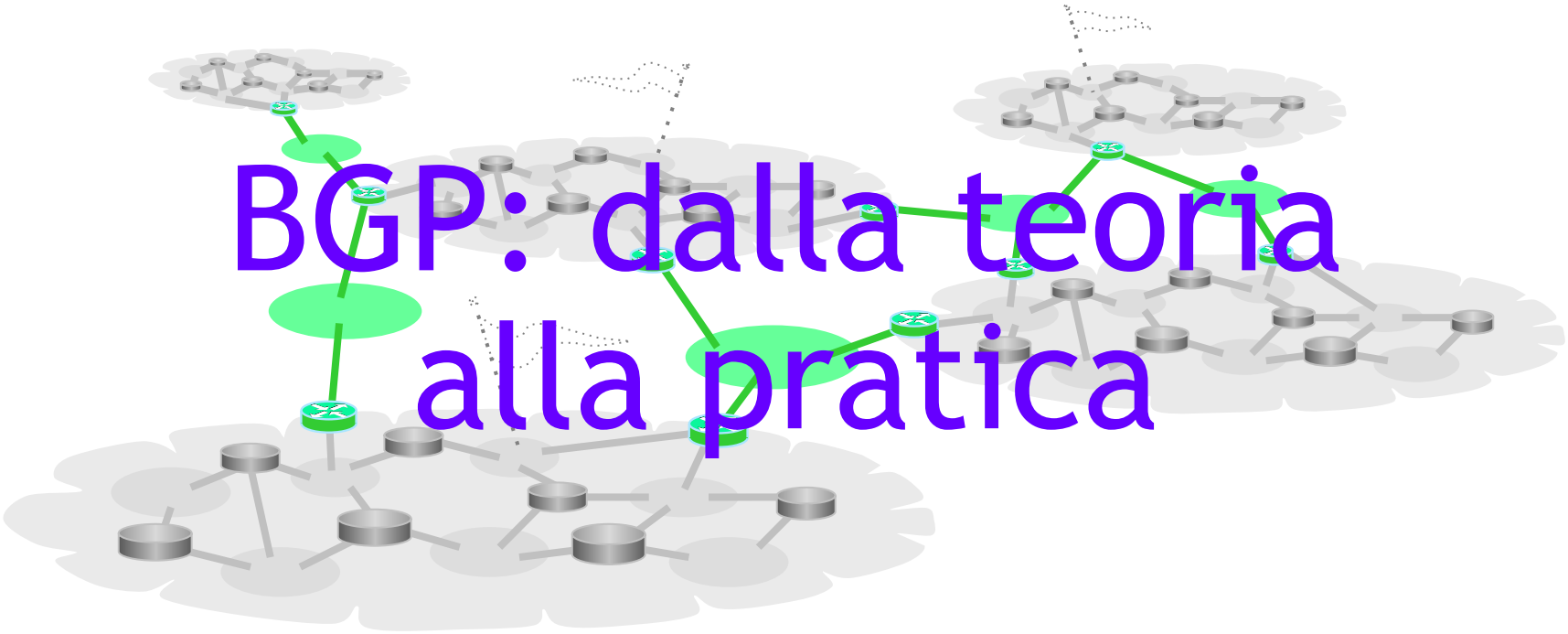
REISS ROMOLI

EUREKALABRIA
Idee in connessione



BGP: dalla teoria alla pratica

Antonio Prado

REISS ROMOLI

Note di *Copyright*

- Questo insieme di diapositive è protetto dalle leggi sul *copyright* e dalle disposizioni dei trattati internazionali. Il titolo ed i *copyright* relativi alle diapositive (ivi inclusi, ma non limitatamente, ogni immagine, fotografia, animazione, video, audio, musica e testo), in accordo con gli artt. 12 e seguenti della Legge 633/1941, sono di proprietà dell'autore Tiziano Tofoni (di seguito 'l'autore').
- Le diapositive possono essere utilizzate esclusivamente per scopi di studio nell'ambito dei corsi tenuti dall'autore.
- Ogni altra utilizzazione o riproduzione (ivi incluse, ma non limitatamente, le riproduzioni su supporti ottici/magnetici, su reti di calcolatori o stampate) in toto o in parte è vietata, se non esplicitamente autorizzata per iscritto, a priori, da parte dell'autore.
- L'informazione contenuta in queste diapositive è ritenuta essere accurata alla data della pubblicazione. Essa è fornita per scopi meramente didattici e non per essere utilizzata in progetti di impianti, prodotti, reti, ecc. In ogni caso essa è soggetta a cambiamenti senza preavviso. L'autore non si assume alcuna responsabilità per il contenuto di queste diapositive (ivi incluse, ma non limitatamente, la correttezza, completezza, applicabilità, aggiornamento dell'informazione).
- In ogni caso non può essere dichiarata conformità all'informazione contenuta in queste diapositive.
- In ogni caso questa nota di *copyright* non deve mai essere rimossa e deve essere riportata anche in utilizzi parziali.

REISS ROMOLI

Agenda (1/3)

- Concetti fondamentali
  - Caratteristiche principali
  - *Autonomous System*
  - Connettività Clienti-ISP
  - Funzionamento di base
  - Sessioni BGP
  - Messaggi e attributi
  - Processo di selezione
- Implementazione base nell'IOS, IOS XE e IOS XR
  - Configurazioni base
  - Propagazione dei prefissi locali
  - Generazione della *default route*
  - BGP per IPv6
  - Controllo della configurazione

Agenda (2/3)

- Strumenti per la manipolazione degli annunci
  - Generalità
  - *Prefix-list*
  - *Route-map*
  - *Route-policy* e *Route Policy Language*
- Filtraggio degli annunci BGP
  - Definizione e motivazioni
  - Filtri basati sui prefissi
  - Filtri basati sulle *Community*
  - Applicazione dei filtri

REISS ROMOLI

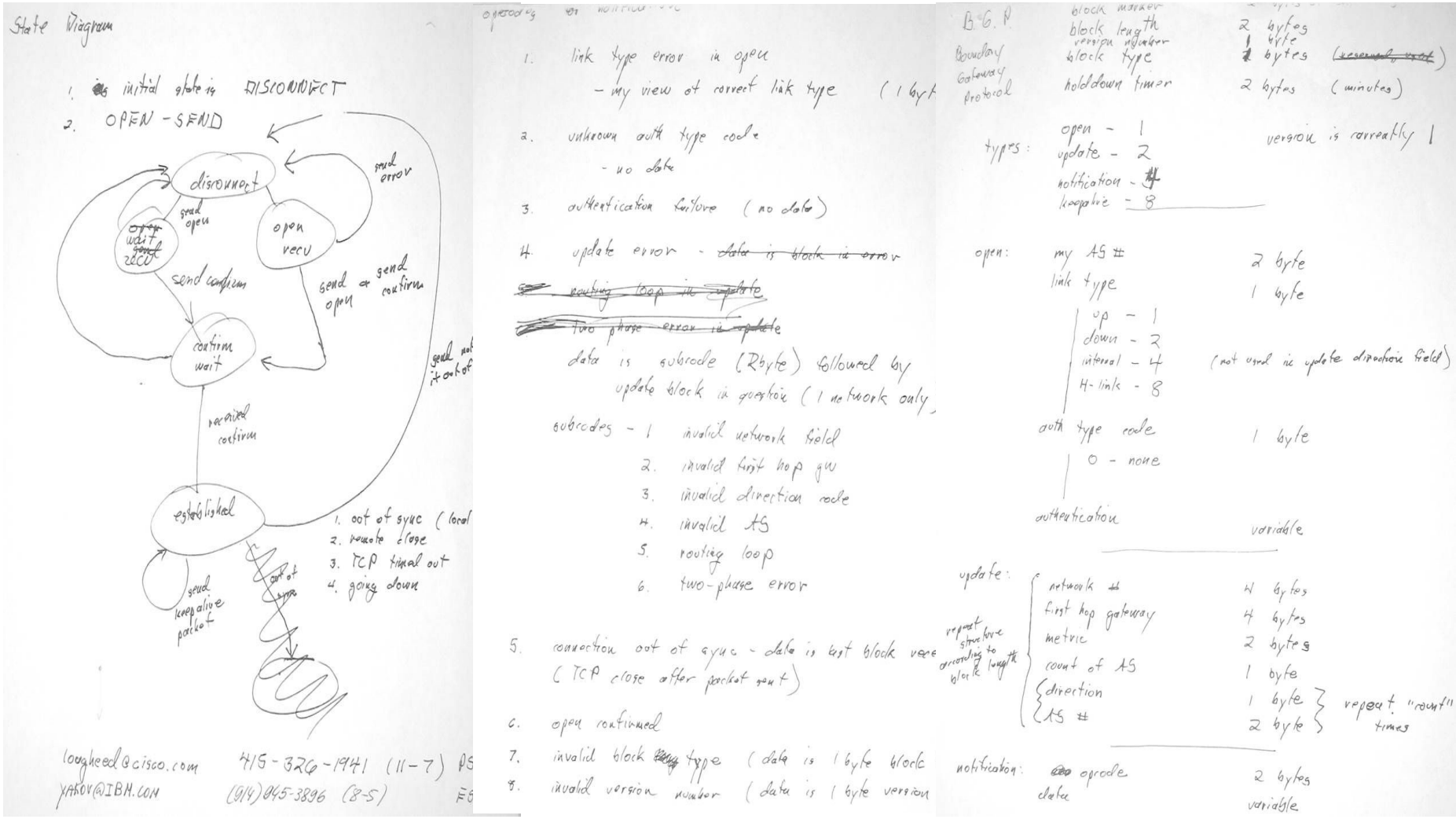
Agenda (3/3)

- Politiche di routing
  - Il processo di selezione nei router Cisco
  - Gestione del traffico *outbound* attraverso l'attributo *Local Preference*
  - Gestione del traffico *inbound* attraverso *AS_PATH prepending*
  - Gestione del traffico *inbound* attraverso l'attributo MED
  - Utilizzo dell'attributo *Community*
- Il BGP nelle reti dei *Service Provider*
  - Architettura di routing delle reti degli ISP
  - *Route Reflector*
- Aspetti di sicurezza
  - Problemi e soluzioni
  - Contromisure
  - *Remote-Triggered Black-Hole Filtering (RTBH)*
  - *RPKI e ROA*

Concetti fondamentali

Caratteristiche principali
Autonomous System
Connettività Clienti-ISP
Funzionamento di base
Sessioni BGP
Messaggi e attributi
Processo di selezione

The “Three Napkins Protocol”



Introduzione (1/2)

- BGP (Border Gateway Protocol) è un protocollo standard sviluppato per lo scambio di informazioni di routing tra reti IP amministrate da Enti diversi (AS, Autonomous System)
  - La versione attuale è la N.ro 4
  - Documento originale: RFC 1771 (Marzo 1995) – *A Border Gateway Protocol 4 (BGP-4)*
  - Nuova versione: RFC 4271 (Gennaio 2006) – *A Border Gateway Protocol 4 (BGP-4)*
- Viene anche utilizzato all'interno di un AS, tipicamente per importare in una rete destinazioni esterne all'AS
- È un protocollo di routing di tipo *Path Vector*
  - Concettualmente simile a un protocollo *Distance Vector* ma con i salti misurati come numero di AS e non come numero di router

Introduzione (2/2)

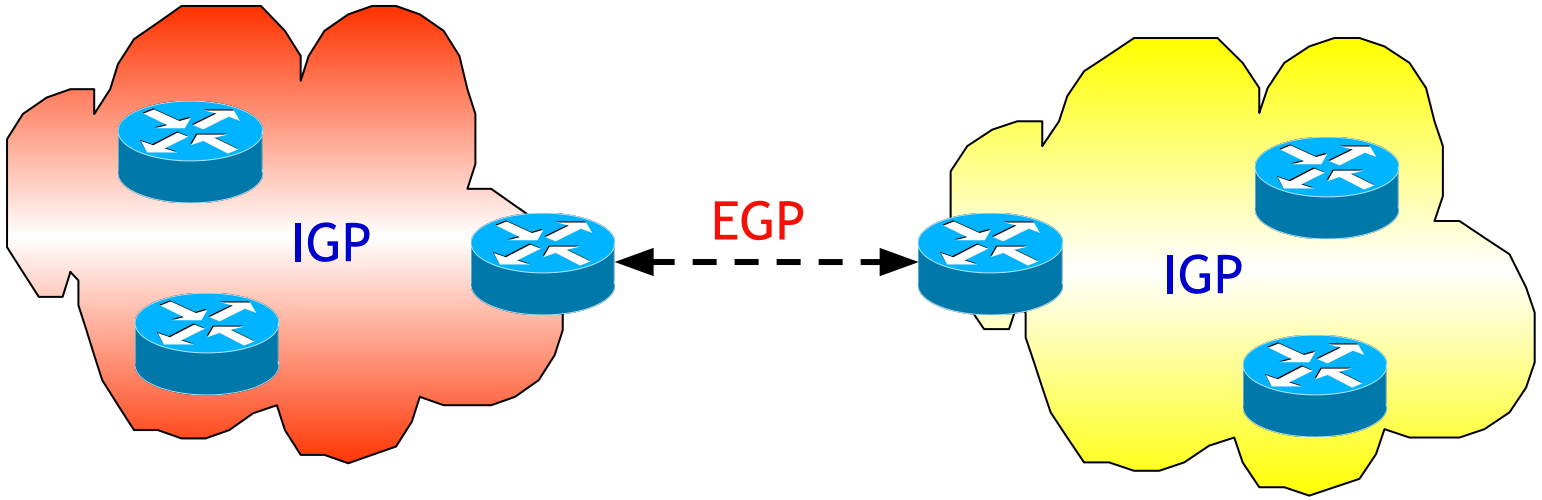
- Determina i percorsi ottimi tramite un processo di selezione molto complesso, basato su metriche di vario tipo
  - La ricchezza delle metriche rende il BGP un protocollo molto complesso in termini di implementazione ma molto potente e flessibile per la realizzazione di politiche di routing
- Caratteristiche principali
  - Scambio affidabile delle informazioni di routing (via sessioni TCP)
  - Ricchezza di metriche (es. Local Preference, MED, AS Path, ecc.)
  - Aggiornamenti solo a fronte di eventi (*triggered update*)
  - Supporta il ***CIDR*** (***C****lassless* ***I****nter* ***D****omain* ***R****outing*)
  - Protocollo scalabile applicabile a reti molto estese (può gestire tabelle con un elevato numero di prefissi IP, dell'ordine delle centinaia di migliaia)

Concetti fondamentali

Caratteristiche principali
Autonomous System
Connettività Clienti-ISP
Funzionamento di base
Sessioni BGP
Messaggi e attributi
Processo di selezione

Definizione

- Un Autonomous System (AS) è un insieme di reti amministrate da un singolo ente
  - Gli AS si scambiano informazioni di routing attraverso protocolli di tipo EGP
  - Al loro interno gli AS scambiano informazioni di routing attraverso protocolli di tipo IGP
- Il BGP è lo standard “de facto” universalmente adottato come protocollo EGP



(*) EGP = Exterior Gateway Protocol ; IGP = Interior Gateway Protocol

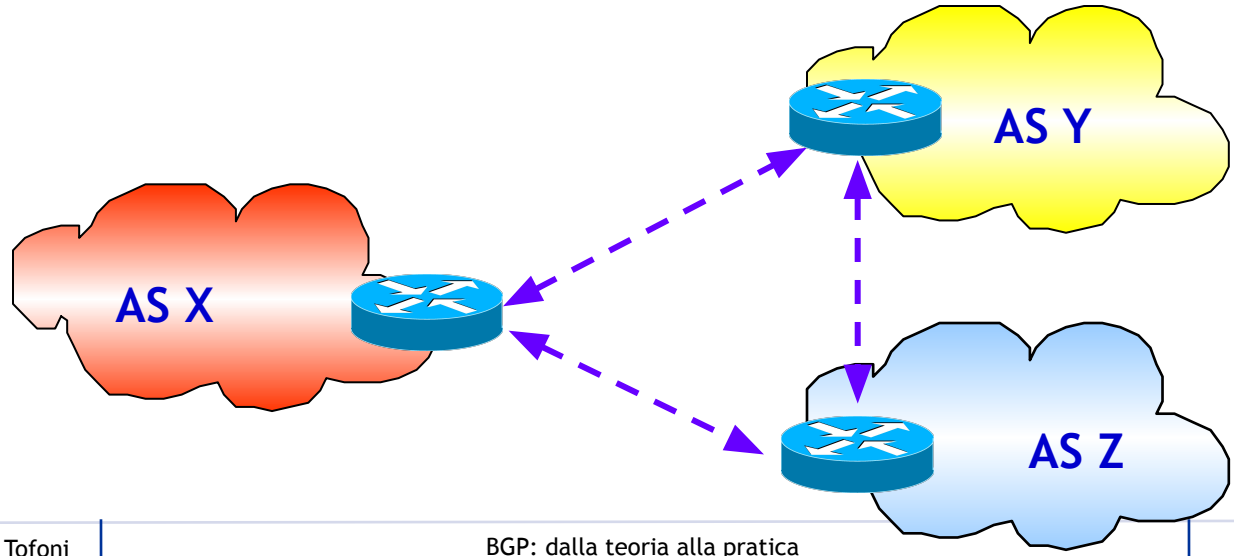
Numerazione

- Ogni AS è identificato universalmente da un numero di 16 bit assegnato dai vari *RIR*(*) (RIPE, APNIC, ARIN, LACNIC, AfriNIC)
  - Numeri disponibili: da 1 a 65.534
    - Il valore di AS = 0 è riservato per alcuni aspetti di sicurezza del BGP
  - I valori da 64.512 a 65.534 non sono assegnabili ad AS pubblici ma sono riservati ad AS privati
  - I valori da 64.496 a 64.511 non sono assegnabili ad AS pubblici ma sono riservati per la documentazione (RFC 5398)
  - Esempi: la rete IP di TIM ha AS=3269, la rete IP di Fastweb ha AS=12874, la rete GIN di NTT ha AS=2914, ecc.
- Considerata l'esiguità dei numeri di AS pubblici disponibili, è stato standardizzato in passato un ampliamento a 32 bit del numero di AS
  - RFC 6793 (ex-4893) , *BGP support for four-octet AS number space*, Maggio 2007

(*) *RIR* = *Regional Internet Registry*

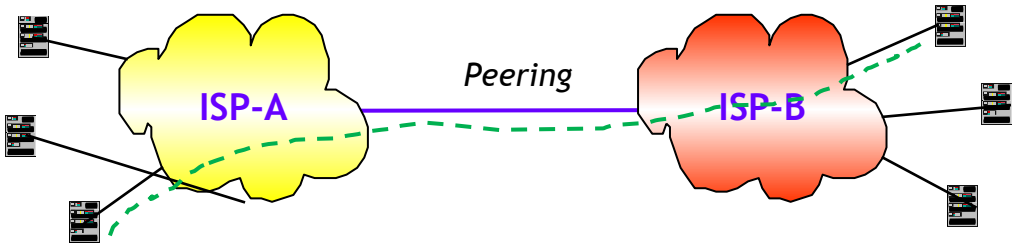
AS *Multi-Homed*

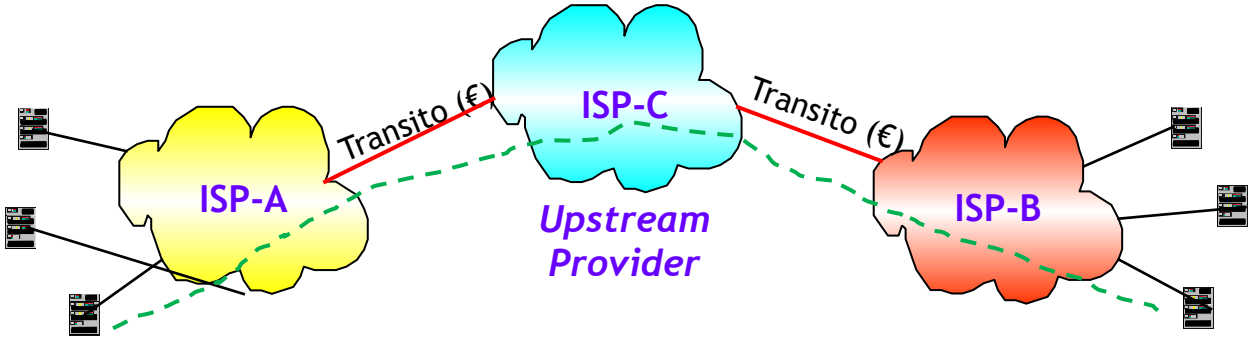
- Un AS è di tipo *Multi-Homed* quando ha più connessioni verso altri AS
  - Esempio tipico: un AS di un Cliente che si collega per motivi di affidabilità alle reti di due ISP
- Due tipi di AS *Multi-Homed*
  - AS di Transito: permettono lo scambio di traffico tra ISP differenti utilizzando come transito risorse proprie
  - AS Non di Transito: non permettono transito di traffico altrui sull'AS
  - NOTA: per un AS, il traffico di transito è definito come l'insieme dei pacchetti IP che hanno indirizzo IP sorgente e destinazione non appartenenti all'AS



Relazioni tra ISP: *peering* e transito

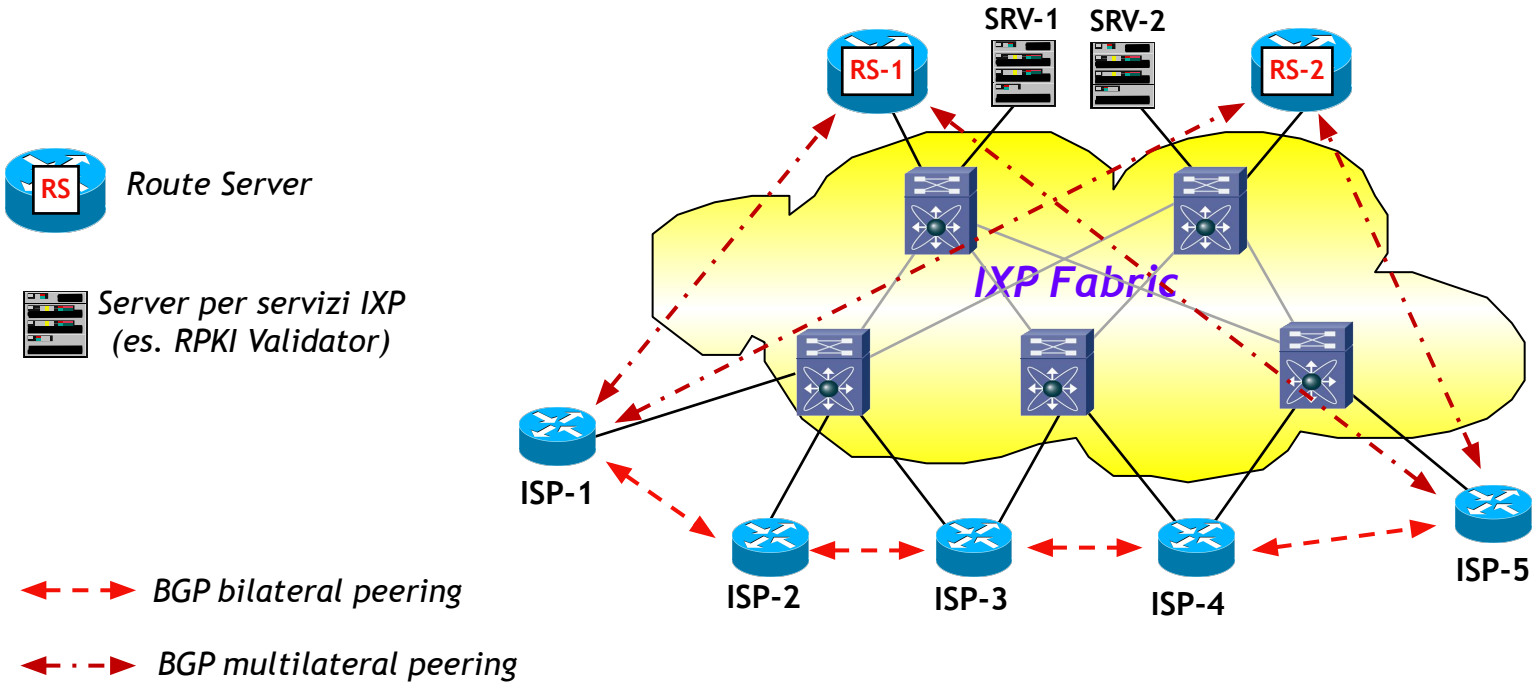
- Due tipi di accordi di interconnessione tra ISP
  - *Peering*: due o più ISP si interconnettono direttamente l'uno con l'altro per scambiare traffico tra i loro clienti
  - Transito: un ISP accetta di trasportare il traffico che fluisce tra un altro ISP e tutti gli altri ISP





Internet eXchange Point (IXP)

- Un *IXP* è un'infrastruttura fisica che consente a diversi ISP di scambiare tra loro traffico Internet
  - L'infrastruttura fisica (*IXP Fabric*) è tipicamente costituita da una rete *Switched Ethernet* o da una *IP Fabric* con *VXLAN+EVPN*
- Lo scambio delle informazioni di routing tra ISP avviene tramite accordi di *peering* bilaterali o multilaterali



Classificazione degli ISP

Big Internet

Tier-1

Tier-1

Tier-2

Tier-2

Tier-2

Tier-3

Tier-3

Tier-3

Tier-3

Le buone regole degli ISP ...

Tipo di annunci	*eBGP-out* verso i clienti	*eBGP-out* verso i *BGP peer*	*eBGP-out* verso gli *Upstream Provider*
Ricevuti da un cliente	Accetta	Accetta	Accetta
Ricevuti da un *BGP peer*	Accetta	Rifiuta	Rifiuta
Ricevuti da un *Upstream Provider*	Accetta	Rifiuta	Rifiuta
Annunci locali	Accetta	Accetta	Accetta
Non classificati	Rifiuta	Rifiuta	Rifiuta

Concetti fondamentali

Caratteristiche principali
Autonomous System
Connettività Clienti-ISP
Funzionamento di base
Sessioni BGP
Messaggi e attributi
Processo di selezione

REISS ROMOLI

Tipologie di connettività

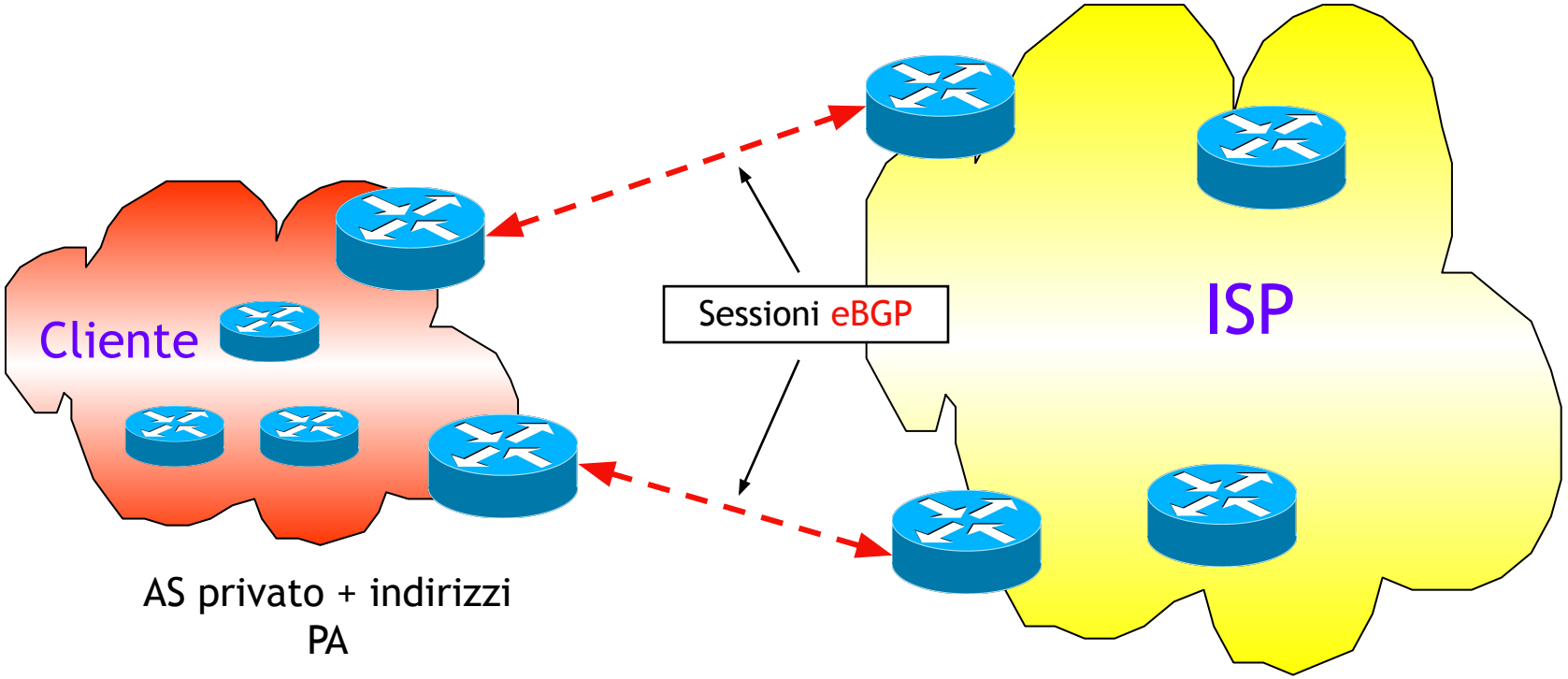
- Connessione a un singolo ISP (*Single-Homed*)
  - Collegamento singolo (*Stub AS*)
  - Collegamento *Fault-tolerant*
    - Modalità primario/backup
    - Con bilanciamento di carico
- Connessione a ISP differenti (*Multi-Homed*)
  - Modalità primario/backup
  - Con bilanciamento di carico
- Tipologie di routing
  - Statico (preferibile data la minore complessità, ma non sempre conveniente)
  - Dinamico (fortemente consigliato l'utilizzo di BGP)

Tipologie di Indirizzi

- Prefissi *Provider Independent* (PI)
  - Sono blocchi di indirizzi pubblici assegnati direttamente dai *RIR* (*Regional Internet Registry*)
  - Il RIR Europeo è il RIPE
  - NOTA: considerata l'endemica scarsità di indirizzi IPv4 pubblici disponibili, il RIPE non assegna più blocchi di indirizzi *PI*

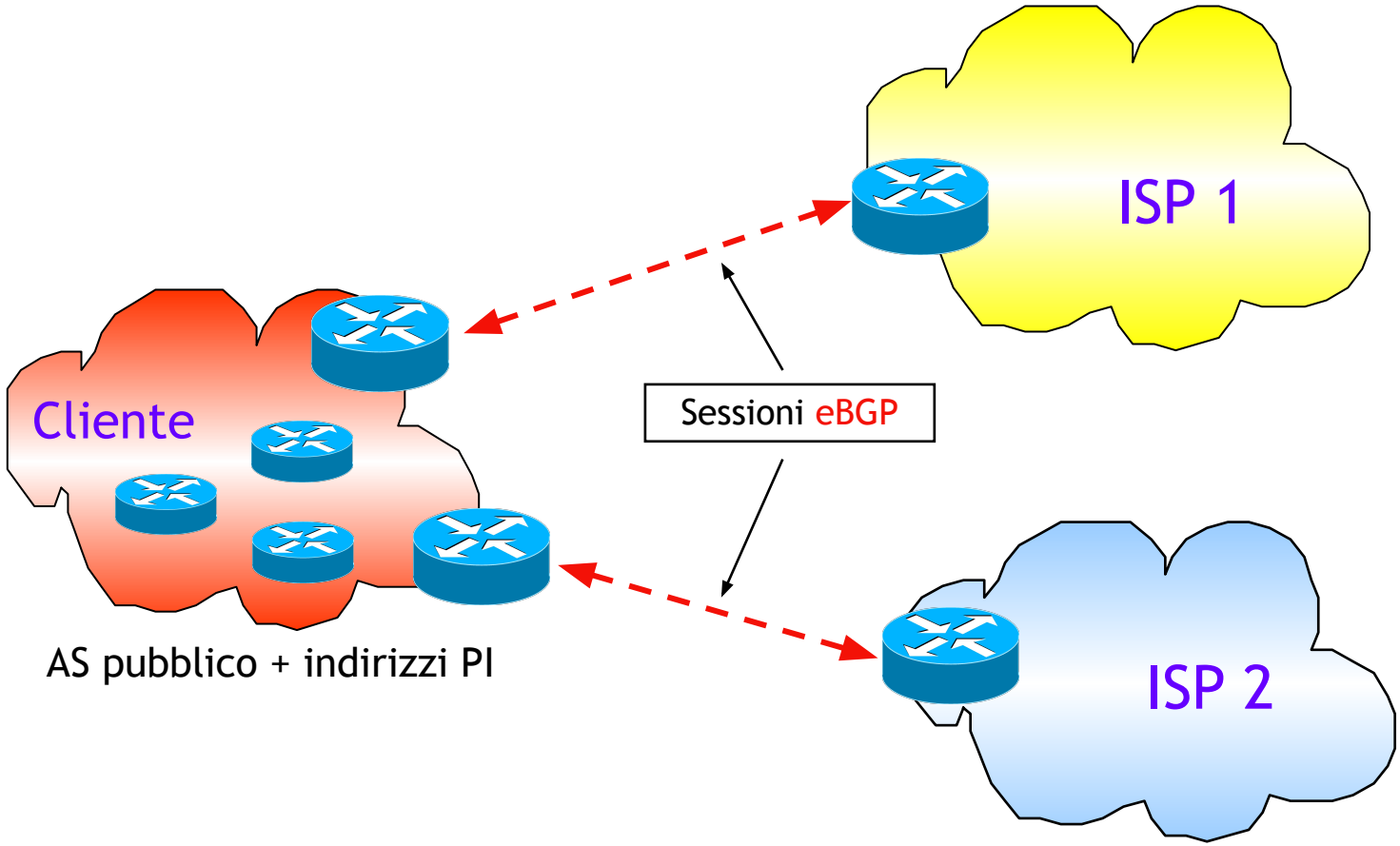
- Prefissi *Provider Aggregatable* (PA)
  - Sono blocchi di indirizzi pubblici assegnati direttamente dagli *ISP* (che assumono il ruolo di *LIR*, *Local Internet Registry*)

Connettività *Single-Homed*



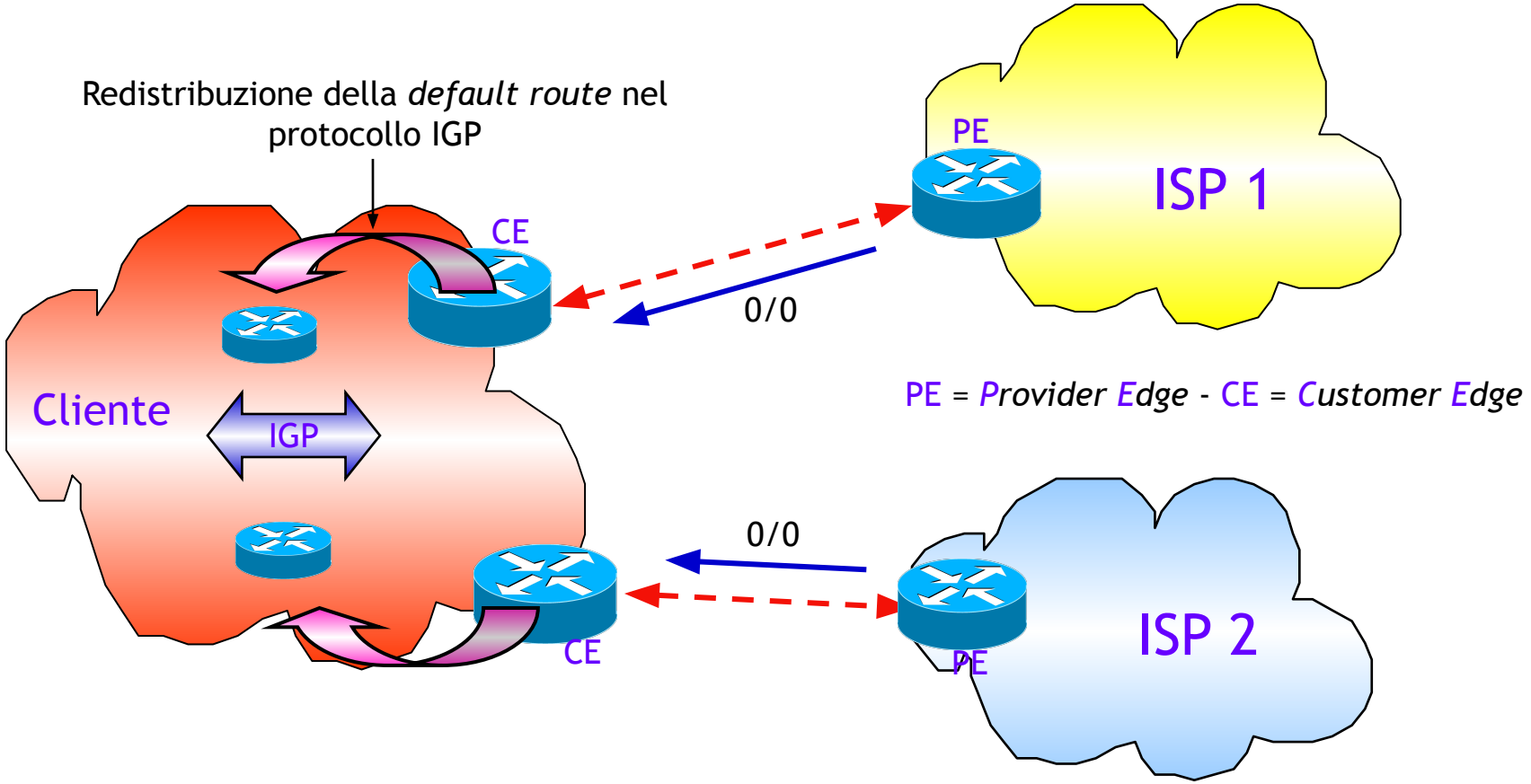
- Per il routing Cliente-ISP è possibile utilizzare route statiche anche se spesso viene utilizzato il BGP per permettere una migliore gestione delle politiche di routing
- Se si utilizza il BGP, al Cliente viene assegnato dall'ISP un numero di AS privato e un blocco di indirizzi PA

Connettività *Multi-Homed*



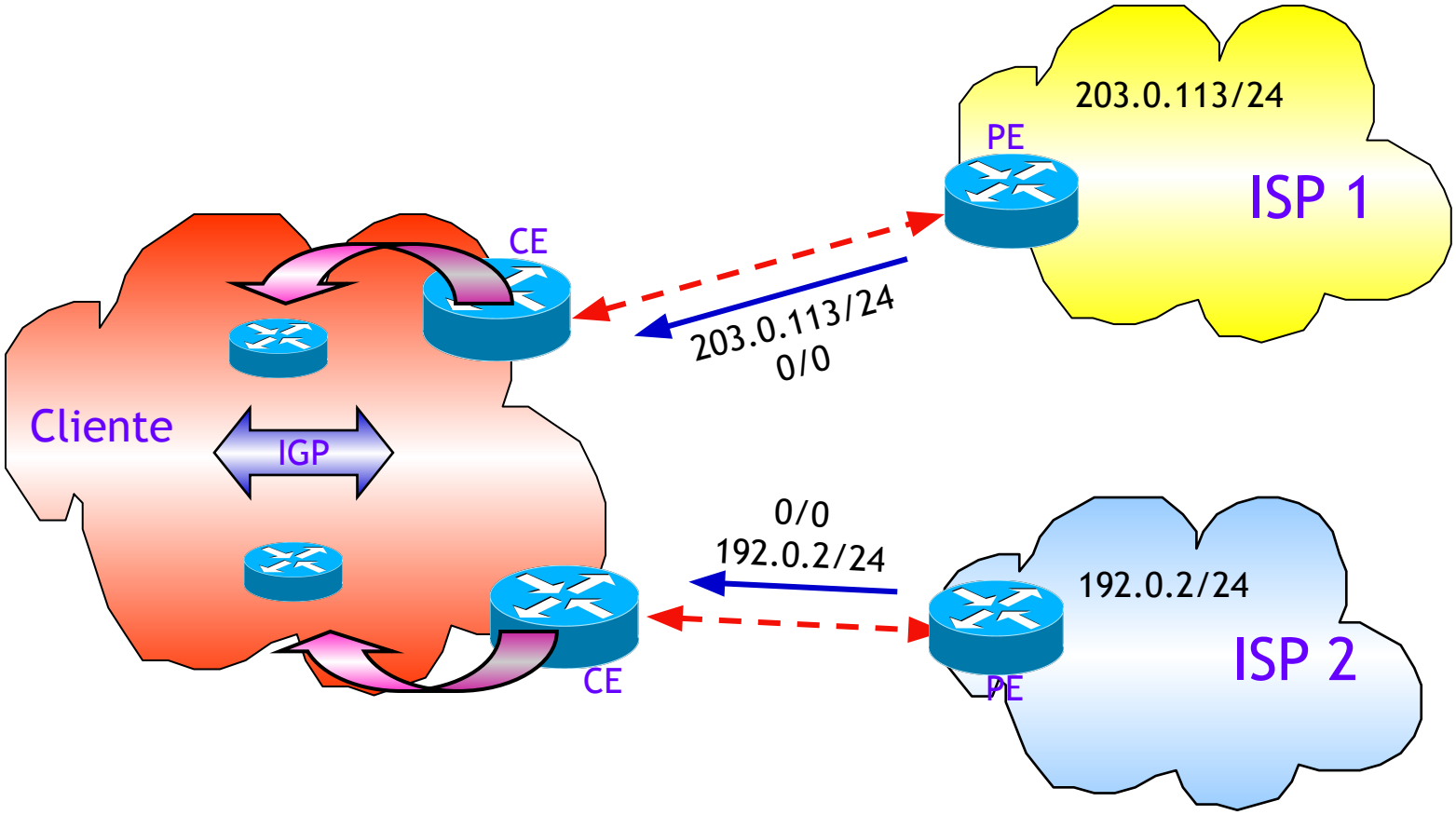
- In questo scenario è di norma preferito l'utilizzo del BGP
- Il Cliente deve utilizzare un numero di AS pubblico e indirizzi PI

