



Routing Optimization, Redistribution, Policy Routing



ROUTE modules 4 + 5

Agenda

- **Characteristic of network design**
- **Redistribute routes into RIP, OSPF, EIGRP**
- **Control routing updates**
 - distribute lists
 - ACLs, prefix lists
 - route maps
- **IP SLA monitoring**

Characteristic of Network Design



Characteristic of Network Design ①

▪ Resiliency

- Ability to maintain an acceptable level of service when faults occur
- Redundancy does not guarantee resiliency
- E.g. fail-over, load balancing

▪ Availability

- Time required for a routing protocol to converge
- E.g. fast converging routing protocols, parameters tuning

▪ Adaptability

- Ability to adapt to changing conditions
- E.g. using redundant path when the primary path becomes congested

▪ Performance

- Routers and other active network devices tuning to achieve best performance
- E.g. load balancing, metrics modification

Characteristic of Network Design ②

- **Support for network and application services**

- Advance path control for specific services – security, QoS, WAAS (Wide Area Application Services)

- **Predictability**

- Implemented path control solution SHOULD provide deterministic and predictable results

- **Asymmetric traffic**

- Traffic flows on the one path in one direction and on a different path in the opposite direction
- Often desirable network trait (satellite link: upstream is very slow compared to downstream)

Redistribution



Multiple IP Routing Protocols on Network

- Interim solution when migrating between IGP
- Merging network under different management
 - ISP and customer networks
- Advanced features in other routing protocol
 - “World beater” routing protocol does not exist
- „Political“ reasons
 - *It's not a rule, it's the policy!*
- Mixed vendor environment
 - Different vendors = different protocols
 - Recommendation is to use open standard protocol

Redistribution ①

- Routing information are exchanged from the one routing protocol to another
- Routing information exchange is called **redistribution**
- *Please note!*
 - Every routing protocol has its own working database
 - Routing protocol populates routing table with the best routes using based on information in the working database
 - Routing protocol transmits only own working database
 - EIGRP spreads only own networks and same behavior have also others (as well as RIP, OSPF and IS-IS)

Redistribution ②

- Several routing protocols MAY run simultaneously on a router
 - Routing information is **NOT** exchanged between routing protocols
 - Additional configuration is always necessary to accomplish this
- **Routing table** is the only place, where information from all routing protocols are “put together”
- **Network MUST be first in the routing table...**
 - ...only after that it can be redistributed
- Routing table is **intact** on redistributing router
- Detailed information about topology are **NOT** sent...
 - ...only information about existence of the network in the routing table

Redistribution ③

- Every protocol has different ways of metric computation
 - Metric in different routing protocol CAN NOT be easily adopted
- Solution:
 - Initial metric (a.k.a. **seed metric**) is set for redistribution in a target protocol...
 - ...and after that the metric is increased as usual
- Link-state protocols use predefined default metric, distance-vector protocols use infinity
 - Initial metric SHOULD be always set by administrator
 - Using default values is not recommended

Default Seed Metrics

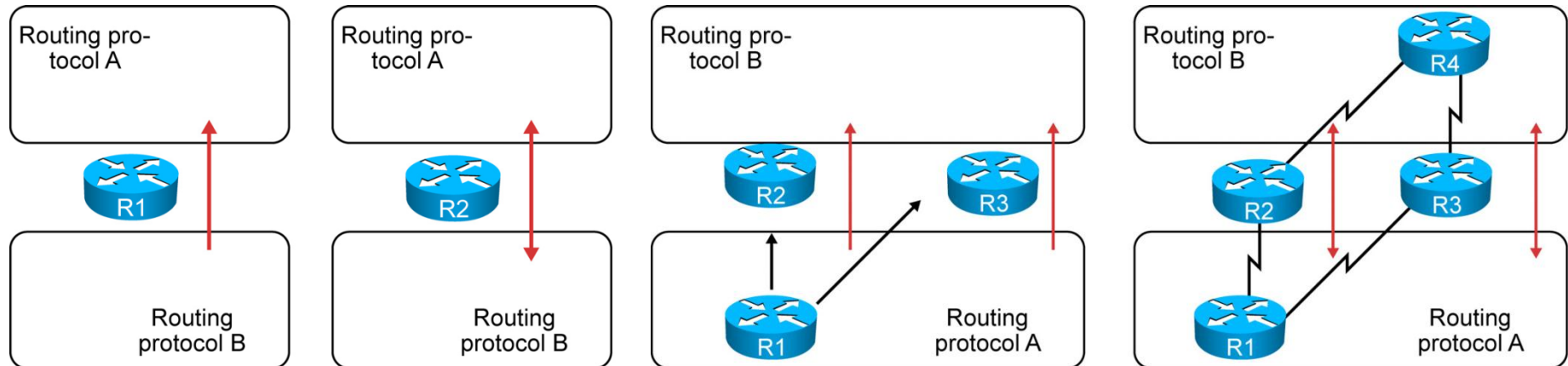
Protocol	Seed metric
RIP	0 interpreted as infinity
IGRP/EIGRP	0 interpreted as infinity
OSPF	20 for all except BGP, 1 for BGP
IS-IS	0
BGP	BGP metric is set to IGP metric value

Redistribution ④

- Any routing protocol can be used as a **source** for redistribution
 - Everything what populates routing table can be redistributed – static routes, connected networks etc.
 - C, L, S, D, D EX, B, O, O IA, O E1/2, O N1/2, R, i
- Any routing protocol (except ODR) can be used as a **destination** for redistribution

Redistribution ⑤

- Redistribution is configured on one or more routers where multiple IGPs are used simultaneously
 - **One-way** or **two-way** redistribution
 - One router (**one-point**) or several routers (**multipoint**)



- Multipoint redistribution – possible cause of routing loops ☹️

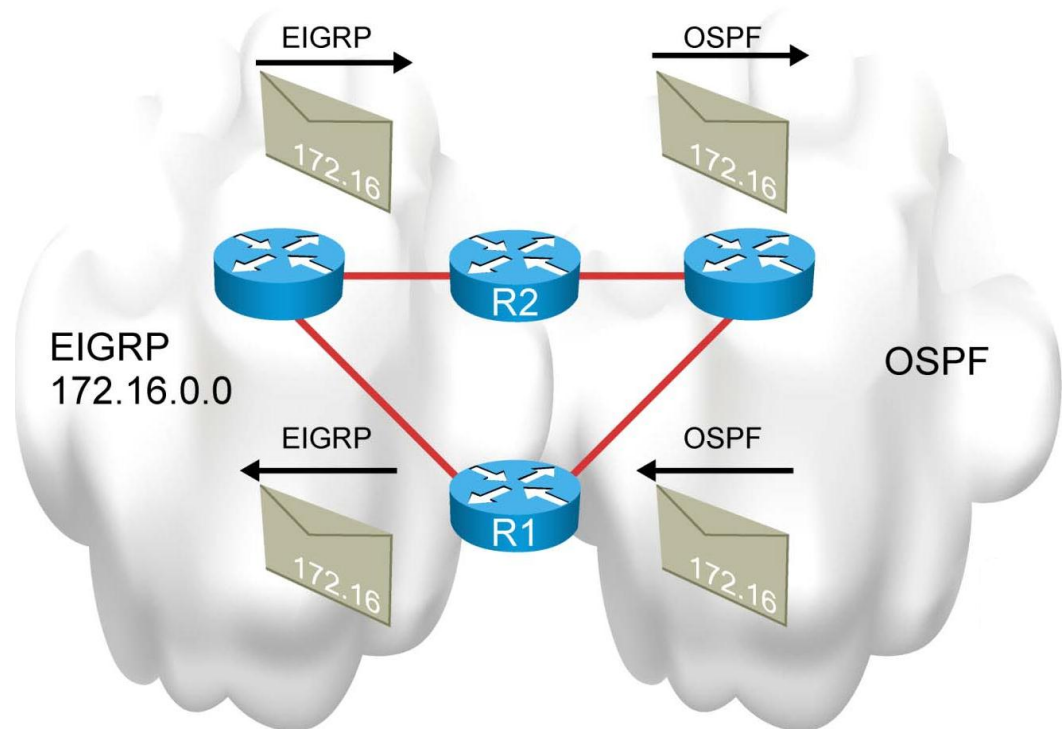
Redistribution: Issues ①

■ Problems:

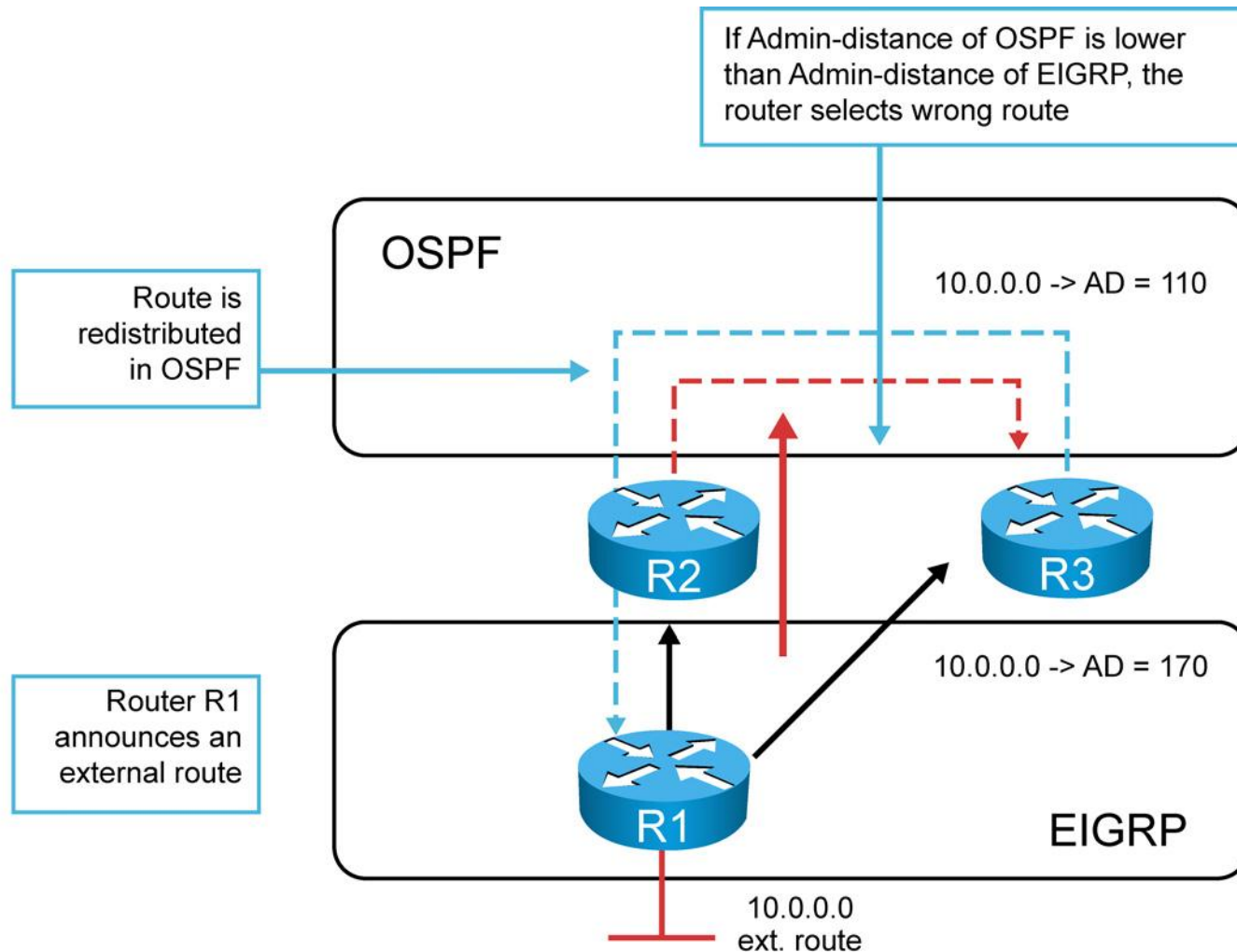
- Routing loop
- Suboptimal path selection
- Incompatible routing information
- Inconsistent convergence time

■ Solutions:

