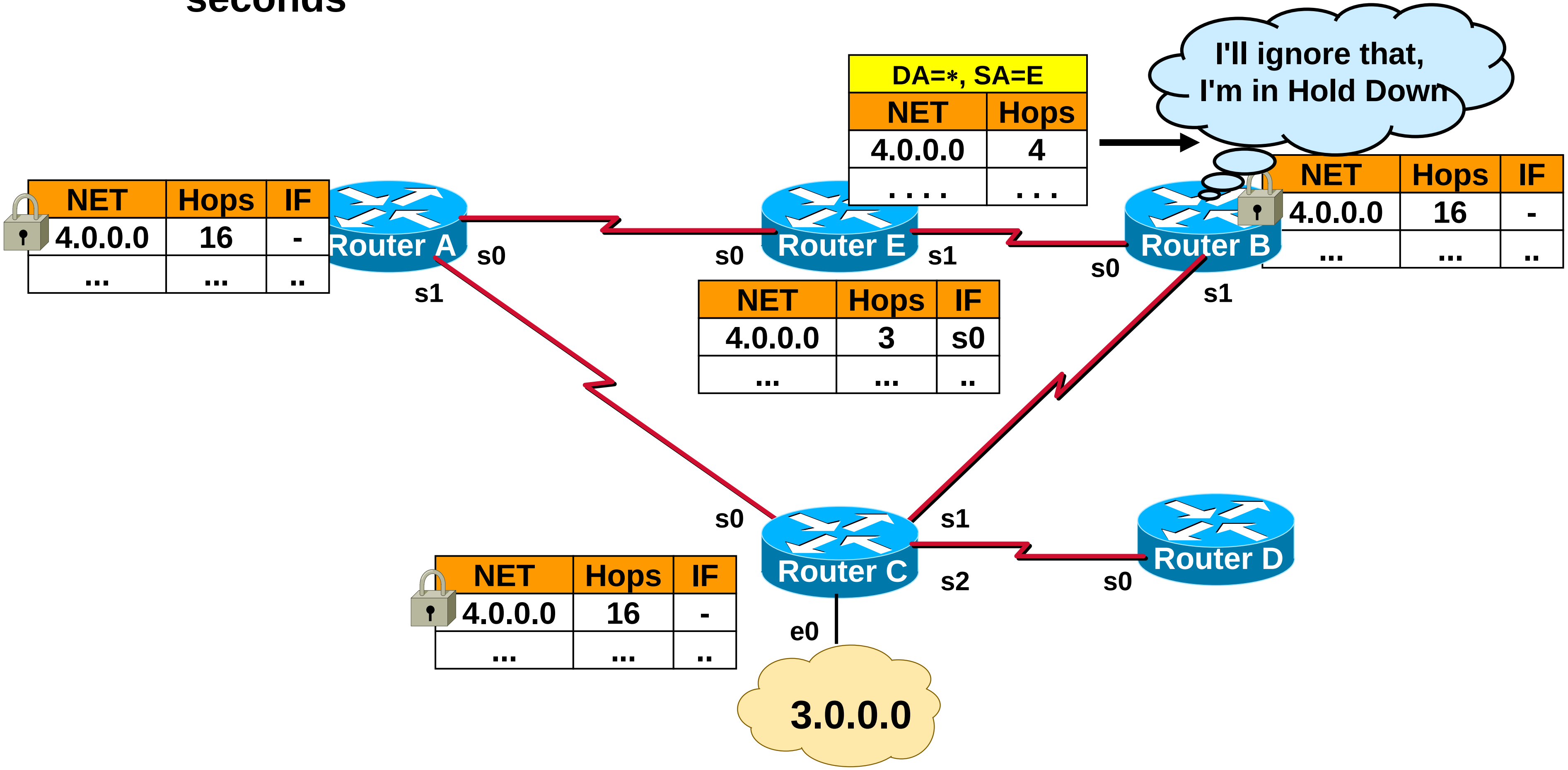
Hold Down (1)

- 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



Hold Down (2)

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



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

DA=∗, SA=E	
NET	Hops
4.0.0.0	4
. . . .	. . .