Solo *default route* dall'ISP



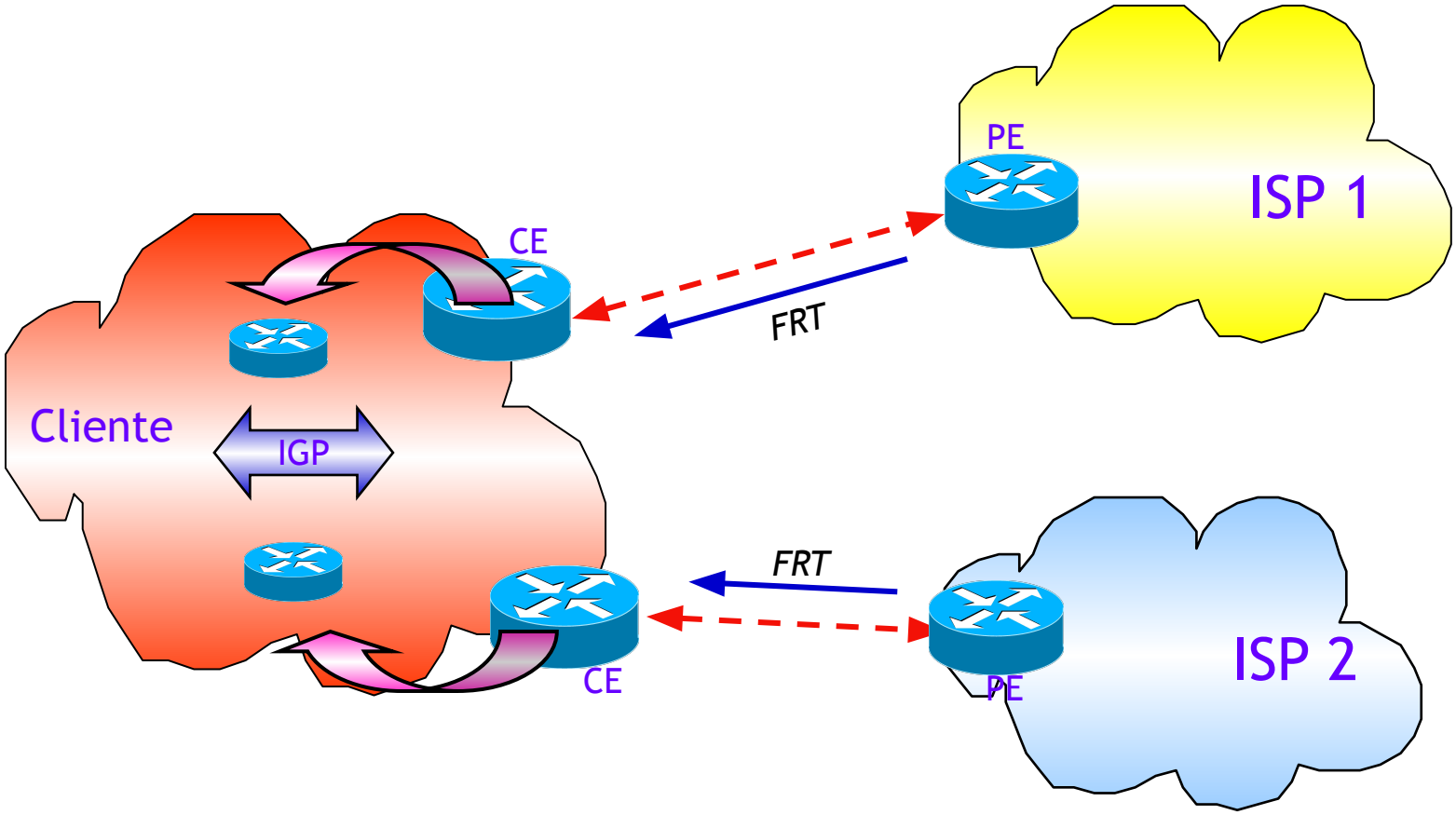
- I router di *edge* dell'ISP propagano ai router di *edge* del Cliente, via BGP, una *default route*, che viene quindi redistribuita nel proprio protocollo IGP

default route + qualche prefisso dall'ISP



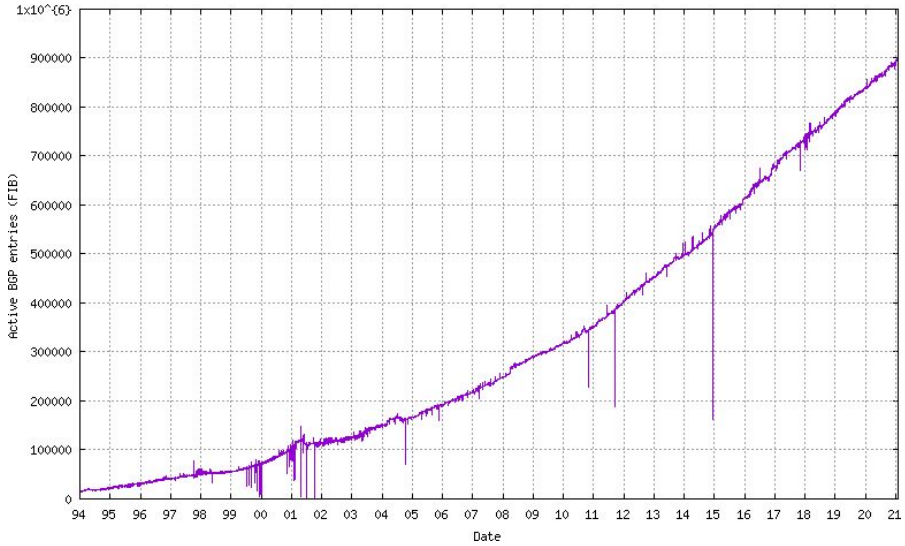
- Vantaggio: il traffico verso i prefissi annunciati dagli ISP segue un percorso "ottimo"

Full Routing table (FRT)



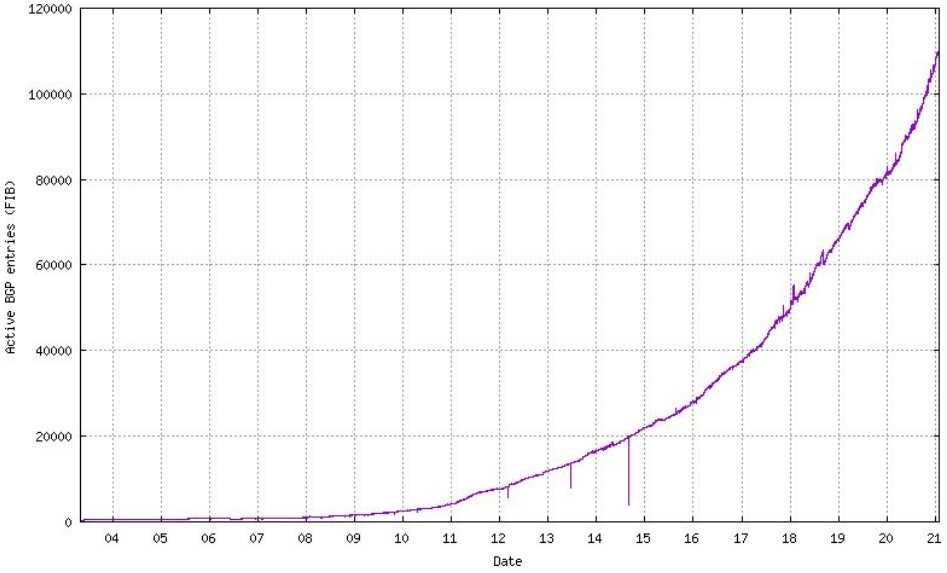
- Vantaggio: il traffico verso tutti i prefissi Internet segue un percorso "ottimo"
- Svantaggio: elevato consumo di risorse interne dei CE

Dimensione della *FRT*



IPv4 (≈ 901k)

IPv6 (≈ 110k)



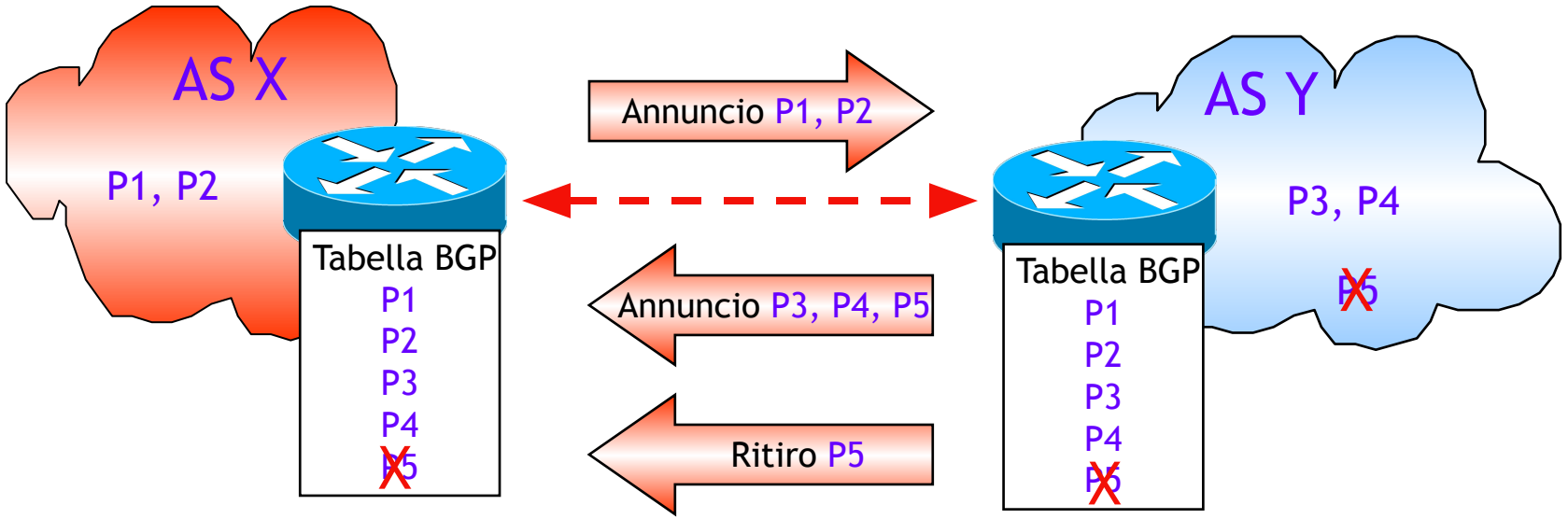
http://bgp.potaroo.net/

Concetti fondamentali

Caratteristiche principali
Autonomous System
Connettività Clienti-ISP
Funzionamento di base
Sessioni BGP
Messaggi e attributi
Processo di selezione

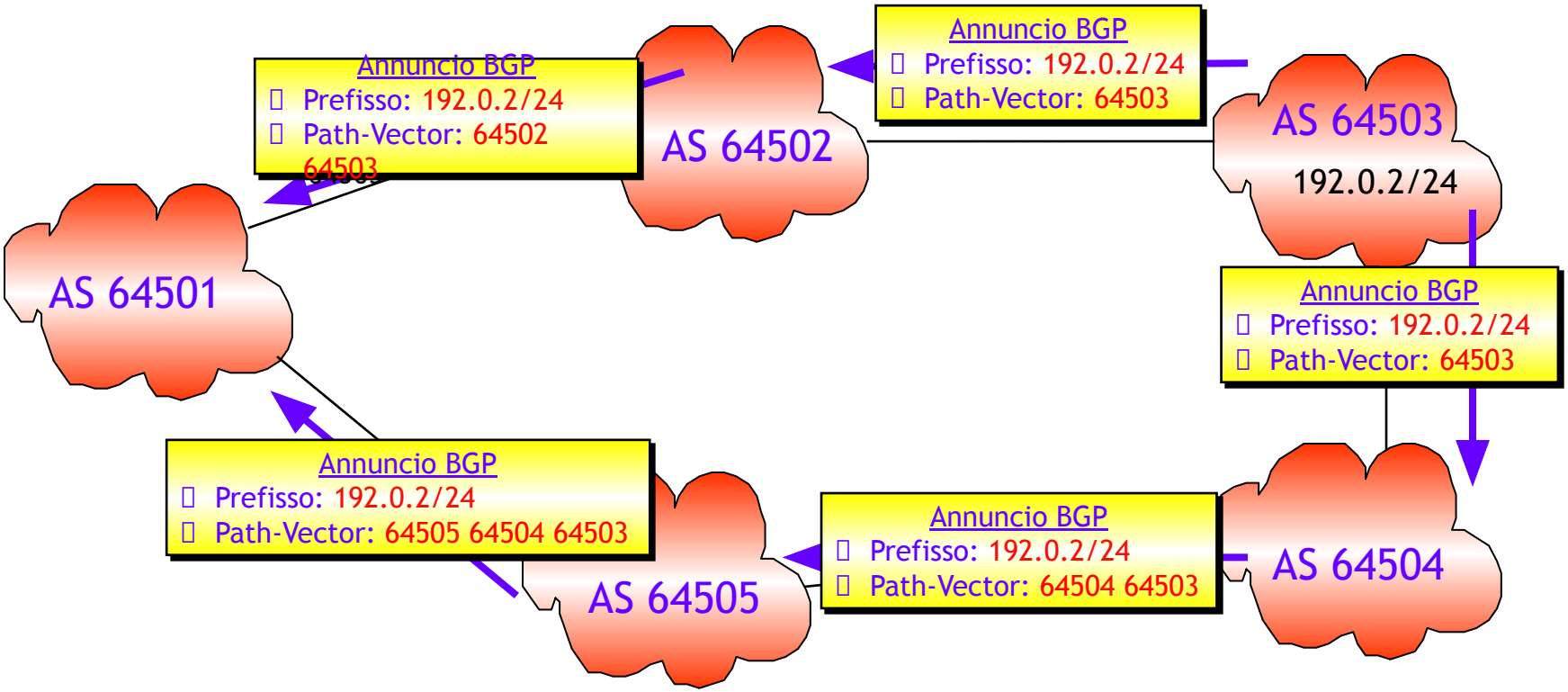
Come funziona

- BGP permette lo scambio di informazioni di routing attraverso un insieme di messaggi scambiati su connessioni TCP
  - Utilizza la porta TCP *well-known* 179
- Le informazioni di routing consistono nell'annuncio di prefissi IP o nel ritiro in caso di perdita di connettività



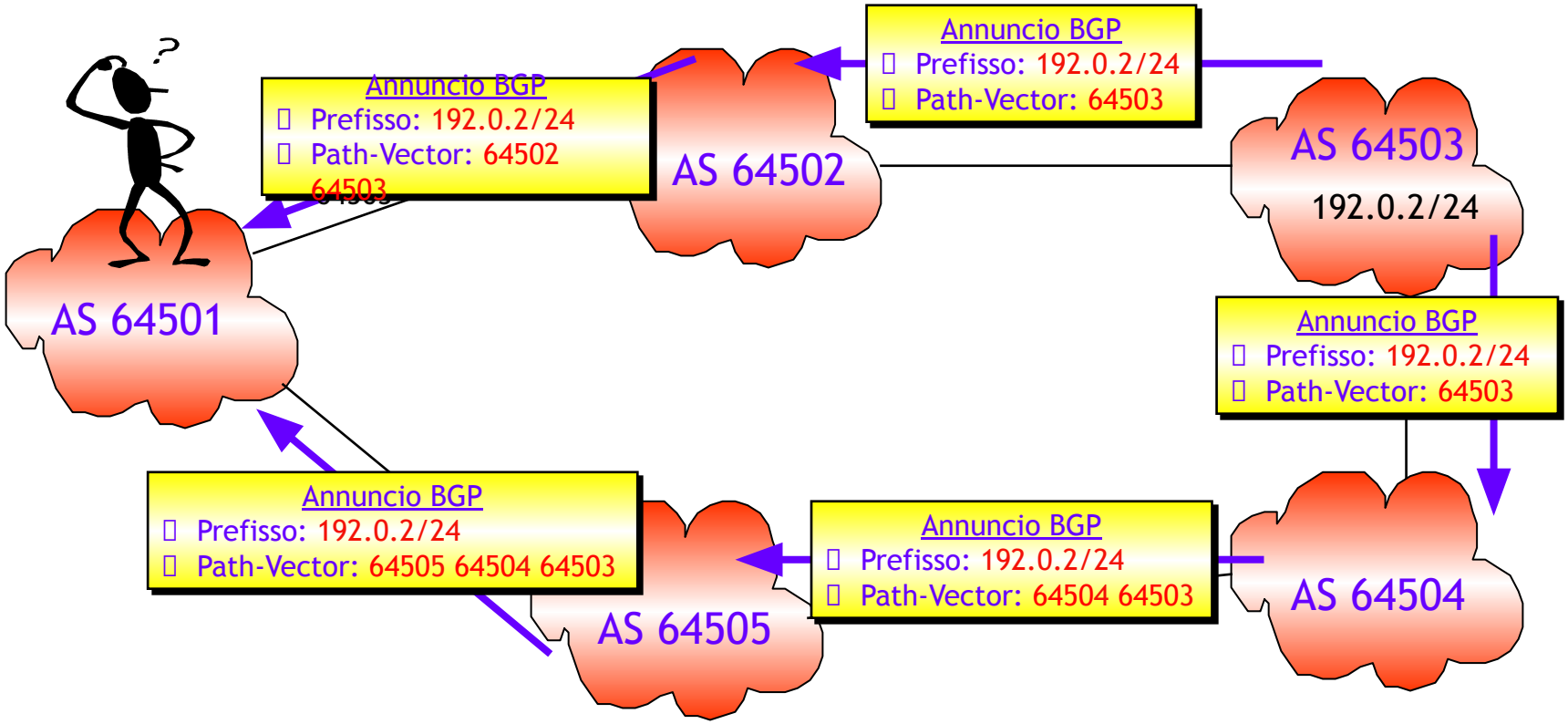
Tipologia di Protocollo

- BGP è un protocollo di tipo *Path Vector*
  - Ogni annuncio BGP contiene (anche) la lista di tutti gli AS attraversati (*Path Vector* o *AS-Path List*)

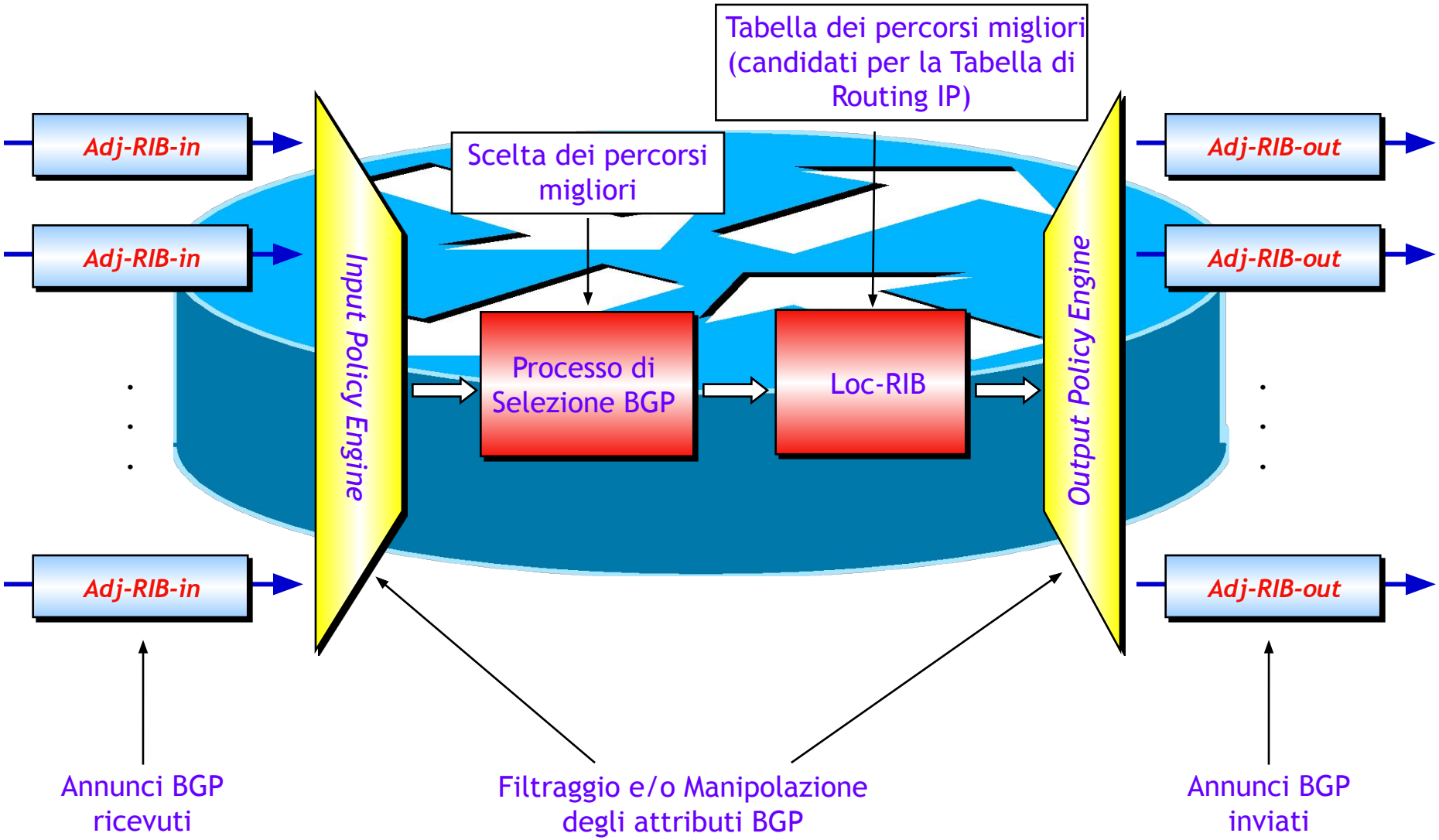


Selezione dei percorsi

- Quando un router riceve più annunci BGP dello stesso prefisso IP, ha un processo di selezione che gli permette di scegliere il percorso migliore
  - Il processo di selezione è molto complesso e dipende da vari fattori (*Path Vector*, metriche, attributi vari)



Modello del processo BGP

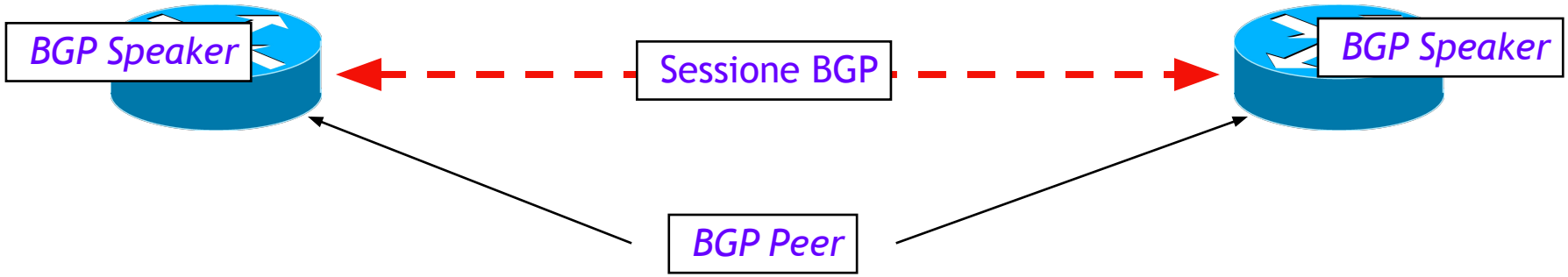


Concetti fondamentali

Caratteristiche principali
Autonomous System
Connettività Clienti-ISP
Funzionamento di base
Sessioni BGP
Messaggi e attributi
Processo di selezione

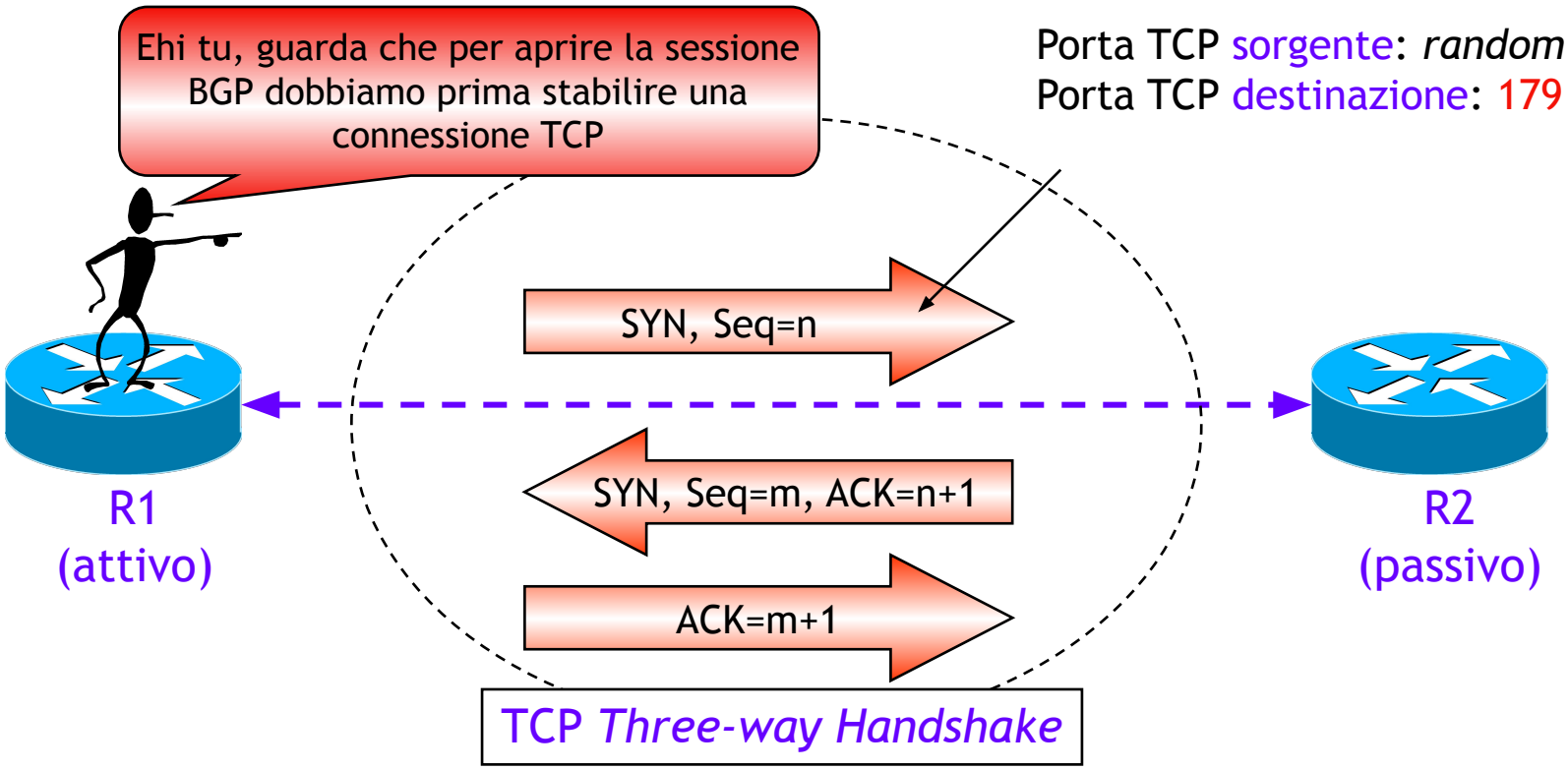
Terminologia

- Un router con un processo BGP attivo viene detto *BGP Speaker*
- Una connessione TCP sulla porta *well-known* 179 viene detta Sessione BGP
  - Le sessioni BGP possono essere stabilite sia tra router direttamente connessi che remoti (purché connessi a livello IP)
- I router agli estremi di una sessione BGP vengono detti *BGP Peer* (o anche *BGP Neighbor*)



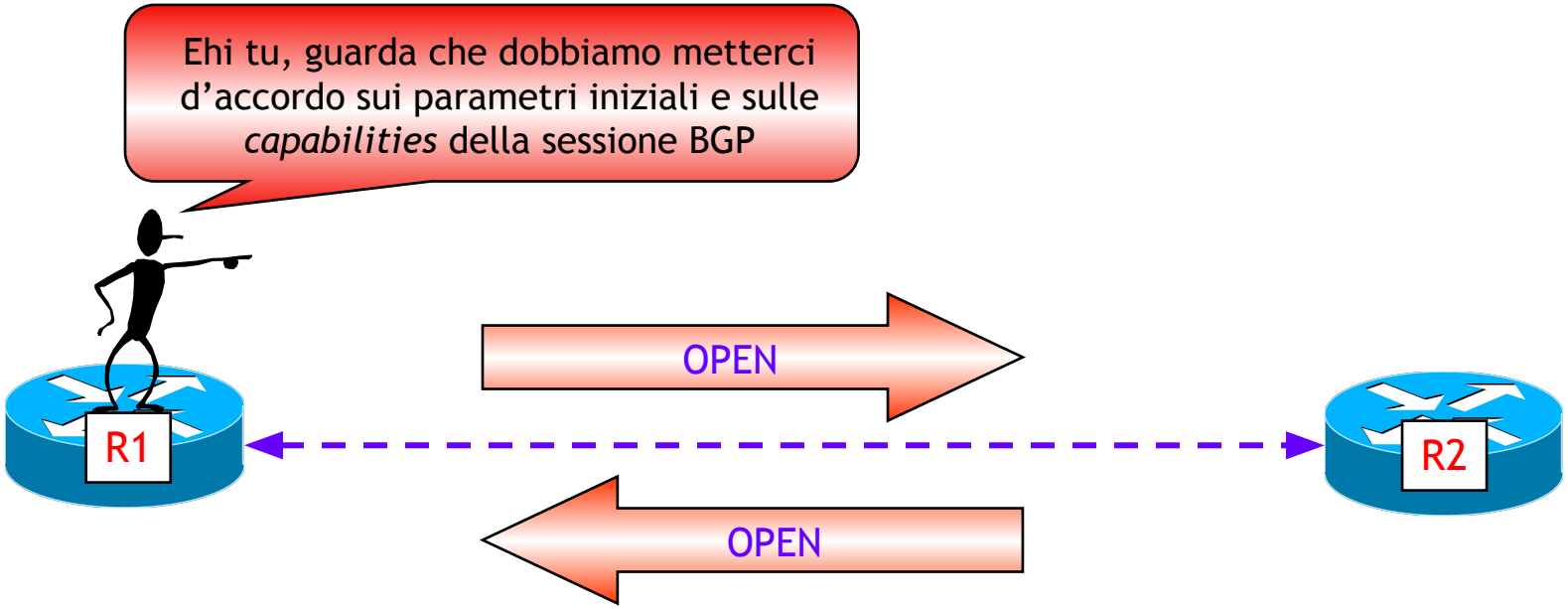
Fasi di costruzione (1/2)

- 1° passo: stabilire una connessione TCP



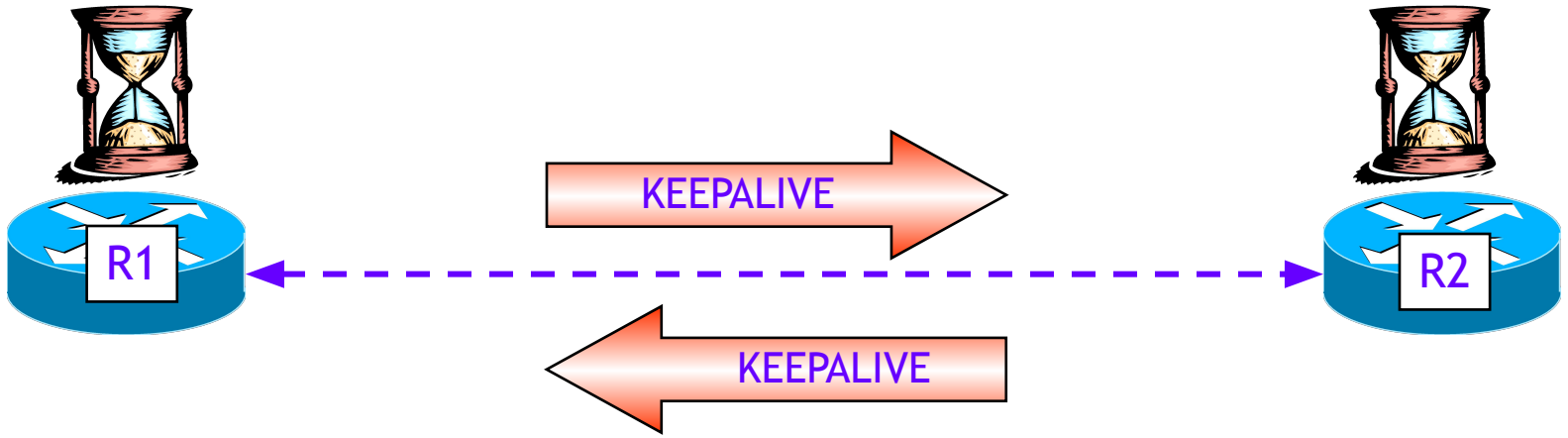
Fasi di costruzione (2/2)

- 2° passo: inizializzazione della sessione
  - Vengono negoziati parametri come Versione del protocollo e *Hold Time* oltre alle *capabilities* supportate (es. *Route Refresh*, *Outbound Route Filtering*, *famiglie di indirizzi*)
    - NOTA 1: differenti versioni del protocollo impediscono la realizzazione della sessione
    - NOTA 2: differenti valori di *Hold Time* non impediscono la realizzazione della sessione; i router scelgono come valore univoco il minimo tra i due
  - Viene utilizzato il messaggio BGP OPEN



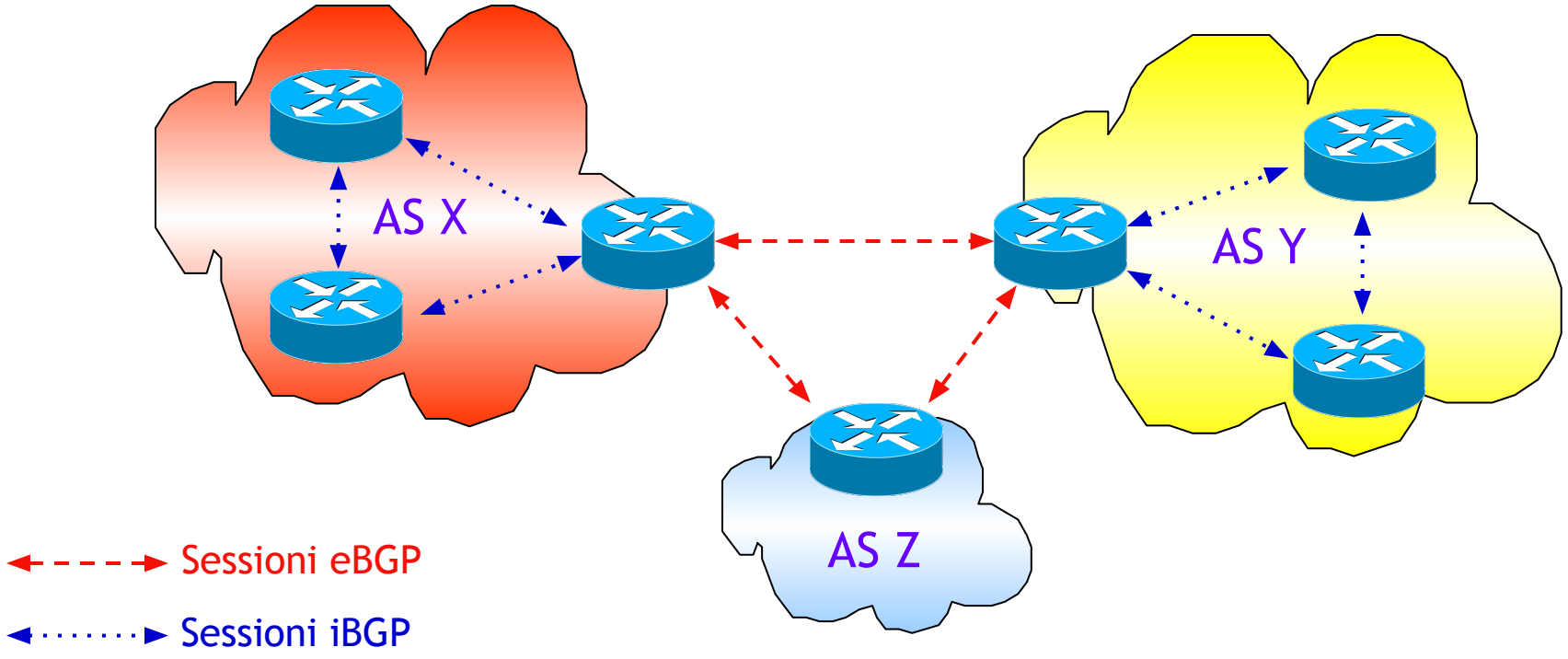
Mantenimento

- La sessione viene mantenuta attiva attraverso l'invio di normali messaggi BGP o in alternativa attraverso l'invio periodico di particolari messaggi BGP detti KEEPALIVE
  - Il periodo di invio dei messaggi KEEPALIVE è di norma configurabile manualmente (default Cisco: 60 *sec*; default Juniper 30 *sec*)
  - In caso di mancata ricezione di messaggi BGP UPDATE o KEEPALIVE per un tempo pari a *Hold Time* (di solito pari al mancato arrivo di 3 KEEPALIVE consecutivi) la sessione viene abbattuta



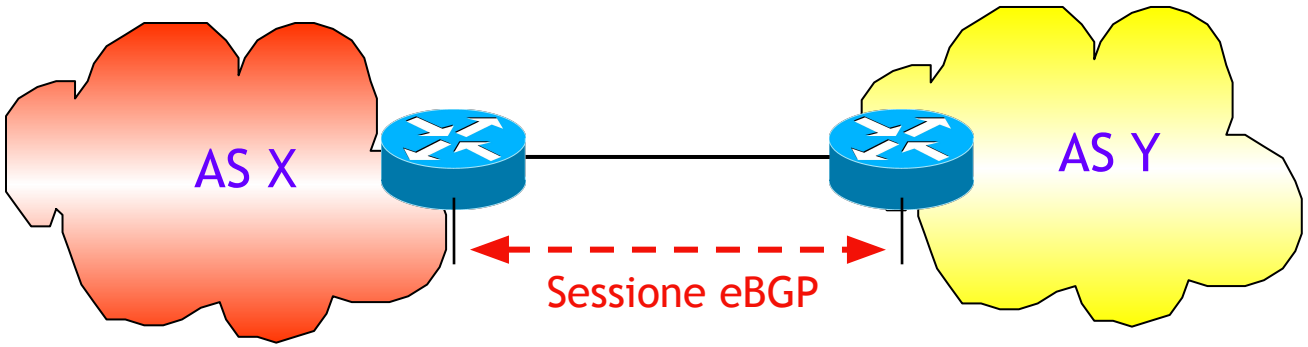
Tipi di Sessioni

- Il BGP prevede due tipi di sessioni
  - Sessioni *external* BGP (eBGP): sono sessioni stabilite tra router appartenenti ad AS diversi
  - Sessioni *internal* BGP (iBGP): sono sessioni stabilite tra router appartenenti allo stesso AS