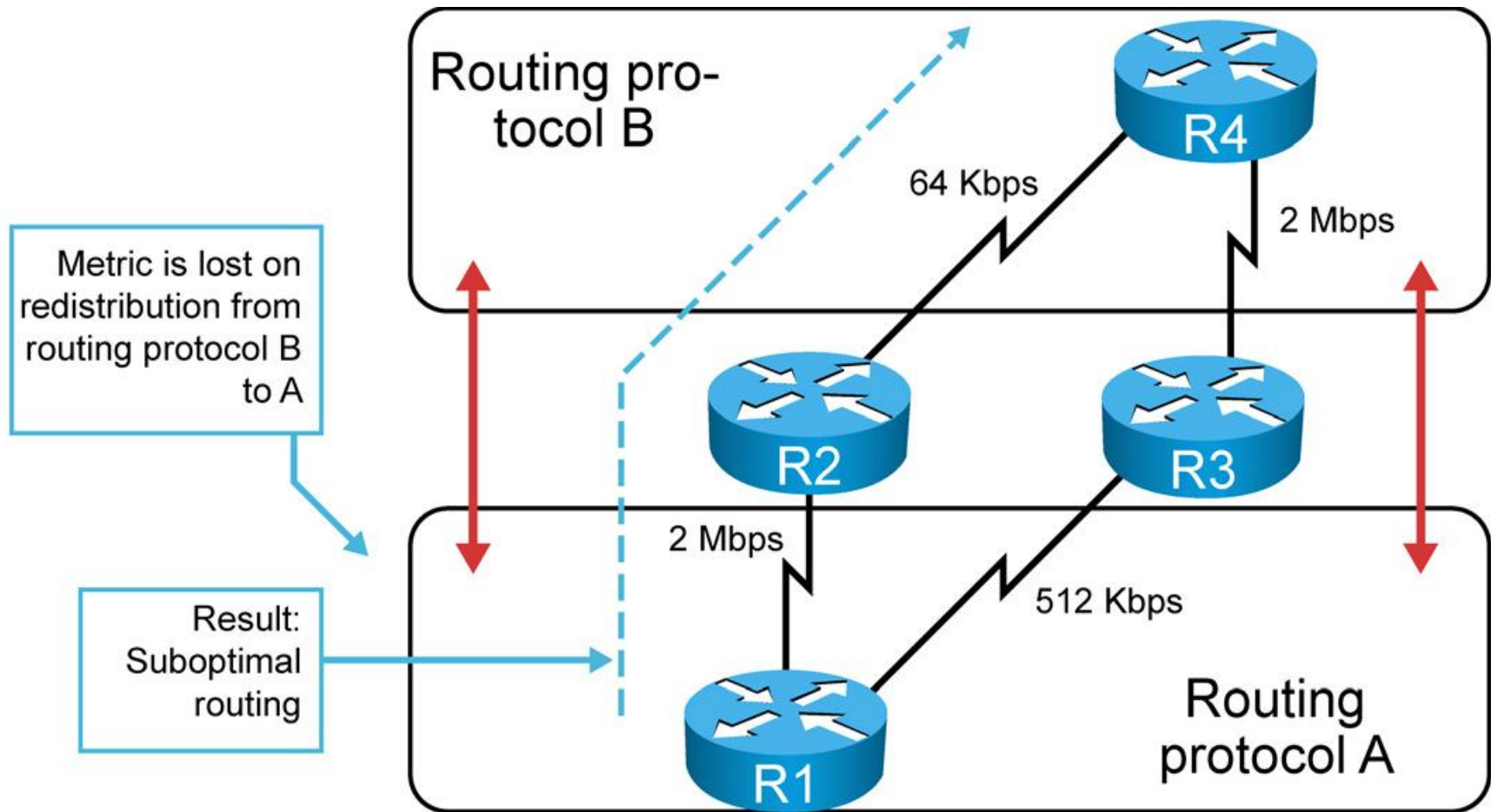
- Administrative distance
- Route maps
- Distribution lists
- Metrics manipulation



Redistribution: Issues ②



Redistribution: Issues ③



Redistribution: Configuration Note

- It is important to note that routes are redistributed **into** a routing protocol, so the **redistribute** command is configured **under** the routing process that is to **receive** the redistributed routes
 - Target routing protocol is **recipient** of routing information
 - Source protocol is **provider** of routing information
- **redistribute** command configures which **source** is used as a provider of routing information
- Redistribution on Cisco routers is configured always as a part of a **target** protocol

Redistribution into RIP

```
Router(config-router)# redistribute protocol [process-id]  
                        [match route-type] [metric metric-value] [route-map map-tag]
```

- Configuration options:

```
R1(config)# router rip  
R1(config-router)# redistribute ospf ?  
  
<1-65535>  Process ID  
R1(config-router)# redistribute ospf 1 ?  
  
match      Redistribution of OSPF routes  
metric     Metric for redistributed routes  
route-map  Route map reference  
...  
<cr>
```

- Default metric is infinity

The redistribute Command for RIP

Parameter	Description
<i>protocol</i>	Source protocol from which routes are redistributed.
<i>process-id</i>	For BGP or EIGRP, this value is an autonomous system number. For OSPF, this value is an OSPF process ID.
match <i>route-type</i>	(Optional) A parameter used when redistributing OSPF routes into another routing protocol. It is the criterion by which OSPF routes are redistributed into other routing domains. It can be any of the following: (internal, external NSSA-external, type 1,2).
metric <i>metric-value</i>	(Optional) A parameter used to specify the RIP seed metric for the redistributed route. If this value is not specified and no value is specified using the default-metric router configuration command, the default metric is 0 (interpreted as infinity). The metric for RIP is hop count.
route-map <i>map-tag</i>	(Optional) Specifies the identifier of a configured route map to be interrogated to filter the importation of routes from the source routing protocol to the current RIP routing protocol.

Redistributing into RIP: Example

```
R1(config)# router rip
R1(config-router)# redistribute ospf 1 metric 3
R1(config-router)#
```

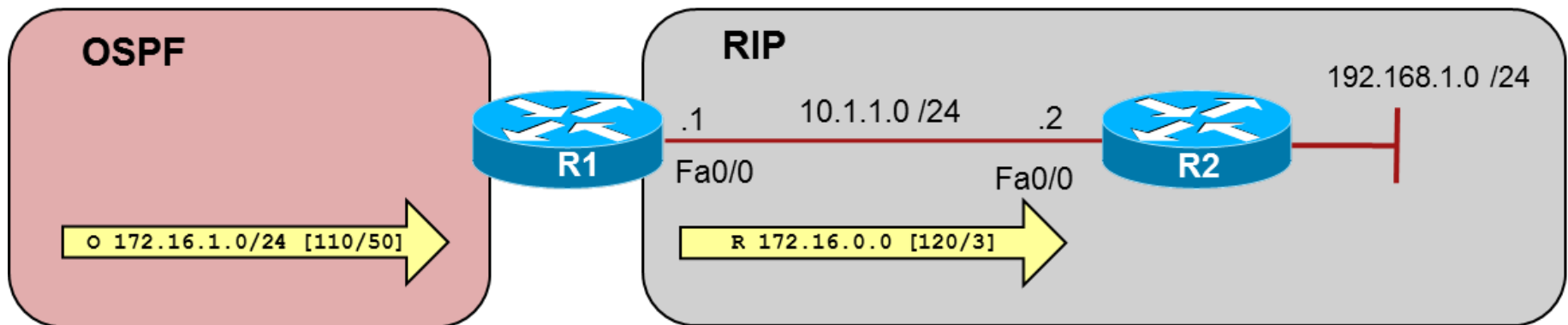


Table R1

```
C 10.1.1.0
R 192.168.1.0 [120/1]
O 172.16.1.0 [110/50]
```

Table R2

```
C 10.1.1.0
C 192.168.1.0
R 172.16.0.0 [120/3]
```

Redistributing into OSPF

```
Router(config-router)# redistribute protocol [process-id]
[metric metric-value] [metric-type type-value]
[route-map map-tag] [subnets] [tag tag-value]
```

- Default route CAN NOT be redistributed into OSPF
- Subnets are not redistributed by default (only classful networks)
 - **subnets** keyword MUST be used for subnets redistribution
- Default metric is 20 (BGP has default metric 1)
- Redistributed networks are sent as an external routes type 1 or 2 (LSA5 or LSA7), default is type 2

```
R1(config-router)# redistribute eigrp 100 ?
metric          Metric for redistributed routes
metric-type     OSPF/IS-IS exterior metric type for
                 redistributed routes
route-map       Route map reference
subnets        Consider subnets for redistribution into OSPF
tag             Set tag for routes redistributed into OSPF
...
<cr>
```

Redistributing into OSPF: Example

```
R1(config)# router ospf 1  
R1(config-router)# redistribute eigrp 100 subnets metric-type 1  
R1(config-router)#
```

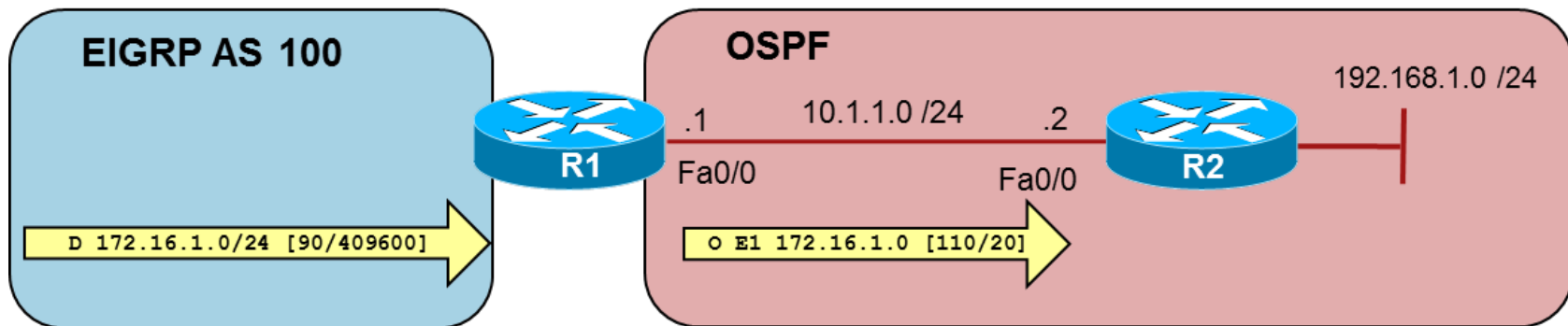


Table R1

```
C 10.1.1.0  
O 192.168.1.0 [110/20]  
D 172.16.1.0 [90/409600]
```

Table R2

```
C 10.1.1.0  
C 192.168.1.0  
O E1 172.16.0.0 [110/20]
```

Redistributing into EIGRP

```
Router(config-router)# redistribute protocol [process-id]
[match {internal | external 1 | external 2}]
[metric metric-value] [route-map map-tag]
```

- Configuration options:

```
R1(config)# router eigrp 100
R1(config-router)# redistribute ospf ?

<1-65535> Process ID
R1(config-router)# redistribute ospf 1 ?

match          Redistribution of OSPF routes
metric         Metric for redistributed routes
route-map      Route map reference
...
<cr>
```

- Default metric is infinity

Redistributing into EIGRP: Example

```
R1(config)# router eigrp 100
R1(config-router)# redistribute ospf 1 metric 10000 100 255 1 1500
R1(config-router)#
```

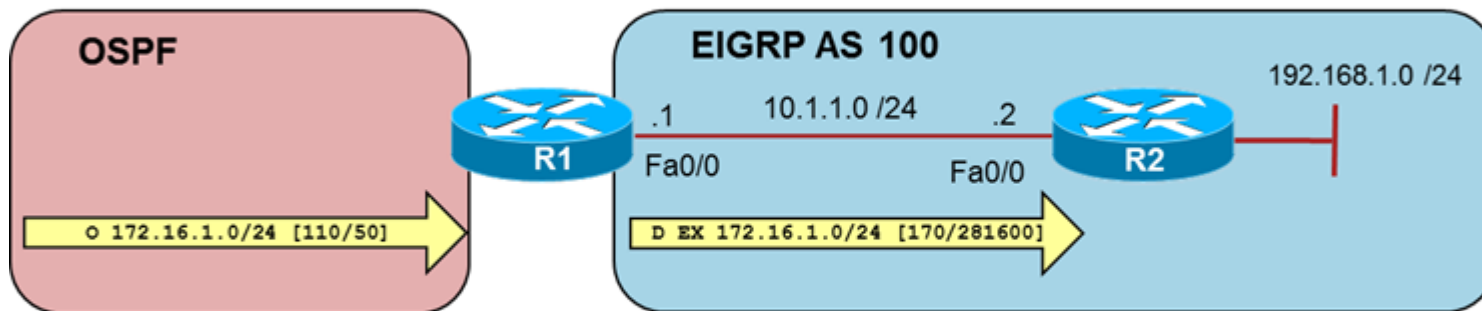


Table R1

```
C 10.1.1.0
D 192.168.1.0 [90/307200]
O 172.16.1.0 [110/50]
```

Table R2

```
C 10.1.1.0
C 192.168.1.0
D EX 172.16.1.0 [170/307200]
```

- **Bandwidth** in kilobits per second = 10000
- **Delay** in tens of microseconds = 100
- **Reliability** = 255 (maximum)
- **Load** = 1 (minimum)
- **MTU** = 1500 bytes