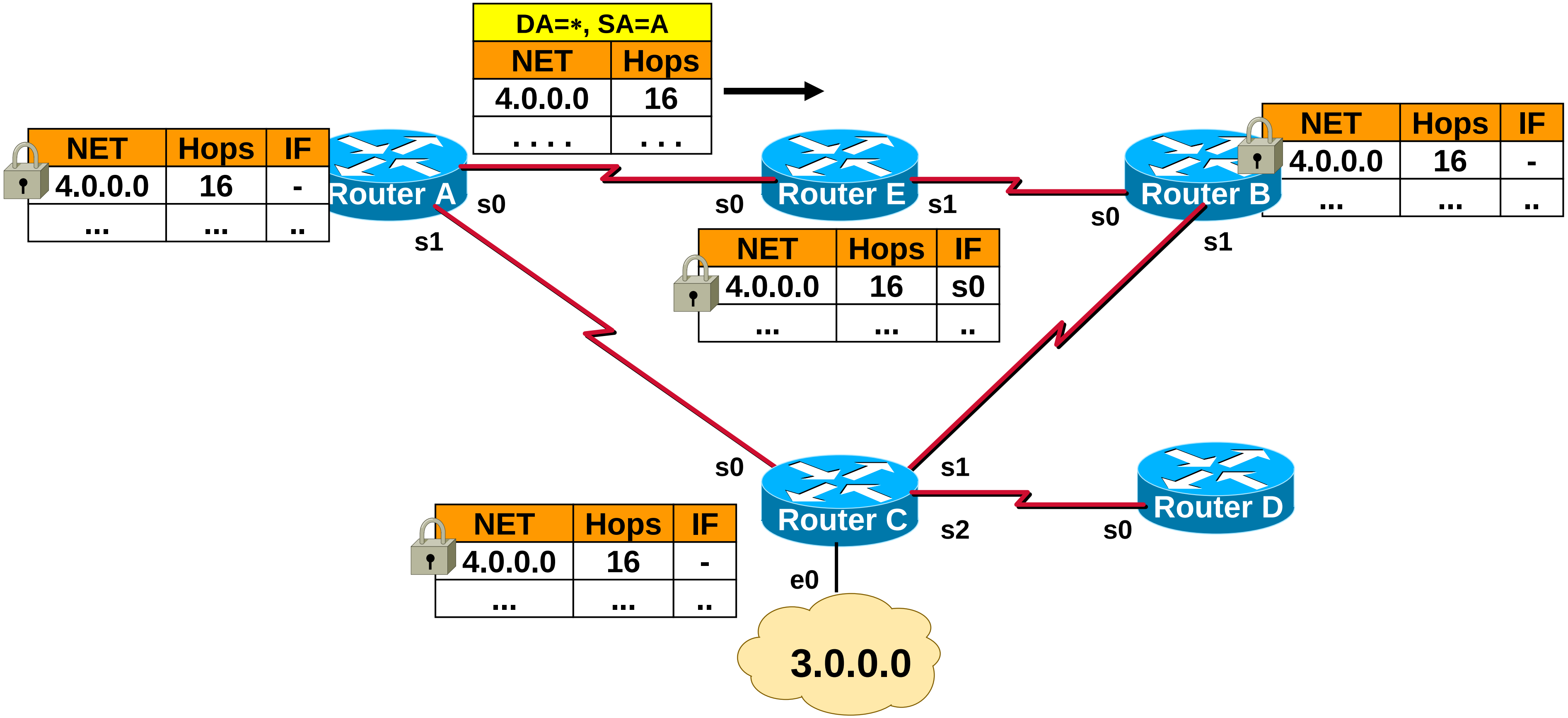
NET	Hops	IF
4.0.0.0	3	s0
...	...	..

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

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

Hold Down (3)

- Time enough to propagate the unreachabilty of network 4.0.0.0



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

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

Message Format

<table>
<tr><td>Command</td><td>Version</td><td colspan="2">Must be zero</td></tr>
<tr><td colspan="2">Address Family Identifier</td><td colspan="2">Must be zero</td></tr>
<tr><td colspan="4">IP Address</td></tr>
<tr><td colspan="4">Must be zero</td></tr>
<tr><td colspan="4">Must be zero</td></tr>
<tr><td colspan="4">Metric</td></tr>
<tr><td colspan="2">Address Family Identifier</td><td colspan="2">Must be zero</td></tr>
<tr><td colspan="4">IP Address</td></tr>
<tr><td colspan="4">Must be zero</td></tr>
<tr><td colspan="4">Must be zero</td></tr>
<tr><td colspan="4">Metric</td></tr>
<tr><td colspan="4">. . . . .</td></tr>
</table>

Up to 25 route entries

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**

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

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)

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 ?