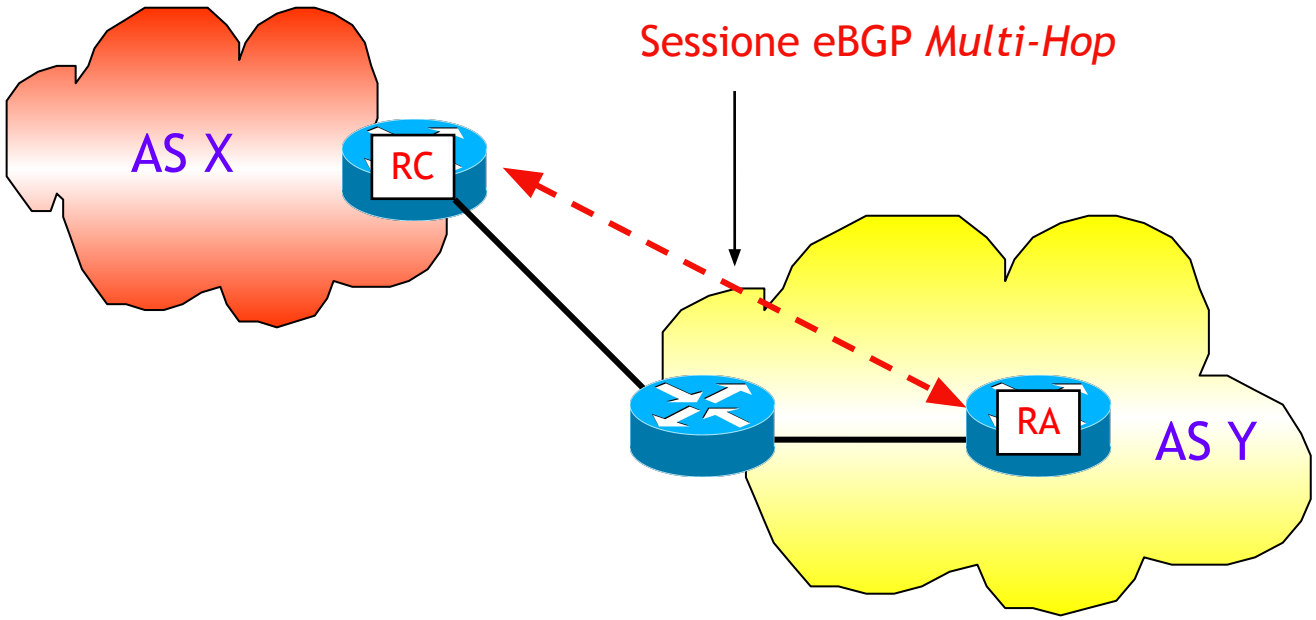
Sessioni eBGP: regole fondamentali (1/2)

- Di default le sessioni eBGP vengono stabilite tra router direttamente connessi (a Livello 2)
  - Di default i messaggi BGP su sessioni eBGP partono con TTL=1
  - È possibile variare il default tramite configurazione manuale (sessioni eBGP *Multi-Hop*)
- Per stabilire la sessione è possibile (e conveniente) utilizzare gli indirizzi IP associati alle interfacce fisiche/logiche agli estremi della sessione
  - Eccezione: collegamenti *multi-link* dove è conveniente utilizzare interfacce di Loopback



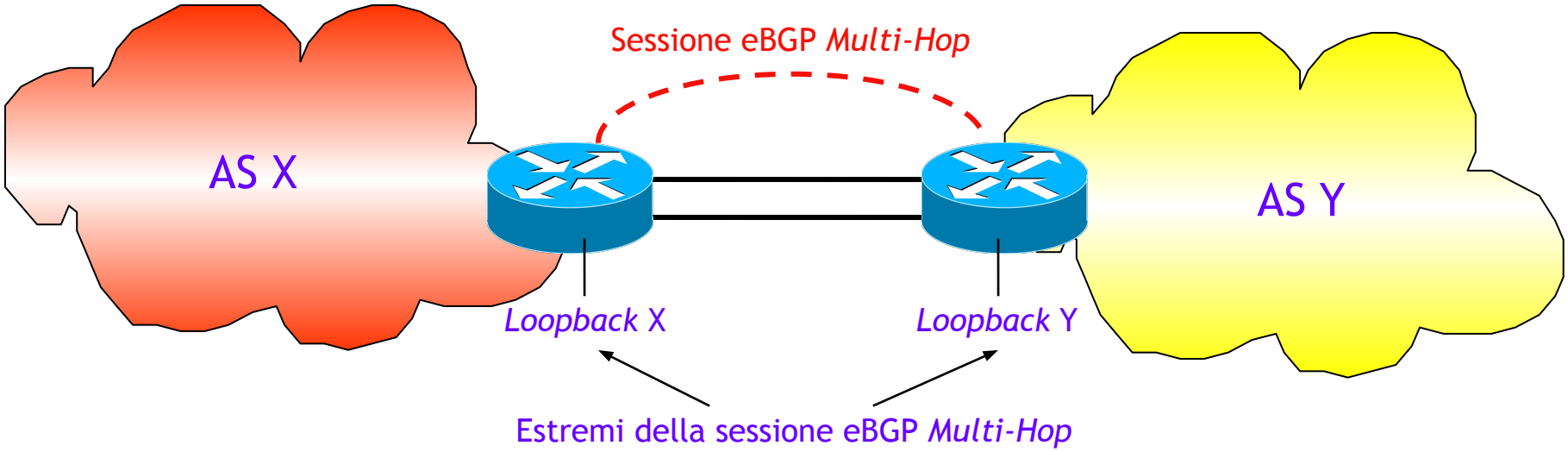
Sessioni eBGP *Multi-Hop*

- Nella configurazione di sessioni eBGP *Multi-Hop* è necessario variare il TTL di default (=1) con cui vengono generati i messaggi BGP
  - Qualora, come nel caso di collegamenti *multi-link*, si utilizzino interfacce di *Loopback* per gli estremi della sessione, è necessario configurare una sessione *Multi-Hop*



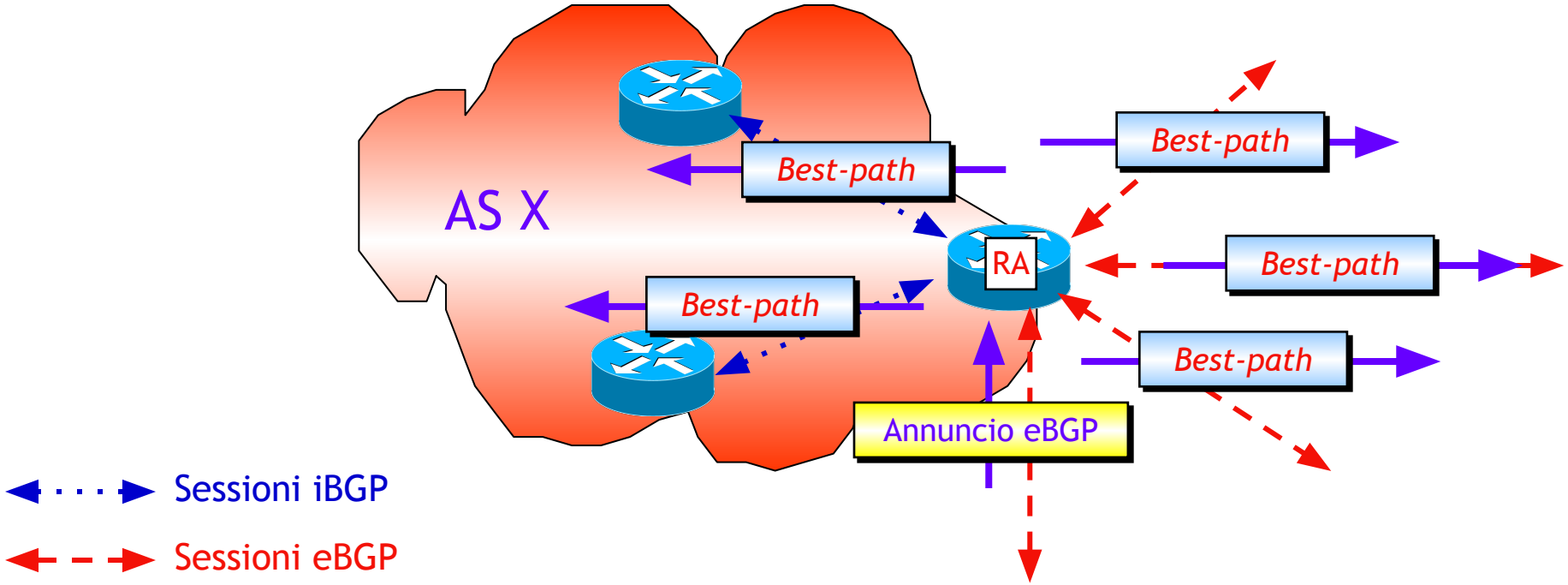
Sessioni eBGP nei collegamenti *Multi-Link*

- Nel caso di router con più collegamenti è conveniente (e consigliato) utilizzare per gli estremi della sessione eBGP due interfacce di *Loopback*
  - Vantaggio: il fuori servizio di uno qualsiasi dei collegamenti fisici non fa cadere la sessione eBGP
  - Necessaria la configurazione di una sessione eBGP *Multi-Hop*



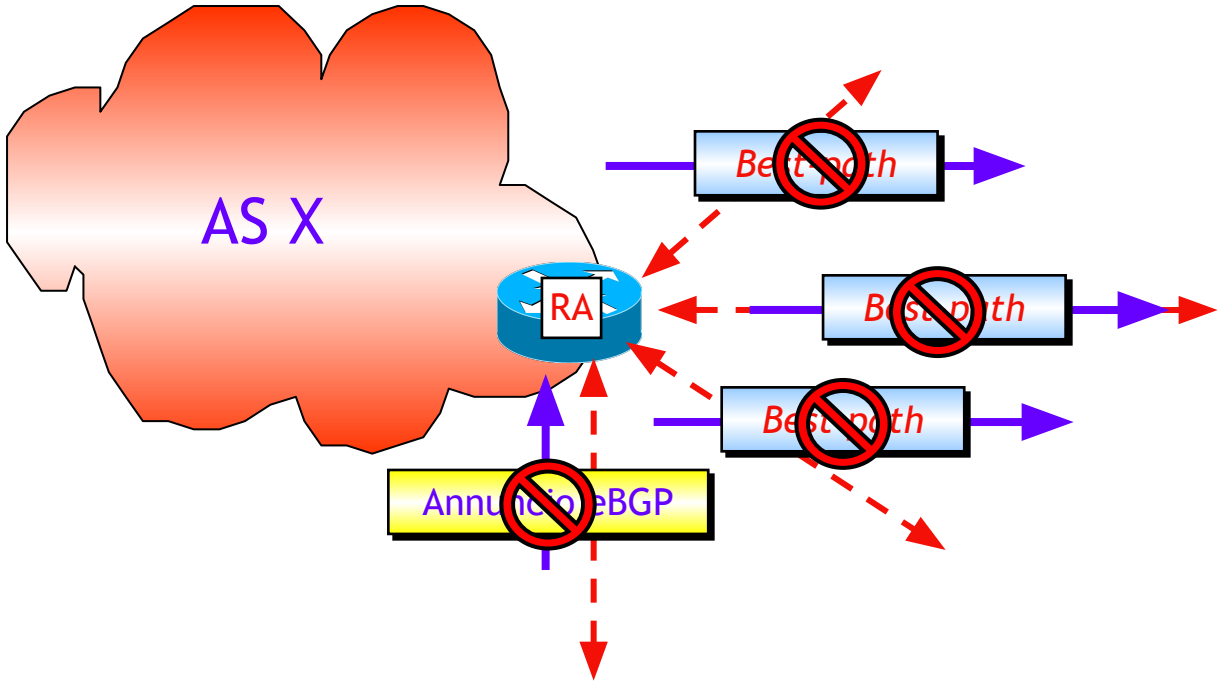
Sessioni eBGP: regole fondamentali (2/2)

- Ogni annuncio BGP ricevuto da un router su una sessione eBGP viene sottoposto al processo di selezione
- Il *best-path* viene propagato automaticamente su tutte le sessioni BGP attive (eBGP e iBGP) ad eccezione quella dalla quale è stato ricevuto l'annuncio



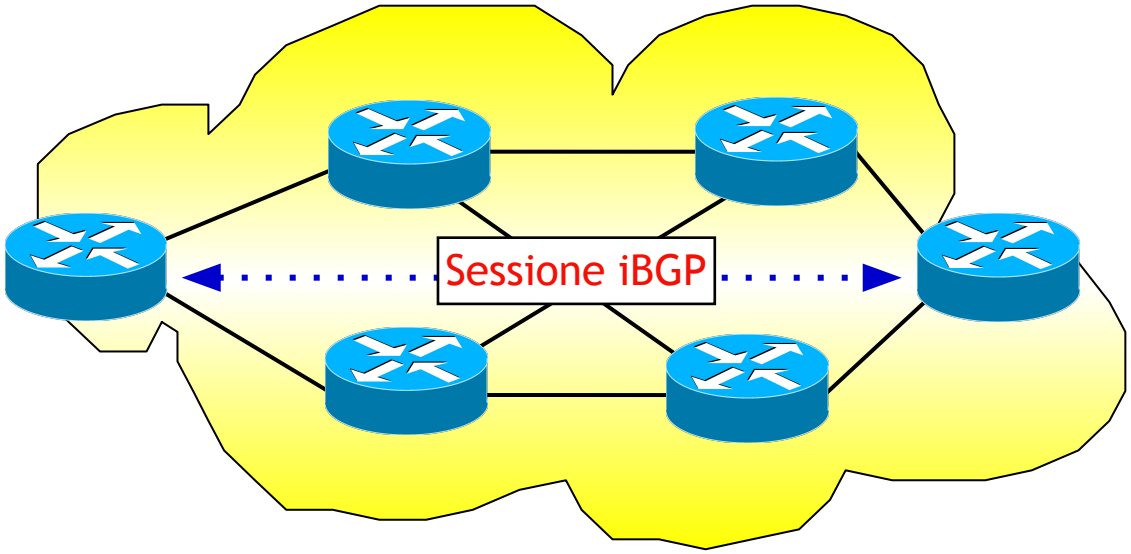
La RFC 8212

- La RFC 8212 - *Default External BGP (EBGP) Route Propagation Behavior without Policies*, luglio 2017, ha variato comportamento di *default* nelle sessioni eBGP come segue: nessun annuncio entrante e/o uscente può essere accettato/propagato se non sia stata esplicitamente configurata una politica di routing *inbound* e/o *outbound*



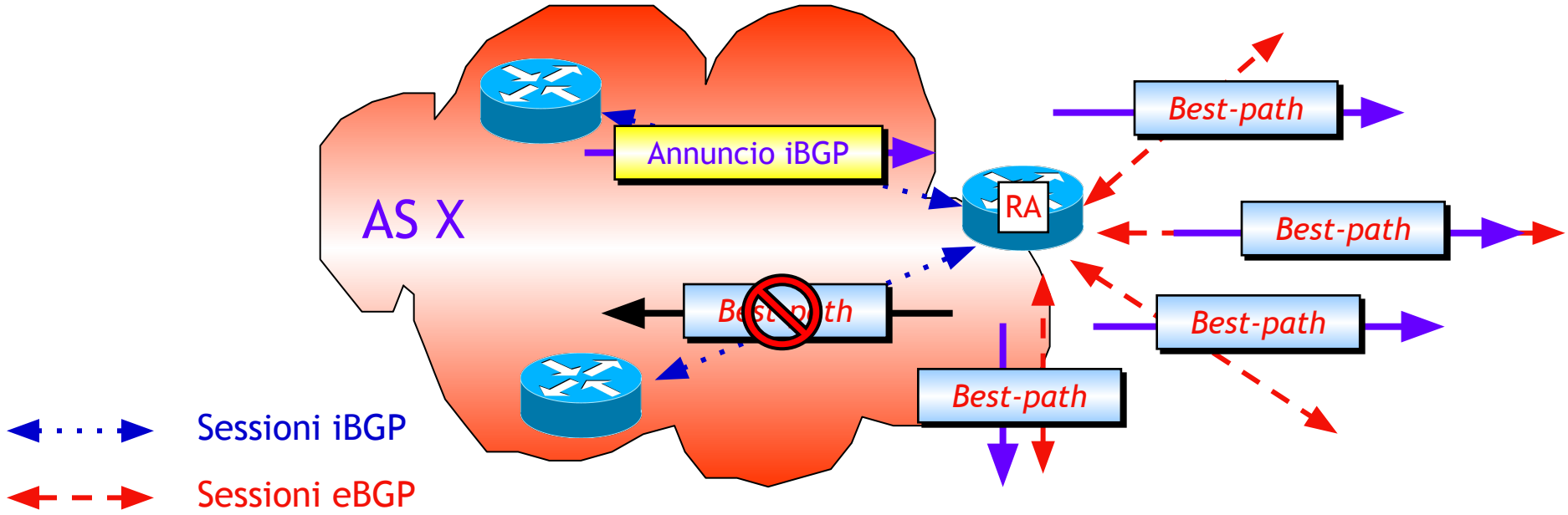
Sessioni iBGP: regole fondamentali (1/2)

- Le sessioni iBGP possono essere stabilite anche tra router non direttamente connessi
  - Requisito fondamentale è che via connettività a livello IP tra i router
- Per stabilire la sessione è possibile (e consigliato) utilizzare delle interfacce di *Loopback*
  - Vantaggio: fino a quando esiste un percorso fisico tra i due router la sessione iBGP non cade

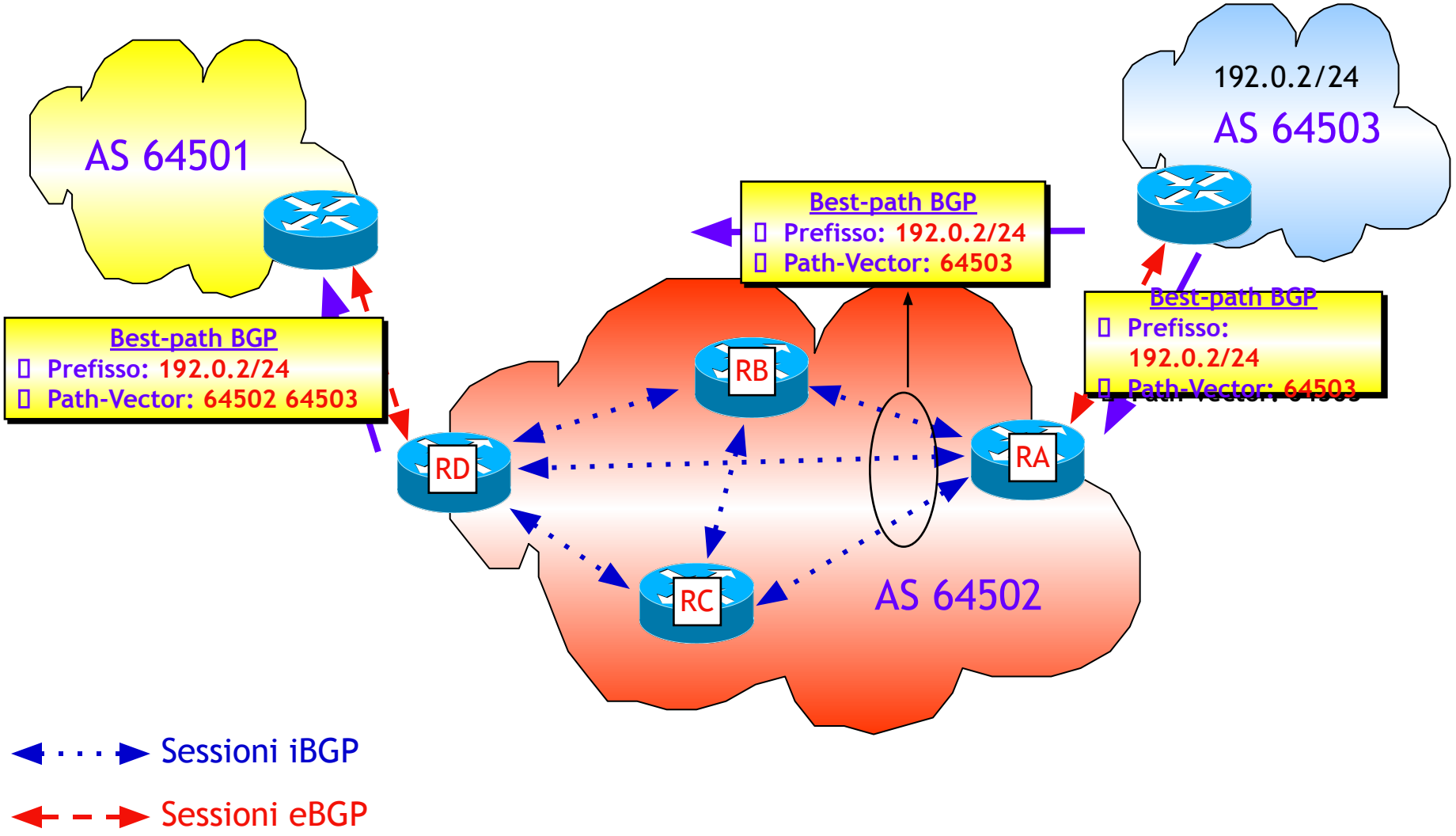


Sessioni iBGP: regole fondamentali (2/2)

- Ogni annuncio BGP ricevuto da un router su una sessione iBGP viene sottoposto al processo di selezione
- Il *best-path* viene propagato automaticamente solo sulle sessioni eBGP attive (regola dello *Split Horizon*)
  - Vi è una eccezione a questa regola, che riguarda i *Route Reflector*



Esempio di propagazione degli annunci



Concetti fondamentali

Caratteristiche principali
Autonomous System
Connettività Clienti-ISP
Funzionamento di base
Sessioni BGP
Messaggi e attributi
Processo di selezione

Tipi di Messaggi

Sessione

- OPEN
- KEEPALIVE

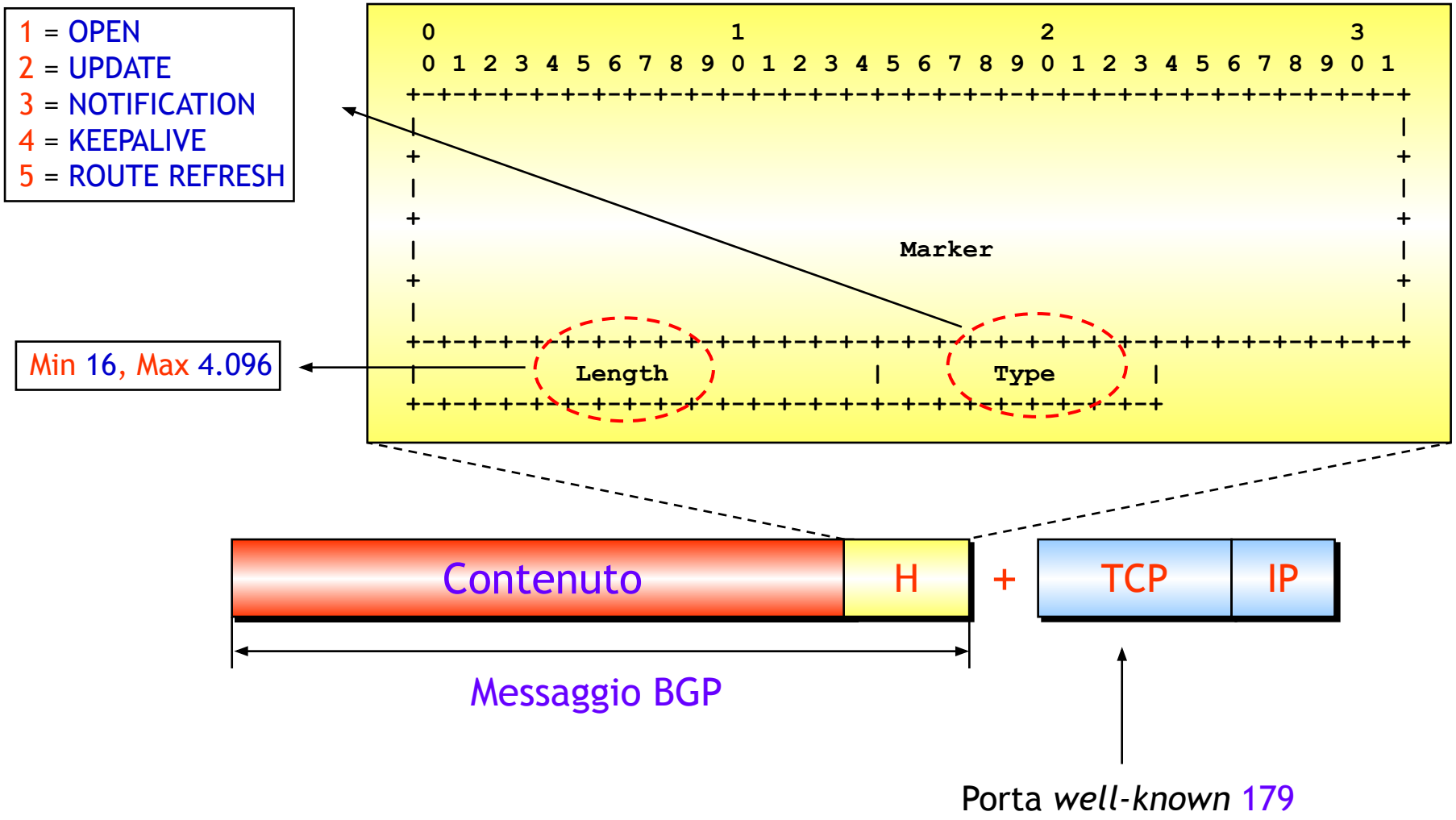
Notifica

- NOTIFICATION

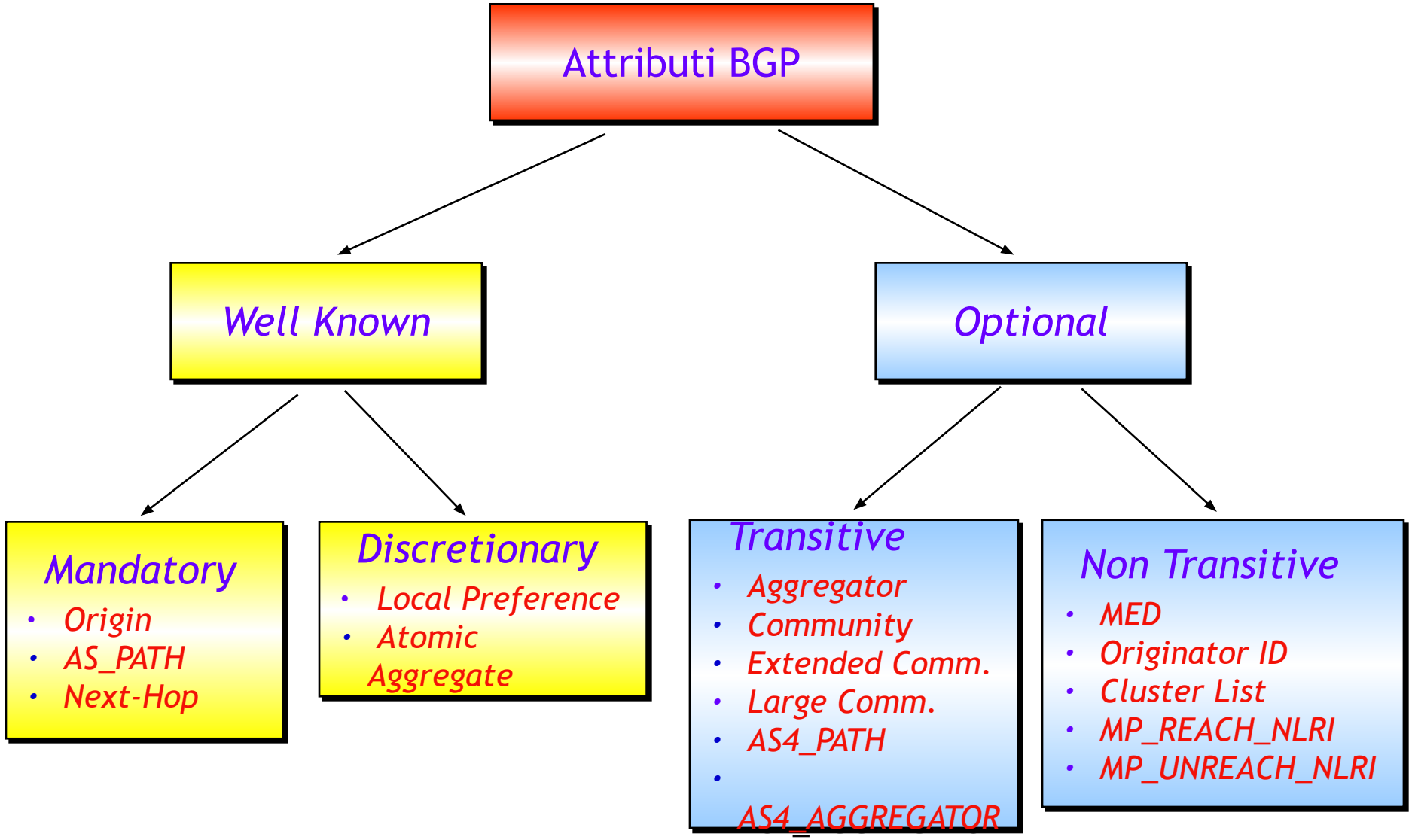
Annunci

- UPDATE
- ROUTE REFRESH

Formato

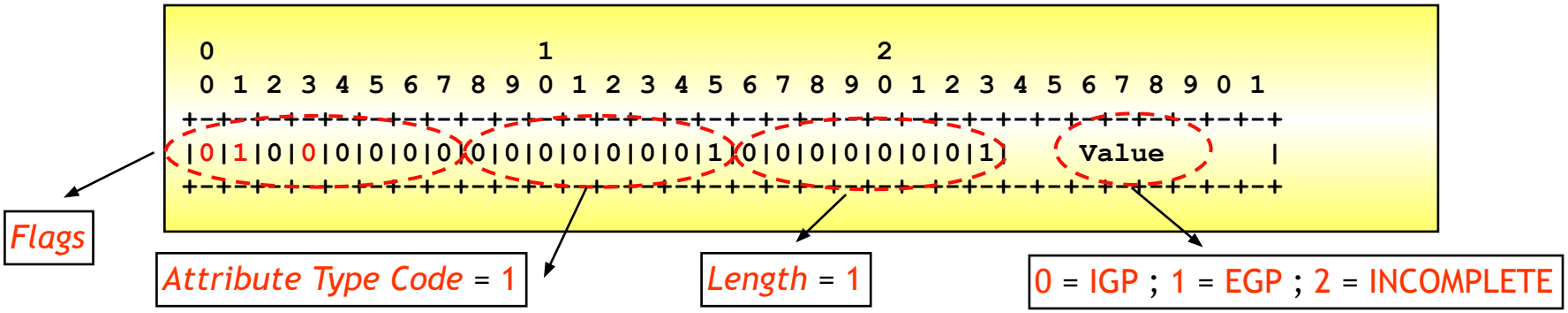

1 = OPEN
2 = UPDATE
3 = NOTIFICATION
4 = KEEPALIVE
5 = ROUTE REFRESH
0 1 2 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
Marker
Length
Type
Min 16, Max 4.096
Contenuto
H
+
TCP
IP
Messaggio BGP
Porta well-known 179

Attributi BGP: classificazione



Attributo *Origin*

- È un attributo *Well Known Mandatory* che specifica l'origine di un prefisso IP (*Attribute Type Code* =1)
- Può assumere tre diversi valori
  - IGP: il prefisso IP è originato direttamente dal router
  - EGP: il prefisso IP è stato appreso attraverso il protocollo EGP (ormai non più usato)
  - INCOMPLETE: il prefisso IP è stato appreso via redistribuzione in BGP
- È utilizzato nel processo di selezione BGP secondo il seguente ordine di preferenza: IGP → EGP → INCOMPLETE



Attributo AS_PATH: regole fondamentali (1/2)

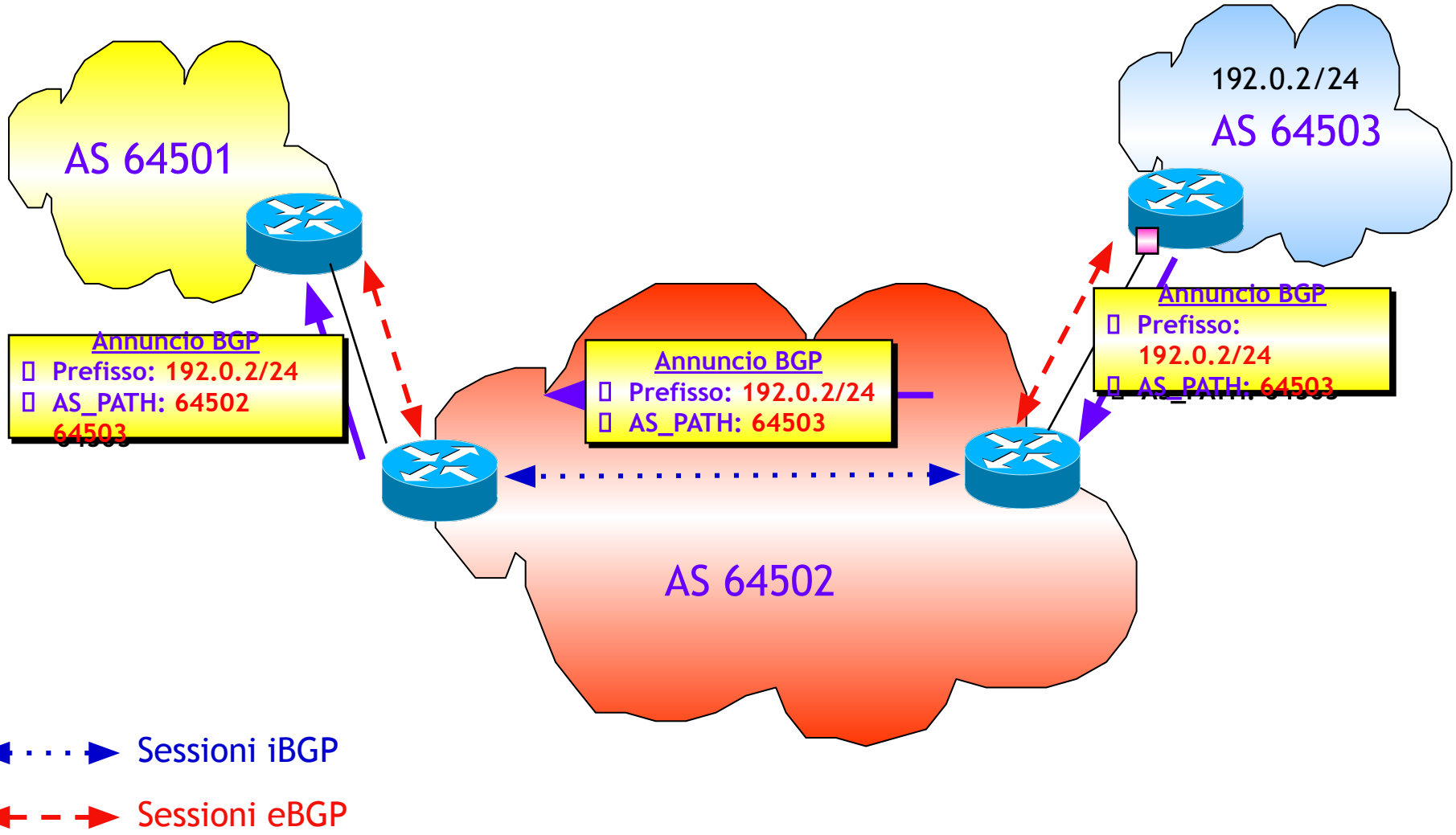
- Quando un *BGP speaker* invia un messaggio UPDATE a un *iBGP peer* l'attributo AS_PATH non viene modificato

- Quando un *BGP speaker* invia un messaggio UPDATE a un *eBGP peer* l'attributo AS_PATH viene modificato nel seguente modo
  - Se il primo segmento dell'AS_PATH è del tipo AS_SEQUENCE, il router che inoltra l'annuncio inserisce il proprio numero di AS in testa alla lista
  - Se il primo segmento dell'AS_PATH è del tipo AS_SET, il router che inoltra l'annuncio inserisce un nuovo segmento di tipo AS_SEQUENCE contenente il solo proprio numero di AS
  - Se l'AS_PATH è vuoto, il router che inoltra l'annuncio inserisce un nuovo segmento di tipo AS_SEQUENCE contenente il solo proprio numero di AS

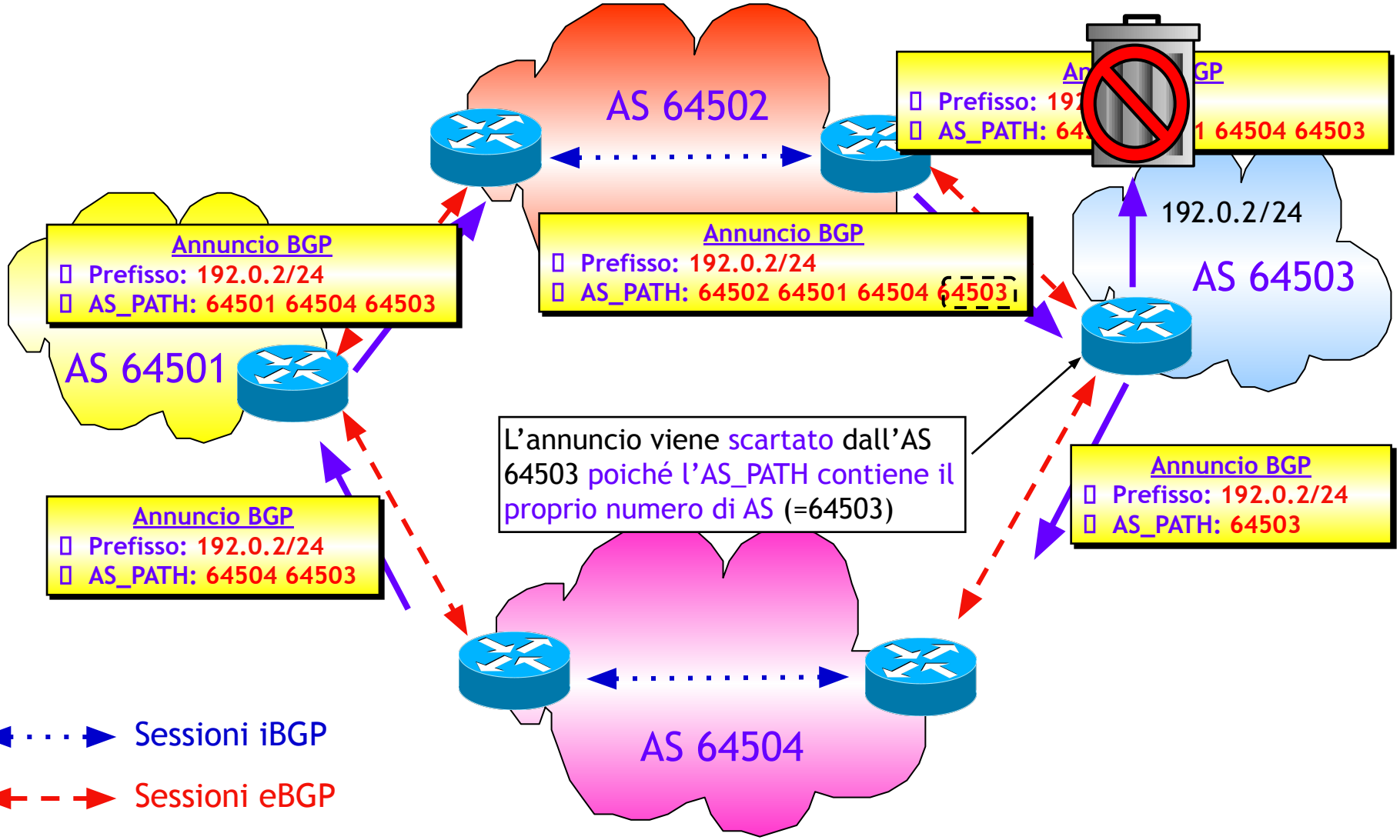
Attributo AS_PATH: regole fondamentali (2/2)

- Qualora un router effettua aggregazione di prefissi è possibile conservare memoria degli AS attraversati dai prefissi oggetto di aggregazione
  - Poiché l'ordine in questo caso non ha importanza, si aggiunge un segmento AS_PATH di tipo AS_SET
- Un router scarta gli annunci ricevuti che contengono nell'AS_PATH il proprio numero di AS (*Loop Prevention*)
- L'attributo AS_PATH è utilizzato dal processo di selezione BGP come possibile metrica
  - Viene scelto il percorso con minimo numero di AS contenuti nell'attributo
  - È possibile aumentare in modo fittizio il numero di AS contenuti nell'attributo per forzare il traffico in entrata ad un AS (*AS_PATH Prepending*)