Default Metric

- Instead of setting default metric in the **redistribute** command, initial metric can be set using **default-metric**

```
Router(config-router) # default-metric number
```

- The *number* parameter is the value of the metric:
 - For RIP it is the number of hops
 - For OSPF it is the assigned cost
 - For EIGRP it is composite metric as on previous slide

- Example:

```
Router(config-router) #  
  redistribute rip metric 10000 1 255 1 1500  
!Same as following..  
Router(config-router) # default-metric 10000 1 255 1 1500  
Router(config-router) # redistribute rip
```

Default Metric: Example

```
R1(config)# router eigrp 100
R1(config-router)# default-metric 10000 100 255 1 1500
R1(config-router)# redistribute ospf 1
R1(config-router)#
```

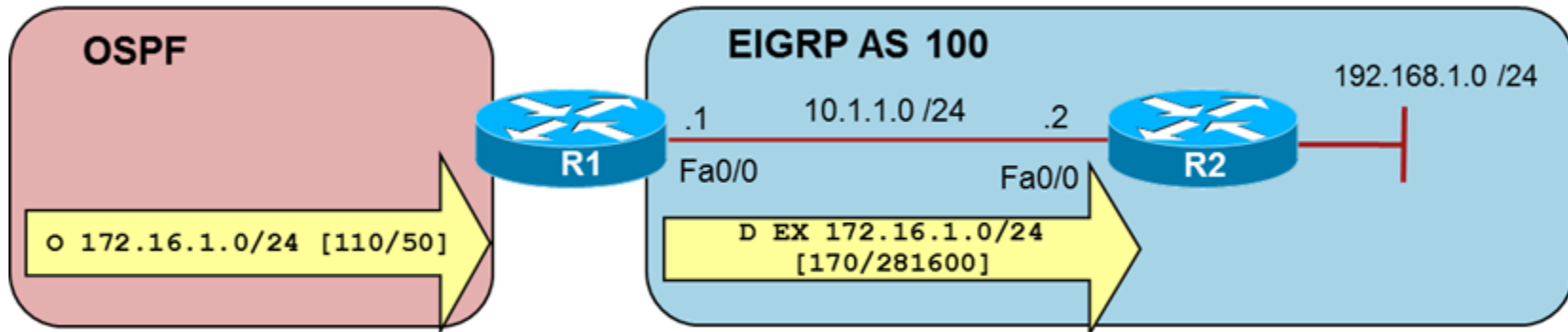


Table R1

```
C 10.1.1.0
D 192.168.1.0 [90/307200]
O 172.16.1.0 [110/50]
```

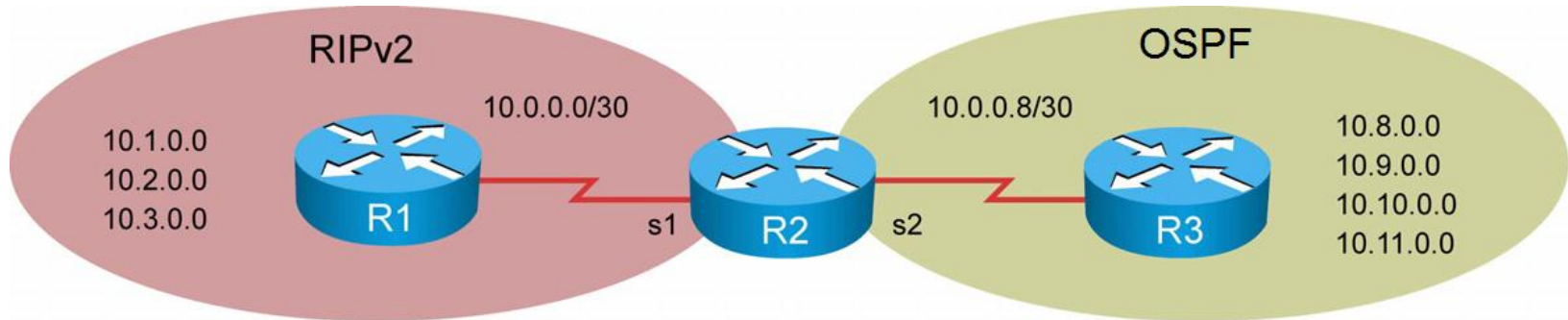
Table R2

```
C 10.1.1.0
C 192.168.1.0
D EX 172.16.1.0 [170/307200]
```

Redistribution Examples



Before Redistribution



R2#

```
router ospf 1
  network 10.0.0.8 0.0.0.3 area 0

router rip
  network 10.0.0.0
  version 2
  passive-interface s2
```

R1 Routing Table

C	10.0.0.0
R	10.1.0.0
R	10.2.0.0
R	10.3.0.0

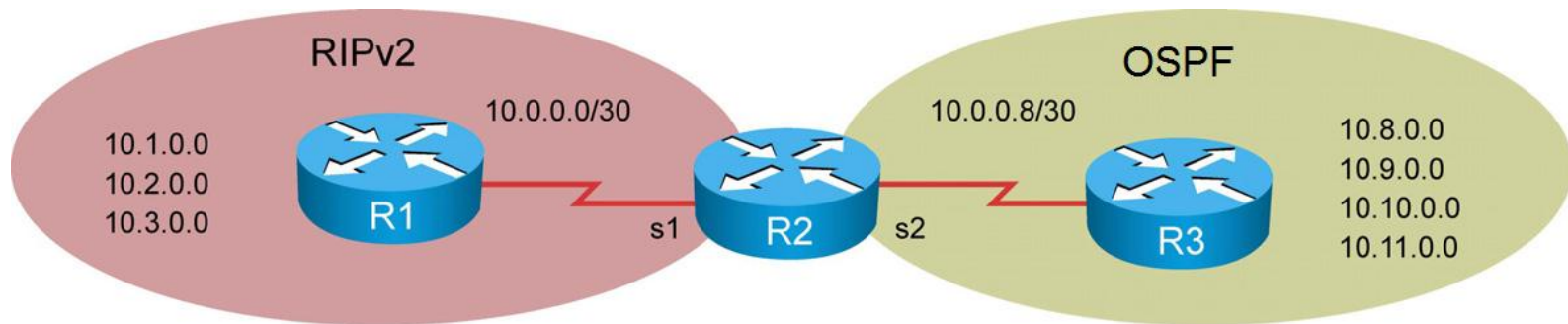
R2 Routing Table

C	10.0.0.0
C	10.0.0.8
R	10.1.0.0
R	10.2.0.0
R	10.3.0.0
O	10.8.0.0
O	10.9.0.0
O	10.10.0.0
O	10.11.0.0

R3 Routing Table

C	10.0.0.8
O	10.8.0.0
O	10.9.0.0
O	10.10.0.0
O	10.11.0.0

Configuration and After Redistribution State



R2#

```
router ospf 1
  network 10.0.0.8 0.0.0.3 area 0
  redistribute rip subnets metric 300

router rip
  network 10.0.0.0
  version 2
  passive-interface s2
  redistribute ospf 1 metric 5
```

R1 Routing Table

C	10.0.0.0
R	10.1.0.0
R	10.2.0.0
R	10.3.0.0
R	10.8.0.0
R	10.9.0.0
R	10.10.0.0
R	10.11.0.0

R2 Routing Table

C	10.0.0.0
C	10.0.0.8
R	10.1.0.0
R	10.2.0.0
R	10.3.0.0
O	10.8.0.0
O	10.9.0.0
O	10.10.0.0
O	10.11.0.0

R3 Routing Table

C	10.0.0.8
O E2	10.1.0.0
O E2	10.2.0.0
O E2	10.3.0.0
O	10.8.0.0
O	10.9.0.0
O	10.10.0.0
O	10.11.0.0

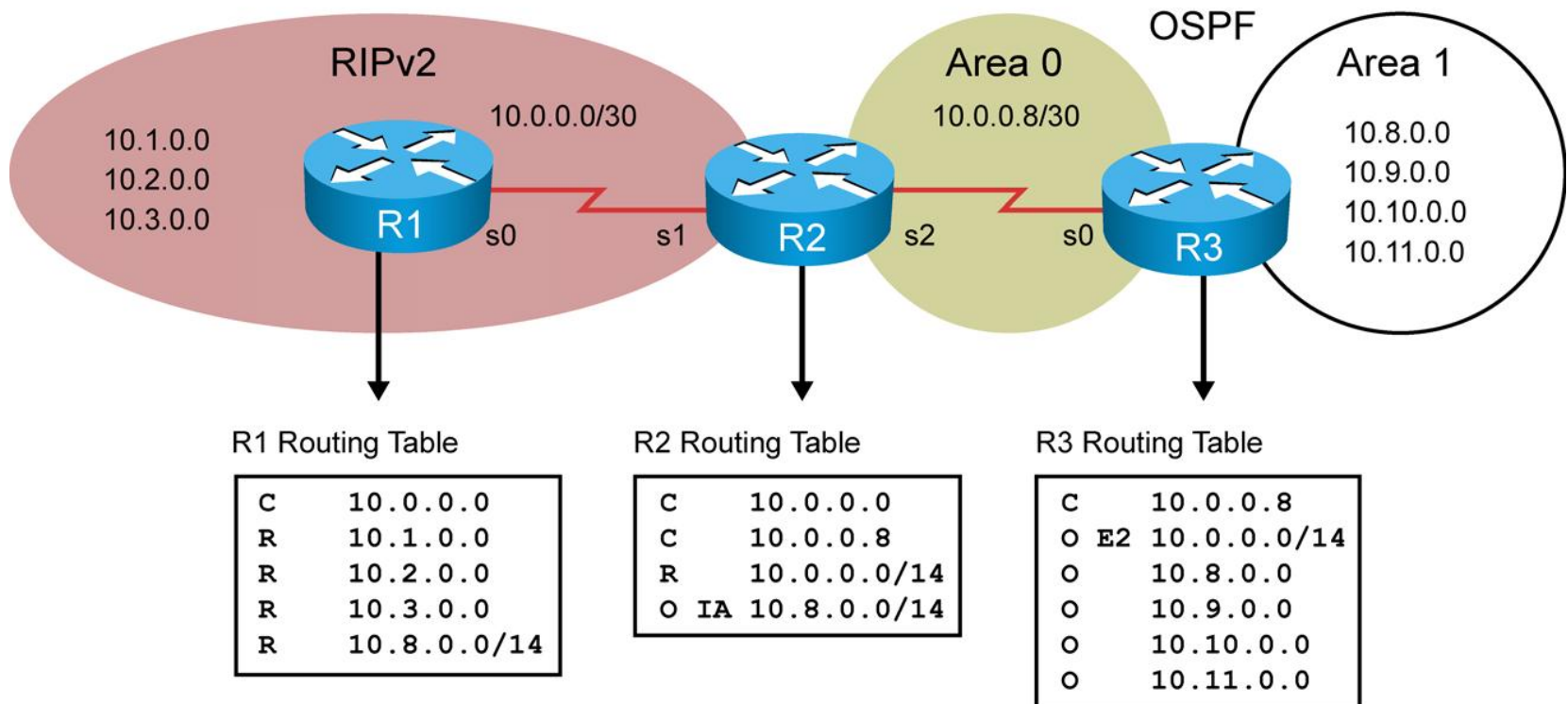
Route Summarization and Final State

R1#

```
interface s0
ip summary-address rip 10.0.0.0 255.252.0.0
```

R3#

```
router ospf 1
area 1 range 10.8.0.0 255.252.0.0
```



Modifying Administrative Distance



Modifying Administrative Distance

- Used in more complicated redistributions
- Proper planning is necessary

```
Router(config-router)# distance administrative-distance  
[address wildcard-mask [access-list-number | NAME]]
```