Attributo AS_PATH: esempio

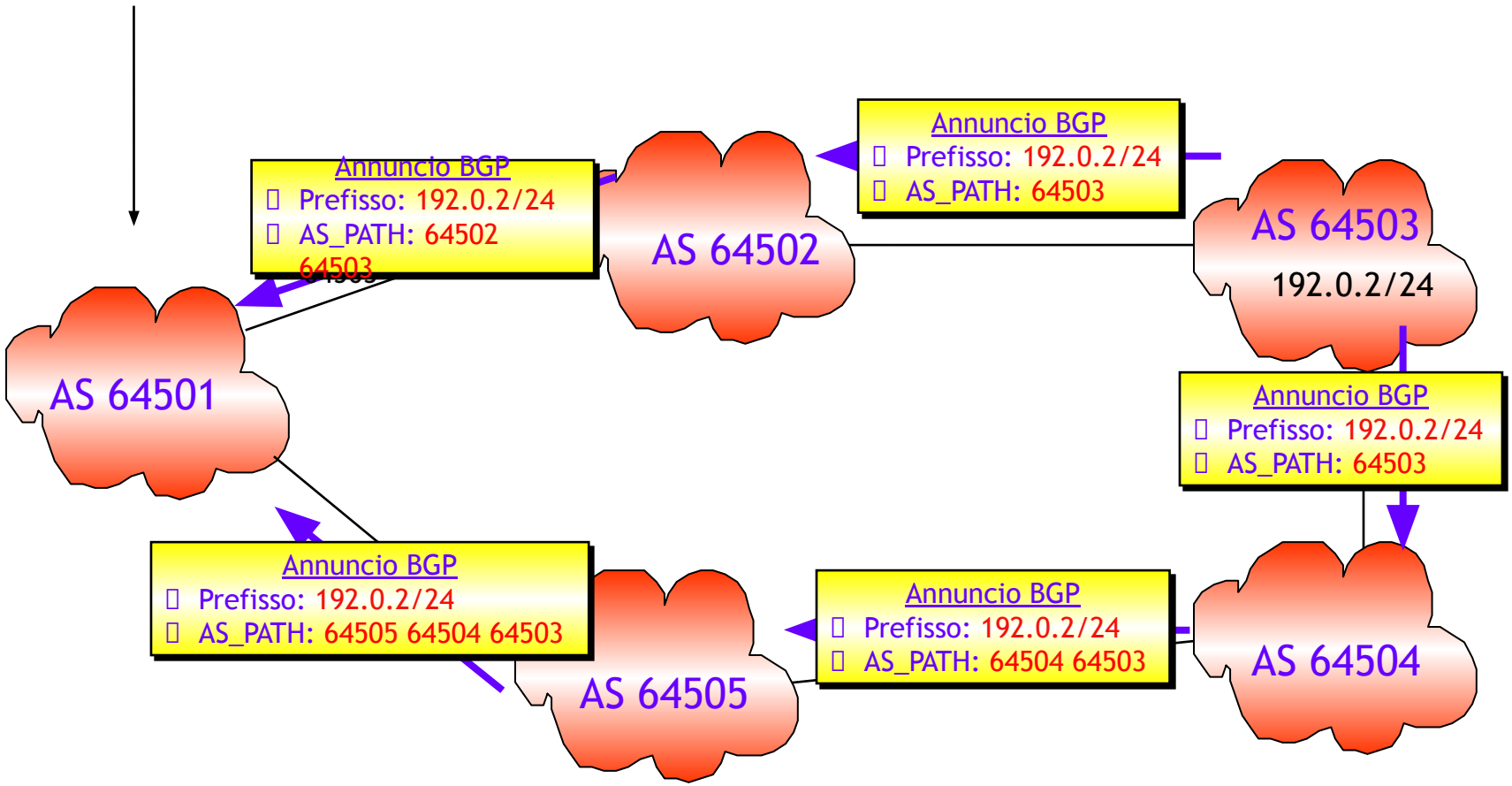


Attributo AS_PATH: *loop prevention*



Attributo AS_PATH: metrica

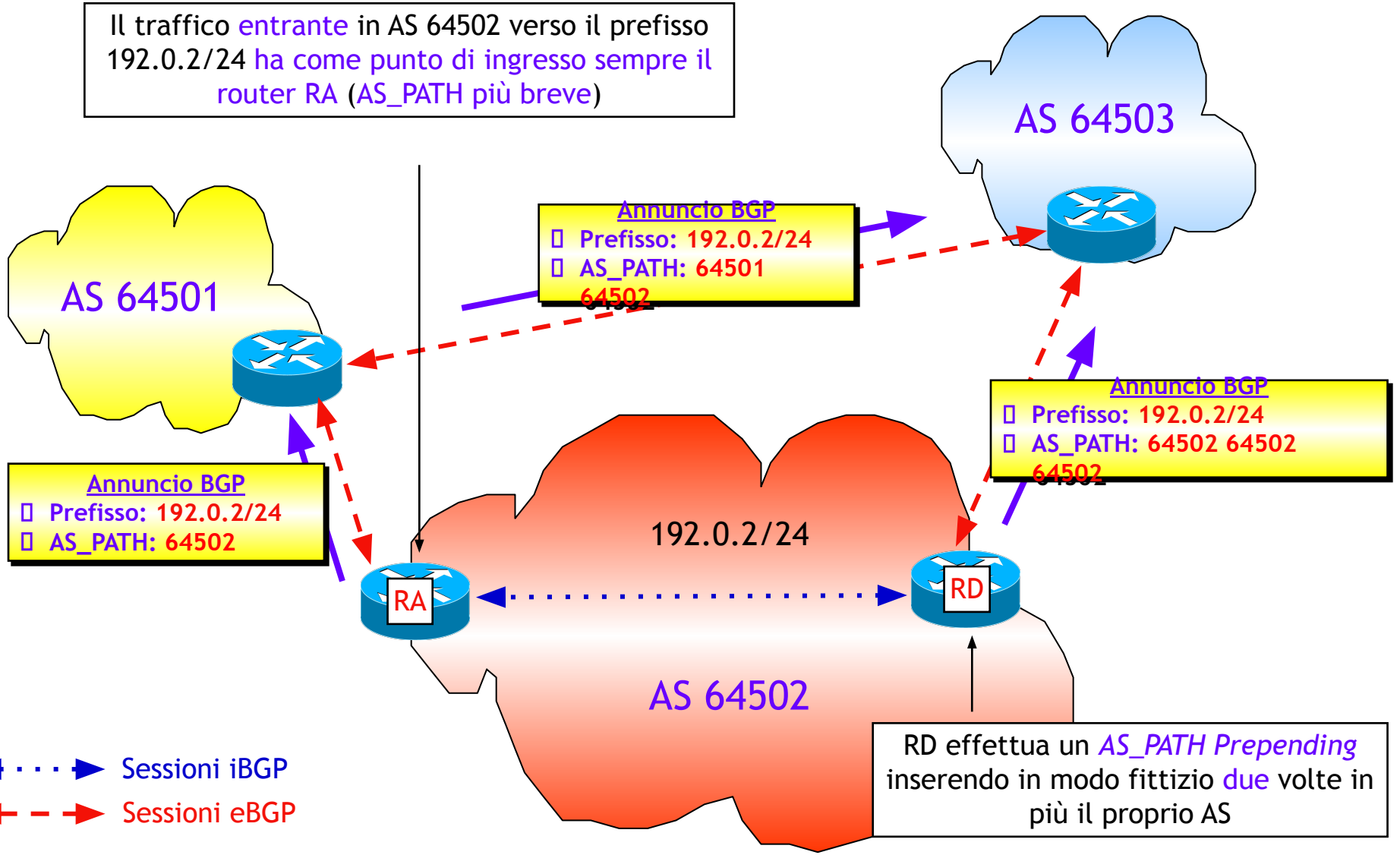
AS 64501 sceglie di raggiungere 192.0.2/24 via AS 64502
poiché attraversa un numero minore di AS



Attributo AS_PATH: *AS_PATH Prepending*

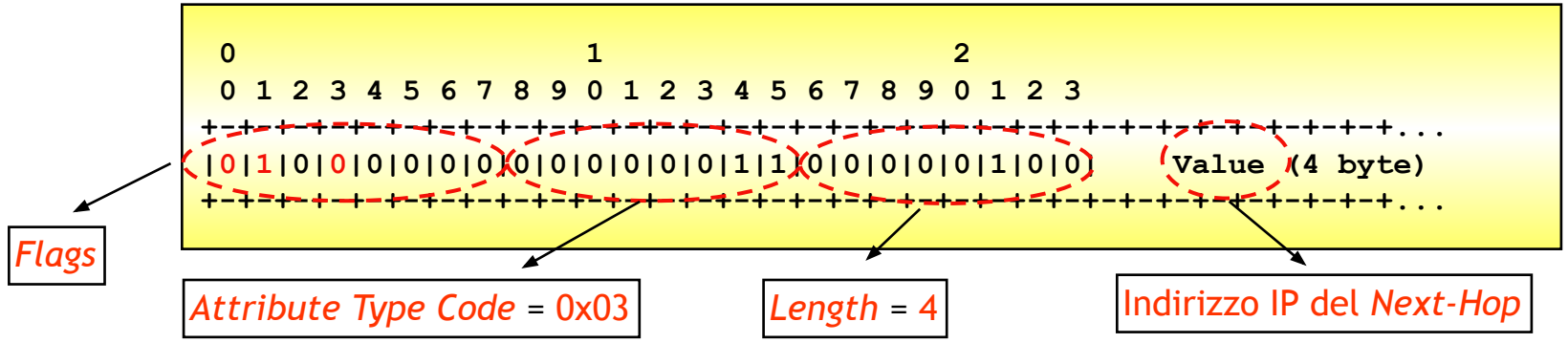
REISS ROMOLI

Il traffico entrante in AS 64502 verso il prefisso 192.0.2/24 ha come punto di ingresso sempre il router RA (AS_PATH più breve)



Attributo *Next-Hop*

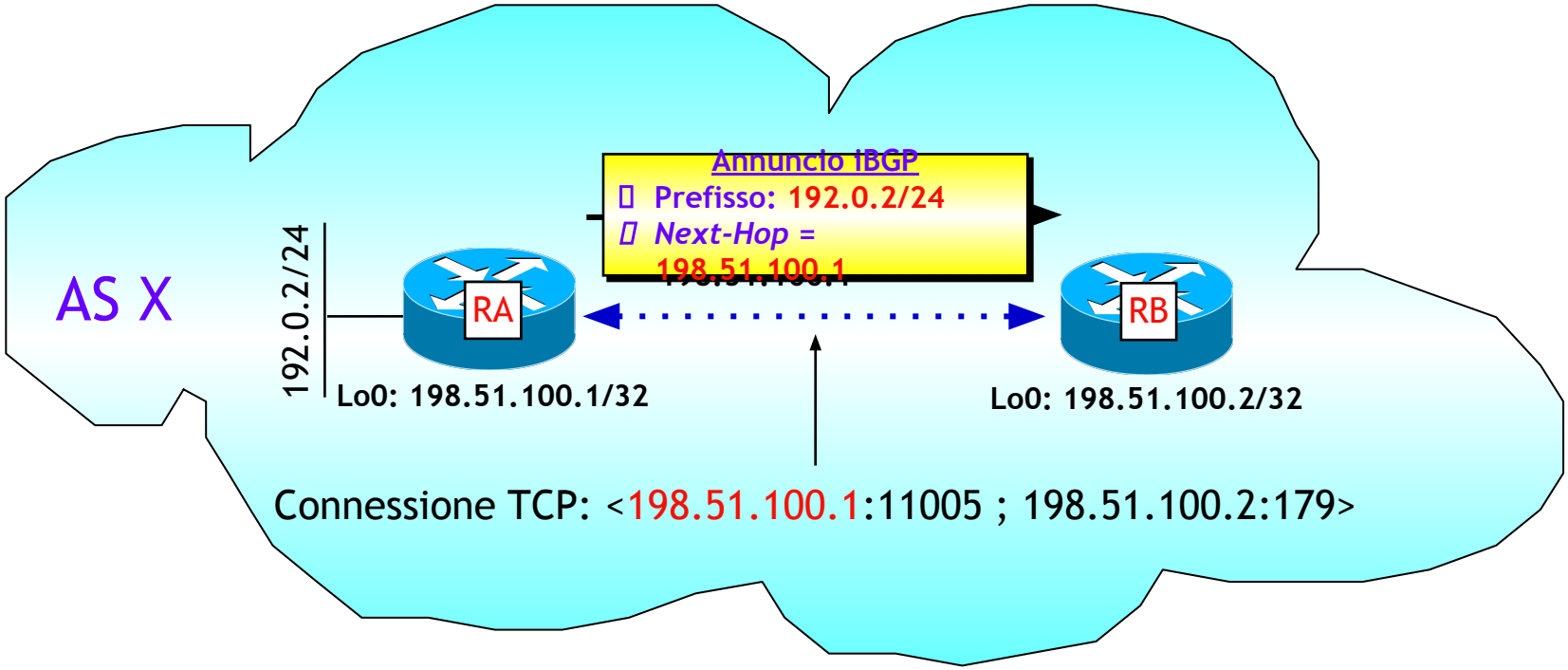
- È un attributo *Well Known Mandatory* che specifica il Next-Hop da utilizzare per i prefissi contenuti nel campo NLRI (*Attribute Type Code* = 3)
- Regole fondamentali
  - Di default il campo *Next-Hop* viene modificato solo quando l'annuncio viene inoltrato su una sessione eBGP
    - È possibile variare questa regola su base configurazione
  - L'indirizzo IP del campo *Next-Hop* deve essere presente nella tabella di routing IP del router che riceve l'annuncio, altrimenti questo viene scartato



Attributo *Next-Hop*: regole base (1/3)

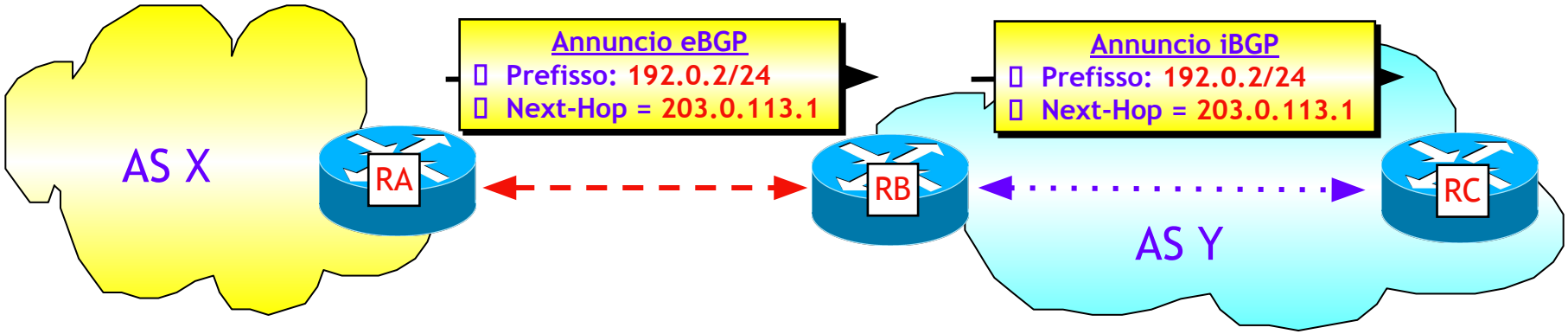
- L'attributo NEXT_HOP viene modificato solo quando gli annunci BGP vengono inoltrati su sessioni eBGP
  - L'indirizzo IP è quello utilizzato dal *BGP speaker* per aprire la connessione TCP verso il *BGP peer*

Attributo *Next-Hop*: regole base (2/3)



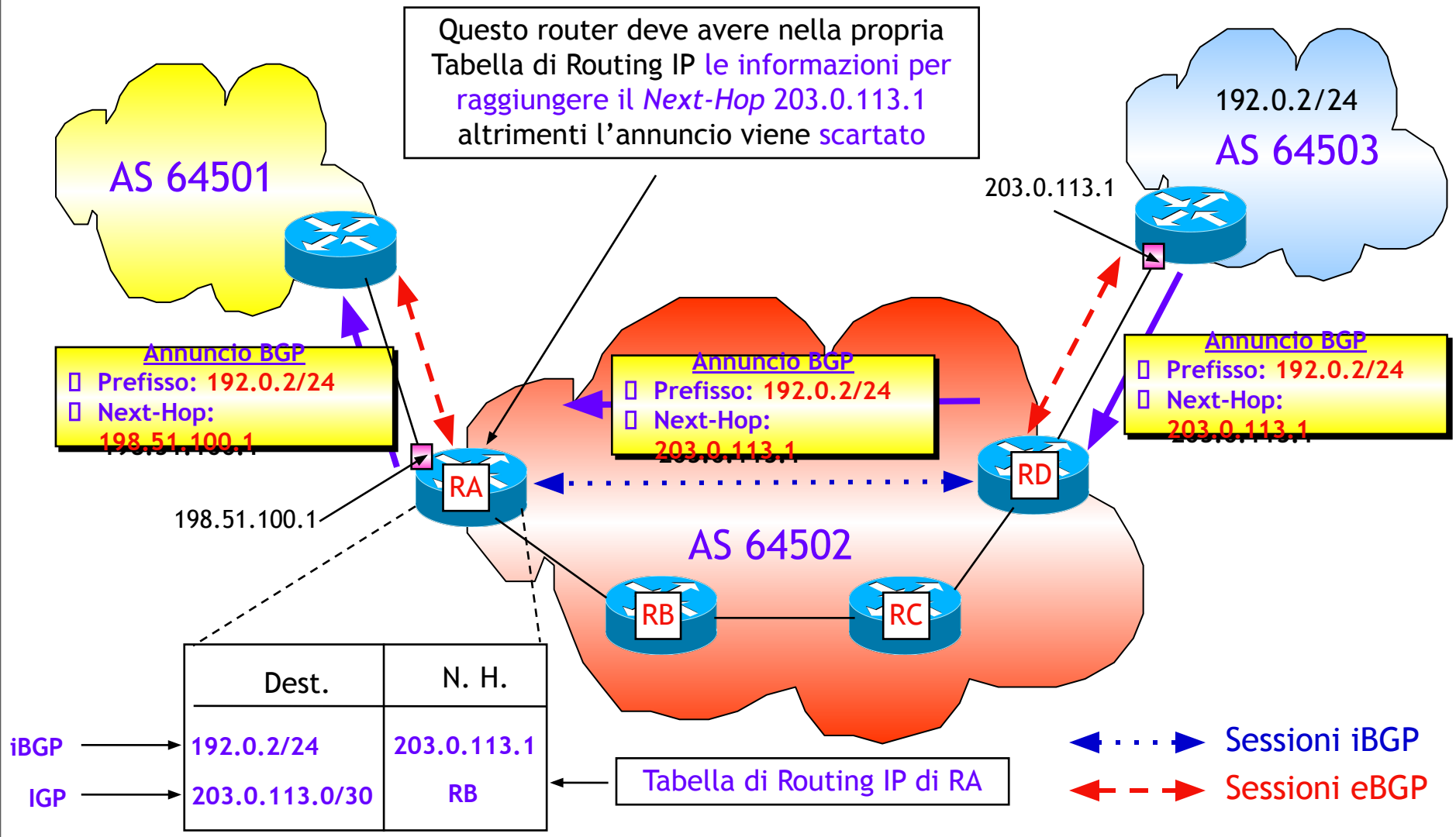
- Se un annuncio BGP di un prefisso locale ad un router viene propagato su una sessione iBGP, l'indirizzo IP inserito nell'attributo *Next-Hop* è quello utilizzato dal *BGP speaker* che propaga l'annuncio per aprire la connessione TCP verso il *BGP peer*

Attributo *Next-Hop*: regole base (3/3)

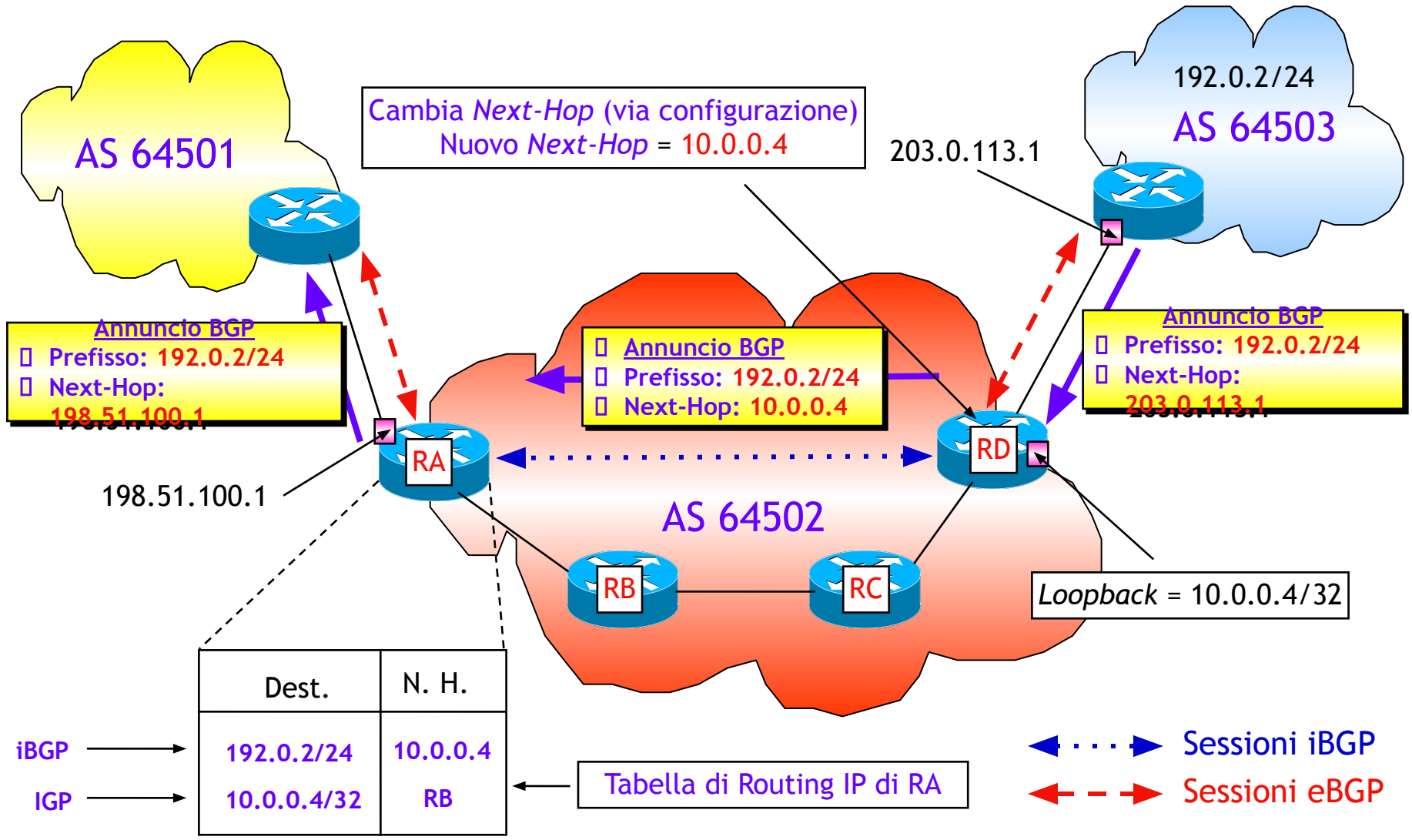


- Se un annuncio di un prefisso esterno ad un AS viene propagato su una sessione iBGP, l'attributo *Next-Hop* non cambia

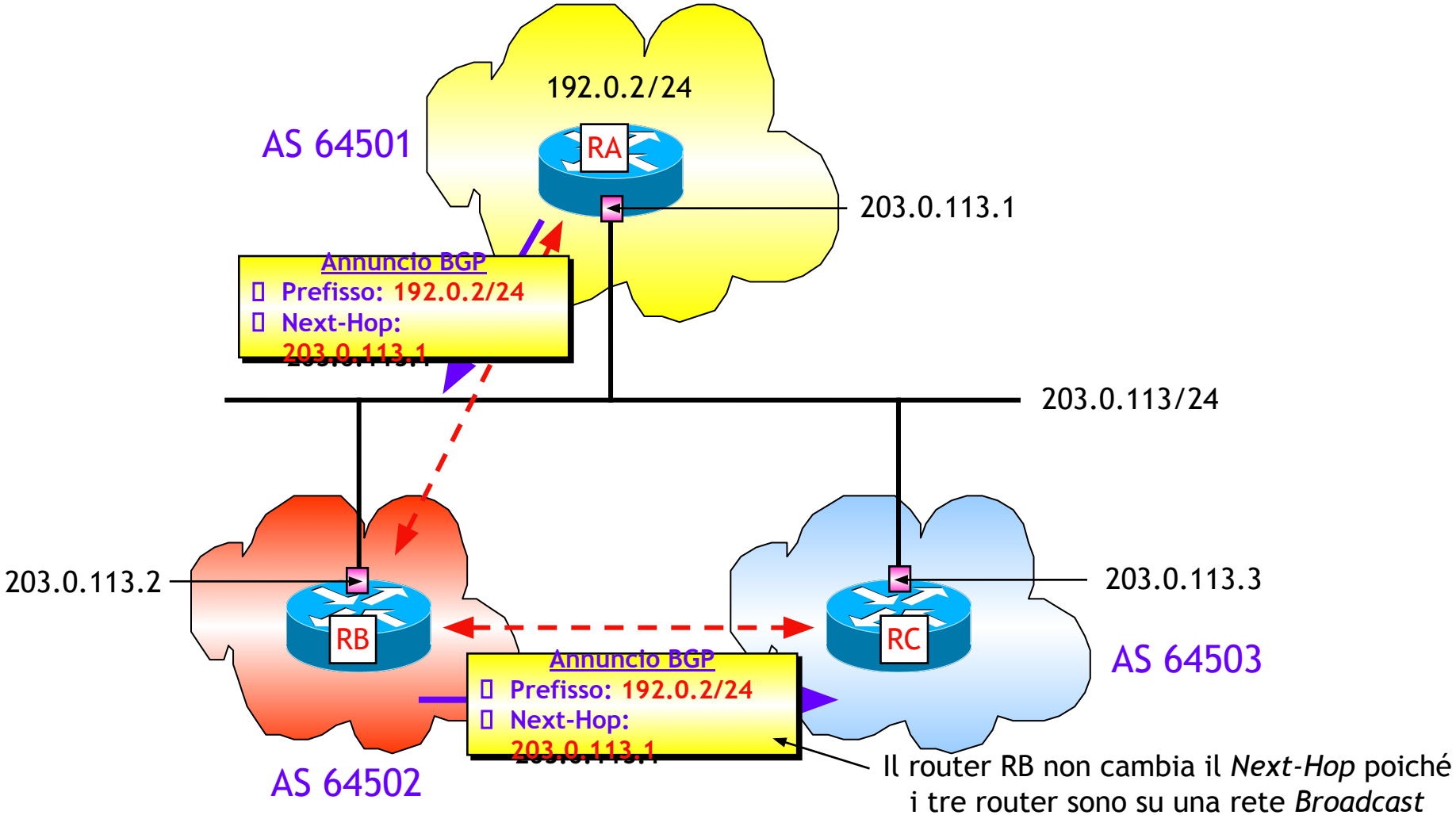
Attributo *Next-Hop*: esempio 1



Attributo *Next-Hop*: esempio 2



Next-Hop nelle reti *Broadcast*

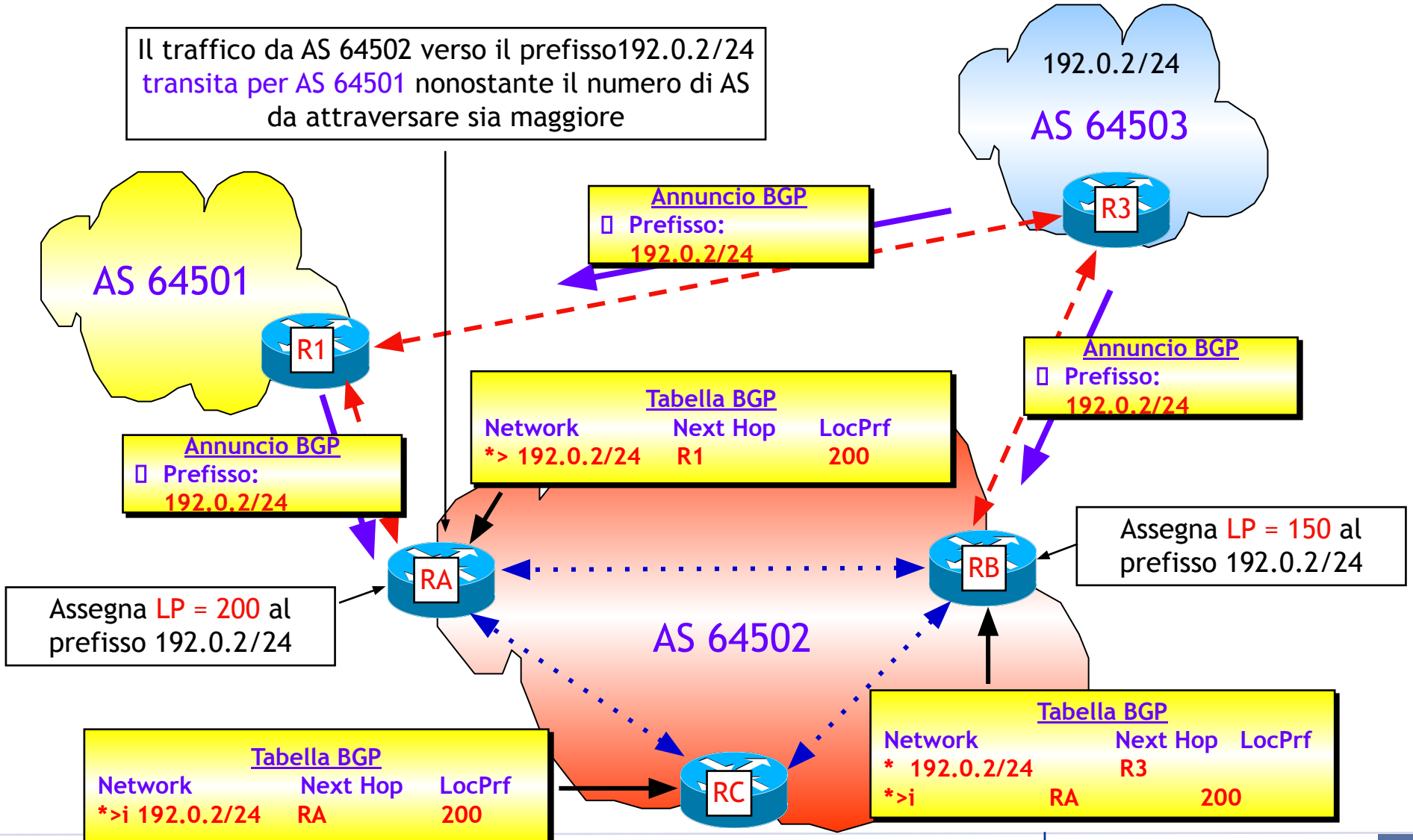


Attributo *Local Preference*: regole fondamentali

- Il valore di LP viene assegnato (su base configurazione o default) da router che ricevono un annuncio e viene propagato solo all'interno dell'AS
  - Questo permette a tutti i router di un AS di scegliere un comune punto di uscita per per il traffico destinato verso un determinato prefisso (o verso tutti)
  - Nei router Cisco e Juniper il valore di default è 100

- Un router seleziona come punto di uscita dall'AS per un determinato prefisso quello che ha valore associato di *Local Preference* più elevato

Attributo *Local Preference*: esempio

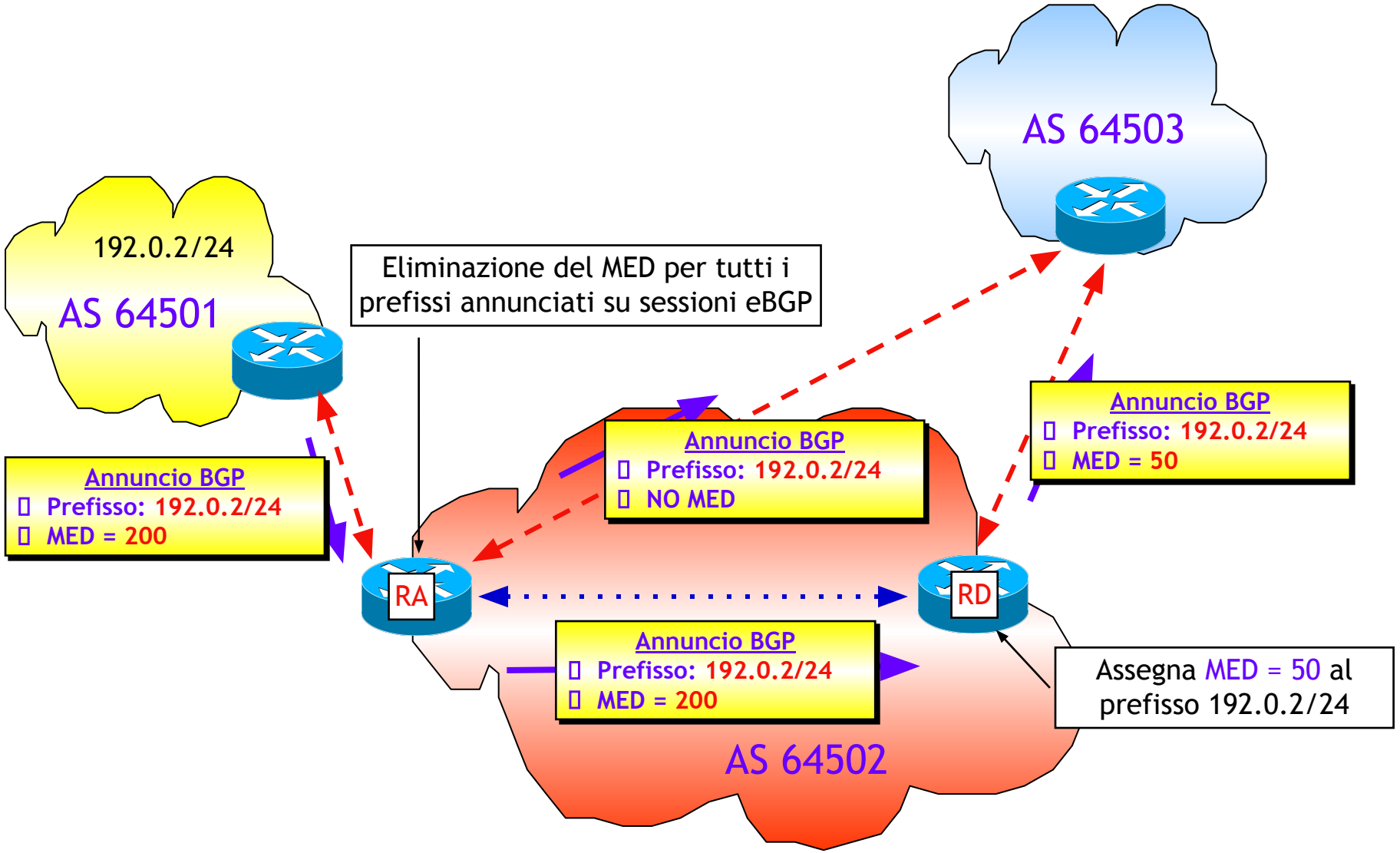


REISS ROMOLI

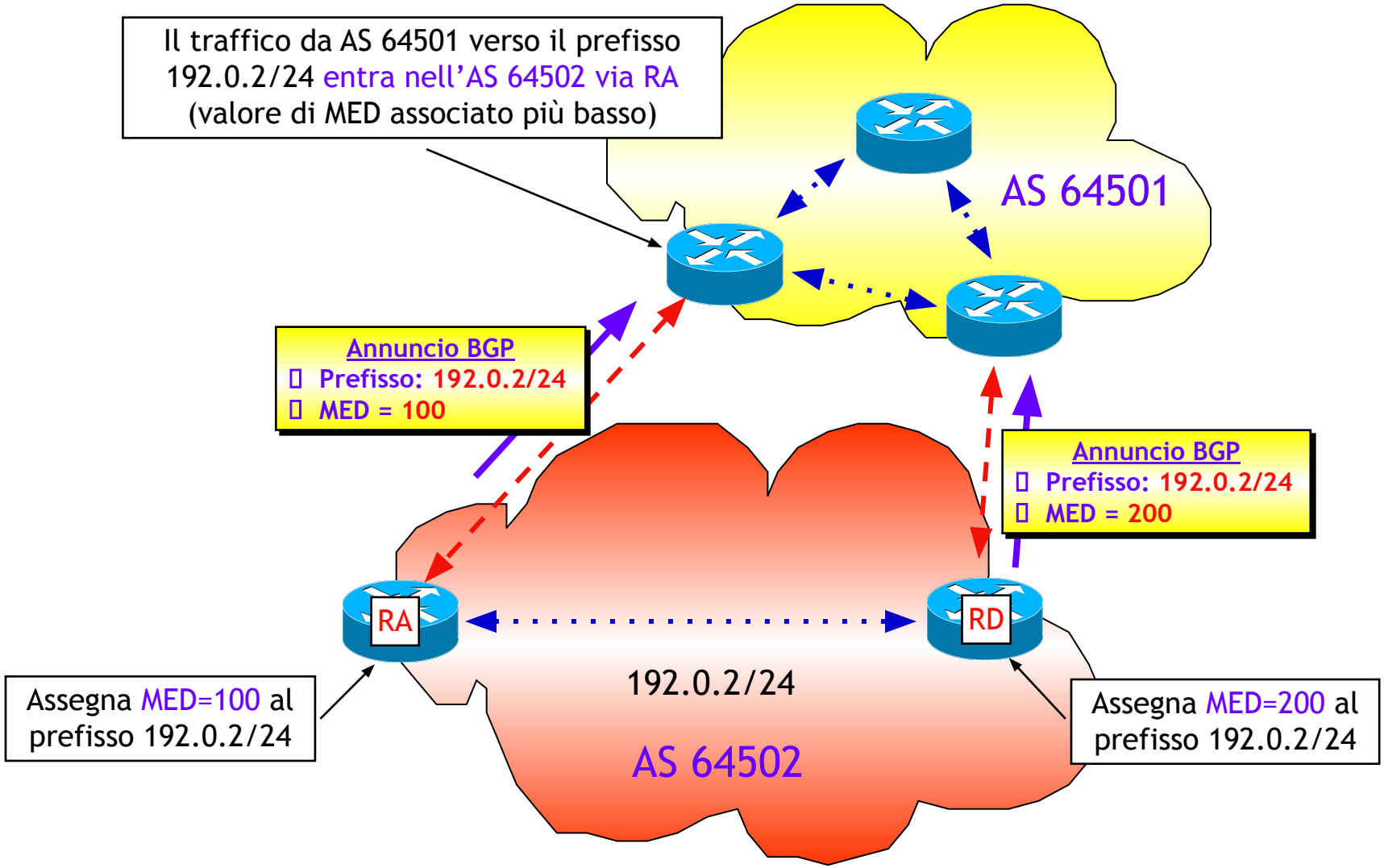
Attributo MED: regole fondamentali

- Il valore del MED viene assegnato su base configurazione (default = 0) da router che emettono un annuncio
- Un router che riceve un annuncio con un determinato valore di MED non propaga il valore all'esterno dell'AS
  - Quando l'annuncio viene propagato ad altri AS il MED viene tolto (a meno che non venga ridefinito su base configurazione)
- Di default un router confronta i valori di MED solo in annunci provenienti dallo stesso AS
  - Questo comportamento può essere variato su base configurazione
- Un router seleziona come punto di ingresso nell'AS per un determinato prefisso quello che ha valore associato di MED più basso

Attributo MED: propagazione

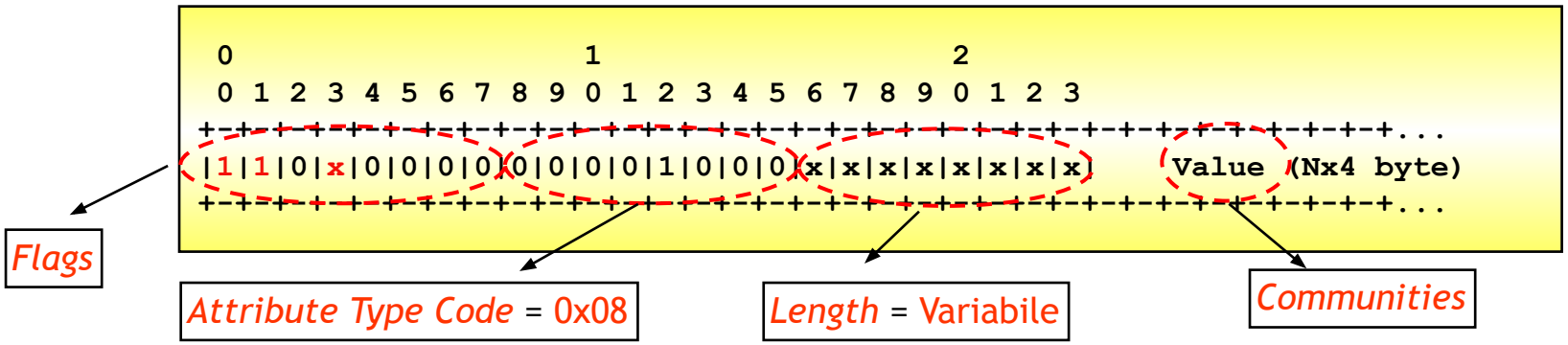


Attributo MED: esempio



Attributo *Community*

- È un attributo *Optional Transitive* che specifica un gruppo di prefissi che condividono le stesse proprietà (*Attribute Type Code* = 8)
  - Ciascun attributo Community ha lunghezza 4 *byte*
    - Esistono anche attributi di tipo *Extended Community* (8 *byte*) e *Large Community* (12 *byte*)
  - NOTA: è possibile associare più *Community* a ciascun prefisso o gruppo di prefissi



Attributo *Community*: regole fondamentali

- I valori dell'attributo *Community* vengono assegnati su base configurazione sia agli annunci in uscita che in ingresso

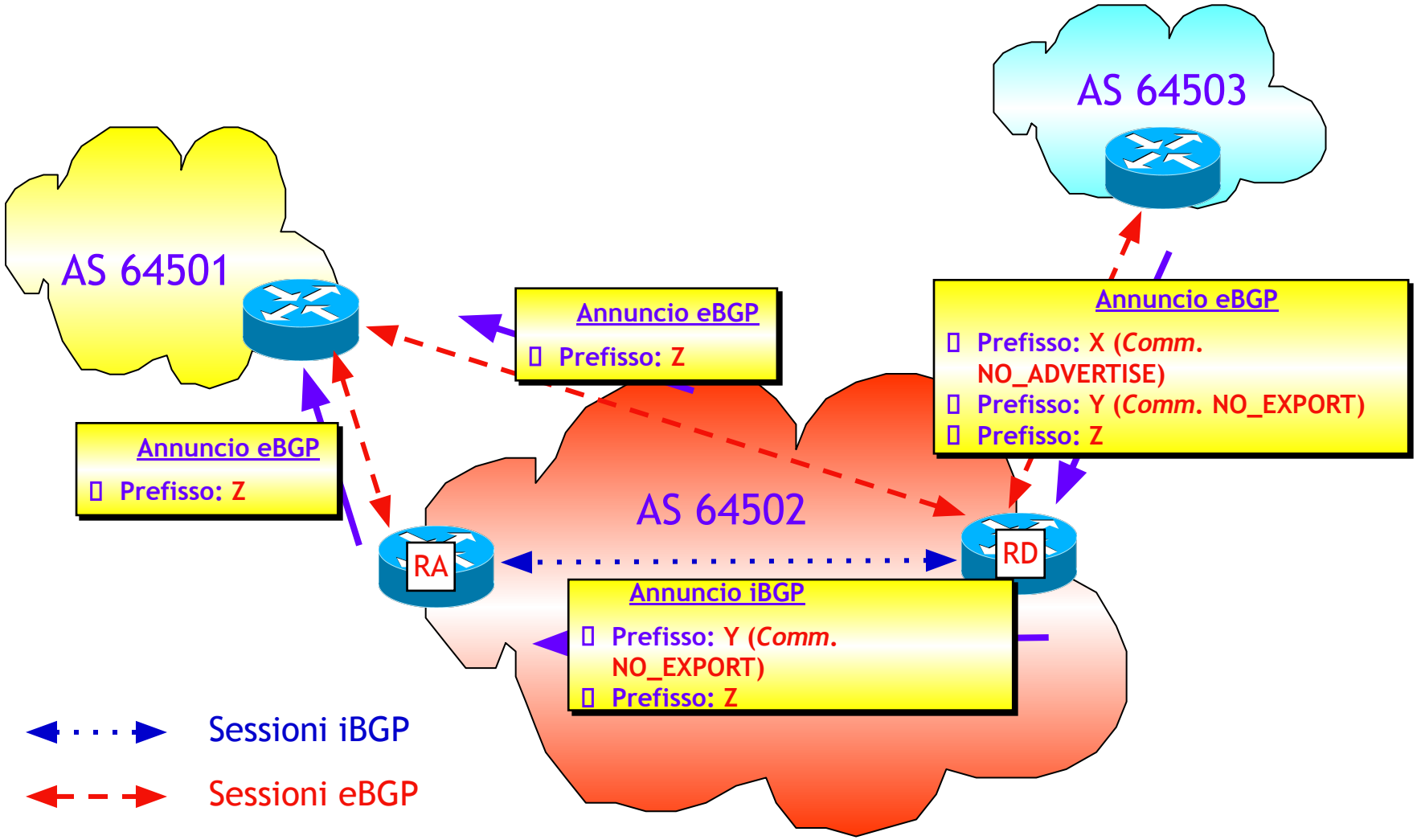
- I valori negli intervalli 0x00000000-0x0000FFFF e 0xFFFF0000-0xFFFFFFFF sono riservati
  - Alcuni di questi valori sono definiti nella RFC 1997 e sono noti come *Well-Known Community*
  - I valori al di fuori degli intervalli riservati dallo standard possono essere utilizzati arbitrariamente

- Tipicamente i valori di *Community* vengono rappresentati nel formato «*AS:NN*», dove «*AS*» è un numero di AS (2 *byte*) e «*NN*» un valore arbitrario (2 *byte*)

Community di tipo *Well-Known*

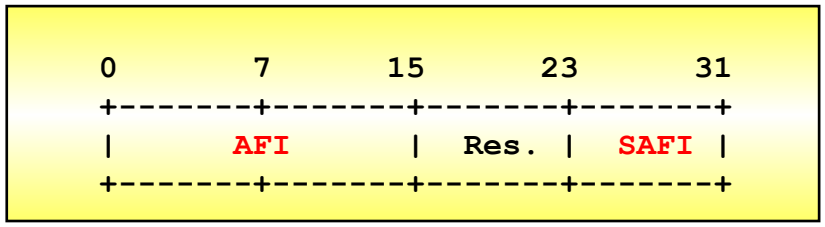
- Nella RFC 1997 sono definite tre *Well-Known Community*
  - NO_EXPORT (0xFFFFFF01=65535:65281): tutti i prefissi che hanno associato questo valore di *Community* non devono essere annunciati al di fuori di un AS (quindi non propagati con annunci su sessioni eBGP)
    - Eccezione: è possibile la propagazione sulle sessioni eBGP tra *sub-AS* di una "Confederazione BGP"
  - NO_ADVERTISE (0xFFFFFF02=65535:65282): tutti i prefissi che hanno associato questo valore di *Community* non devono essere annunciati su alcuna sessione BGP (iBGP o eBGP)
  - NO_EXPORT_SUBCONFED (0xFFFFFF03=65535:65283): tutti i prefissi che hanno associato questo valore di *Community* non devono essere annunciati su alcuna sessione eBGP (incluse le sessioni eBGP tra *sub-AS* di una confederazione BGP)

Utilizzo delle *well-known Community*



Multi-Protocol BGP (MP-BGP)

- Il protocollo BGP è stato inizialmente standardizzato per il trasporto delle informazioni di routing IPv4
- Generalizzato nella RFC 4760 (ex 2858) per il trasporto di informazioni di routing non-IPv4: *Multi-Protocol* BGP (MP-BGP)
  - Esempi: trasporto di informazioni di routing IPv6, VPN-IPv4, VPN-IPv6, ecc.
  - La funzionalità MP-BGP viene negoziata da due *BGP peer* tramite il messaggio di OPEN, utilizzando la *BGP Capability* «*Multi Protocol BGP*»
- Ciascun protocollo identificato dalla coppia AFI/SAFI
  - La coppia AFI/SAFI è contenuta nel campo *Capability Value* della *BGP Capability* «*Multi Protocol BGP*»



Concetti fondamentali

Caratteristiche principali
Autonomous System
Connettività Clienti-ISP
Funzionamento di base
Sessioni BGP
Messaggi e attributi
Processo di selezione

Il processo di selezione

Caratteristiche

- Ogni *BGP speaker* ha un processo di selezione interno che gli permette di determinare, per ciascun prefisso ricevuto, il percorso migliore per raggiungerlo (*best-path*)
  - I percorsi migliori sono inseriti dal router nella *Local-RIB* della Tabella BGP
  - Solo i percorsi migliori sono candidati ad essere inseriti nella Tabella di Routing IP
  - Solo i percorsi migliori sono annunciati agli altri *BGP Peer* (a valle di eventuali politiche di filtraggio in uscita)

- Il processo di selezione è basato su vari fattori (metriche, attributi vari, ecc.) ed alla fine produce un unico percorso migliore

- NOTA IMPORTANTE: solo i prefissi che hanno un *Next-Hop* raggiungibile sono candidati ad essere esaminati dal processo di selezione

Sequenza di decisione standard

1) Preferire gli annunci con il più alto valore di *Local Preference*

2) Preferire gli annunci con il minimo numero di elementi nell'attributo AS_PATH

3) Preferire gli annunci con il minimo valore nell'attributo *Origin* (0=IGP; 1=EGP; 2=INCOM.)

4) Se gli annunci arrivano dallo stesso AS preferire quello con il minimo valore di MED

α

α

5) Preferire annunci provenienti da sessioni eBGP rispetto a quelli provenienti da sessioni iBGP

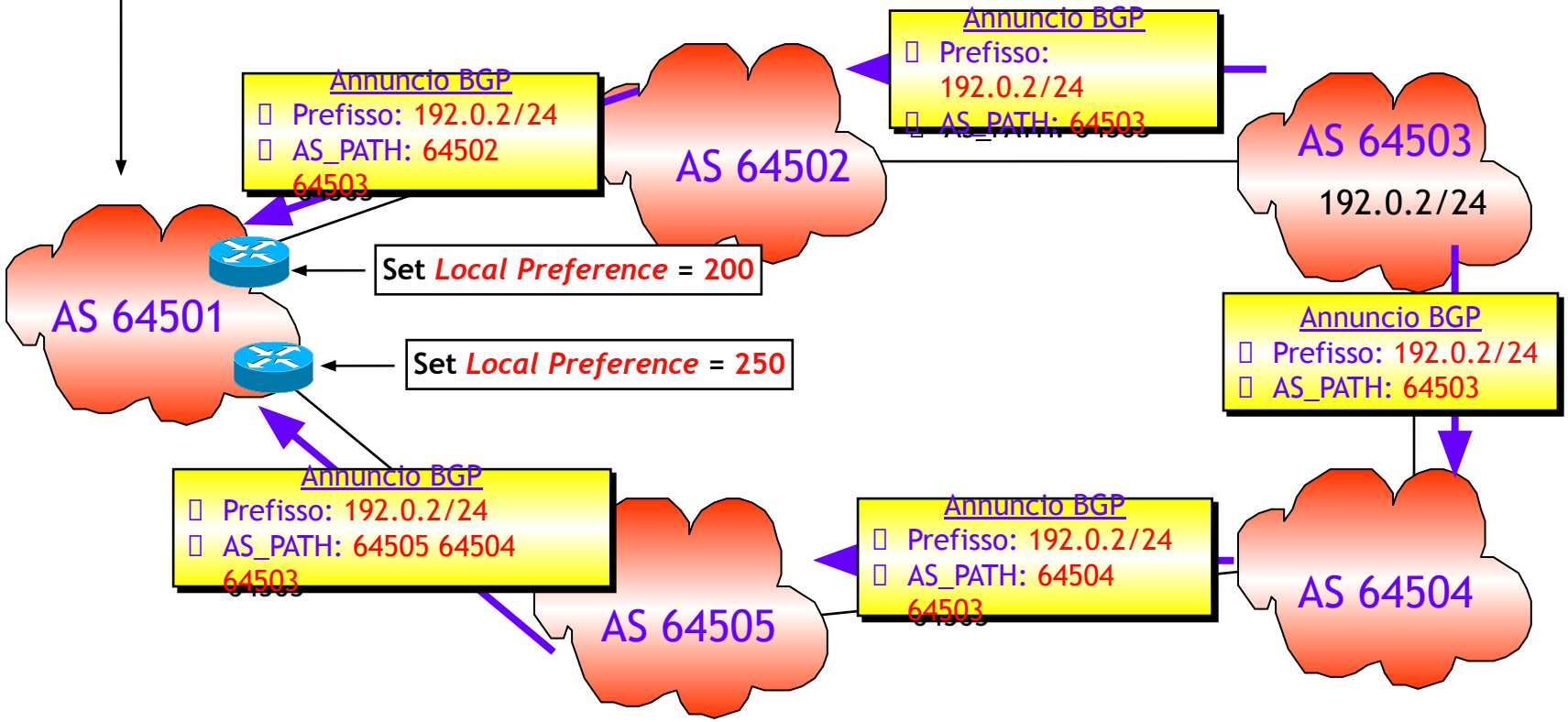
6) Preferire gli annunci che hanno il *Next-Hop* più "vicino" secondo la metrica IGP

7) Preferire l'annuncio proveniente dal BGP peer con *Router ID* più basso

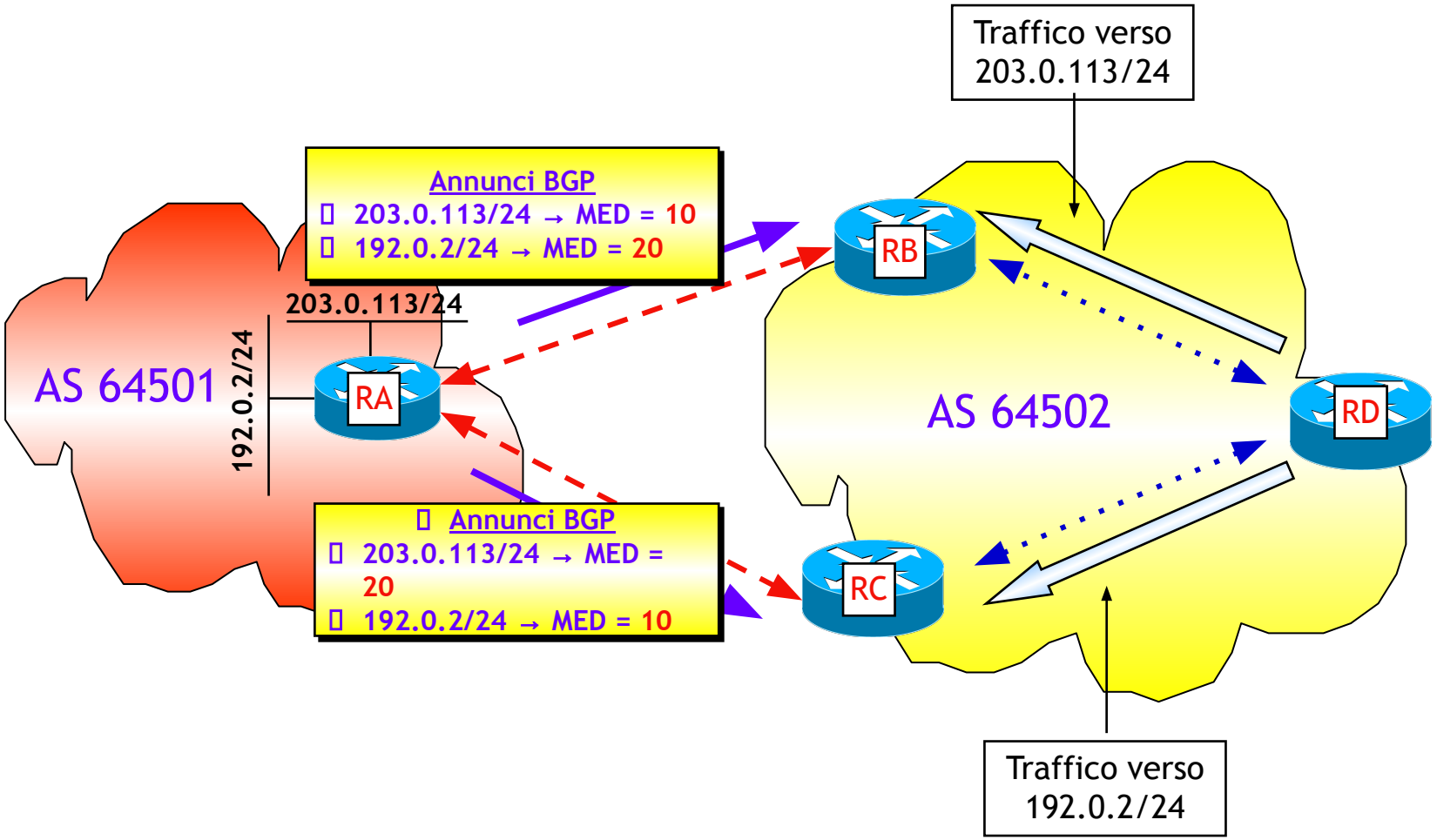
8) Preferire l'annuncio proveniente dal BGP peer con *Neighbor ID* più basso

Esempio 1

AS 64501 sceglie di raggiungere 192.0.2/24 via AS 64505 poiché all'annuncio proveniente da AS 64505 è stato associato un *Local Preference* più elevato. Questo nonostante il numero di AS da attraversare sia maggiore

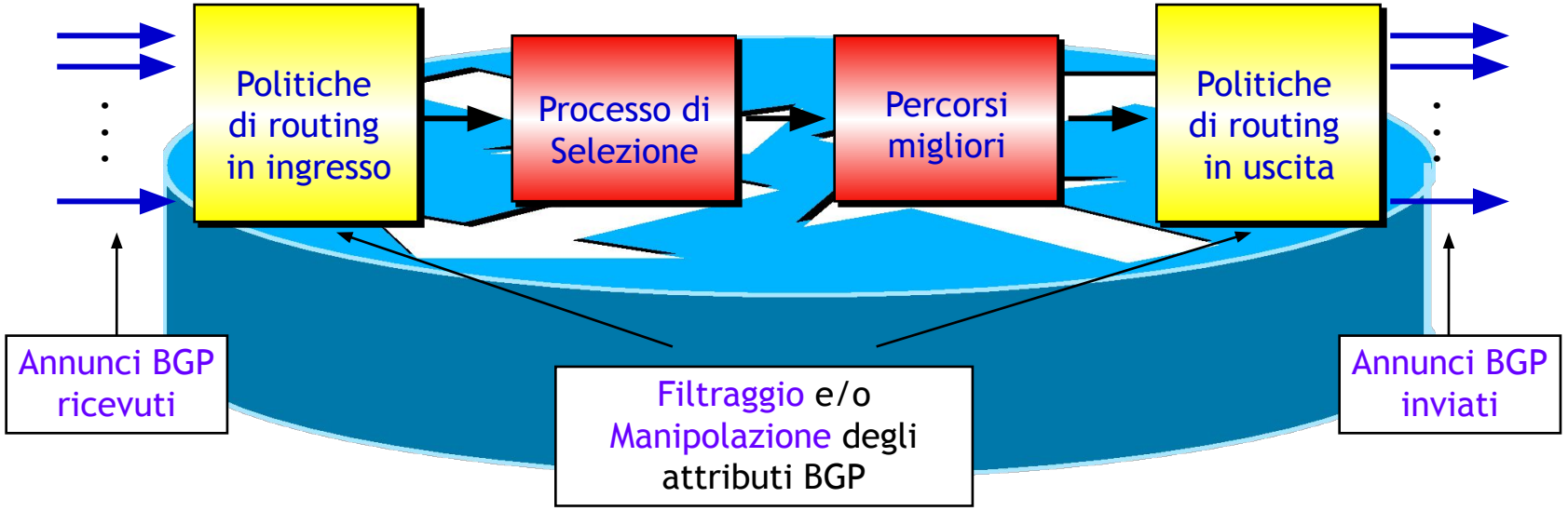


Esempio 2


Traffico verso
203.0.113/24
Annunci BGP
203.0.113/24 → MED = 10
192.0.2/24 → MED = 20
203.0.113/24
AS 64501
192.0.2/24
RA
RB
AS 64502
RD
Annunci BGP
203.0.113/24 → MED = 20
192.0.2/24 → MED = 10
RC
Traffico verso
192.0.2/24

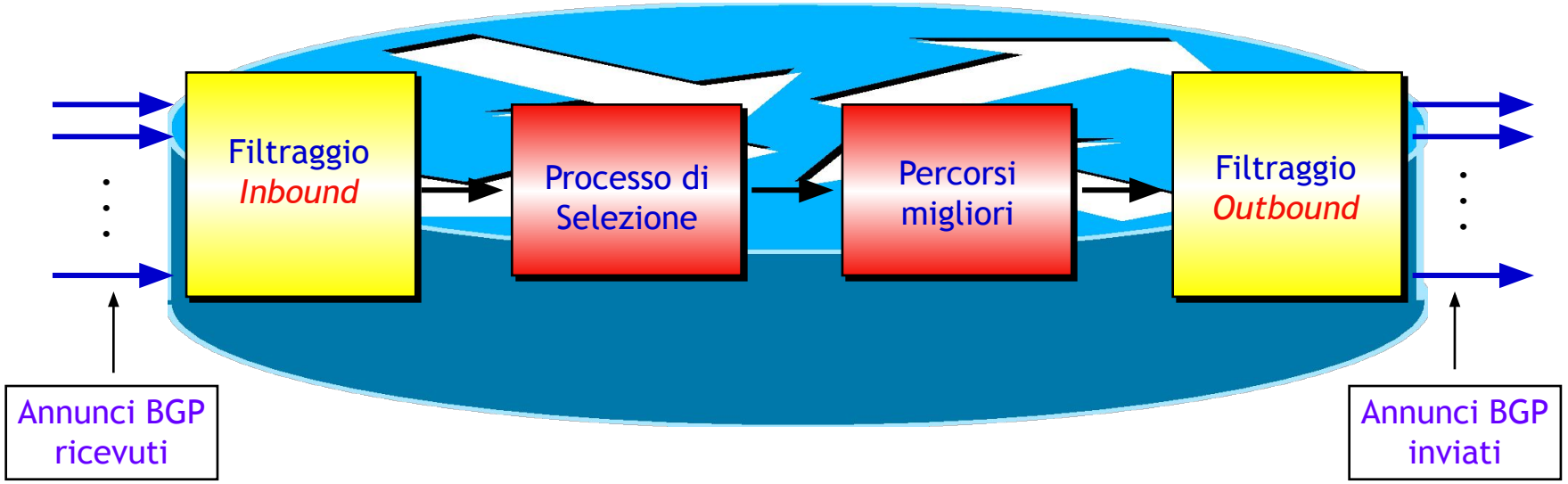
Politiche di routing

- Una politica di routing definisce le regole adottate da un AS per la gestione del traffico entrante (*inbound*) ed uscente (*outbound*) e le regole di accettazione ed invio dei prefissi
- Strumenti
  - Filtraggio dei prefissi
  - Manipolazione degli attributi BGP



Filtraggio dei prefissi

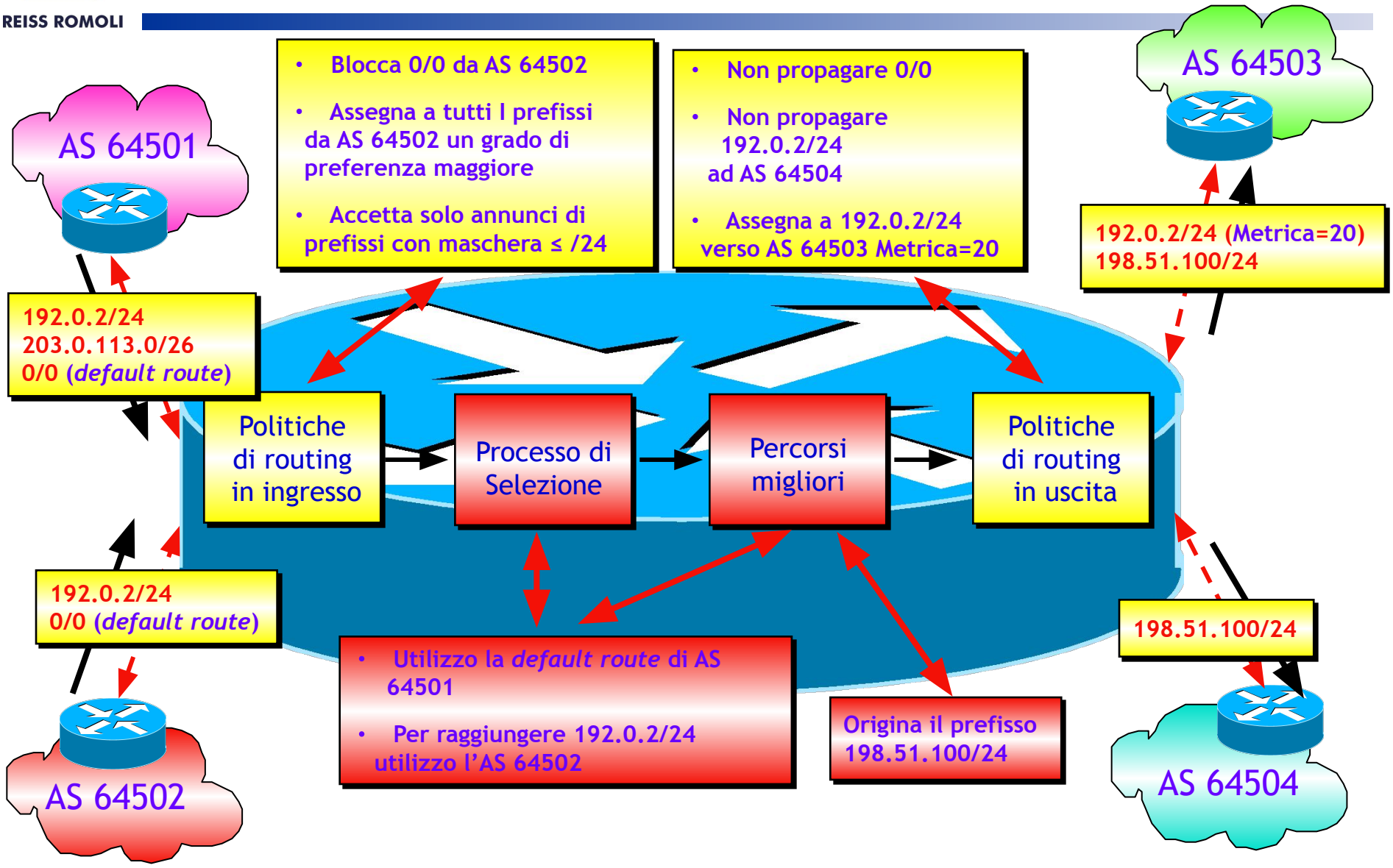
- Permette a un *BGP speaker* di scegliere quali prefissi accettare dai *BGP peer* e quali inviare ai *BGP peer*
  - Filtraggio *Inbound*: consente di definire politiche di routing sul traffico entrante agendo sui prefissi ricevuti dai vari *BGP peer*
  - Filtraggio *Outbound*: consente di definire politiche di routing sul traffico uscente agendo sui percorsi migliori per ciascun prefisso (quindi a valle del processo di selezione)
  - È utile anche per limitare i prefissi redistribuiti da un protocollo IGP in BGP e viceversa



Manipolazione degli attributi

- I prefissi che passano i filtri possono avere gli attributi BGP manipolati
  - *Origin*
  - AS_PATH
  - *Next-Hop*
  - *Local Preference*
  - MED
  - …
- La manipolazione degli attributi BGP permette la definizione di politiche di routing sia per il traffico entrante (*Inbound Routing Policy*) che per il traffico uscente (*Outbound Routing Policy*)

Esempio


AS 64501
AS 64503
AS 64502
AS 64504
• Blocca 0/0 da AS 64502
• Assegna a tutti I prefissi da AS 64502 un grado di preferenza maggiore
• Accetta solo annunci di prefissi con maschera ≤ /24
• Non propagare 0/0
• Non propagare 192.0.2/24 ad AS 64504
• Assegna a 192.0.2/24 verso AS 64503 Metrica=20
192.0.2/24 (Metrica=20)
198.51.100/24
192.0.2/24
203.0.113.0/26
0/0 (default route)
Politiche di routing in ingresso
Processo di Selezione
Percorsi migliori
Politiche di routing in uscita
192.0.2/24
0/0 (default route)
198.51.100/24
• Utilizzo la default route di AS 64501
• Per raggiungere 192.0.2/24 utilizzo l'AS 64502
Origina il prefisso 198.51.100/24

REISS ROMOLI

Implementazione base nell'IOS, IOS XE e IOS XR

Configurazioni base

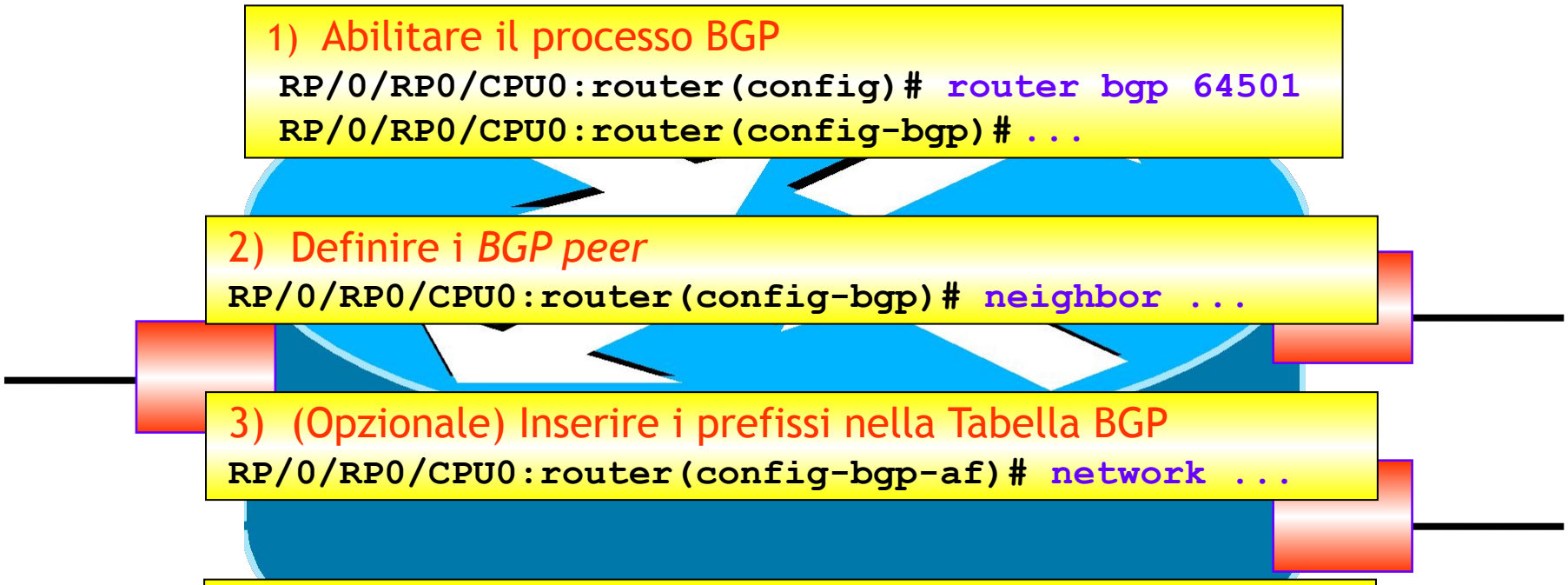
Propagazione dei prefissi locali

Generazione della *default route*

BGP per IPv6

Controllo della configurazione

Cosa bisogna fare …



1) Abilitare il processo BGP

```
RP/0/RP0/CPU0:router(config)# router bgp 64501
RP/0/RP0/CPU0:router(config-bgp)# ...
```

2) Definire i *BGP peer*

```
RP/0/RP0/CPU0:router(config-bgp)# neighbor ...
```

3) (Opzionale) Inserire i prefissi nella Tabella BGP

```
RP/0/RP0/CPU0:router(config-bgp-af)# network ...
```

4) (Opzionale) Definire filtri, politiche di routing, timer, ecc.

```
RP/0/RP0/CPU0:router(config-bgp)# neighbor ...
```

Abilitazione del processo BGP

- IOS e IOS XE

```
router(config)# router bgp numero-AS
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
```

- Il valore "*numero-AS*" identifica il numero di AS a cui appartiene il router
  - Nel caso di AS a 16 bit è un valore nell'intervallo 1÷65535
    - Nel caso di AS a 32 bit, vedi note
  - È essenziale per distinguere le sessioni eBGP da quelle iBGP
  - NOTA IMPORTANTE: in un router non è possibile attivare più processi BGP

REISS ROMOLI

Definizione del *Router ID* (1/2)

- Il *BGP Router ID* può essere o configurato manualmente o definito automaticamente dal router

- Criterio di scelta
  1. RID configurato manualmente
  2. Indirizzo IP più alto tra tutte le interfacce di *Loopback*
  3. Indirizzo IP più alto tra tutte le interfacce fisiche attive (solo IOS/IOS XE)

- NOTA: l'eventuale variazione del RID in un router con un processo BGP già attivo comporta il *reset* e successivo riavvio delle sessioni BGP
  - È una operazione da effettuare con molta cautela poiché nei router il *reset* e successivo riavvio delle sessioni BGP può comportare tempi elevati di fuori servizio

Definizione del *Router ID* (2/2)

- Configurazione manuale
  - L'indirizzo IP può essere qualsiasi, non necessariamente quello di una interfaccia

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# bgp router-id IP-address
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# bgp router-id IP-address
```

Definizione dei *BGP peer* (1/3)

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# neighbor IP-peer remote-as AS-peer
```

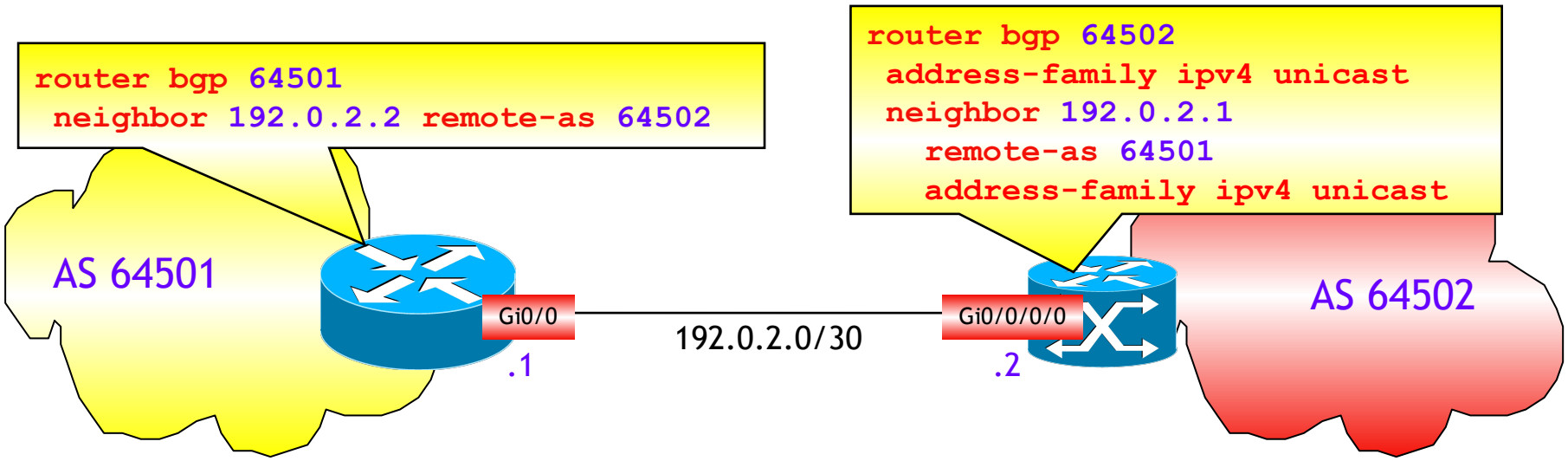
- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# exit
RP/0/RP0/CPU0:router(config-bgp)# neighbor IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as AS-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
```

- I *BGP peer* vanno definiti manualmente in entrambi i router agli estremi della sessione BGP
  - L'indirizzo IP del *BGP peer* viene utilizzato come indirizzo IP destinazione per tutti i messaggi BGP
- NOTA: nell'IOS XR di default non è abilitata alcuna *address-family*

Definizione dei *BGP peer* (2/3)

- ATTENZIONE: affinché la sessione BGP tra due *BGP peer* venga stabilita devono verificarsi le seguenti condizioni
  - Il numero di AS contenuto nell'intestazione comune dei messaggi BGP deve coincidere con quello configurato come *numero-as-peer* nell'altro *BGP peer*
  - L'indirizzo IP sorgente dei pacchetti TCP/IP che trasportano i messaggi BGP, deve coincidere con quello configurato come *IP-peer* nell'altro *BGP peer*
    - Di default l'indirizzo IP sorgente è quello dell'interfaccia che inoltra il pacchetto
  - Per le sessioni eBGP inoltre i due router devono essere (di default) direttamente connessi



Definizione dei *BGP peer* (3/3)

- Qualora un router per stabilire una sessione BGP utilizzi una interfaccia diversa da quella che inoltra i messaggi BGP, è necessario informarlo per permettergli di cambiare l'indirizzo IP sorgente dei pacchetti TCP/IP che trasportano i messaggi BGP
  - Esempio: utilizzo di interfacce di *Loopback* nella realizzazione di sessioni iBGP o eBGP *multi-hop*

- IOS e IOS XE

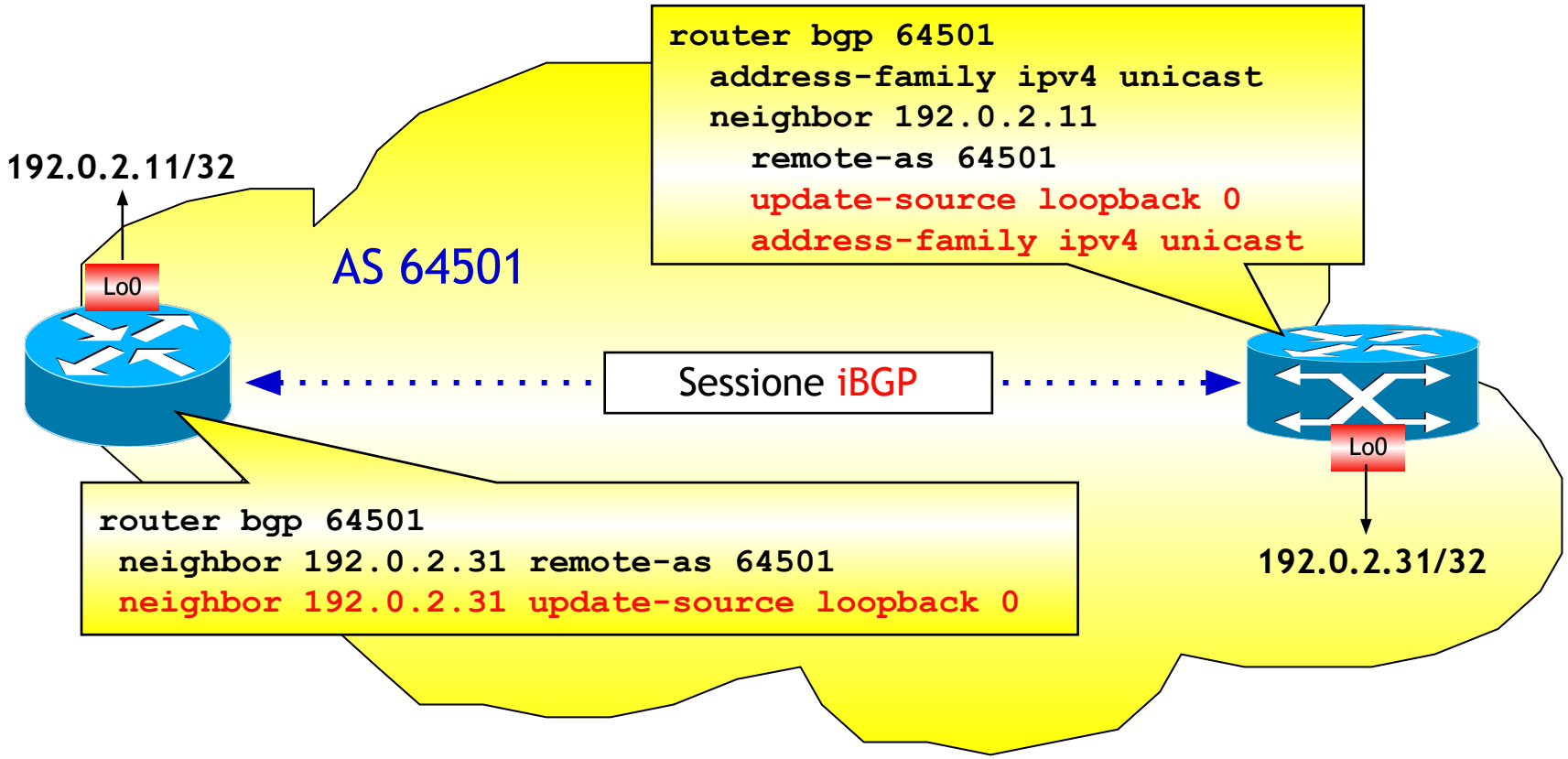
```
router(config)# router bgp numero-AS
router(config-router)# neighbor IP-peer update-source interfaccia
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
. . .
RP/0/RP0/CPU0:router(config-bgp)# neighbor IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# update-source interfaccia
```

Definizione dei *BGP peer*: esempio

- Utilizzo di interfacce di *Loopback* per stabilire una sessione iBGP
  - NOTA IMPORTANTE: Le due interfacce di Loopback utilizzate per stabilire la sessione devono essere tra loro connesse a livello IP



Sessioni eBGP nell'IOS XR (1/2)

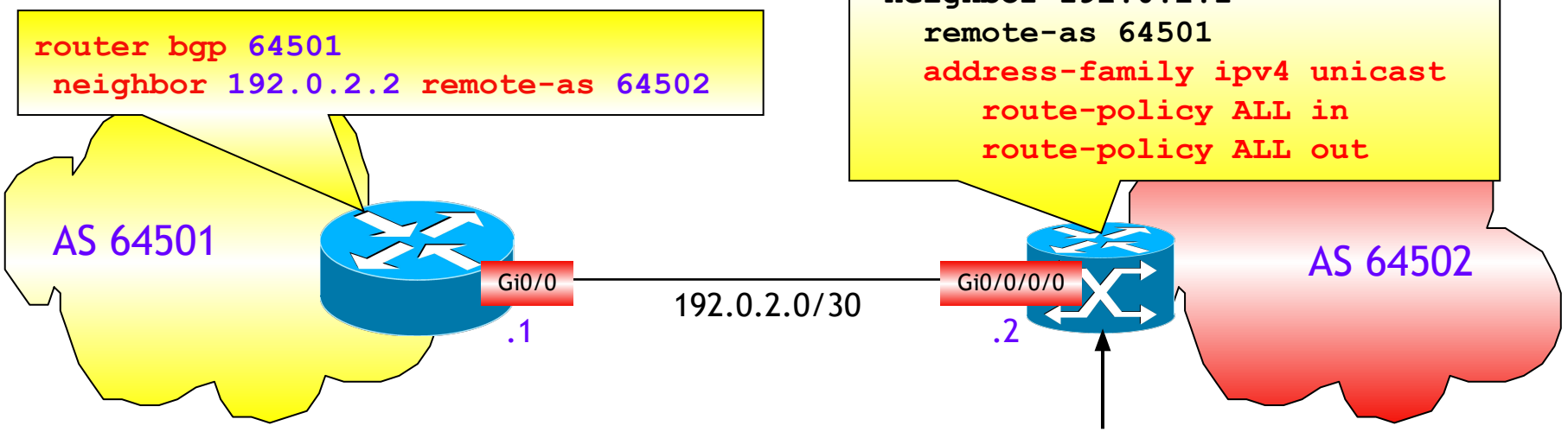
- Nell'IOS XR un *eBGP speaker* di default non accetta e non invia annunci
  - La sessione BGP raggiunge comunque lo stato *Established*
  - Nelle sessioni eBGP è quindi necessario abilitare delle *routing policy* sia *inbound* che *outbound*
  - Questa regola può essere disabilitata con il comando (sconsigliato) a livello di processo BGP: **bgp unsafe-ebgp-policy**
- L'IOS XR segnala la mancanza delle *routing policy* con un messaggio nel comando «**show bgp summary**»

```
RP/0/3/CPU0:RGTW-2# show bgp summary
. . .
Some configured eBGP neighbors (under default or non-default vrfs)
do not have both inbound and outbound policies configured for IPv4 Unicast
address family. These neighbors will default to sending and/or
receiving no routes and are marked with '!' in the output below.
Use the 'show bgp neighbor <nbr_address>' command for details.