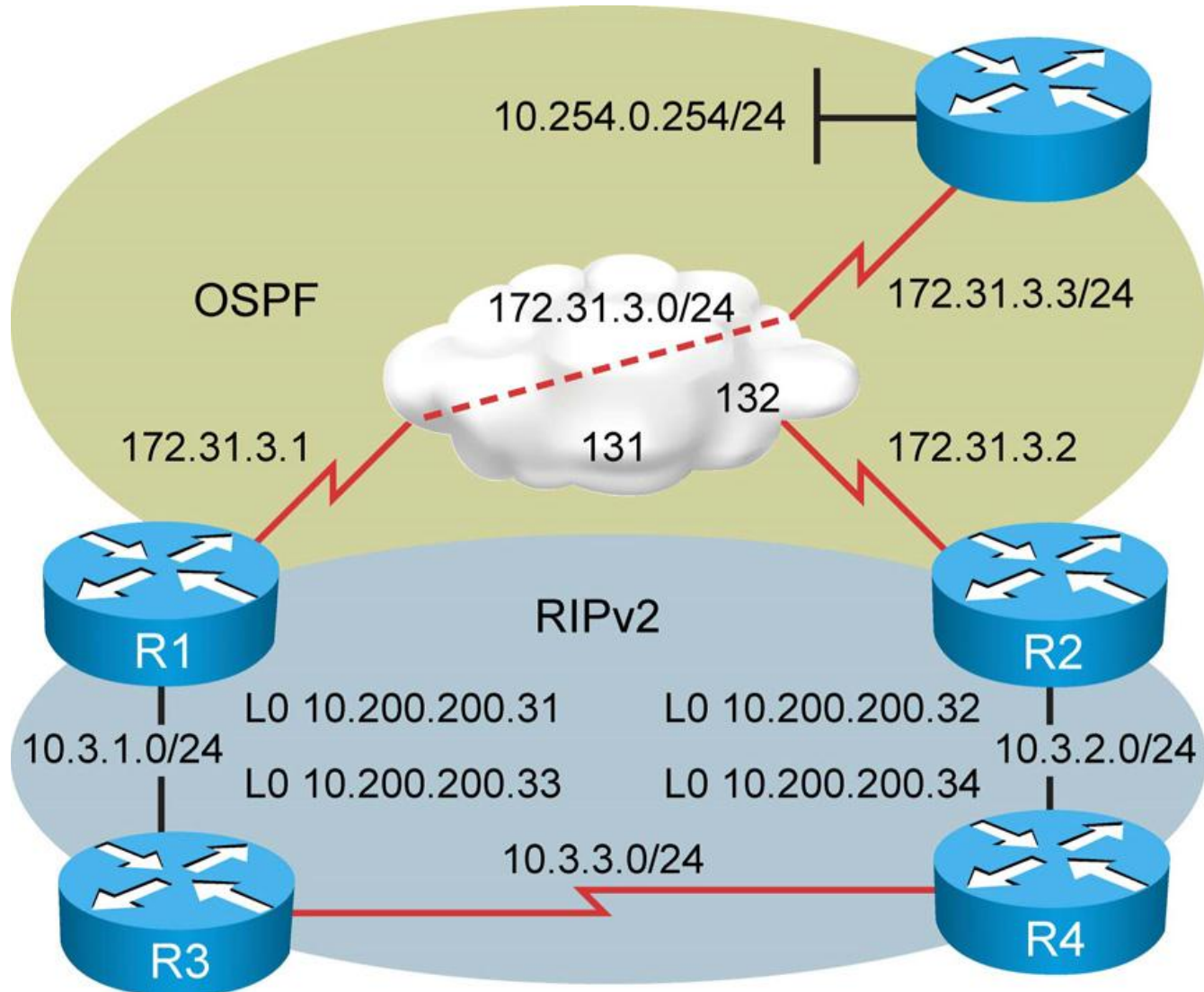
- Usage for OSPF

```
Router(config-router)# distance ospf {[intra-area dist1]  
[inter-area dist2] [external dist3]}
```

- Usage for EIGRP

```
Router(config-router)# distance eigrp internal external
```


Topology



Routers Configurations

R1#

```
router ospf 1
  redistribute rip metric 10000 metric-type 1 subnets
  network 172.31.0.0 0.0.255.255 area 0
!
router rip
  version 2
  redistribute ospf 1 metric 5
  network 10.0.0.0
  no auto-summary
```

R2#

```
router ospf 1
  redistribute rip metric 10000 metric-type 1 subnets
  network 172.31.3.2 0.0.0.0 area 0
!
router rip
  version 2
  redistribute ospf 1 metric 5
  network 10.0.0.0
  no auto-summary
```

R2 Routing Table After Redistribution

With OSPF and RIP running



```
R2#show ip route
<Output Omitted>
```

```
Gateway of last resort is not set
```

```
    172.31.0.0/24 is subnetted, 7 subnets
```

```
O    172.31.55.0 [110/2343] via 172.31.3.3, 00:09:46, Serial0/0/0
```

```
C    172.31.3.0 is directly connected, Serial0/0/0
```

```
O    172.31.2.0 [110/1562] via 172.31.3.3, 00:09:46, Serial0/0/0
```

```
10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
```

```
O E1  10.3.1.0/24 [110/10781] via 172.31.3.1, 00:09:47, Serial0/0/0
```

```
O E1  10.3.3.0/24 [110/10781] via 172.31.3.1, 00:04:51, Serial0/0/0
```

```
C    10.3.2.0/24 is directly connected, fastethernet0/0
```

```
O E1  10.200.200.31/32 [110/10781] via 172.31.3.1, 00:09:48, Serial0/0/0
```

```
O E1  10.200.200.34/32 [110/10781] via 172.31.3.1, 00:04:52, Serial0/0/0
```

```
C    10.200.200.32/32 is directly connected, Loopback0
```

```
O E1  10.200.200.33/32 [110/10781] via 172.31.3.1, 00:04:52, Serial0/0/0
```

```
O E2  10.254.0.0/24 [110/50] via 172.31.3.3, 00:09:48, Serial0/0/0
```

- R2 includes suboptimal paths and loops.


Modifying Administrative Distance

```
hostname R1
!
router ospf 1
  redist rip metric 10000 metric-t 1 subnets
  network 172.31.0.0 0.0.255.255 area 0
  distance 125 0.0.0.0 255.255.255.255 64
!
router rip
  version 2
  redistribute ospf 1 metric 5
  network 10.0.0.0
  no auto-summary
!
access-list 64 permit 10.3.1.0 0.0.0.255
access-list 64 permit 10.3.3.0 0.0.0.255
access-list 64 permit 10.3.2.0 0.0.0.255
access-list 64 permit 10.200.200.31
access-list 64 permit 10.200.200.34
access-list 64 permit 10.200.200.32
access-list 64 permit 10.200.200.33
```

```
hostname R2
!
router ospf 1
  redist rip metric 10000 metric-t 1 subnets
  network 172.31.3.2 0.0.0.0 area 0
  distance 125 0.0.0.0 255.255.255.255 64
!
router rip
  version 2
  redistribute ospf 1 metric 5
  network 10.0.0.0
  no auto-summary
!
access-list 64 permit 10.3.1.0 0.0.0.255
access-list 64 permit 10.3.3.0 0.0.0.255
access-list 64 permit 10.3.2.0 0.0.0.255
access-list 64 permit 10.200.200.31
access-list 64 permit 10.200.200.34
access-list 64 permit 10.200.200.32
access-list 64 permit 10.200.200.33
```


R2 Routing Table After Modifying AD

With OSPF changing administrative distance



Gateway of last resort is not set

	172.31.0.0/16 is variably subnetted, 8 subnets, 2 masks
O	172.31.55.4/32 [110/781] via 172.31.33.4, 00:00:01, Serial10/0/0
C	172.31.33.0/24 is directly connected, Serial10/0/0
O	172.31.33.1/32 [110/1562] via 172.31.33.4, 00:00:01, Serial10/0/0
O	172.31.33.4/32 [110/781] via 172.31.33.4, 00:00:01, Serial10/0/0
O	172.31.44.4/32 [110/781] via 172.31.33.4, 00:00:01, Serial10/0/0
O	172.31.22.4/32 [110/781] via 172.31.33.4, 00:00:01, Serial10/0/0
O	172.31.11.4/32 [110/781] via 172.31.33.4, 00:00:03, Serial10/0/0
O	172.31.66.4/32 [110/781] via 172.31.33.4, 00:00:03, Serial10/0/0
	10.0.0.0/8 is variably subnetted, 8 subnets, 2 masks
R	10.3.1.0/24 [120/2] via 10.3.2.4, 00:00:03, FastEthernet0/0
R	10.3.3.0/24 [120/1] via 10.3.2.4, 00:00:03, FastEthernet0/0
C	10.3.2.0/24 is directly connected, FastEthernet0/0
R	10.200.200.31/32 [120/3] via 10.3.2.4, 00:00:04, FastEthernet0/0
R	10.200.200.34/32 [120/1] via 10.3.2.4, 00:00:04, FastEthernet0/0
C	10.200.200.32/32 is directly connected, Loopback0
R	10.200.200.33/32 [120/2] via 10.3.2.4, 00:00:04, FastEthernet0/0
O E2	10.254.0.0/24 [110/50] via 172.31.33.4, 00:00:04, Serial10/0/0

- Router R2 prefers RIP routes.

Controlling Routing Update Traffic



Controlling Routing Update Traffic

- Several ways how to control routing updates
- **Passive interfaces**
 - No routing updates are sent through this kind of interface
- **Distribute list**
 - A distribute list allows a filter (ACL or prefix-list) to be applied to routing updates
- **Route-map**
 - Conditions CAN be tested against a route in routing update. Actions CAN be taken to modify attributes of the packet or route.
 - Some routing protocols allow using route-map even in distributed list or neighbor command

The `passive-interface` Command ①

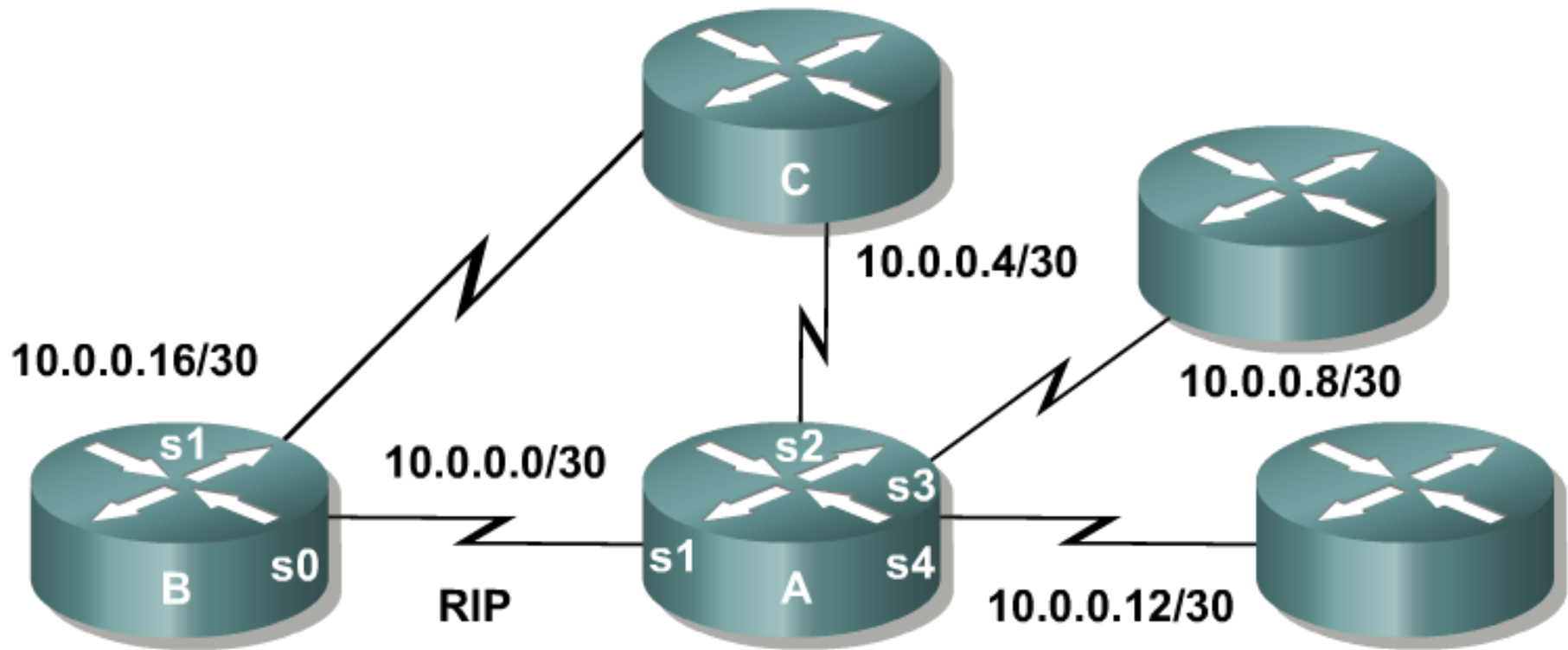
- Passive interface prevents routing updates from being sent through an interface.
 - Passive interface configured for `RIP` protocol however accepts `RIP packets` – IF this is undesired behavior THEN ACL MUST be used
 - Passive interface configured for routing protocols using Hello packets (EIGRP, OSPF, IS-IS) `doesn't receive or send packets`

Routing protocol	Suppresses outgoing routing updates	Suppresses incoming routing updates	Stops neighbor adjacency
RIP	✓		
EIGRP	✓	✓	✓
OSPF	✓	✓	✓
IS-IS	✓	✓	✓