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down  St/PfxRcd
192.0.2.1         0 64501      15      12        2    0    0 00:10:04           0!
```

Sessioni eBGP nell'IOS XR (2/2)

```
route-policy ALL
  pass
end-policy
!
router bgp 64502
 address-family ipv4 unicast
 neighbor 192.0.2.1
   remote-as 64501
   address-family ipv4 unicast
      route-policy ALL in
      route-policy ALL out
```

```
router bgp 64501
 neighbor 192.0.2.2 remote-as 64502
```




```
RP/0/3/CPU0:RGTW-2# show bgp summary
. . .

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down   St/PfxRcd
192.0.2.1         0 64501      31       28        3    0    0 00:25:03           1
```

1 prefisso ricevuto

Shutdown di una sessione

- È possibile disabilitare temporaneamente una sessione BGP senza il reset dell'intero processo
  - Molto comodo nei momenti di *troubleshooting* e durante una modifica consistente delle politiche di routing

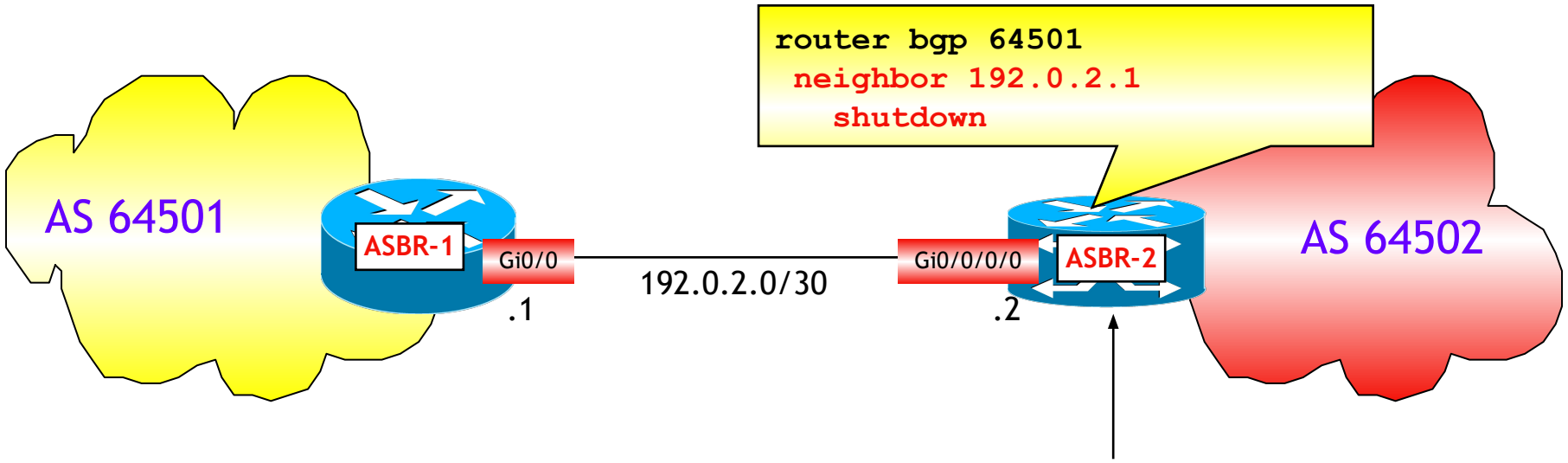
- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# neighbor IP-peer shutdown
! Per ristabilire la sessione
router(config-router)# no neighbor IP-peer shutdown
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# shutdown
! Per ristabilire la sessione
RP/0/RP0/CPU0:router(config-bgp-nbr)# no shutdown
```

Shutdown di una sessione: esempio




```
RP/0/RP0/CPU0:ASBR-2# show bgp ipv4 unicast summary 192.0.2.1
. . .
Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down   St/PfxRcd
192.0.2.1         0 64501     107      104        0    0    0 00:07:28 Idle (Admin)
```

Variazione del *Next-Hop*

- È utile e quasi sempre consigliato variare la regola di definizione dell'attributo *Next-Hop* imponendo come *Next-Hop* un proprio indirizzo
  - L'indirizzo utilizzato è quello utilizzato come indirizzo IP sorgente nei pacchetti TCP/IP che trasportano i messaggi BGP

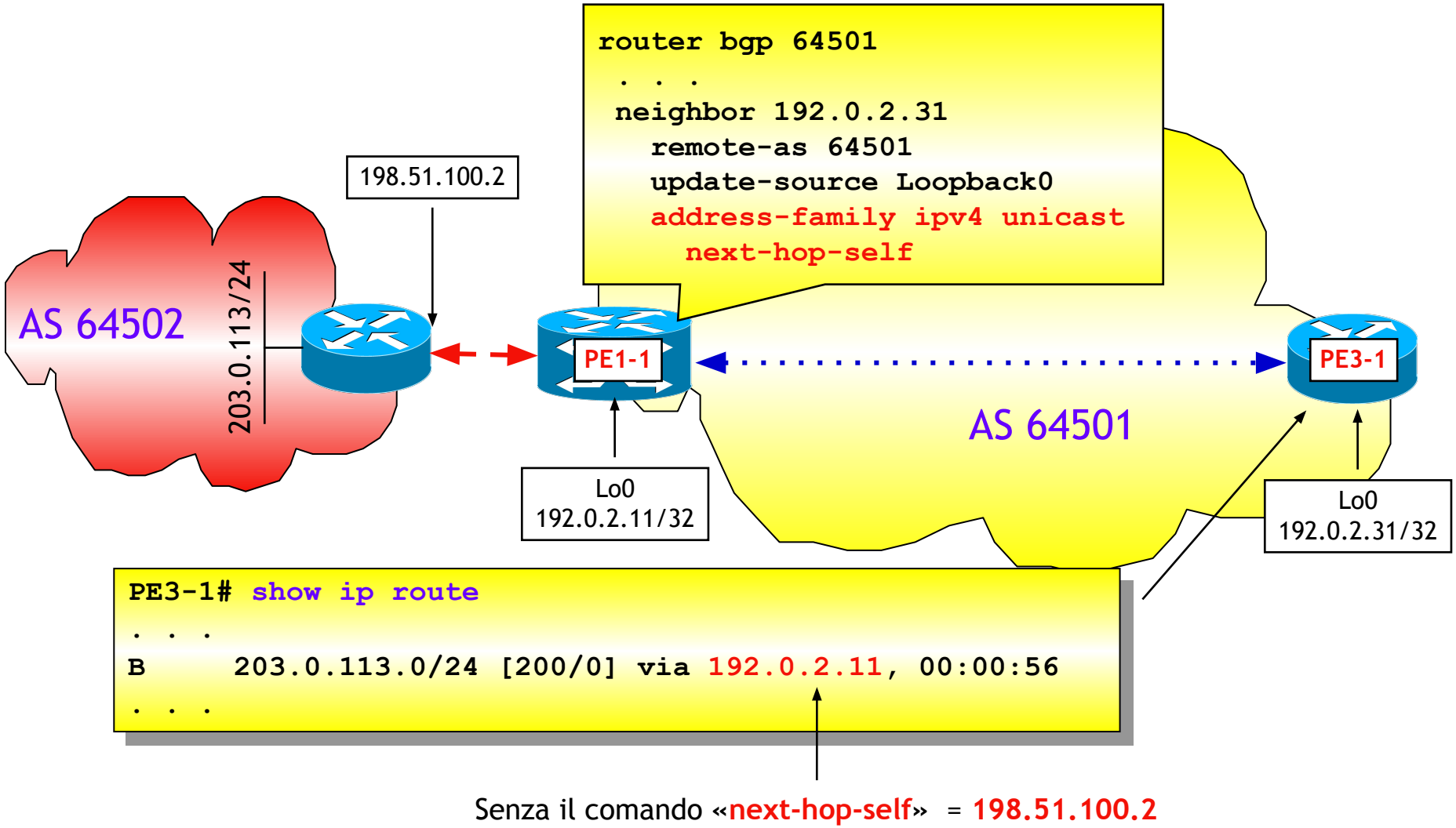
- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# neighbor IP-peer next-hop-self
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# next-hop-self
```

Variazione del *Next-Hop*: esempio



Senza il comando «next-hop-self» = 198.51.100.2

Sessioni eBGP *Multi-Hop*

- IOS e IOS XE

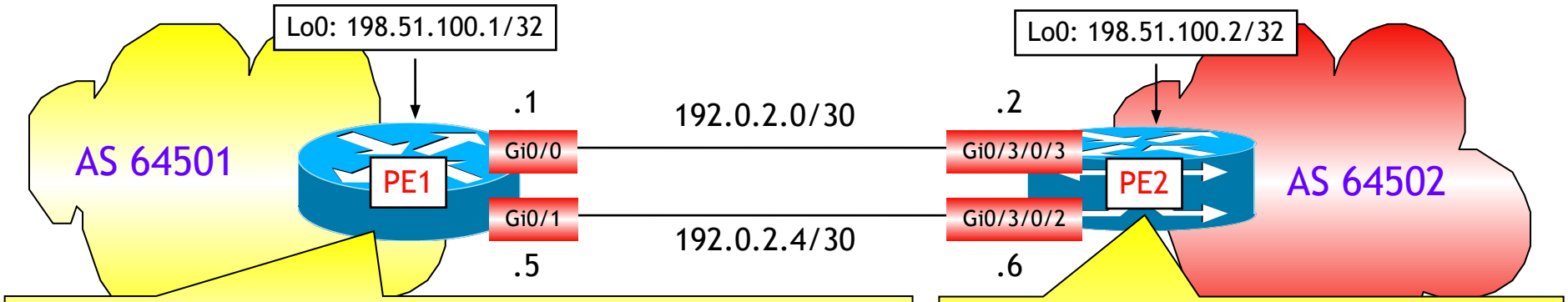
```
router(config)# router bgp numero-AS
router(config-router)# neighbor <IP-peer> ebgp-multihop [TTL]
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor <IP-peer>
RP/0/RP0/CPU0:router(config-bgp-nbr)# ebgp-multihop [TTL]
```

- Il valore TTL indica con quale TTL IP verranno inoltrati i messaggi BGP
  - L'omissione del valore TTL implica TTL = 255

Sessioni eBGP *Multi-Hop*: esempio




```
router bgp 64501
 neighbor 198.51.100.2 remote-as 64502
 neighbor 198.51.100.2 update-source loopback
0
 neighbor 198.51.100.2 ebgp-multihop 2
```

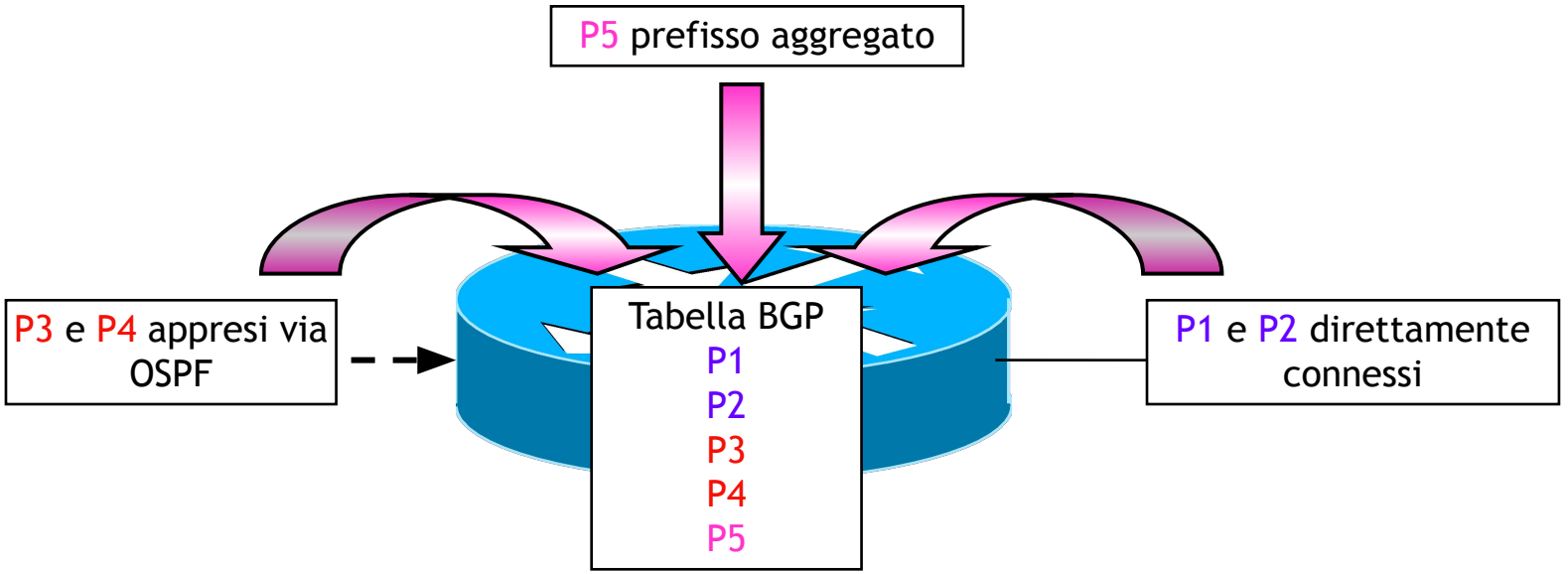
```
router bgp 64502
  address-family ipv4 unicast
  neighbor 198.51.100.1
    remote-as 64501
    ebgp-multihop 2
    update-source Loopback0
    address-family ipv4 unicast
      route-policy ALL in
      route-policy ALL out
```

Implementazione base nell'IOS, IOS XE e IOS XR

Configurazioni base
Propagazione dei prefissi locali
Generazione della *default route*
BGP per IPv6
Controllo della configurazione

Prefissi IP locali (1/2)

- Un prefisso IP è locale quando viene inserito nella tabella BGP secondo una delle seguenti modalità
  - Manualmente
  - Tramite un processo di aggregazione di prefissi
  - Tramite un processo di redistribuzione da un altro protocollo di routing



Prefissi IP locali (2/2)

- Ogni prefisso locale inserito nella tabella BGP appare come un nuovo prefisso con i seguenti attributi *well-known mandatory*
  - *Next-Hop*
    - identico a quello presente nella tabella di routing IP se inserito tramite redistribuzione
    - 0.0.0.0 se inserito manualmente o attraverso aggregazione (indica che il NEXT-HOP è il router stesso)
  - AS_PATH vuoto
  - ORIGIN
    - ORIGIN = IGP se inserito manualmente o attraverso aggregazione
    - ORIGIN = INCOMPLETE se inserito tramite redistribuzione

- NOTA: La distanza amministrativa di un prefisso locale è pari a 200

Inserimento manuale

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# network prefisso-IP mask maschera
                                          [route-map nome]
```

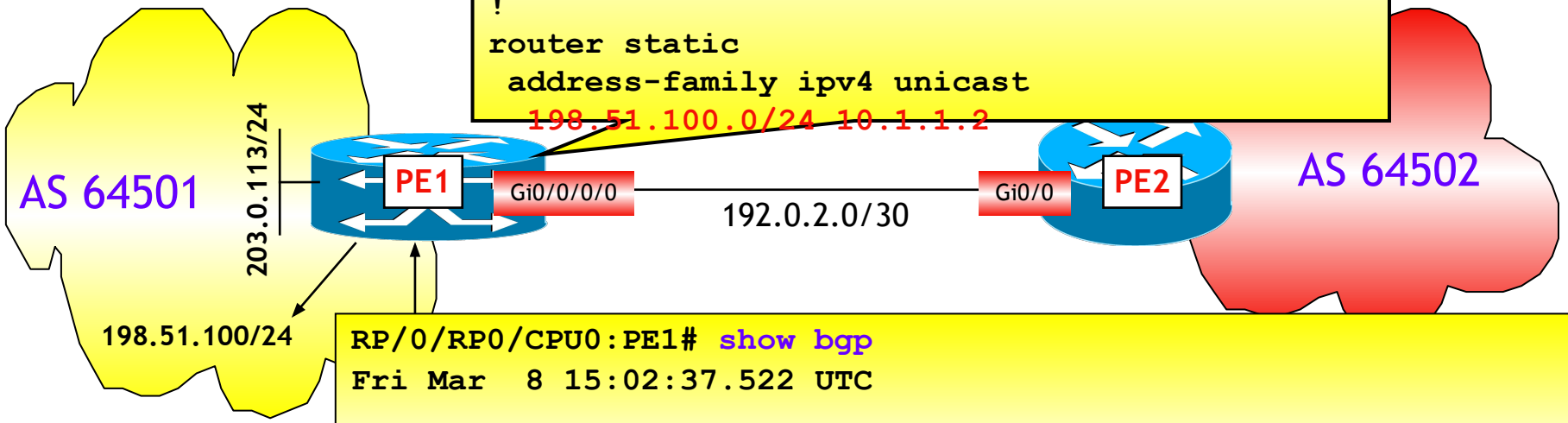
- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# network prefisso-IP/maschera
                                          [route-policy nome]
```

- Tramite la *route-map* (IOS e IOS XE) o *route-policy* (IOS XR), è possibile definire degli attributi BGP (MED, *Origin*, *Community*, ecc.)
- REGOLA FONDAMENTALE: affinché il comando abbia effetto è necessaria la presenza esatta (prefisso e maschera) nella Tabella di Routing IP del prefisso da inserire

Inserimento manuale: esempio

```
router bgp 64501
 address-family ipv4 unicast
   network 203.0.113.0/24 route-policy
SET-MED
   network 198.51.100.0/24
!
route-policy SET-MED
  set med 10
end-policy
!
router static
 address-family ipv4 unicast
  198.51.100.0/24 10.1.1.2
```




```
RP/0/RP0/CPU0:PE1# show bgp
Fri Mar  8 15:02:37.522 UTC
. . .
   Network          Next Hop            Metric LocPrf Weight Path
*> 203.0.113.0      0.0.0.0                 10         32768 i
*> 198.51.100.0     10.1.1.2                 0         32768 i
. . .
```

Inserimento via aggregazione

- IOS e IOS XE

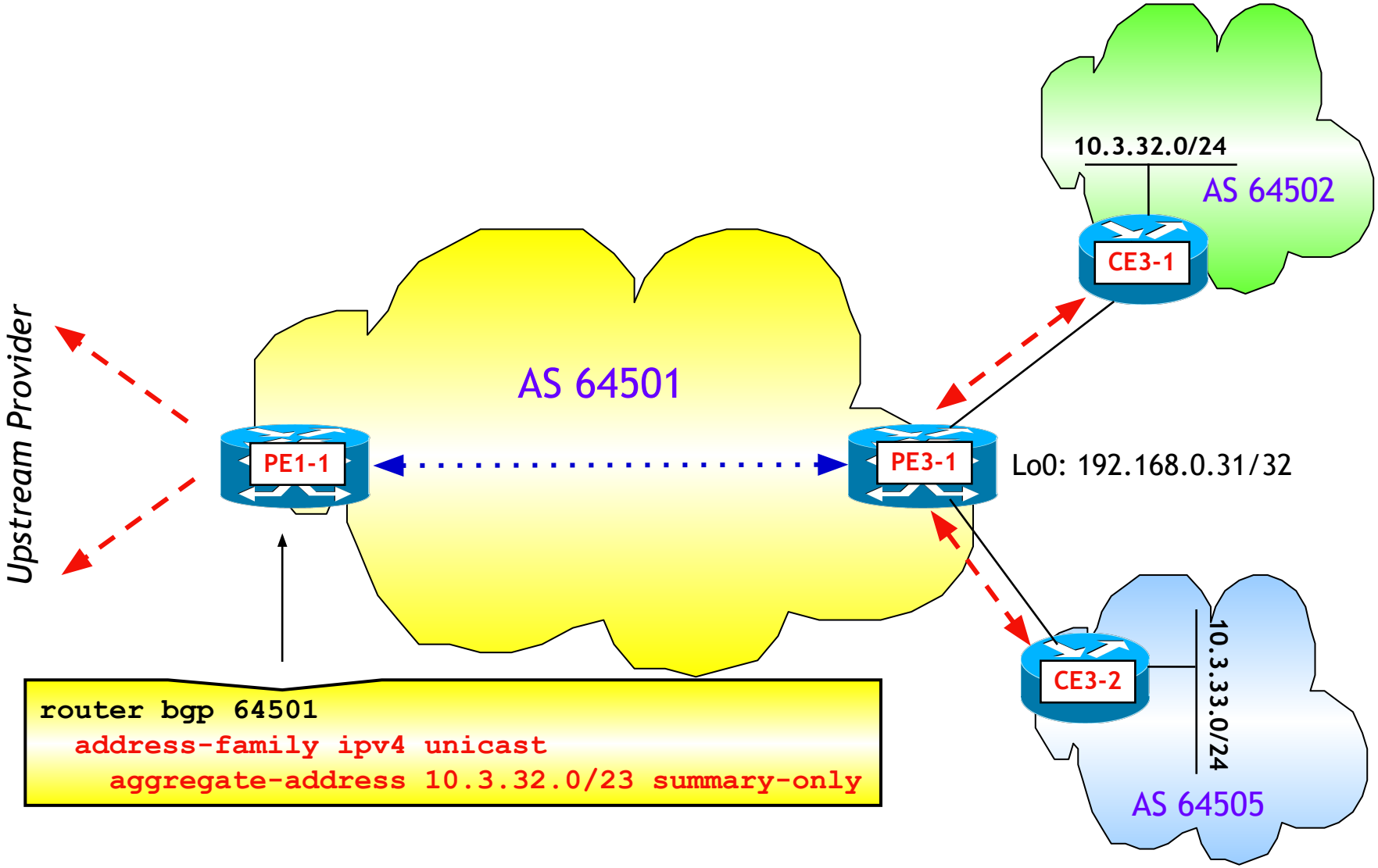
```
router(config)# router bgp numero-AS
router(config-router)# aggregate-address prefisso-IP maschera
                        [summary-only] [as-set] [suppress-map nome]
                                                [attribute-map nome]
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# aggregate-address
  prefisso-IP/maschera [summary-only] [as-set] [route-policy nome-RP]
```

- REGOLA FONDAMENTALE: affinché l'aggregazione abbia successo, deve essere presente nella Tabella BGP almeno un prefisso più specifico

Esempio (1/2)


10.3.32.0/24
AS 64502
CE3-1
AS 64501
Upstream Provider
PE1-1
PE3-1
Lo0: 192.168.0.31/32
CE3-2
10.3.33.0/24
AS 64505
router bgp 64501
address-family ipv4 unicast
aggregate-address 10.3.32.0/23 summary-only

Esempio (2/2)

```
RP/0/RP0/CPU0:PE1-1# show bgp
Fri Mar  8 15:51:37.027 UTC
. . .
Status codes: s suppressed, d damped, h history, * valid, > best
              i - internal, r RIB-failure, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network          Next Hop            Metric  LocPrf Weight Path
*> 10.3.32.0/23     0.0.0.0                              32768 i
s> 10.3.32.0/24     192.168.0.31   0                         0 64502 i
s> 10.3.33.0/24     192.168.0.31   0                         0 64505 i
```

- L'opzione «**summary-only**» consente la soppressione di tutti i prefissi più specifici

Redistribuzione in BGP

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# redistribute protocollo ...
                                [metric metrica] [route-map nome-RM]
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# redistribute protocollo ...
                                [metric metrica] [route-policy nome-RM]
```

- NOTA: il formato completo del comando di redistribuzione dipende dal tipo di protocollo redistribuito

Redistribuzione e attributo MED

- Il comando «**default-metric …**» consente di associare manualmente un valore di MED (*seed metric*) a tutti i prefissi appresi via redistribuzione
  - Valori ammessi: min=0, max=2^{32} - 1 (=4.294.967.295)
  - NOTA IMPORTANTE: l'eventuale MED conseguente all'utilizzo dell'opzione «**metric**» nel comando «**redistribute …**» ha sempre la precedenza

- IOS e IOS XE

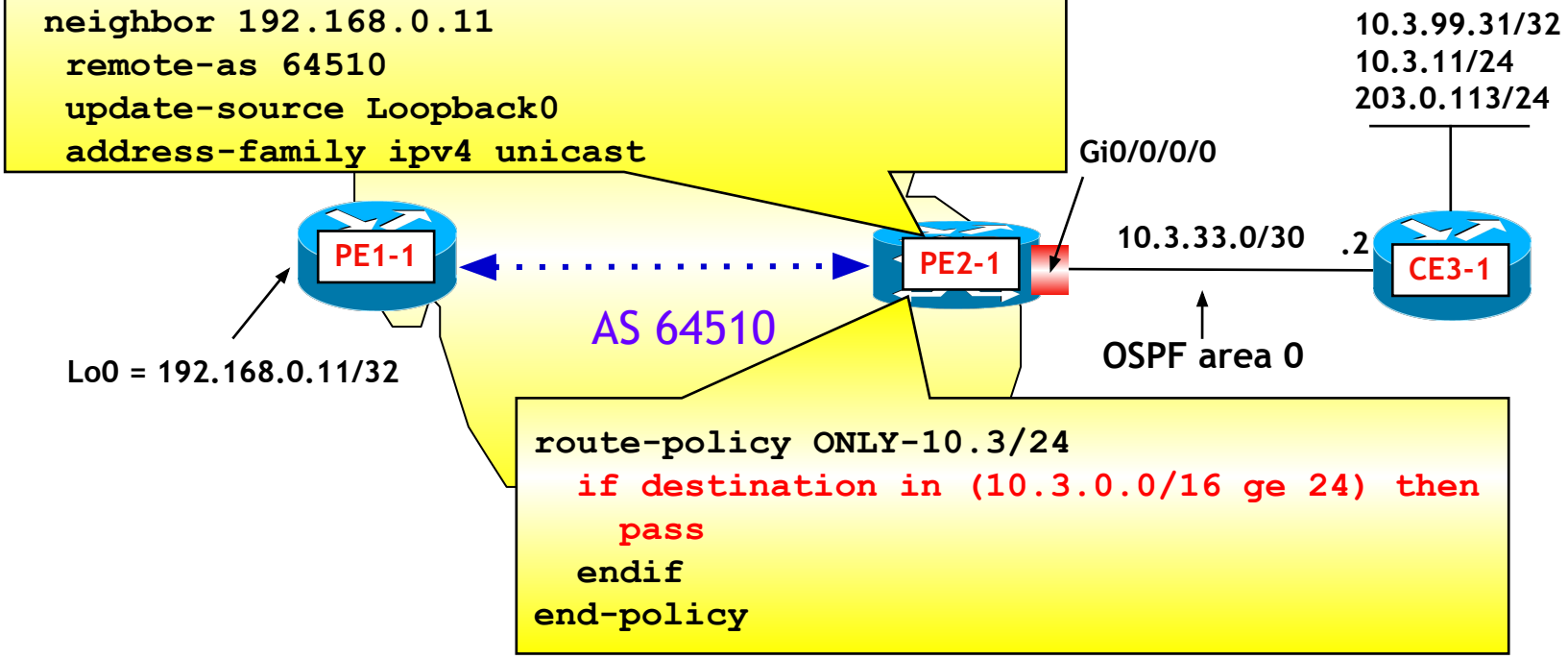
```
router(config)# router bgp numero-AS
router(config-router)# default-metric MED
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# default-metric MED
```

Esempio: redistribuzione da OSPF in BGP (1/2)

```
router ospf TT
 area 0
  interface GigabitEthernet0/0/0/0
!
router bgp 64510
 address-family ipv4 unicast
       redistribute    ospf    TT    route-policy
ONLY-10.3/24
 !
 neighbor 192.168.0.11
  remote-as 64510
  update-source Loopback0
  address-family ipv4 unicast
```




```
route-policy ONLY-10.3/24
  if destination in (10.3.0.0/16 ge 24) then
    pass
  endif
end-policy
```

Esempio: redistribuzione da OSPF in BGP (2/2)

```
RP/0/6/CPU0:PE2-1# show route ipv4 ospf TT
. . .

O    10.3.11.0/24 [110/2] via 10.3.33.2, 00:00:47, GigabitEthernet0/0/0/0
O    10.3.99.31/32 [110/2] via 10.3.33.2, 00:00:47, GigabitEthernet0/0/0/0
O    203.0.113.0/24 [110/2] via 10.3.33.2, 00:00:26, GigabitEthernet0/0/0/0
```

```
RP/0/6/CPU0:PE2-1# show bgp | include 10.3
. . .

*> 10.3.11.0/24       10.3.33.2                2         32768 ?
*> 10.3.33.0/30       0.0.0.0                  0         32768 ?
*> 10.3.99.31/32      10.3.33.2                2         32768 ?

RP/0/6/CPU0:PE2-1# show bgp | include 203.0
Wed Apr  3 13:43:47.446 UTC
```

Prefisso direttamente connesso ma di una interfaccia abilitata OSPF

Implementazione base nell'IOS, IOS XE e IOS XR

Configurazioni base
Propagazione dei prefissi locali
Generazione della *default route*
BGP per IPv6
Controllo della configurazione

Modalità di generazione

- È possibile propagare la *default route* su tutte le sessioni BGP utilizzando due modalità di configurazione
  - Utilizzare il comando "**network ...**"
    - Richiede la configurazione preliminare di una *default route*
  - Utilizzare il comando "**default-information originate**"
    - Richiede la configurazione preliminare di una *default route* e del comando "**redistribute** *protocollo*"

- In alternativa è possibile propagare la *default route* su una singola sessione BGP
  - Utilizzare il comando "**neighbor** *IP-peer* **default-originate**"
    - NON richiede la configurazione preliminare di una *default route* né del comando "**redistribute ...**"
    - L'annuncio della *default route* può essere condizionato dalla presenza/assenza nella tabella di routing IP di una o più reti

Propagazione della *default route* su tutte le sessioni BGP

- IOS/IOS XE

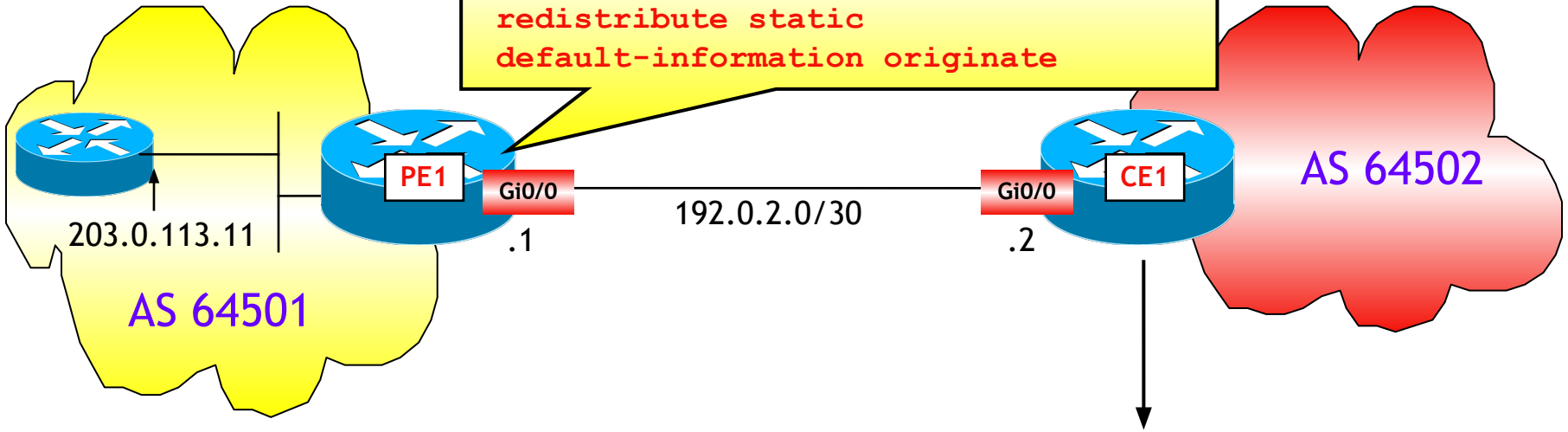
```
 router(config)# router bgp numero-AS
! Modalità 1
 router(config-router)# network 0.0.0.0
! Modalità 2
 router(config-router)# redistribute protocollo
 router(config-router)# default-information originate
```

- IOS XR

```
 RP/0/RP0/CPU0:router(config)# router bgp numero-AS
! Modalità 1
 RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
 RP/0/RP0/CPU0:router(config-bgp-af)# network 0.0.0.0/0
! Modalità 2
 RP/0/RP0/CPU0:router(config-bgp)# default-information originate
 RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
 RP/0/RP0/CPU0:router(config-bgp-af)# redistribute protocollo
```

Esempio

```
ip route 0.0.0.0 0.0.0.0 203.0.113.11
!
router bgp 64501
 neighbor 192.0.2.2 remote-as 64502
 network 0.0.0.0
! OPPURE
 redistribute static
 default-information originate
```




```
CE1# show bgp ipv4 unicast
BGP table version is 3, local router ID is 192.168.1.2
. . .
   Network          Next Hop            Metric LocPrf Weight Path
*> 0.0.0.0          192.0.2.1                0             0 64501 i
```

Propagazione della *default route*

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# neighbor indirizzo-IP-peer
                        default-originate [route-map nome]
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# default-originate
                                          [route-policy nome]
```

- È possibile condizionare l'annuncio della *default route* tramite una *route-map* o *route-policy*
  - Regola: la *default route* viene generata se e solo se sono presenti nella tabella di routing IP i prefissi permessi dalla *route-map* o *route-policy*

Esempio

```
route-policy COND
  if rib-has-route in (192.0.2.1/32) then
     pass
  endif
end-policy
!
router bgp 64510
  address-family ipv4 unicast
  neighbor 10.3.3.1
   remote-as 64511
   address-family ipv4 unicast
     default-originate route-policy COND
```

192.0.2.1

GTW

AS 64510

ASBR-X

10.3.3.0/30

.1

ASBR-Y

AS 64511

Sessioni iBGP

Sessioni eBGP

Implementazione base nell'IOS, IOS XE e IOS XR

Configurazioni base
Propagazione dei prefissi locali
Generazione della *default route*
BGP per IPv6
Controllo della configurazione

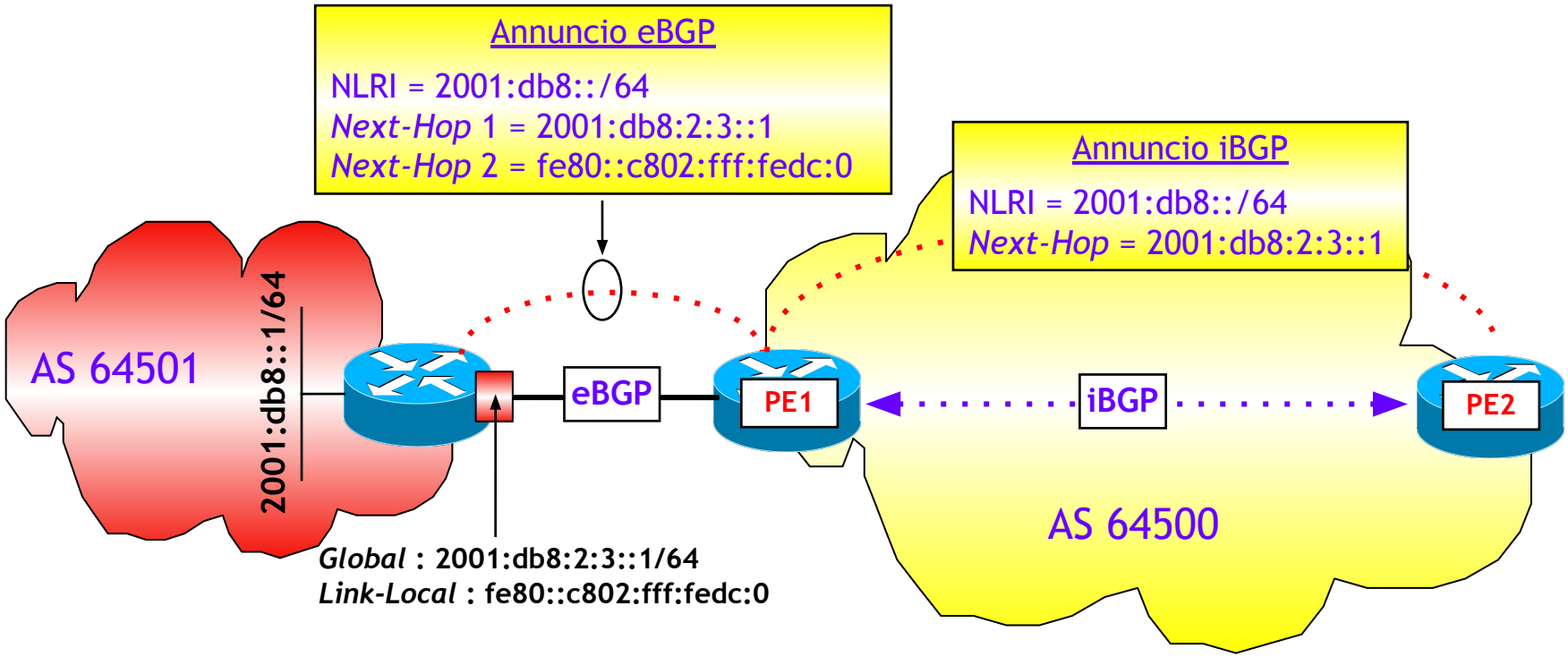
Generalità

- Il BGP per IPv6 ha la stessa identica struttura del BGP per IPv4 (stessi attributi, processo di selezione, funzionalità, timer ecc.)

- Il trasporto delle informazioni di routing IPv6 avviene sfruttando l'estensione multi protocollo del BGP
  - RFC 4760: *Multiprotocol Extensions for BGP-4*, Gennaio 2007

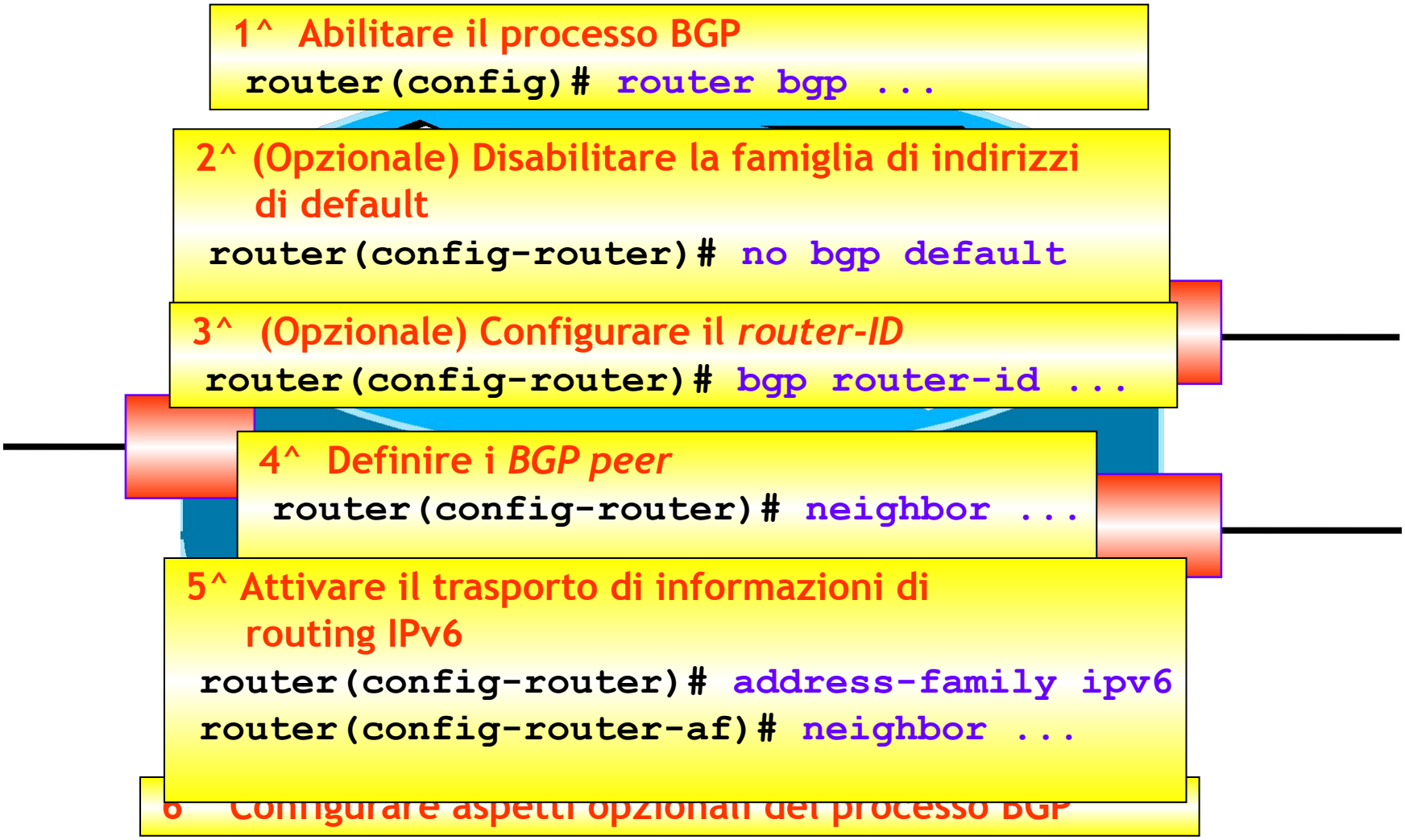
- Codici AFI/SAFI utilizzati
  - AFI = 2
  - SAFI
    - 1 = unicast
    - 2 = multicast
    - 4 = IPv6 + label
    - 128 = VPN-IPv6 + VPN label

Gestione del *BGP Next-Hop*



- La RFC 2545 consiglia di includere:
  - Nel caso in cui gli indirizzi IPv6 utilizzati per la sessione BGP siano sulla stessa *subnet*, due *Next-Hop*, l'indirizzo *non Link-Local* utilizzato per attivare la sessione BGP e l'indirizzo *Link-Local* dell'interfaccia sul segmento di rete
  - Quando gli indirizzi IPv6 utilizzati per la sessione BGP non appartengono alla stessa *subnet*, un solo *Next-Hop* gestito secondo le classiche regole del BGP

Configurazione: cosa bisogna fare ...



Disabilitazione del trasporto di default e attivazione del trasporto IPv6

REISS ROMOLI

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# no bgp default ipv4-unicast
router(config-router)# neighbor IPv6-peer remote-as AS-peer
router(config-router)# address-family ipv6
router(config-router-af)# neighbor IPv6-peer activate
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv6 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# exit
RP/0/RP0/CPU0:router(config-bgp)# neighbor IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as AS-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv6 unicast
```

- NOTA (solo per IOS/IOS XE): qualora si volesse definire una sessione per il solo trasporto di informazioni di routing IPv6 è necessario
  1. Disabilitare il trasporto di default
  2. Attivare il trasporto IPv6

Configurazione del *router-ID*

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# bgp router-id RID
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# bgp router-id RID
```

- Il *router-ID* ha lunghezza 32 bit e viene rappresentato con la stessa notazione degli indirizzi IPv4
- NOTA: la procedura di selezione del BGP RID per IPv6 è identica all'analoga selezione del BGP RID per IPv4
  - Questo implica che nei router solo IPv6 (ossia con nessuna interfaccia a cui è stato assegnato un indirizzo IPv4) è obbligatoria la configurazione manuale

Comandi *Show*, *Debug* e *Clear*

- Struttura dei comandi *Show* (validi per tutte le tipologie di IOS)

```
router# show bgp ipv6 unicast ...
```

  - **show bgp ipv6 unicast summary** : permette di visualizzare lo stato delle sessioni BGP
  - **show bgp ipv6 unicast** : permette di visualizzare la tabella BGP per IPv6
  - **show bgp ipv6 unicast** *prefisso*: permette di visualizzare il dettaglio degli annunci di un determinato prefisso IPv6 presenti nella tabella BGP IPv6

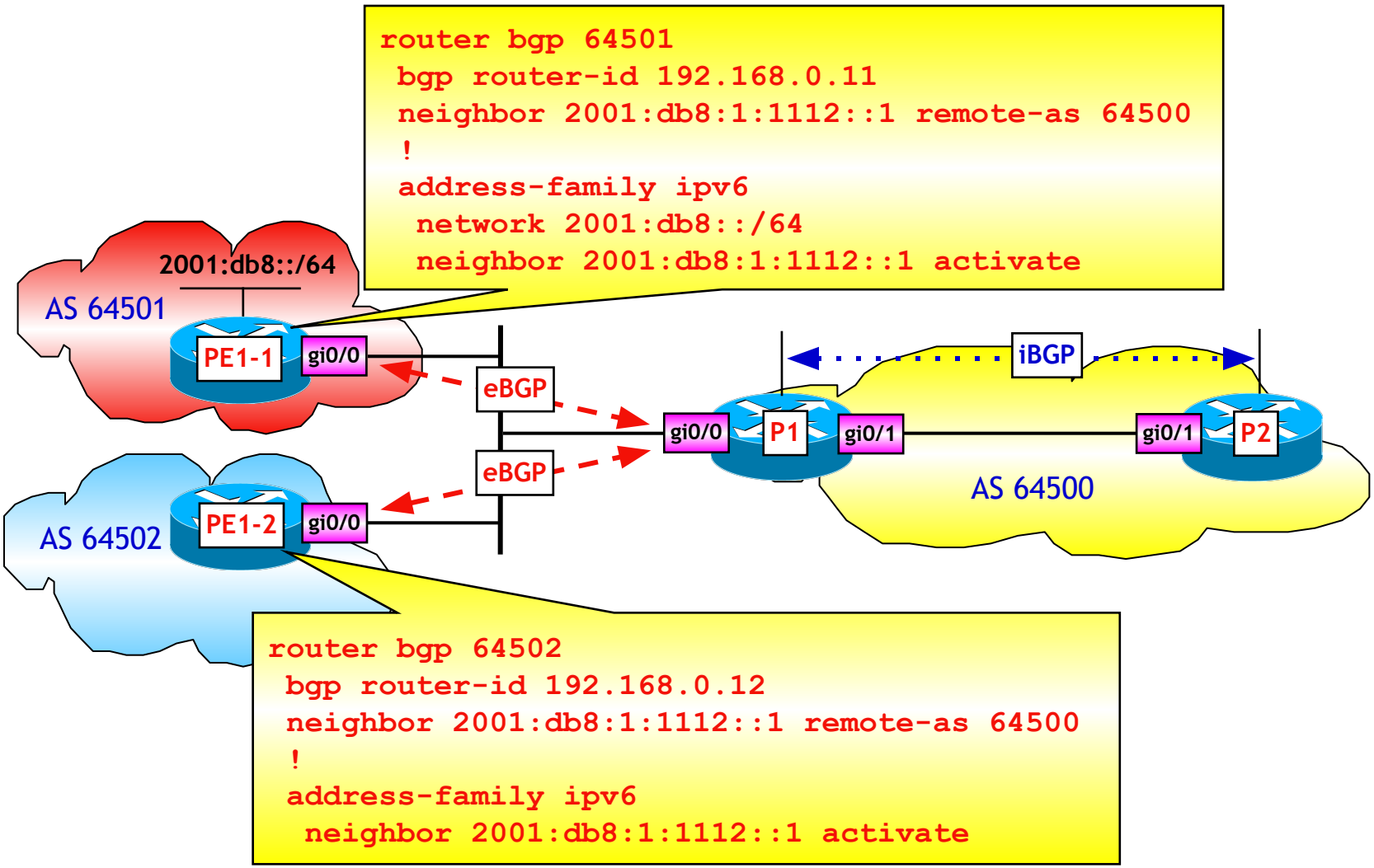
- Struttura dei comandi *Debug*

```
router# debug bgp ipv6 unicast ...
```

- Struttura dei comandi *Clear*

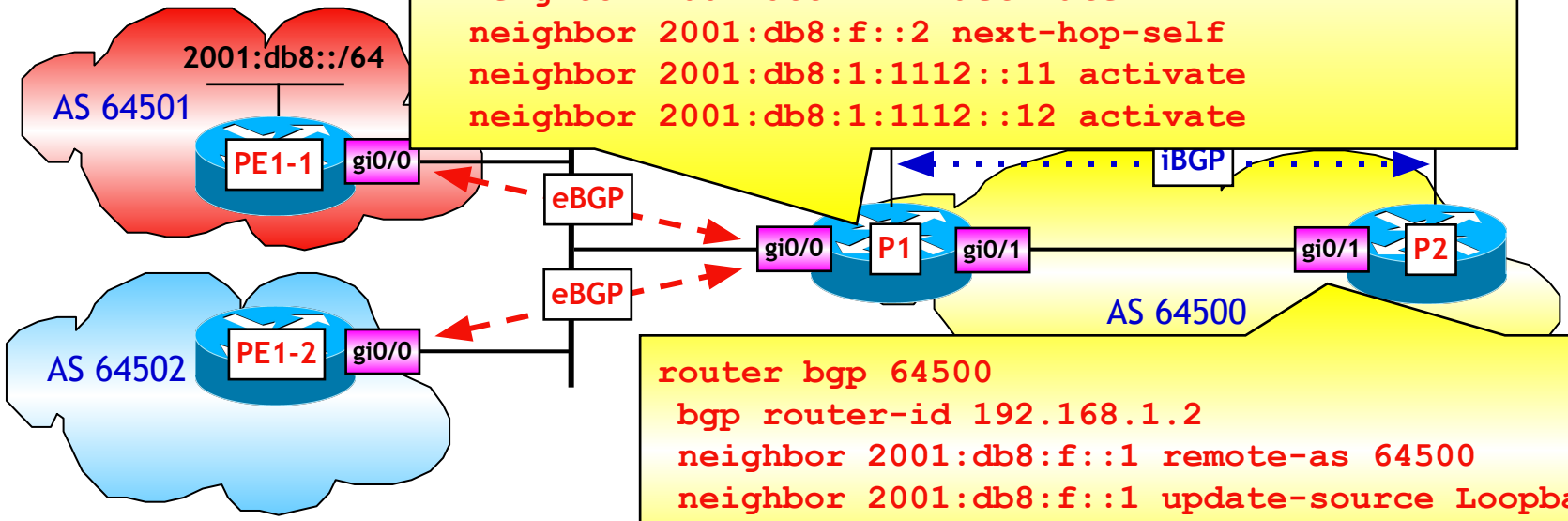
```
router# clear bgp ipv6 unicast ...
```

Esempio: configurazione base (1/4)



Esempio: configurazioni base (2/4)

```
router bgp 64500
 bgp router-id 192.168.1.1
 neighbor 2001:db8:f::2 remote-as 64500
 neighbor 2001:db8:f::2 update-source Loopback0
 neighbor 2001:db8:1:1112::11 remote-as 64501
 neighbor 2001:db8:1:1112::12 remote-as 64502
!
 address-family ipv6
  neighbor 2001:db8:f::2 activate
  neighbor 2001:db8:f::2 next-hop-self
  neighbor 2001:db8:1:1112::11 activate
  neighbor 2001:db8:1:1112::12 activate
```




```
router bgp 64500
 bgp router-id 192.168.1.2
 neighbor 2001:db8:f::1 remote-as 64500
 neighbor 2001:db8:f::1 update-source Loopback0
!
 address-family ipv6 2001:db8:f::1 activate
```

Esempio: configurazione base (3/4)

```
P1# show bgp ipv6 unicast summary
BGP router identifier 192.168.1.1, local AS number 64500
BGP table version is 3, main routing table version 3
2 network entries using 288 bytes of memory
2 path entries using 152 bytes of memory
2/2 BGP path/bestpath attribute entries using 248 bytes of memory
2 BGP AS-PATH entries using 48 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory
0 BGP filter-list cache entries using 0 bytes of memory
BGP using 736 total bytes of memory
BGP activity 4/2 prefixes, 4/2 paths, scan interval 60 secs

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
2001:db8:f::2
                4 64500      47      50        3    0    0 00:37:42            0
2001:db8:1:1112::11
                4 64501      48      49        3    0    0 00:39:34            1
2001:db8:1:1112::12
                4 64502      48      49        3    0    0 00:42:55            0
```

Esempio: configurazione base (4/4)

```
P1# show bgp ipv6 unicast 2001:db8::/64
BGP routing table entry for 2001:db8::/64, version 2
Paths: (1 available, best #1, table Global-IPv6-Table)
Flag: 0x820
  Advertised to update-groups:
        1    2
  64501
  2001:db8:1:1112::11 (FE80::C803:16FF:FE0C:8) from 2001:db8:1:1112::11
   (192.168.0.11)
      Origin IGP, metric 0, localpref 100, valid, external, best
```

```
P2# show bgp ipv6 unicast 2001:db8::/64
BGP routing table entry for 2001:db8::/64, version 2
Paths: (1 available, best #1, table Global-IPv6-Table)
Flag: 0x820
  Not advertised to any peer
  64501
    2001:db8:f::1 (metric 64) from 2001:db8:f::1 (192.168.1.1)
    Origin IGP, metric 0, localpref 100, valid, internal, best
```

Implementazione base nell'IOS, IOS XE e IOS XR

Configurazioni base
Propagazione dei prefissi locali
Generazione della *default route*
BGP per IPv6
Controllo della configurazione

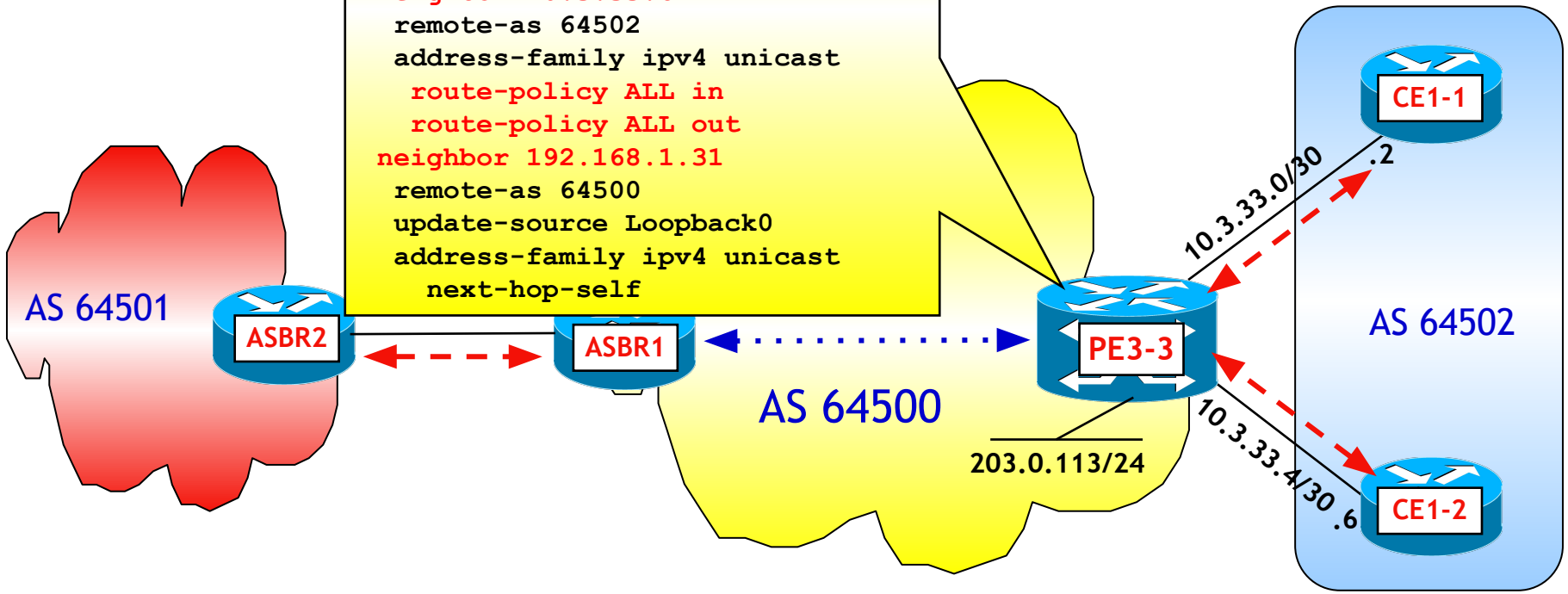
Cosa visualizzare

- Caratteristiche del protocollo
  - **show protocols**
- Caratteristiche e stato delle sessioni
  - **show bgp summary …**
  - **show bgp neighbor …**
- Contenuto della Tabella BGP
  - **show bgp …**
- Prefissi in tabella di routing appresi via BGP
  - **show route ipv4 bgp …**

Rete esempio

```
router bgp 64500
 address-family ipv4 unicast
  network 203.0.113.0/24
 neighbor 10.3.33.2
  remote-as 64502
  address-family ipv4 unicast
   route-policy ALL in
   route-policy ALL out
 neighbor 10.3.33.6
  remote-as 64502
  address-family ipv4 unicast
   route-policy ALL in
   route-policy ALL out
 neighbor 192.168.1.31
  remote-as 64500
  update-source Loopback0
  address-family ipv4 unicast
    next-hop-self
```

```
route-policy ALL
  pass
end-policy
```



Loopback 0 di PE3-3 = 192.168.0.33
Loopback 0 di ASBR1 = 192.168.1.31

Caratteristiche e stato delle sessioni (1/5)

- IOS e IOS XE

```
router# show bgp [(ipv4 | ipv6) unicast] summary
```

- IOS XR

```
RP/0/RP0/CPU0:router# show bgp [(ipv4 | ipv6) unicast] summary
```

- Permette la visualizzazione di
  - BGP RID e proprio AS
  - *BGP neighbor-ID*
  - Versione del protocollo
  - AS remoto
  - Messaggi BGP inviati/ricevuti
  - Stato della sessione e, se *up*, da quanto tempo
  - Numero di prefissi ricevuti

- È il comando più semplice e veloce per verificare caratteristiche e stato delle sessioni

Caratteristiche e stato delle sessioni (2/5)

REISS ROMOLI

```
RP/0/6/CPU0:PE3-3# show bgp summary
Wed Apr  3 08:59:13.074 UTC
BGP router identifier 192.168.0.33, local AS number 64500
BGP generic scan interval 60 secs
BGP table state: Active
Table ID: 0xe0000000   RD version: 21
BGP main routing table version 21
BGP scan interval 60 secs

BGP is operating in STANDALONE mode.


Process       RcvTblVer   bRIB/RIB   LabelVer  ImportVer  SendTblVer  StandbyVer
Speaker              21         21         21         21          21          21

Neighbor        Spk    AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down  St/PfxRcd
10.3.33.2         0 64502      28      36       21    0    0 00:24:41        0
10.3.33.6         0 64502      26      34       21    0    0 00:22:15        0
192.168.1.31      0 64500      43      36       21    0    0 00:32:07        8
```

Versione della Tabella BGP.
Incrementa di 1 ad ogni variazione

Stato
(Vuoto = *Established*)

BGP peer

Vedi note

AS dei
BGP peer

Messaggi
inviati/ricevuti

N.ro di messaggi
BGP in coda

Tempo
di *up*

Prefissi
ricevuti

Caratteristiche e stato delle sessioni (3/5)

- IOS e IOS XE

```
router# show bgp [(ipv4 | ipv6) unicast] neighbors [neighbor-ID]
```

- IOS XR

```
RP/0/RP0/CPU0:router# show bgp [(ipv4 | ipv6) unicast]
                                neighbors [neighbor-ID]
```

- Permette la visualizzazione di (tra l'altro)
  - Indirizzo IP, RID e numero di AS del/dei *BGP peer*
  - Stato della sessione
  - *Holdtime*, *Keepalive* e tempo minimo tra due annunci
  - Numero di messaggi BGP inviati e ricevuti
  - Estremi della sessione BGP

Caratteristiche e stato delle sessioni (4/5)

```
RP/0/6/CPU0:PE3-3# show bgp neighbors 192.168.1.31
Wed Apr  3 09:08:22.908 UTC

BGP neighbor is 192.168.1.31
 Remote AS 64500, local AS 64500, internal link
 Remote router ID 192.168.1.31
  BGP state = Established, up for 00:41:17
  Last read 00:00:17, Last read before reset 00:00:00
  Hold time is 180, keepalive interval is 60 seconds
  Configured hold time: 180, keepalive: 60, min acceptable hold time: 3

  ... < output omesso > ...

  Multi-protocol capability received
  Neighbor capabilities:
    Route refresh: advertised (old + new) and received (old + new)
    4-byte AS: advertised
    Address family IPv4 Unicast: advertised and received

  Received 52 messages, 0 notifications, 0 in queue
  Sent 45 messages, 0 notifications, 0 in queue
  Minimum time between advertisement runs is 0 secs

                                                          (SEGUE)
```

Sessione iBGP

Caratteristiche e stato delle sessioni (5/5)

```
For Address Family: IPv4 Unicast
  BGP neighbor version 21
  Update group: 0.3 Filter-group: 0.1  No Refresh request being processed
  NEXT_HOP is always this router
  Route refresh request: received 0, sent 0
  8 accepted prefixes, 8 are bestpaths
  Cumulative no. of prefixes denied: 0.
  Prefix advertised 2, suppressed 0, withdrawn 1
  Maximum prefixes allowed 1048576
  Threshold for warning message 75%, restart interval 0 min
  AIGP is enabled
  An EoR was received during read-only mode
  Last ack version 21, Last synced ack version 0
  Outstanding version objects: current 0, max 2
  Additional-paths operation: None

  Connections established 1; dropped 0
  Local host: 192.168.0.33, Local port: 22712
  Foreign host: 192.168.1.31, Foreign port: 179
  Last reset 00:00:00
```

Contenuto della Tabella BGP (1/3)

- IOS e IOS XE

```
router# show bgp [(ipv4 | ipv6) unicast] ...
```

- IOS XR

```
RP/0/RP0/CPU0:router# show bgp [(ipv4 | ipv6) unicast] ...
```

- Permette la visualizzazione di (tra l'altro)
  - Annunci di ciascun prefisso, ricevuti dai *BGP peer*
  - Validità di ciascun annuncio
  - Se l'annuncio è iBGP o eBGP
  - Quale annuncio è stato eletto *best-path*
  - *BGP Next-Hop*, *Neighbor* ID e RID del *BGP peer* dal quale ciascun annuncio è stato ricevuto
  - Attributi BGP

Contenuto della Tabella BGP (2/3)

REISS ROMOLI

```
RP/0/6/CPU0:PE3-3# show bgp
Wed Apr  3 12:06:42.089 UTC

BGP router identifier 192.168.0.33, local AS number 64500
BGP generic scan interval 60 secs
BGP table state: Active
Table ID: 0xe0000000   RD version: 21
BGP main routing table version 21
BGP scan interval 60 secs

Status codes: s suppressed, d damped, h history, * valid, > best
              i - internal, r RIB-failure, S stale
Origin codes: i - IGP, e - EGP, ? - incomplete

   Network          Next Hop            Metric LocPrf Weight Path
*> 203.0.113.0/24   0.0.0.0                  0         32768 i
*>i111.111.0.0/16   192.168.1.31             0    100      0 64501 i
*>i142.2.0.0/16     192.168.1.31             0    100      0 64501 420 i
*>i190.213.0.0/16   192.168.1.31             0    100      0 64501 213 i
*>i220.37.1.0/24    192.168.1.31             0    100      0 64501 420 370 i
*>i220.42.1.0/24    192.168.1.31             0    100      0 64501 420 i
*>i220.213.1.0/24   192.168.1.31             0    100      0 64501 213 i
*>i222.151.1.0/24   192.168.1.31             0    100      0 64501 420 260 510
 i
*>i222.222.0.0/16   192.168.1.31             0    100      0 64501 420 260 222
 i

Processed 9 prefixes, 9 paths
```

MED

Origin

Prefisso appreso via iBGP

Local Preference

AS_PATH

Contenuto della Tabella BGP (3/3)

```
RP/0/6/CPU0:PE3-3# show bgp 222.222.0.0/16
Wed Apr  3 12:19:52.226 UTC
BGP routing table entry for 222.222.0.0/16
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 17          17
Last Modified: Apr  3 08:29:05.030 for 03:50:47
Paths: (1 available, best #1)
  Advertised to update-groups (with more than one peer):
    0.2
  Path #1: Received by speaker 0
  Advertised to update-groups (with more than one peer):
    0.2
  64501 420 260 222
    192.168.1.31 (metric 2) from 192.168.1.31 (192.168.1.31)
      Origin IGP, metric 0, localpref 100, valid, internal, best,
      Received Path ID 0, Local Path ID 1, version 17
```

AS_PATH

RID del *BGP peer*

BGP Next-Hop

Metrica IGP verso il *BGP Next-Hop* 192.168.1.31

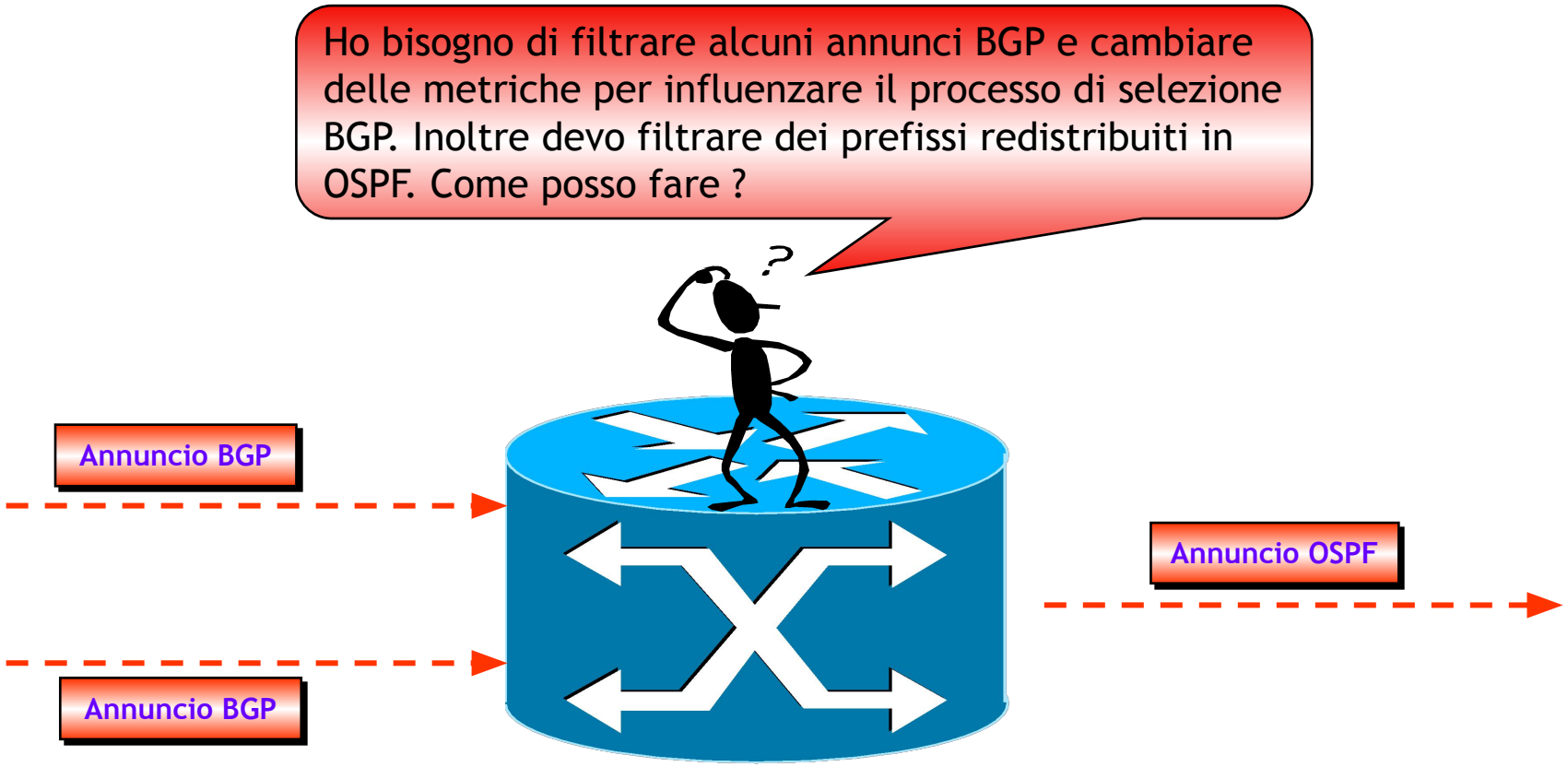
Indirizzo IP utilizzato per stabilire la sessione BGP (= *Neighbor ID*)

Strumenti per la manipolazione degli annunci

Generalità
Prefix-list
Route-map
Route-policy e *Route Policy Language*

Obiettivi

- Filtraggio degli annunci
- Manipolazione delle metriche e/o parametri vari



Strumenti

- *Prefix-list*: disponibili in tutte le versioni dell'IOS, sono uno strumento per il filtraggio degli annunci sulla base dei prefissi IPv4/IPv6
- *Access-list* basate sull'AS_PATH: disponibili in tutte le versioni dell'IOS, sono uno strumento per il filtraggio degli annunci sulla base del campo AS_PATH
- *Route-map*: disponibili nell'IOS e IOS XE, sono uno strumento sia per il filtraggio che per la manipolazione degli annunci
  - Sono l'unico strumento nell'IOS e IOS XE che consente di manipolare i campi degli annunci dei protocolli di routing
- *Route-policy*: sono l'analogo delle *route-map* nell'IOS XR
  - Utilizzano come strumento di configurazione il *Route Policy Language*

Strumenti per la manipolazione degli annunci

Generalità
Prefix-list
Route-map
Route-policy e *Route Policy Language*

REISS ROMOLI

Generalità

- Due criteri di filtraggio dei prefissi
  - Tradizionale: utilizzare ACL IP standard/estese congiuntamente al comando «**distribute-list ...**» (solo IOS e IOS XE)
  - Moderno: utilizzare le *Prefix-list*
- Entrambi i criteri
  - Permettono di filtrare i prefissi ricevuti e annunciati
  - Agiscono sui prefissi annunciati via BGP
- Le *Prefix-list* sono molto più flessibili e più semplici da realizzare

Configurazione di *Prefix-list*

- IOS e IOS XE

```
router(config)# ip prefix-list nome [seq numero-sequenza]
                                    permit|deny prefisso/maschera
                                    [ge lunghezza-minima-maschera]
                                    [le lunghezza-massima-maschera]
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# ipv4 prefix-list nome
RP/0/RP0/CPU0:router(config-ipv4_pfx)# [numero-sequenza]
                                    permit|deny prefisso/maschera
                                    [ge lunghezza-minima-maschera]
                                    [le lunghezza-massima-maschera]
                                    [eq lunghezza-maschera]
```

Esempi

```
 ip prefix-list A permit 172.16.1.0/24
!
 ip prefix-list B permit 172.16.0.0/16 le 20
!
 ip prefix-list C permit 192.168.1.0/24 ge 26
!
 ip prefix-list D permit 0.0.0.0/0
!
 ip prefix-list E permit 0.0.0.0/0 ge 8 le 24
!
 ip prefix-list F permit 0.0.0.0/0 le 32
!
 ip prefix-list NO-RFC1918 deny 10.0.0.0/8 le 32
 ip prefix-list NO-RFC1918 deny 172.16.0.0/12 le 32
 ip prefix-list NO-RFC1918 deny 192.168.0.0/16 le 32
 ip prefix-list NO-RFC1918 permit 0.0.0.0/0 le 32
```

Configurazione incrementale

- Configurazione di partenza

```
ip prefix-list XYZ seq 10 deny 198.51.100.0/24 le 32
ip prefix-list XYZ seq 15 deny 203.0.113.0/24 ge 25
ip prefix-list XYZ seq 20 permit 0.0.0.0/0 le 32
```

- Aggiungere una riga che nega tutte le *subnet* della rete 192.0.1.0/24

```
ip prefix-list XYZ seq 18 deny 192.0.2.0/24 ge 25
```

- Eliminare la riga che nega la rete 203.0.113/24 e tutte le sue *subnet*

```
no ip prefix-list XYZ seq 15 deny 203.0.113.0/24 ge 25
```

- *Prefix-list* finale

```
ip prefix-list XYZ seq 10 deny 198.51.100.0/24 le 32
ip prefix-list XYZ seq 18 deny 192.0.2.0/24 ge 25
ip prefix-list XYZ seq 20 permit 0.0.0.0/0 le 32
```

Verifica della Configurazione

- IOS e IOS XE

```
router# show ip prefix-list [detail|summary]
                                    nome-prefix-list [seq numero-seq]
```

- IOS XR

```
RP/0/RP0/CPU0:router# show prefix-list ipv4 nome-prefix-list
                                                  [numero-seq]
```

- Consentono di visualizzare la struttura del filtro *Prefix-list* o di parti di esso

Strumenti per la manipolazione degli annunci

Generalità
Prefix-list
Route-map
Route-policy e Route Policy Language

Definizione

- Sono gruppi di comandi che permettono di effettuare operazioni del tipo

  if

  {*condizioni*}

  then

  {*azioni*}

- Le condizioni possono essere di diversi tipi e basate su ACL tradizionali, filtri di diversi tipi, ecc.

- Le azioni (opzionali) consentono di modificare uno o più attributi BGP, o, in altri contesti, altri tipi di parametri

- NOTA: le *route-map* trovano applicazione in altri settori come redistribuzione tra protocolli di routing, classificazione del traffico, *import/export-map* nelle VPN IP BGP/MPLS, ecc.

Route-map nel BGP: condizioni e azioni

- Nelle applicazioni delle *route-map* al BGP le condizioni (*match*) possono essere di svariati tipi
  - ACL
  - *Prefix-list*
  - *Router ID* del router che origina un annuncio
  - Indirizzo del *BGP Next-Hop*
  - AS_PATH
  - *BGP Community* associate all'annuncio BGP
  - . . .
- Le azioni (*set*) possono agire su vari attributi BGP e non
  - *Origin*
  - *BGP Next-Hop*
  - *Weight* (metrica proprietaria Cisco)
  - *BGP Community*
  - *Local Preference*
  - MED

Logica di Elaborazione

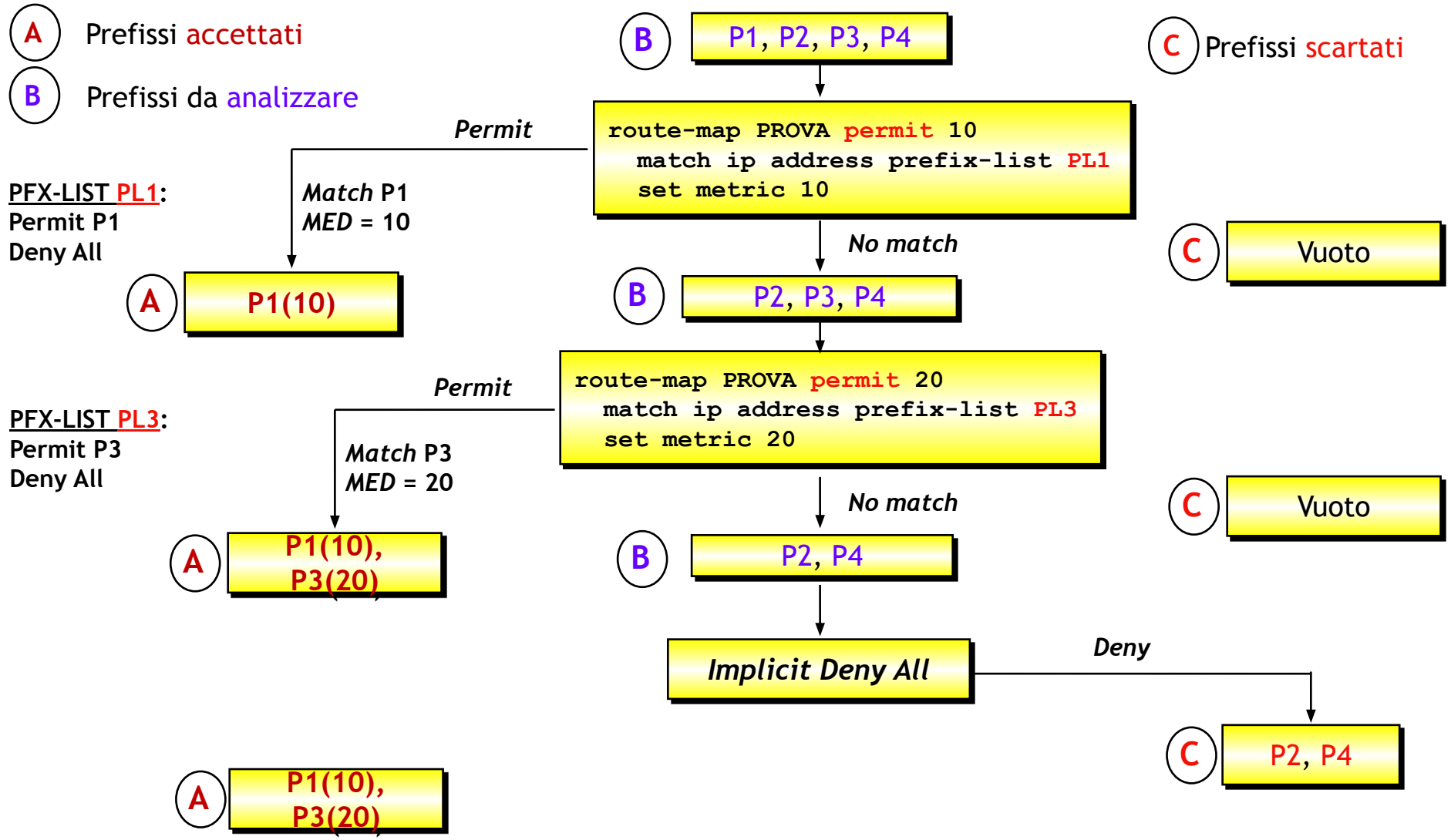
if {condizioni "*match*" soddisfatte}
then {
    if {route-map è di tipo "*permit*"}
        then {esegui le azioni "*set*"}
    else {non eseguire le azioni "*set*"}
    exit route-map
}
else {elabora il gruppo di istruzioni successivo
    con numero di sequenza maggiore}

Configurazione

```
router(config)# route-map nome [permit|deny numero-sequenza]
router(config-route-map)# match condizione-1
. . .
router(config-route-map)# match condizione-N
router(config-route-map)# set attributo-1
. . .
router(config-route-map)# set attributo-N
router(config-route-map)# [continue [numero-sequenza]]
```

- Regole fondamentali
  - Qualora non venga specificata alcuna parola chiave *permit* o *deny* il default è *permit*
  - Esiste un *implicit deny all* finale, ossia una ultima riga del tipo «**route-map** *nome* **deny** *numero-sequenza*»
  - La condizione *permit all* è definita da una linea iniziale di tipo *permit* senza *match*
  - Più condizioni *match* dello stesso tipo sono trattate in OR, mentre più condizioni di *match* di tipo diverso sono trattate in AND
  - La prima riga della *route-map* pone in *permit*/*deny* le condizioni di *match*
    - Quando utilizzate come filtro, *permit* indica accetta/invia gli annunci che soddisfano le condizioni di *match*, *deny* indica il filtraggio (scarto)

Esempio di elaborazione



Route-map nel BGP: esempi

1

```
route-map SET-METRIC permit 10
  match ip address prefix-list ABC
  set metric 5
route-map SET-METRIC permit 20
!
ip prefix-list ABC permit 198.51.100.0/24
```

2

```
route-map SET-METRIC permit 10
  match community 1 2
  set metric 5
route-map SET-METRIC permit 20
!
ip community-list 1 permit 64501:1
ip community-list 2 permit 64501:2
```

3

```
route-map SET-METRIC permit 10
  match ip address prefix-list ABC
  match community 3
  set metric 5
route-map SET-METRIC permit 20
!
ip prefix-list ABC permit 198.51.100.0/24
ip community-list 3 permit 64501:3
```

Strumenti per la manipolazione degli annunci

Generalità
Prefix-list
Route-map
Route-policy e Route Policy Language

REISS ROMOLI

Caratteristiche

- Sono l'analogo delle *route-map* per l'IOS XR
  - Stessa struttura del tipo «**if** {*condizioni*} **then** {*azioni*}»
- Stesso utilizzo per filtraggio e manipolazione degli annunci, ma modalità di configurazione completamente diversa
  - Sintassi simile a quella dei comuni linguaggi di programmazione
- Rispetto alle *route-map*
  - Migliore modularità e riusabilità
  - Possibilità di parametrizzazione
  - Nidificazione di *policy* e condizioni
  - Opzioni per le condizioni più potenti
  - Insiemi di valori (es. prefissi, *Community*, AS_PATH, ecc.) riutilizzabili

Struttura generale di configurazione

- Ogni *Routing Policy* è identificata da un nome (sensibile alle maiuscole/minuscole) e definita tra i comandi «**route-policy**» e «**end-policy**»

```
route-policy nome-RP
   ...
   ...
   ...
end-policy
```

- Funzioni principali
  - Filtraggio di annunci (comandi «**pass**» e «**drop**»)
  - Manipolazione di attributi (comandi «**set**»)

Azioni «pass» e «drop»

- L'azione «**pass**» implica l'accettazione di un annuncio
  - L'utilizzo esplicito dell'azione «**pass**» implica la continuazione dell'elaborazione della *Routing Policy*
- L'azione «**drop**» implica il rifiuto di un annuncio
  - L'utilizzo esplicito dell'azione «**drop**» implica la terminazione dell'elaborazione della *Routing Policy*
  - L'azione «**drop**» è l'azione di default (*implicit deny all*)
- Ogni azione di modifica è un «**pass**» implicito

```
route-policy TEST-1
end-policy
```

Drop

```
route-policy TEST-2
  pass
end-policy
```

Pass

```
route-policy TEST-3
  drop
end-policy
```

Drop

```
route-policy TEST-4
  set med 10
end-policy
```

Pass

```
route-policy TEST-5
  pass
  drop
  pass
end-policy
```

Drop

Condizioni

- Permettono di individuare annunci di protocolli di routing sulla base di vari campi, dipendenti dal tipo di protocollo
- Sono inserite dopo le parole chiave «**if**» o «**elseif**» e sono del tipo «*attributo operatore valore*»
- Sintassi generale di configurazione

```
route-policy nome-RP
  if condizione-1 then
    azione-1
    [ elseif condizione-2 then
       azione-2 ]
  else
    azione-3
  endif
end-policy
```

Operatori

- Consentono di confrontare un attributo con un valore dell'attributo
  - **eq:** un attributo uguale a uno specifico valore
  - **le:** un attributo numericamente inferiore o al più uguale a uno specifico valore
  - **ge:** un attributo numericamente superiore o al più uguale a uno specifico valore
  - **is:** un attributo non numerico uguale a uno specifico valore
  - **in:** un attributo non numerico contenuto in un insieme di valori
  - Altre opzioni specifiche di particolari attributi

```
route-policy TEST
  if med eq 20 then
    set local-preference 150
  elseif med eq 30 then
    set local-preference 140
  else
    set local-preference 50
  endif
end-policy
```

Operatori booleani (1/2)

- Consentono di combinare più condizioni
  - **and:** tutte le condizioni devono essere soddisfatte
  - **or:** almeno una condizione deve essere soddisfatta
  - **not:** negazione della condizione

```
route-policy TEST
  if med eq 20 and not local-preference eq 100 then
    set local-preference 150
    elseif med eq 30 or local-preference eq 200 then
       set local-preference 140
  else
    set local-preference 50
  endif
end-policy
```

REISS ROMOLI

Operatori booleani (2/2)

- Precedenza degli operatori booleani
  - **not**: precedenza più elevata
  - **and**: precedenza più elevata di «**or**» e più bassa di «**not**»
  - **or**: precedenza più bassa

- Per evitare dubbi è possibile utilizzare le parentesi «(...)»