The `passive-interface` Command ②

- This command is used to set either a particular interface or all router interfaces to passive
 - Set particular interface as a passive interface using `passive-interface` *IFACE*
 - Set all interface using `passive-interface default` and then enable particular interface using `no passive-interface` *IFACE*

The passive-interface Command ③



Router B Configuration

```
router rip
 network 10.0.0.0
 passive-interface s1
```

Router A Configuration

```
router rip
 network 10.0.0.0
 passive-interface default
 no passive-interface s1
```

Distribute List ①

- Distribute list allows filtering to be applied to routing updates so unwanted networks can go through
 - Distribute list makes reference to an access list or a prefix list
- Different behavior for different IGP
 - For distance-vector protocols (RIP, EIGRP, IGRP)
 - Distribute list CAN filter any sent or received network
 - OSPF **in**
 - Routes CAN NOT be filtered from entering the OSPF LSDB, they are only filtered from entering the routing table
 - OSPF **out**
 - Only external routes redistributed by ASBR CAN be filtered, but NOT intra-area or inter-area routes

Distribute List ②

- Configuration snippet:

```
Router(config-router)# distribute-list  
    { 1-199 | 1300-2699 | NAME | prefix NAME | route-map NAME }  
    {in | out} [interface]
```

- Parameters:

- Reference to ACL number, named ACL, prefix-list or route-map
 - **in/out**: applies the distribution list to incoming/outgoing direction or interface
- NOT all parameters or combination of parameters are valid

Distribute List together with ACL

- According to the ACL type, different field in a packet is checked
 - Standard ACL checks only network IP address
 - Extended ACL checks next-hop (source part in the ACL) and destination IP network address (destination part in the ACL)
 - Useful for debugging is rule: **permit ip any any log**
- Actions:
 - **permit**: permits sending or receiving given network
 - **deny**: denied sending or receiving given network
- Example:

“Remove all networks 10..*.* and network 192.168.1.0”*

```
access-list 1 deny 10.0.0.0 0.255.255.255
access-list 1 deny 192.168.1.0 0.0.0.0
access-list 1 permit any
```

- Example:

“Only network 172.16..* is allowed from next-hop address 1.2.3.4”*

```
access-list 100 permit ip host 1.2.3.4 172.16.0.0 0.0.255.2
```

Distribute List together with Prefix List ①

- Using ACL in distribute list as route filter has several drawbacks
- Prefix list allows same filtering but with performance improvement and more user-friendly command-line interface
- **Prefix list** is list of network prefixes together with netmasks and permit/deny rules

```
Router(config)# ip prefix-list NAME [seq N] {permit | deny}  
                NETWORK/LENGTH [ge D] [le U]
```

- Parameters:
 - **seq** *N*: sequence number of prefix-list statement
 - *NETWORK/LENGTH*: prefix to be matched and the length of the prefix
 - **ge** *D*: network mask has to be at least /*D* long
 - **le** *U*: network mask has to be at most /*U* long
 - *LENGTH* <= *D* <= *U* <= 32

Distribute List together with Prefix List ②

- Example:

```
(conf)# ip prefix-list PL permit 192.0.2.0/26 ge 28 le 31
```

- Satisfy every network A.B.C.D/M that meets following requirements
 - $A.B.C.D \& 255.255.255.192 = 192.0.2.0$
 - $255.255.255.240 \leq M \leq 255.255.255.254$
 - An exact match is assumed when neither **ge** nor **le** is specified
- *Which PL matches all routes?*

```
ip prefix-list PL_ALL permit 0.0.0.0/0 le 32
```

Distribute List: Prefix List vs. ACL

- Prefix lists are more effective than ACL – PL is internally transformed into tree structure
- PL should be used for filtering routing updates instead of ACL
- PL is only useful for routing information control management
 - CAN NOT be used as an ACL replacement for e.g. control user traffic
- Prefix list and ACL ends with implicit `deny any` rule
 - Every network that is NOT allowed is rejected

Route-map

- **Route-maps** work like ACL but more sophisticated – they allow some conditions to be tested against the packet or route using `IF-THEN-ELSE` rules
- Each route-map consists of several statements
 - Route-map statement has `test`, `change`, `action` blocks
 - `IF` statement's conditions are matched `THEN` permit/deny is executed and some actions `MAY` be taken to modify attributes of the packet `ELSE` next statement is going to be evaluated
- Processing is similar to ACL
 - The route-map statements are executed in a sequence
 - First match execute the action together with changes and route-map evaluation ends
 - **Implicit default** `match any / deny` **at the end of a route-map**
 - Route-map statements are numbered

Route-map Applications

- **Route filtering during redistribution**

- Route maps offer an easy way how to manipulate routing metrics

- **Policy-based routing (PBR)**

- PBR allows the operator to define routing policy other than basic destination-based routing using the routing table

- **BGP**

- Route maps are the primary tools for implementing BGP policy and manipulation of BGP path attributes

- **NAT**

- Route maps have better control which private addresses are translated to public addresses

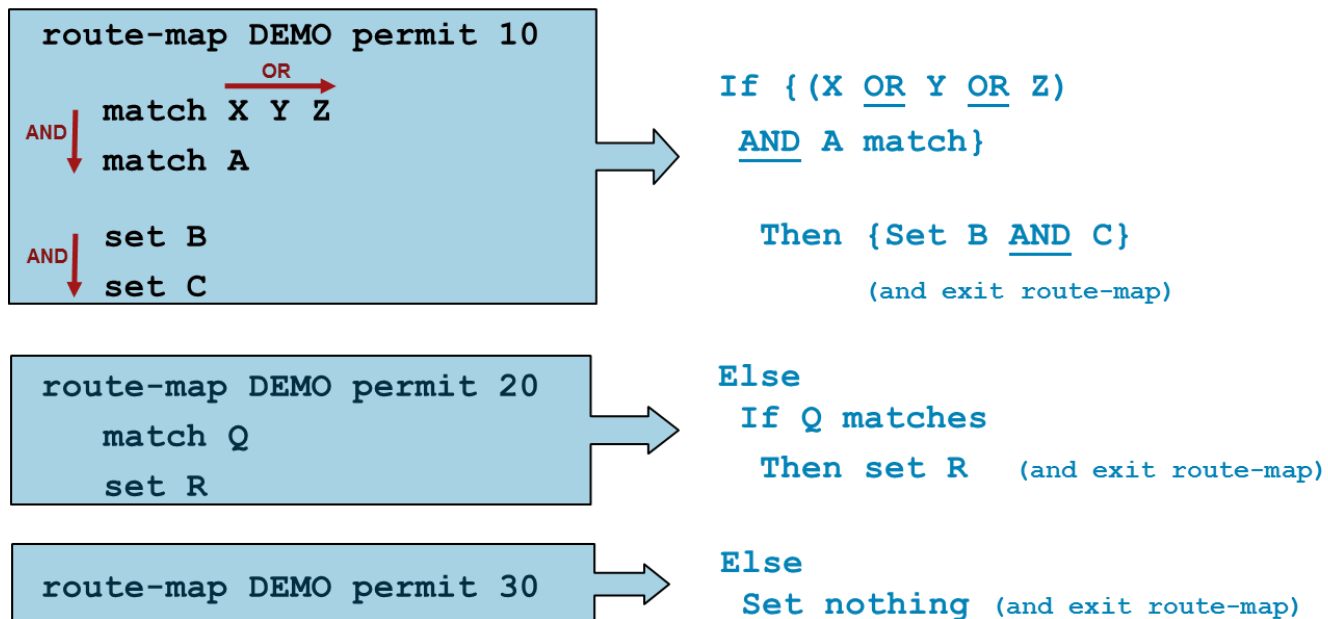
Route-map Statement

- Route-map statement has several blocks
 - Optional part for testing the condition (**match**), optional part for change the packet (**set**) and mandatory part for action (**permit/deny**)
- Blocks are checked from line to line (**top-down**) looking for a match
- Each line is sequence-numbered

```
route-map my_bgp permit 10
    { match statements }
    { match statements }
    { set statements }
    { set statements }
route-map my_bgp deny 20
    ::          ::  ::
    ::          ::  ::
route-map my_bgp permit 30
    ::          ::  ::
    ::          ::  ::
```

Route-map Statement

- Multiple match criteria in the same line are allowed
 - The criteria are processed using **OR logic**
- Separate match criteria can also be applied vertically under a route map line
 - In this case, each match uses **AND logic**
- Successful match executes the permit/deny action



The route-map command

- Defines the route map statement action

```
Router(config) #  
    route-map map-tag [permit | deny] [sequence-number]
```

- Defines the conditions to match

```
Router(config-route-map) # match {conditions}
```

- Defines the action to be taken on a match

```
Router(config-route-map) # set {actions}
```

- Allows for detailed control of routes being redistributed into a routing protocol

```
Router(config-router) #  
    redistribute protocol [process-id] route-map map-tag
```

The match command

- The **match** commands specify criteria to be matched
- The associated route map statement permits or denies the matching routes

Command	Description
match community	Matches a BGP community
match interface	Matches any routes that have the next hop out of one of the interfaces specified
match ip address	Matches any routes that have a destination network number address that is permitted by an ACL or prefix list
match ip next-hop	Matches any routes that have a next-hop router address that is passed by one of the ACLs specified
match ip route-source	Matches routes that have been advertised by routers and access servers at the address that is specified by the ACLs
match length	Matches based on the layer 3 length of a packet
match metric	Matches routes with the metric specified
match route-type	Matches routes of the specified type
match tag	Matches tag of a route

The set Command

- The **set** commands modify matching routes (parameters of redistributed routes)

Command	Description
set as-path	Modifies an AS path for BGP routes
set automatic-tag	Computes automatically the tag value
set community	Sets the BGP communities attribute
set default interface	Indicates where to output packets that pass a match clause of a route map for policy routing and have no explicit route to the destination
set interface	Indicates where to output packets that pass a match clause of a route map for policy routing
set ip default next-hop	Indicates where to output packets that pass a match clause of a route map for policy routing and for which the Cisco IOS software has no explicit route to a destination
set ip next-hop	Indicates where to output packets that pass a match clause of a route map for policy routing
set level	Indicates where to import routes for IS-IS and OSPF
set local-preference	Specifies a BGP local preference value
set metric	Sets the metric value for a routing protocol
set metric-type	Sets the metric type for the destination routing protocol
set tag	Sets tag value for destination routing protocol
set weight	Specifies the BGP weight value

Route-map: Example

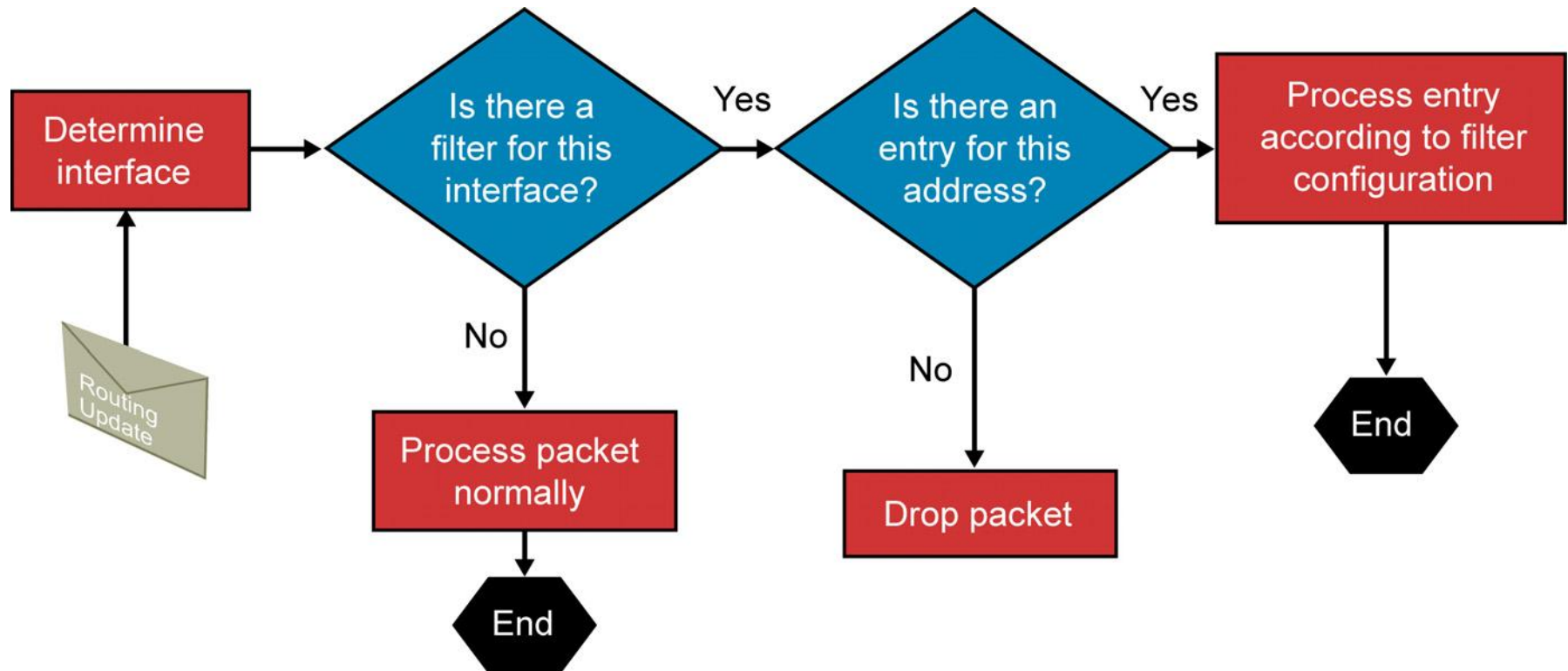
```
R1(config)# access-list 23 permit 10.1.0.0 0.0.255.255
R1(config)# access-list 29 permit 172.16.1.0 0.0.0.255
R1(config)# access-list 37 permit 10.0.0.0 0.255.255.255
R1(config)#
R1(config)# route-map REDIS-RIP permit 10
R1(config-route-map)# match ip address 23 29
R1(config-route-map)# set metric 500
R1(config-route-map)# set metric-type type-1
R1(config-route-map)#
R1(config-route-map)# route-map REDIS-RIP deny 20
R1(config-route-map)# match ip address 37
R1(config-route-map)#
R1(config-route-map)# route-map REDIS-RIP permit 30
R1(config-route-map)# set metric 5000
R1(config-route-map)# set metric-type type-2
R1(config-route-map)#
R1(config-route-map)# router ospf 10
R1(config-router)# redistribute rip route-map REDIS-RIP subnets
R1(config-router)#
```

- Routes matching either access list 23 or 29 are redistributed with an OSPF cost of 500, external type 1
- Routes permitted by access list 37 are not redistributed
- All other routes are redistributed with an OSPF cost metric of 5000, external type 2

Route-map together with ACL

- Route-map used for redistribution often use ACL for network filtering
 - Network = IP address and netmask
- Once again different behavior according to ACL type
- Redistributed networks are matched according to ACL type
 - Standard ACL matches only IP network address not the netmask
 - Extended ACL matches IP network address and netmask
- *This mess only leads to recommendation of using prefix-lists instead of ACLs for network filtering ☺*

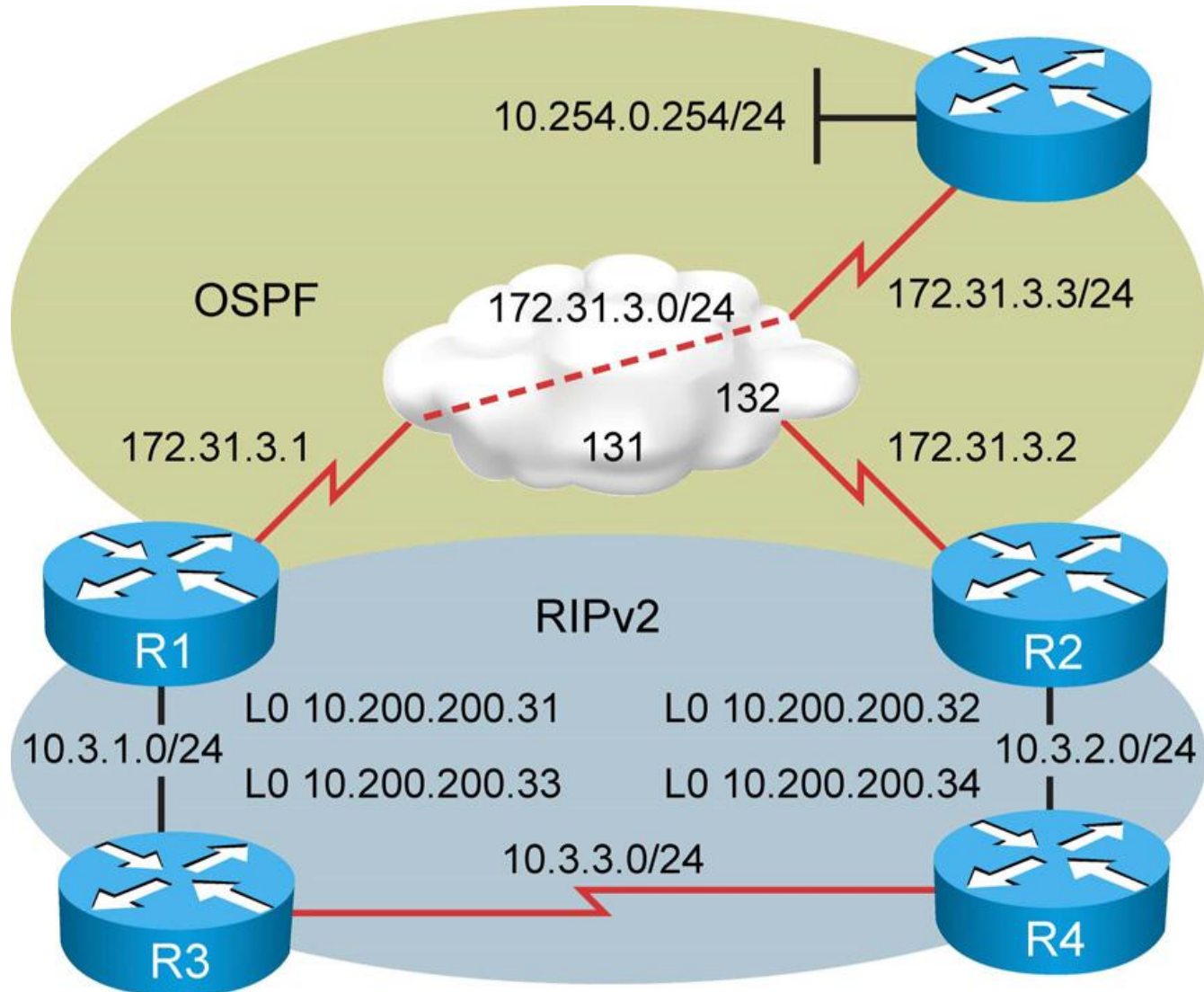
Filtering Routing Updates



Loop Prevention with Tagging



Topology



Tagging Redistributed Direction

```
hostname R1
!
router ospf 1
  redistribute rip metric 10000 metric-type 1
    route-map r2o subnets
  network 172.31.0.0 0.0.255.255 area 0
!
router rip
  version 2
  redistribute ospf 1 metric 5 route-map o2r
  network 10.0.0.0
  no auto-summary
!
route-map r2o deny 10
  match tag 110

route-map r2o permit 20
  set tag 120

route-map o2r deny 10
  match tag 120

route-map o2r permit 20
  set tag 110
```

```
hostname R2
!
router ospf 1
  redistribute rip metric 10000 metric-type 1
    route-map r2o subnets
  network 172.31.3.2 0.0.0.0 area 0
!
router rip
  version 2
  redistribute ospf 1 metric 5 route-map o2r
  network 10.0.0.0
  no auto-summary
!
route-map r2o deny 10
  match tag 110

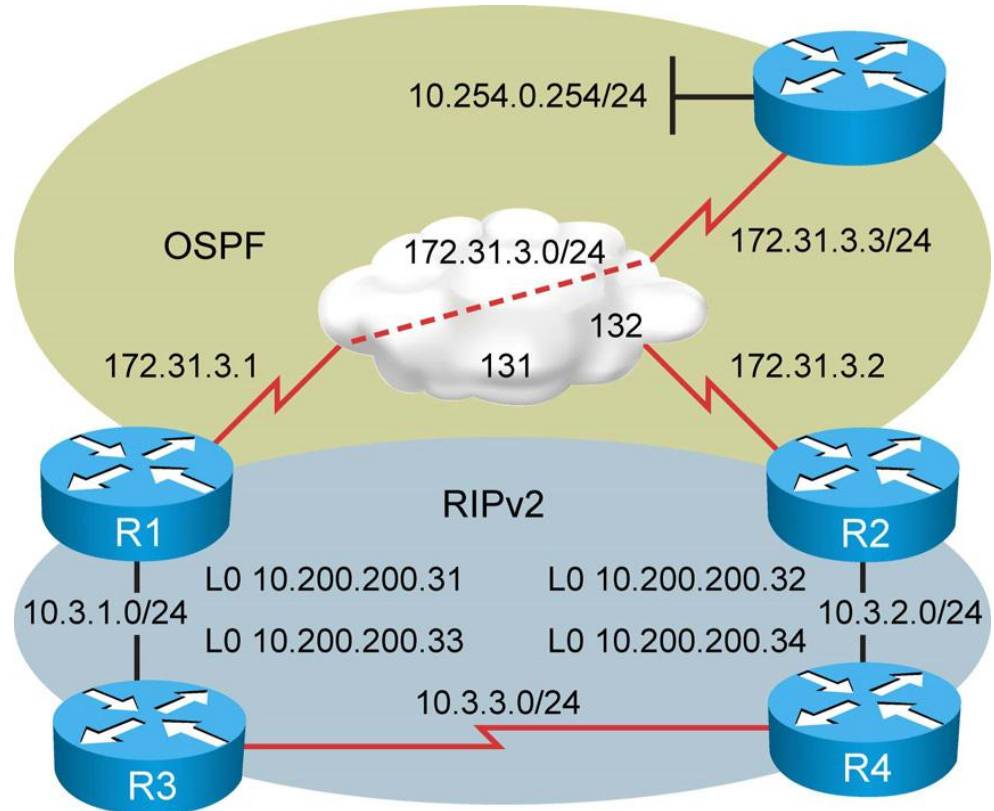
route-map r2o permit 20
  set tag 120

route-map o2r deny 10
  match tag 120

route-map o2r permit 20
  set tag 110
```

Know Your Network

- IF the network from one IGP is distributed in other IGP with higher AD as in previous IGP THEN routing loops MAY occur
- Be very familiar with your network BEFORE implementing redistribution
- Focus on routers border routers with redundant paths
- Make sure no path information is lost when using the **distance** command



Offset List



Offset List

- Offset list can increase the metric of given network in RIP or EIGRP protocols
 - Network metric can be increased for announced or received networks
 - Offset list can be used for routing protocol or for announced/received networks on particular interface
- Created by **offset-list** referencing an ACL
 - Offset value is added to RIP metric or in case of EIGRP to Delay parameter

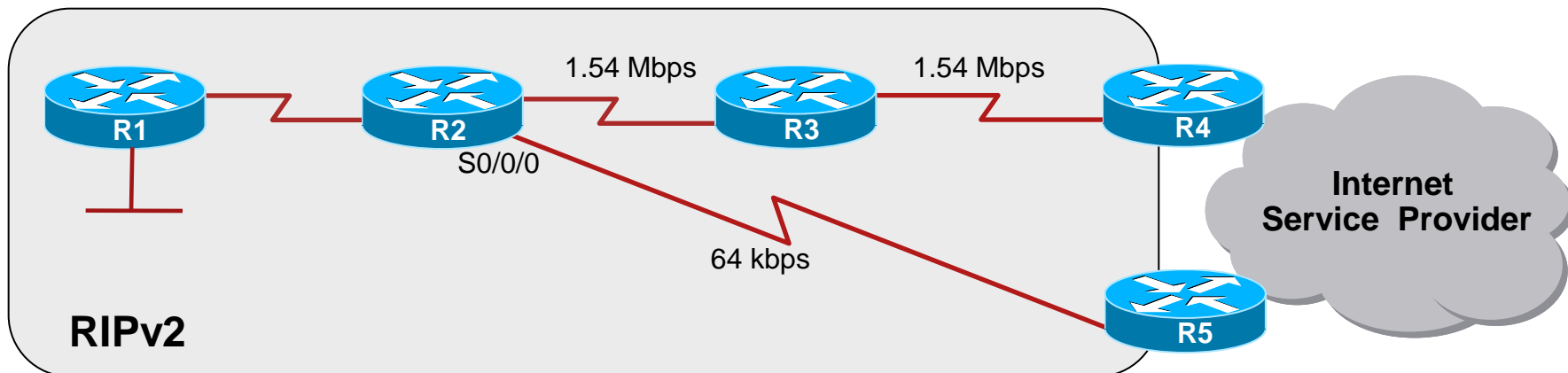
The offset-list Command

- Offset list is configured under routing protocol subsection:

```
Router(config-router)# offset-list {acl-number | acl-name}  
{in | out} offset [interface-type interface-number]
```

Parameter	Description
<i>acl-number</i> <i>aclname</i>	Standard access list number or name to be applied. Access list number 0 indicates all networks.
in	Offset list is use for incoming RIP/EIGRP messages
out	Offset list is use for outgoing RIP/EIGRP messages
<i>offset</i>	Positive offset to be applied to metrics for networks matching the access list. If the offset is 0, no action is taken.
<i>interface-type</i> <i>interface-number</i>	(Optional) Interface type and number to which the offset list is applied.

Offset List: Example



```
R2 (config)# access-list 21 permit 192.0.2.0 0.0.0.255
R2 (config)# router rip
R2 (config-router)# offset-list 21 in 2 serial 0/0/0
```

- Two paths exist for end devices in the R1 LAN network to the ISP network 192.0.2.0/24
- R5 has lower metric, however with very slow link
- Offset-list can increase the metric so the upper path through R3 will be preferred
 - Offset-list increases the metric of 192.0.2.0/24 network received from R5 by 2

IP Policy Based Routing

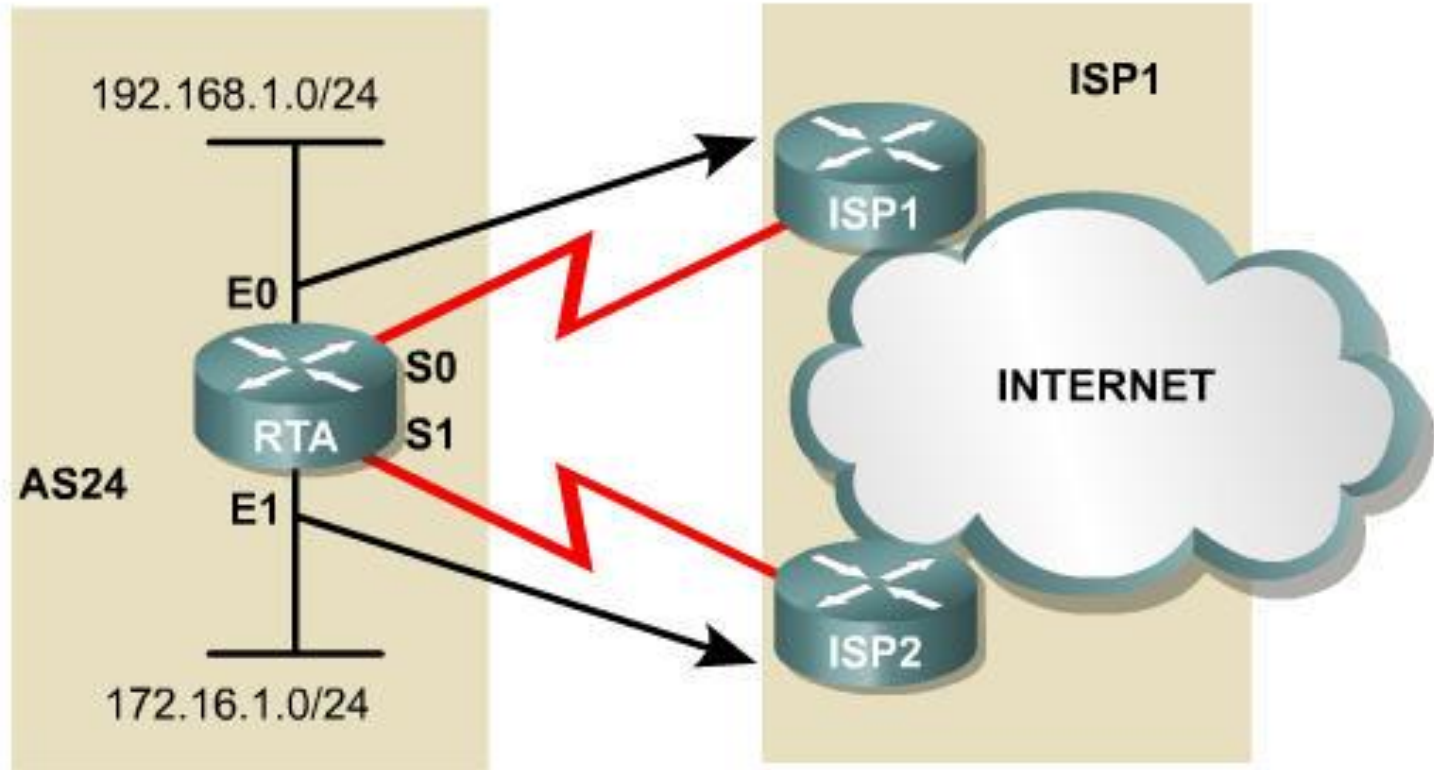


Policy Based Routing

- **PBR** allows administrator to define a routing policy other than basic destination-based routing using the routing table
 - Source and destination address match
 - Protocol type, size of packets ...
- PBR uses route-map
 - **match** selects the packets
 - Permit action – packets are forwarded **according to PBR**
 - **set** defines how the packet should be forwarded
 - **set ip next-hop, set interface,**
set ip default next-hop, set default interface
 - Deny action – packets are forwarded **according to the routing table**
- Route-map is applied on **ingress interface**

```
Router(config-if) # ip policy route-map NAME
```

Example of PBR



Policy routing on RTA can be used to route traffic based on its source network, in addition to its destination. Traffic from the 192.168.1.0 network will use the link to ISP1. Traffic from the 172.16.1.0 network will use the link to ISP2.

PBR Configuration

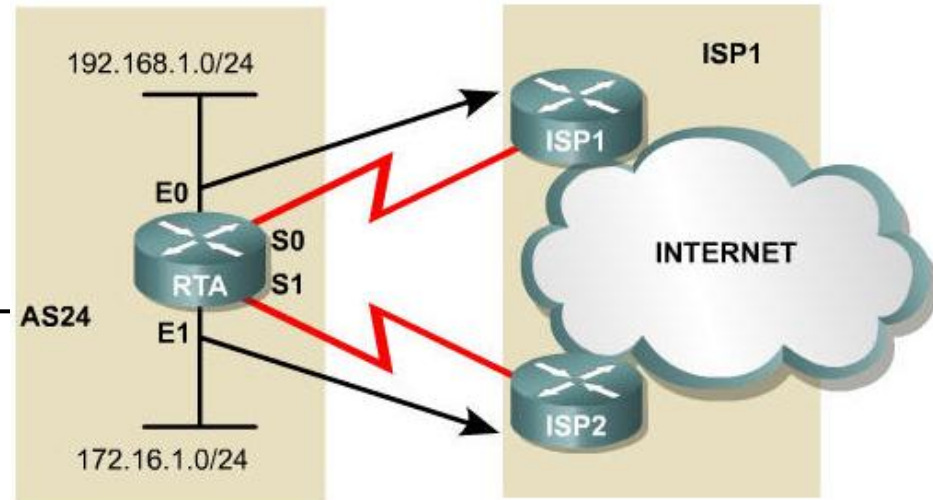
```
hostname RTA
!
interface Ethernet 0
  ip address 192.168.1.1 255.255.255.0
  ip policy route-map ISP1

interface Ethernet 1
  ip address 172.16.1.1 255.255.255.0
  ip policy route-map ISP2

route-map ISP1 permit 10
  match ip address 1
  set interface Serial0

route-map ISP2 permit 10
  match ip address 2
  set interface Serial1

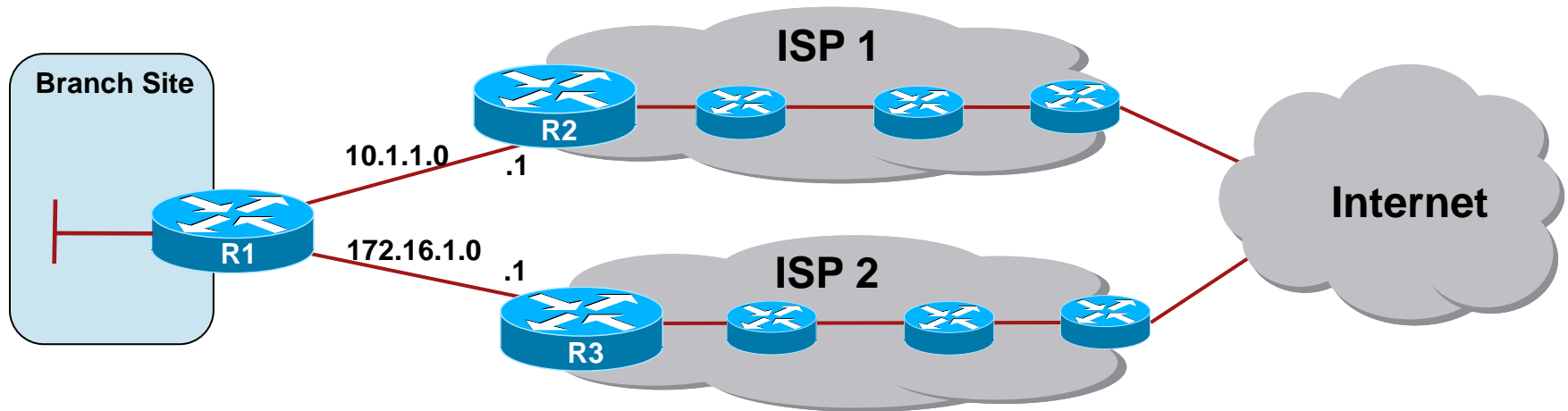
access-list 1 permit 192.168.1.0 0.0.0.255
access-list 2 permit 172.16.1.0 0.0.0.255
```



IP Service Level Agreement (IP SLA)

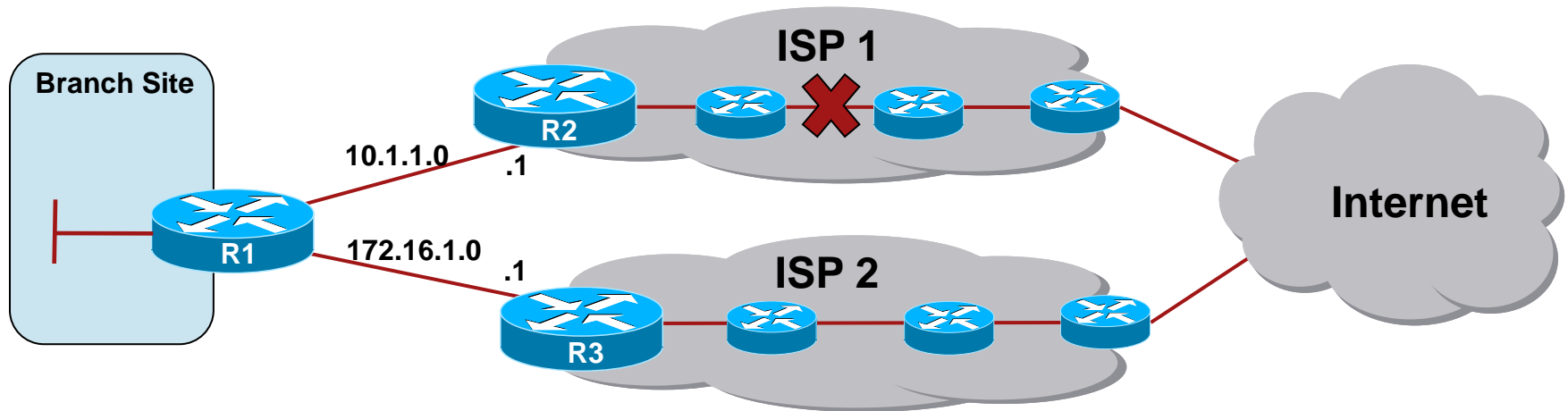


Multi-homed Connection ①



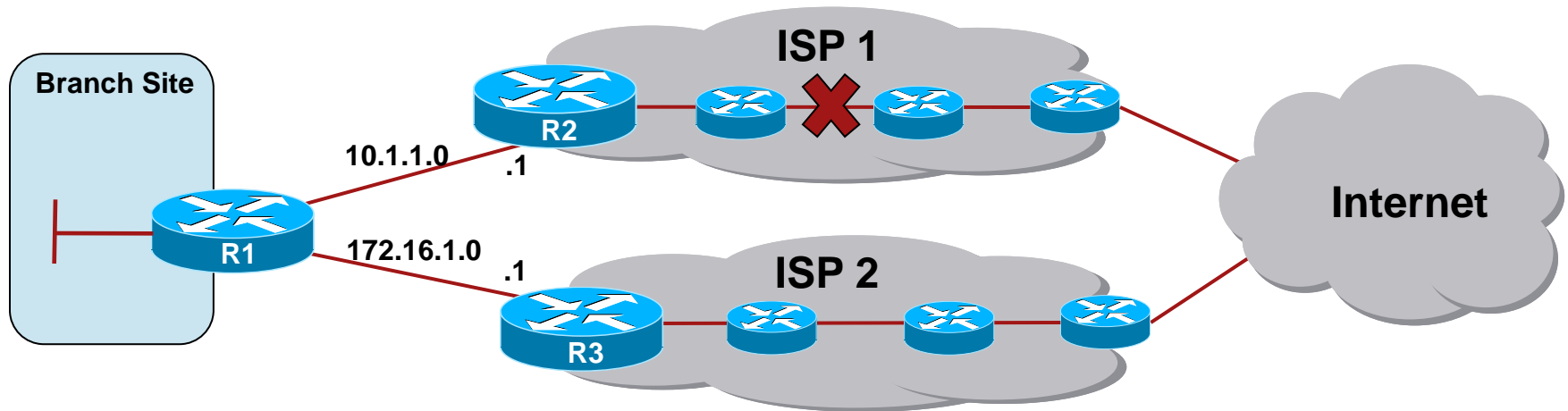
- The R1 edge router is configured to perform NAT, and has two default routes for outbound traffic to two ISPs
- The two static default routes are configured on R1
 - Router will use load balancing over the links by default
 - IF primary link fails THEN secondary will be used

Multi-homed Connection ②



- Problem can occur in ISP1 inner network
 - R1 sees the link to ISP1 as functional
 - Forwarded data will be lost
- Possible solution:
 - Routing protocol between ISP and customer
 - Impractical for smaller branch offices, and in any case requires interaction and integration with the ISPs

Multi-homed Connection ③



- Another solution is to use static routes or PBR, but make them **subject to reachability tests**
 - E.g. DNS server of mail server
 - IF the test fails THEN static route will be removed from routing table
- **IP Service Level Agreements** provides availability tests

Managing Routing Using IP SLAs

- Cisco IOS IP Service Level Agreements (SLAs) use **active traffic monitoring**
- Cisco IOS IP SLAs tests send simulated data and measure performance between network locations
- Administrator can set the requirements needed to satisfy the test
- IP SLA tests CAN run between:
 - Cisco devices
 - Cisco device and an IP host (only limited set of tests)

Cisco IOS IP SLAs

- Possible parameters:
 - **Network resource availability**
 - **Response time**
 - **One-way latency**
 - **Jitter**
 - **Packet lost**
 - **Voice quality scoring**
 - **Application performance**

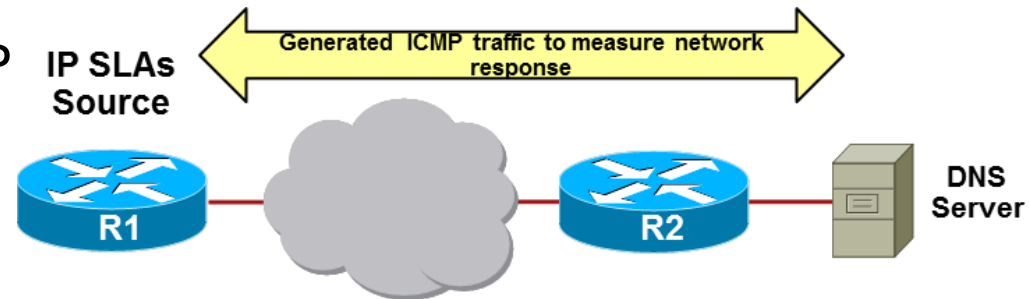
IP SLA Sources, Responders and Operation

- **IP SLA operation** is measurement that includes protocol, frequency, traps and thresholds
- **IP SLA source** sends testing data to destination
 - All test are configured on SLA source
 - SLA source use **control protocol** to communication with responder before the test starts (agreement on TCP/UDP ports, test type etc.)
 - Source and responder have to use time synchronization (NTP)
- **IP SLA responder** allows to anticipate and respond to IP SLAs request packets

IP SLAs Operations

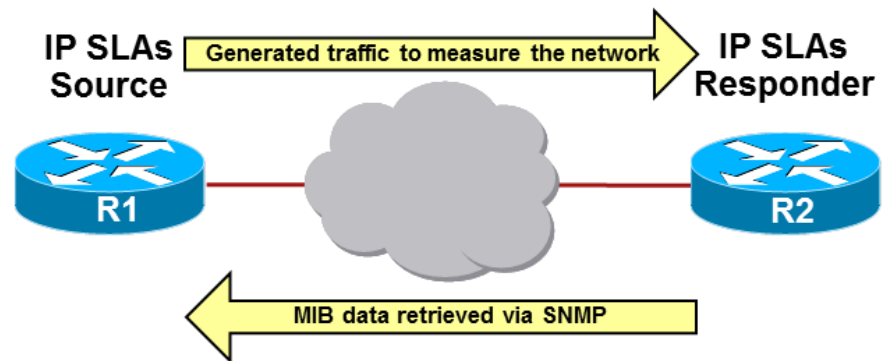
- IP SLA against device **without SLA responder** (web server or IP station)

- E.g. ping test



- IP SLA against device **with SLA responder** running (e.g. Cisco router)

- More powerful test or more accurate results



Configuration Steps

1. Define SLA operation (probe)
2. Scheduling IP SLA operation
3. Define at least one tracking object and define action associated with the tracking object

- Note:

- Available from IOS 12.4(4)T, 12.2(33)SB and 12.2(33)SXI ,
command **ip sla monitor** is replaced by command **ip sla**

Step 1 – Define SLA Operation ①

1. Defining SLA operation

```
Router(config)# ip sla operation-number
```

- Parameter *operation-number* is ID of operation

```
R1(config)# ip sla 1  
R1(config-ip-sla)# ?  
IP SLAs entry configuration commands:  
  
  dhcp          DHCP Operation  
  dns           DNS Query Operation  
  exit          Exit Operation Configuration  
  icmp-echo    ICMP Echo Operation  
  icmp-jitter   ICMP Jitter Operation  
  ...  
R1(config-ip-sla)#
```


Step 1 – Define SLA Operation ②

- PING probe against non-responder node:

```
Router(config-ip-sla)# icmp-echo {destination-ip-address |  
destination-hostname} [source-ip {ip-address | hostname} |  
source-interface interface-name]
```

Parameter	Description
<i>destination-ip-address</i> <i>destination-hostname</i>	Destination IPv4/IPv6 address
source-ip <i>{ip-address hostname}</i>	(Optional) Sets source IPv4/IPv6 address
source-interface <i>interface-name</i>	(Optional) Sets the interface IP address as a source

- Note:
 - Starting from IOS 12.4(4)T, 12.2(33)SB and 12.2(33)SXI command **type echo protocol ipIcmpEcho** is replaced by command **icmp-echo**

The icmp-echo Command

■ Configuration options:

```
R1(config-ip-sla)# icmp-echo 209.165.201.30
```

```
R1(config-ip-sla-echo)# ?
```

IP SLAs echo Configuration Commands:

default	Set a command to its defaults
exit	Exit operation configuration
frequency	Frequency of an operation
history	History and Distribution Data
no	Negate a command or set its defaults
owner	Owner of Entry
request-data-size	Request data size
tag	User defined tag
threshold	Operation threshold in milliseconds
timeout	Timeout of an operation
tos	Type Of Service
verify-data	Verify data
vrf	Configure IP SLAs for a VPN Routing/Forwarding in-stance

```
R1(config-ip-sla-echo)#
```

```
Router(config-ip-sla-echo)# frequency seconds
```

```
Router(config-ip-sla-echo)# timeout milliseconds
```

Step 2 – Scheduling IP SLA Operation

2. IP SLA operation needs to be scheduled

```
Router(config)#
```

```
ip sla schedule operation-number [life {forever |  
seconds}] [start-time {hh:mm[:ss] [month day | day  
month] | pending | now | after hh:mm:ss}]  
[ageout seconds] [recurring]
```

■ Note:

- Starting from IOSu 12.4(4)T, 12.2(33)SB and 12.2(33)SXI the command **ip sla monitor schedule** is replaced by **ip sla schedule**

Step 3 – Creating Tracking Object

3. Creating tracking object:

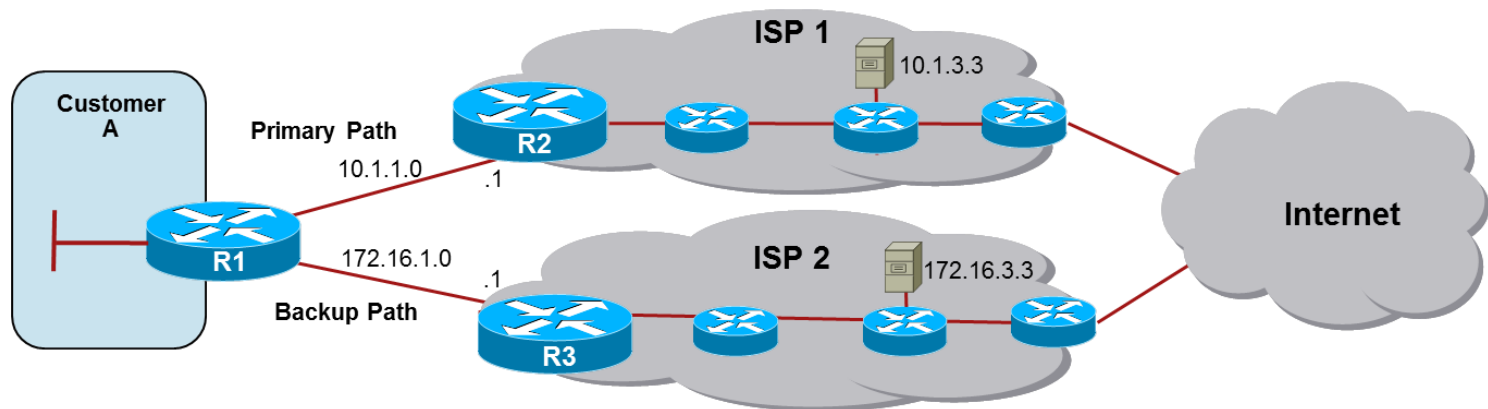
```
Router(config)# track object-number ip sla operation-number {state | reachability}
```

Parameter	Description
<i>object-number</i>	Tracking object number from 1 to 500
<i>operation-number</i>	Number used for the identification of the IP SLAs operation you are tracking
state	Tracks the operation return code
reachability	Tracks the reachability

■ Note:

- Starting from IOSu 12.4(20)T, 12.2(33)SXI1 and 12.2(33)SRE the command **track rtr** is replaced by **track ip sla**

IS SLAs on Multi-homed Connection Example



```
R1(config)# ip sla 11
R1(config-ip-sla)# icmp-echo 10.1.3.3
R1(config-ip-sla-echo)# frequency 10
R1(config-ip-sla-echo)# exit ! 2x
R1(config)# ip sla 22
R1(config-ip-sla)# icmp-echo 172.16.3.3
R1(config-ip-sla-echo)# frequency 10
R1(config-ip-sla-echo)# exit ! 2x
R1(config)# track 1 ip sla 11 reachability
R1(config-track)# delay down 10 up 1
R1(config-track)# exit
R1(config)# track 2 ip sla 22 reachability
R1(config-track)# delay down 10 up 1
R1(config-track)# exit
R1(config)# ip sla schedule 11 life forever start-time now
R1(config)# ip sla schedule 22 life forever start-time now
R1(config)# ip route 0.0.0.0 0.0.0.0 10.1.1.1 2 track 1
R1(config)# ip route 0.0.0.0 0.0.0.0 172.16.1.1 3 track 2
```

Useful Commands

`show ip protocols`

`show ip policy`

`show route-map [map-name]`

`debug ip policy`

`show ip sla configuration`

`show ip sla statistics`

Where to go next?

- Commonly Used IP ACLs

- http://cisco.com/en/US/tech/tk648/tk361/technologies_configuration_example09186a0080100548.shtml

- Default Passive Interface Feature

- http://cisco.com/en/US/products/sw/iosswrel/ps1830/products_feature_guide09186a008008784e.html

- Route-Maps for IP Routing Protocol Redistribution Configuration

- http://cisco.com/en/US/tech/tk365/technologies_tech_note09186a008047915d.shtml

- IP SLA Configurations

- http://www.cisco.com/en/US/docs/ios/12_4/ip_sla/configuration/guide/hsoverv.html

- The Cisco IOS IP SLAs Command Reference:

- http://www.cisco.com/en/US/docs/ios/ipsla/command/reference/sla_book.html

- Cisco OER

- http://www.cisco.com/en/US/tech/tk1335/tsd_technology_support_sub-protocol_home.html



Slides adapted by Matěj Grégr (and extended by [Vladimír Veselý](#))
partially from official course materials
but the most of the credit goes to CCIE#23527 Ing. Peter Palúch, Ph.D.

Last update: 2012-09-09