```
if med eq 20 and not local-preference eq 100 or weight eq 30 then ...
```

=

```
if ((med eq 20) and (not (local-preference eq 100)))
                                    or (weight eq 30) then ...
```

Azioni di tipo «set» (1/2)

- Consentono di variare attributi e/o parametri contenuti negli annunci
  - Quando più azioni «**set**» riguardano lo stesso attributo/parametro, vale l'ultima azione configurata

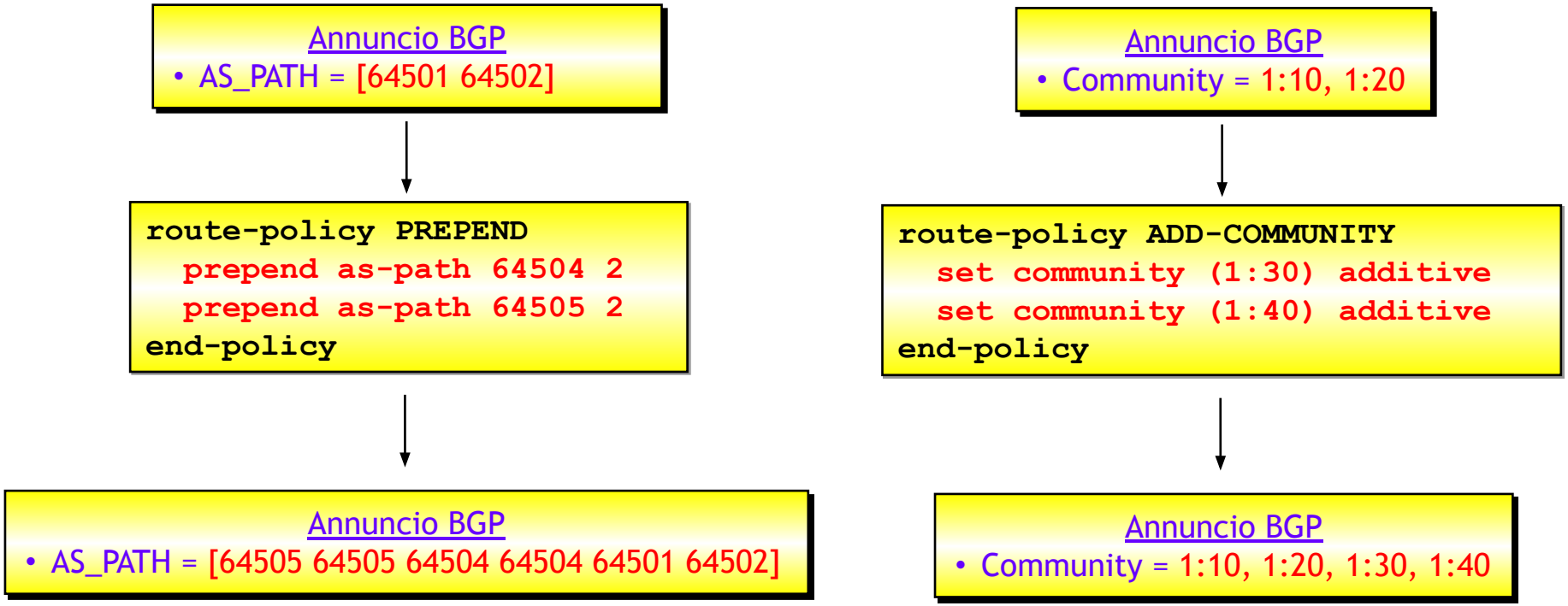
Tabella BGP
- MED = 20
- Local Preference = 100
- Weight = 0

```
route-policy TEST
  if weight eq 0 then
    set local-preference 150
  endif
  if local-preference eq 100 then
    set med 100
  endif
  if med eq 20 then
    set med 200
  endif
end-policy
```

Annuncio BGP
- MED = 200
- Local Preference = 150

Azioni di tipo «set» (2/2)

- Quando un comando «**set**» elabora un attributo costituito da un insieme di valori (es. AS_PATH, Community), tutti i comandi «**set**» possono avere effetto



REISS ROMOLI

Alcune azioni di tipo «**set**» nel BGP

- Manipolazione delle *Community standard*
  - Formati ammessi: decimale, esadecimale, *AS:NN*

```
set community (valore-1 [valore-2] ...) [additive]
```

- Manipolazione delle *Community estese*
  - Formati ammessi: decimale, esadecimale, *AS:NN*

```
set extcommunity rt (valore-1 [valore-2] ...) [additive]
```

- Manipolazione del *Local Preference*

```
set local-preference valore
```

- Manipolazione del MED

```
set med { [+ | -] valore | igp-cost | max-reachable }
```

Altre azioni nel BGP

- Eliminazione delle *Community standard*

```
delete community { all | [not] in community-set }
```

- Eliminazione delle *Community estese* di tipo *Route Target*

```
delete extcommunity rt { all | [not] in extcommunity-set }
```

- Manipolazione dell'AS_PATH per ***AS_PATH prepending***

```
prepend as-path { AS | most-recent } [numero-volte]
```

- Sostituzione nell'AS_PATH di una sequenza di AS o di AS privati con il proprio AS

```
replace as-path { 'AS1 AS2 ...' | private-as }
```

Parametrizzazione (1/2)

- Utilizzo di parametri globali
  - I parametri globali si definiscono nella sezione «**policy-global ... end-global**» e devono essere separati da una virgola
  - I parametri hanno un nome e i valori vanno definiti tra apici singoli
  - Il riferimento ai parametri globali all'interno della *route-policy* si fa anteponendo al nome del parametro il simbolo «$»
  - NOTA : lo stesso insieme di parametri può essere utilizzato da più *route-policy*

- Esempio

```
policy-global
 # PARAMETRI GLOBALI
 AS '65201',
 Lo0 '192.168.0.11',
 DefaultWeight '0',
 DefaultLP '150',
 DefaultMED '0'
end-global
```

```
route-policy SETLP
  if as-path originates-from '$AS' then
    set local-preference $DefaultLP
  endif
end-policy
```

Parametrizzazione (2/2)

- Passaggio di parametri
  - I parametri vanno dichiarati durante la creazione di una *route-policy*
  - Concetto analogo alle classiche *subroutine* dei linguaggi di programmazione

- Esempio

```
route-policy SETLP($LP)
   set local-preference $LP
end-policy
!
route-policy TEST
  if as-path neighbor-is '100' then
    apply SETLP(150)
  elseif as-path neighbor-is '200' then
    apply SETLP(200)
  endif
end-policy
```

Insiemi di valori

- Le condizioni di una *route-policy* possono utilizzare insiemi di valori
  - Gli insiemi di valori vanno specificati tra parentesi tonde (*inline set*), o raggruppati in un insieme separato identificato da un nome (*named set*) per maggiore riusabilità
  - Insiemi di valori permessi: AS_PATH, *Community* standard ed estese, prefissi, ...

Insieme di valori «*inline*»

```
route-policy SETLP
  if attributo in (valore-1, valore-2, ...) then
    set local-preference 150
  endif
end-policy
```

```
as-path-set
community-set
extcommunity-set
prefix-set
...
```

```
Tipo-set nome-value-set
  valore-1,
  valore-2,
  ...
end-set
!
route-policy SETLP
  if attributo in nome-value-set then
    set local-preference 150
  endif
end-policy
```

Insiemi di valori: «prefix-set»

- I «prefix-set» sono costituiti da insiemi di prefissi
  - È possibile utilizzare le clausole «le», «ge» e «eq» già viste nelle *prefix-list*

- Esempio

```
prefix-set RFC1918
  10.0.0.0/8 le 32,
  172.16.0.0/12 le 32,
  192.168.0.0/16 le 32
end-set
!
route-policy NO-RFC1918
  if destination in RFC1918 then
    drop
  else
    pass
  endif
end-policy
```

Editing delle *routing policy* (1/2)

- È possibile utilizzare la CLI oppure 3 diversi tipi di *editor*
  - *GNU nano, Emacs, VIM*
- Dopo aver modificato la *routing policy* ...
  - Salvare le modifiche
  - Uscire dall'*editor*
  - Mandare in esecuzione le modifiche tramite «**commit**»
- NOTA IMPORTANTE: la modifica di una *routing policy* esistente attraverso l'utilizzo della CLI comporta la riscrittura completa della *routing policy*

```
RP/0/RP0/CPU0:router(config)# route-policy TEST
% Warning: Policy object 'route-policy TEST' exists! Reconfiguring
it via CLI will replace current definition. Use 'abort' to cancel.
RP/0/RP0/CPU0:router(config-rpl)# abort
```

Editing delle *routing policy* (2/2)

- È possibile utilizzare uno dei 3 *editor* disponibili sia per l'*editing* delle *routing policy* che degli insiemi di valori e parametri

```
RP/0/RP0/CPU0:router# edit ?
  as-path-set         edit an as-path-set
  community-set       edit a community-set
  extcommunity-set    edit an extended-community-set
  ospf-area-set       edit a ospf-area-set
  policy-global       edit policy-global definitions
  prefix-set          edit a prefix-set
  rd-set              edit a rd-set
  route-policy        edit a route-policy
  tag-set             edit a tag-set
```

- Scelta dell'*editor*

```
RP/0/RP0/CPU0:router# edit route-policy TEST ?
  emacs         to use Micro Emacs editor
  nano          to use nano editor
  vim           to use Vim editor
  <cr>
```

REISS ROMOLI

Applicazione delle *routing policy* (1/2)

- Passi da eseguire
  1. Progettazione
  2. Configurazione
  3. Test utilizzando i comandi «**show** ...»
  4. Applicazione
     - Annunci dei protocolli di routing (BGP, OSPF, IS-IS, ...)
     - Redistribuzione
     - Comando «**network** ...» nel BGP
     - Inserimento di una *route* nella Tabella di Routing
     - Comandi «**show** ...» per il filtraggio della visualizzazione

- In funzione del contesto, l'applicazione può essere nella direzione *input* o nella direzione *output*
  - Esempio: *routing policy* utilizzata per il filtraggio *inbound*/*outbound* di annunci BGP

Applicazione delle *routing policy* (2/2)

- La validità di una *routing policy* è verificata in due fasi:
  - Controllo della sintassi e dei valori durante l'*editing* e durante la fase di «**commit**»

```
RP/0/RP0/CPU0:router(config-rpl)# set local-preference 12345678909876543321
                                                       ^

% Invalid input detected at '^' marker.
```

```
RP/0/RP0/CPU0:router(config)# commit
Fri Apr 19 10:15:13.034 UTC

% Failed to commit one or more configuration items during a pseudo-atomic
operation. All changes made have been reverted. Please issue 'show
configuration failed' from this session to view the errors

RP/0/RP0/CPU0:router(config)# show configuration failed
Fri Apr 19 10:15:23.856 UTC
... < output omesso > ...
router bgp 11
 neighbor 172.16.1.11
  address-family ipv4 unicast
   route-policy TEST-2 in
!!% Could not find entry in list: Policy [TEST-2] uses the 'metric-type'
attribute. There is no 'metric-type' attribute at the bgp neighbor-in-dflt
attach point.
```

Verifica della configurazione

REISS ROMOLI

```
RP/0/RP0/CPU0:router# show rpl route-policy [nome-RP] [detail]
```

```
RP/0/6/CPU0:PE3-3# show rpl route-policy TEST detail
Fri Apr 19 11:36:30.708 UTC
prefix-set 222
  222.0.0.0/8 le 32
end-set
!
route-policy TEST
  if destination in 222 then
    pass
  endif
end-policy
```

```
RP/0/RP0/CPU0:router# show rpl route-policy nome-RP attachpoints
```

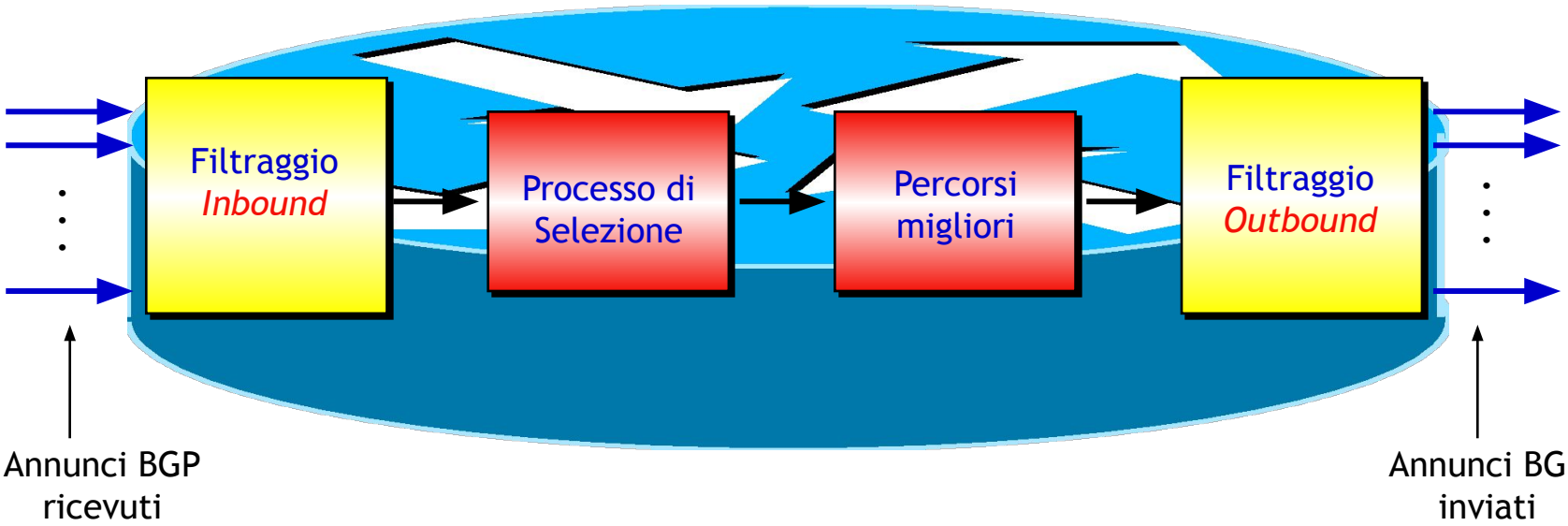
```
RP/0/6/CPU0:PE3-3# show rpl route-policy TEST attachpoints
Fri Apr 19 11:40:16.355 UTC

BGP Attachpoint: Neighbor

Neighbor/Group          type   afi/safi    in/out     vrf name   bound by
-------------------------------------------------------------------------

172.16.1.11             --     IPv4/uni    in         default    TEST
```

Filtraggio degli annunci BGP

Definizione e motivazioni
Filtri basati sui prefissi
Filtri basati sulle *Community*
Applicazione dei filtri

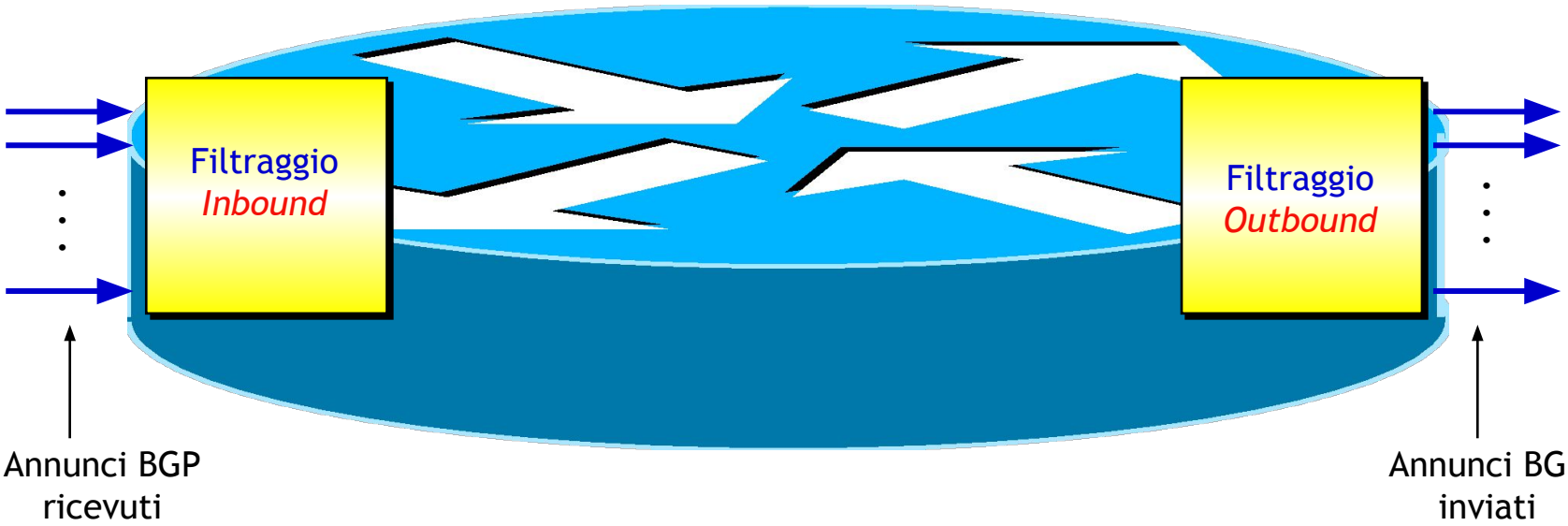
A cosa serve il filtraggio

- Permette a un *BGP speaker* di scegliere quali annunci accettare dai *BGP peer* e quali inviare ai *BGP peer*
- Tipi di filtraggio
  - Filtraggio *Inbound*
  - Filtraggio *Outbound*
  - Filtraggio dei prefissi locali redistribuiti da protocolli IGP in BGP



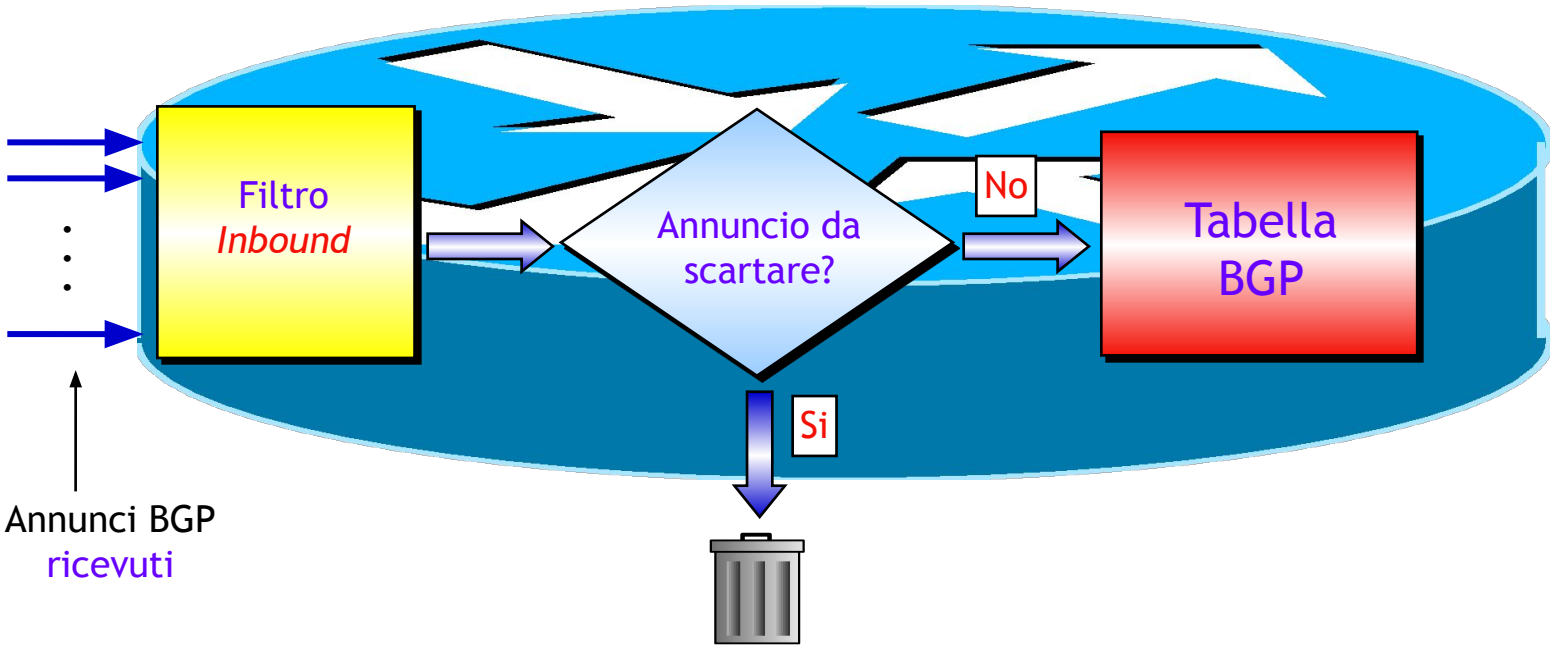
Perché filtrare?

- È la vostra prima linea di difesa
- Consente di controllare ciò che si annuncia ai *BGP peer* (filtraggio *outbound*)
  - Un AS non ha alcun controllo su che annunciano gli altri AS
- Consente di controllare ciò che si accetta da altri AS (filtraggio *inbound*)



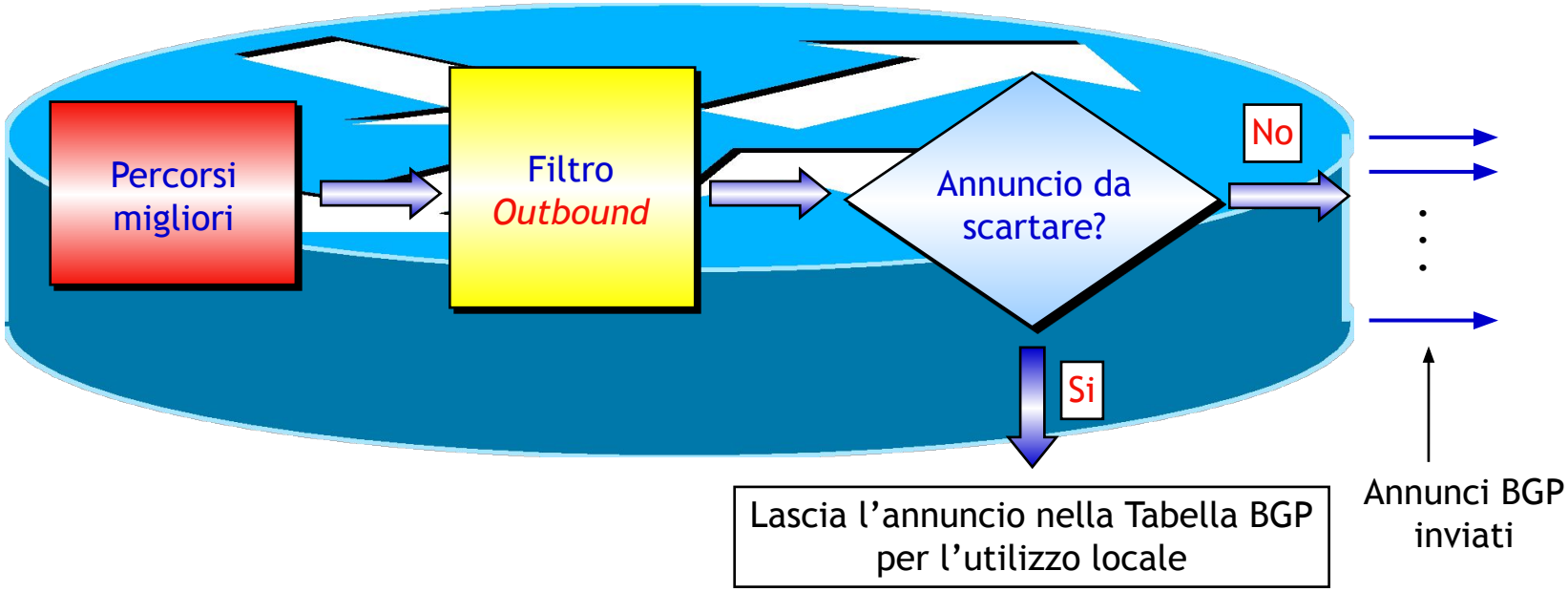
Filtraggio *Inbound*

- Funzionamento: gli annunci ricevuti dai vari *BGP peer* e non permessi dal filtro NON vengono inseriti nella tabella BGP



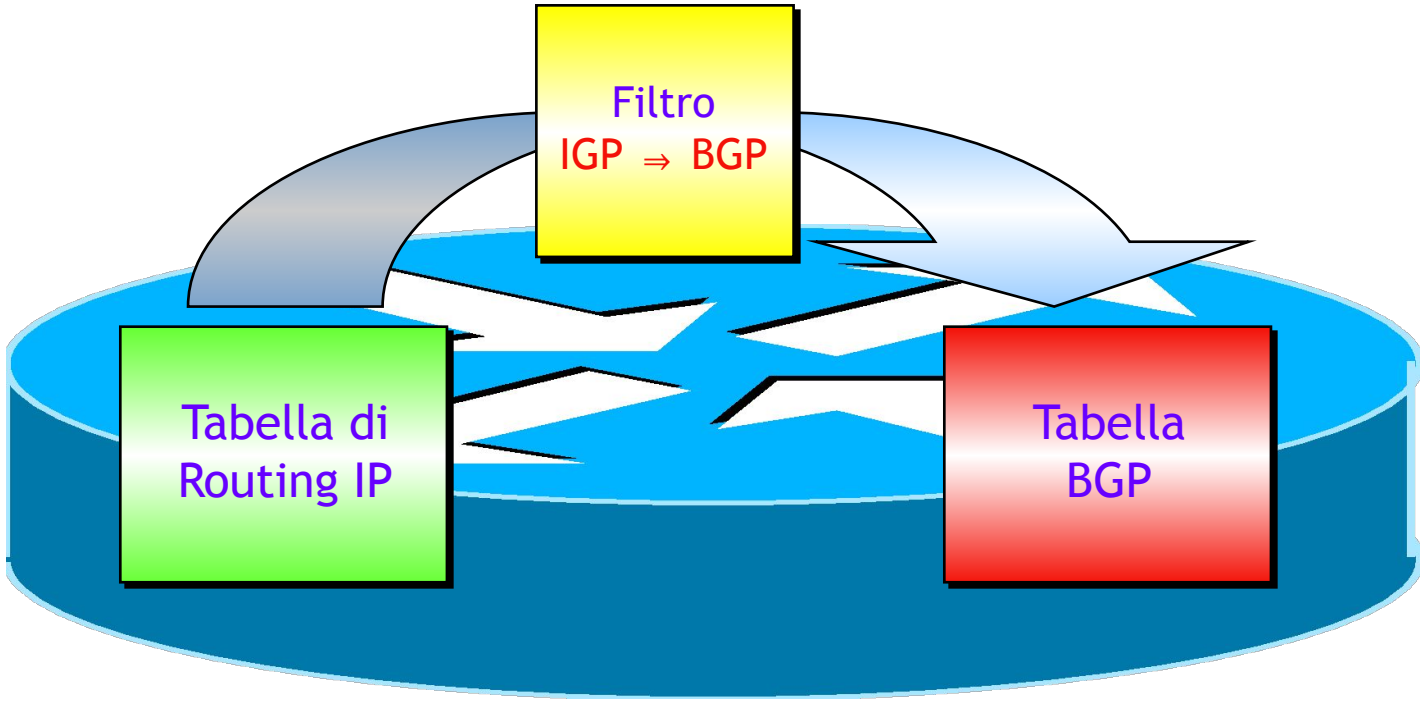
Filtraggio *Outbound*

- Funzionamento: i prefissi da inoltrare ai vari *BGP peer* e conformi al filtro NON vengono annunciati



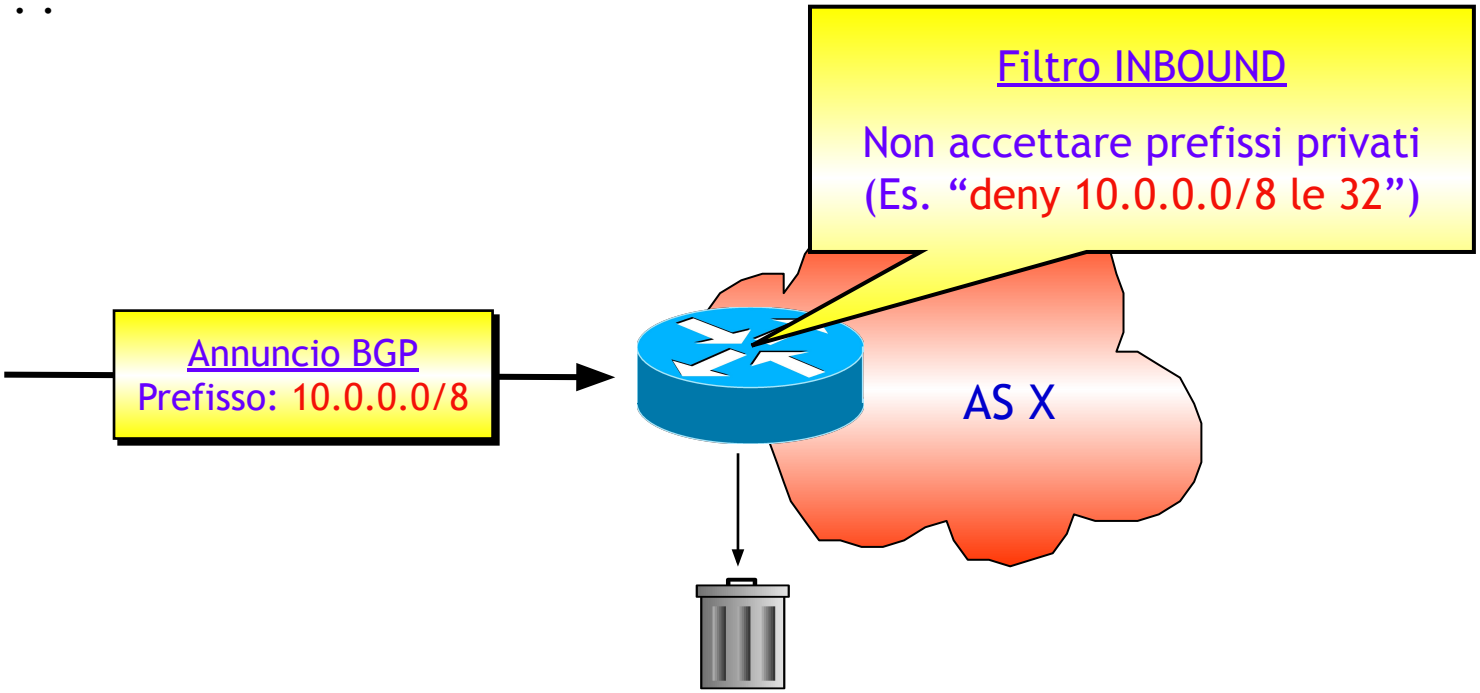
Filtraggio dei prefissi redistribuiti

- Permette di filtrare i prefissi presenti nella Tabella di Routing IP e redistribuiti da un protocollo IGP in BGP



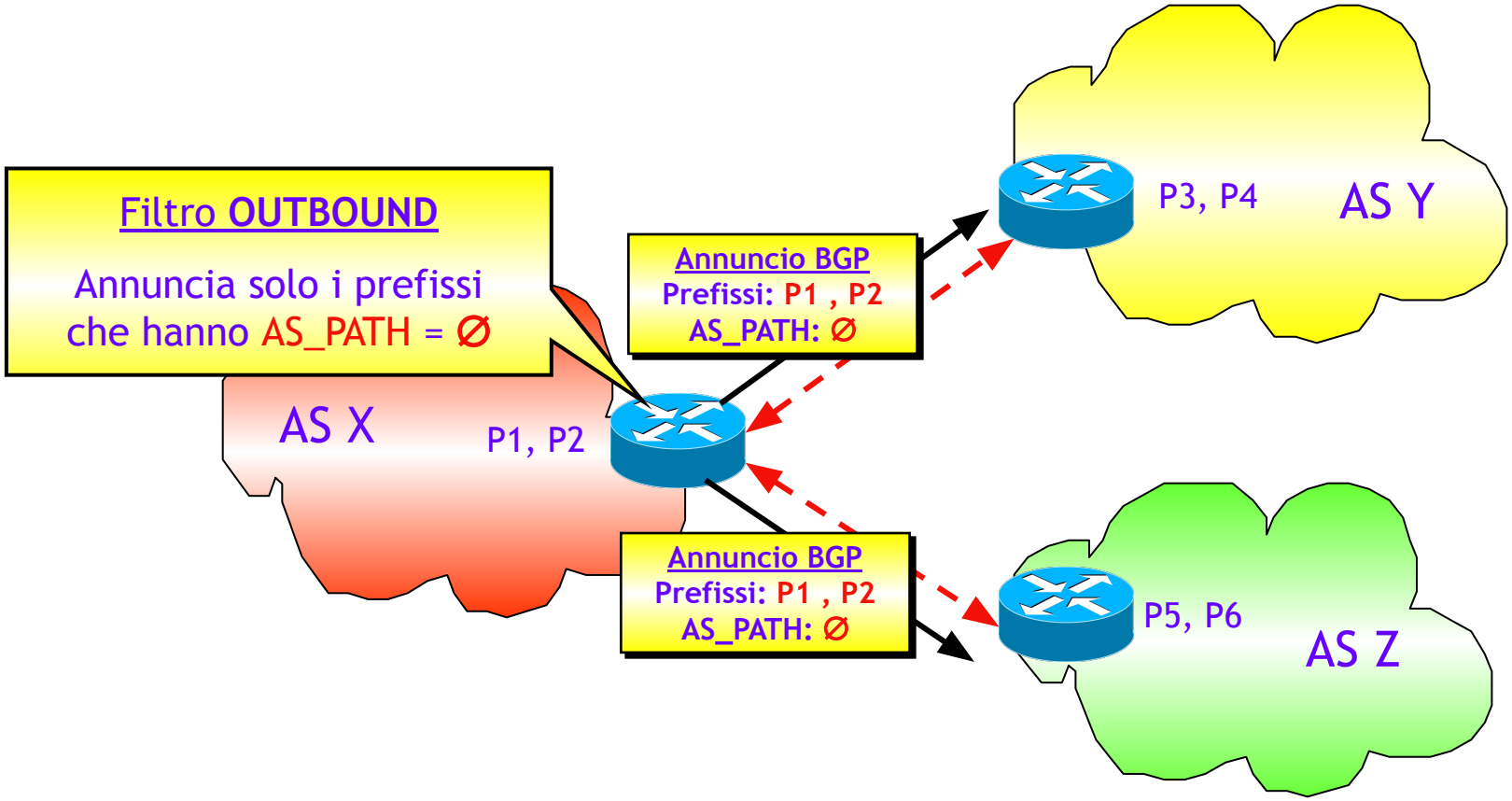
Esempio 1

- Non accettare annunci di prefissi riservati o con strutture particolari
  - Prefissi della RFC 1918
  - *martian list*
  - *default route*
  - Prefissi con maschera superiore ad un determinato valore (es. /24 per IPv4 e /48 per IPv6)
  - . . .



Esempio 2

- Impedire che un AS di un cliente diventi di transito
  - Annuncia solo i prefissi locali (identificati da un AS_PATH vuoto)



Le buone regole del filtraggio *inbound* ...

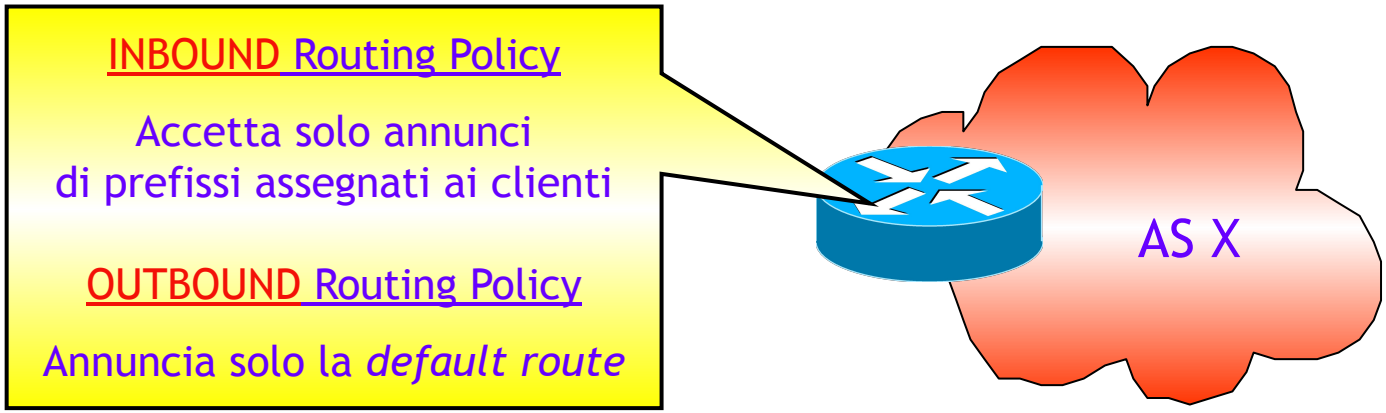
- NON ACCETTARE annunci con AS *bogon* (non permessi)
  - Es. annunci con AS privati, con AS riservati, ecc.
- NON ACCETTARE annunci con prefissi *bogon* (non permessi)
  - Es. annunci di prefissi privati, *martian list*, non ancora allocati da IANA, ecc.
- NON ACCETTARE annunci di prefissi propri
  - Es. prefissi allocati da un RIR all'AS
- NON ACCETTARE annunci della *default route* (a meno che non sia esplicitamente richiesta)
- NON ACCETTARE annunci troppo specifici (ossia di prefissi con maschera superiore ad un determinato valore
  - Es. /24 per IPv4 e /48 per IPv6
- NON ACCETTARE annunci con AS_PATH troppo lunghi
  - Es. AS_PATH di lunghezza ≥ 15
- Creare i filtri sulla base delle informazioni contenute negli IRR

Filtraggio degli annunci BGP

Definizione e motivazioni
Filtri basati sui prefissi
Filtri basati sulle *Community*
Applicazione dei filtri

Scenari di applicazione

- Scartare in ingresso annunci di prefissi particolari
  - *bogons* (vedi note)
  - *default route*
  - annunci di prefissi troppo specifici
  - annunci di prefissi propri
  - . . .
- Propagare solo annunci di determinati prefissi



Configurazione di filtri *Prefix-list*

- La configurazione di un filtro *Prefix-list* avviene in due fasi
  - Definizione del filtro *Prefix-list*
  - Applicazione del filtro agli annunci *Inbound*/*Outbound* di un particolare *BGP peer*
    - NOTA: l'applicazione di un filtro *Prefix-list* non definito implica l'accettazione/invio di tutti i prefissi (*permit any*)

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# neighbor indirizzo-IP-peer
                              prefix-list nome-prefix-list in|out
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy filtro in|out
```

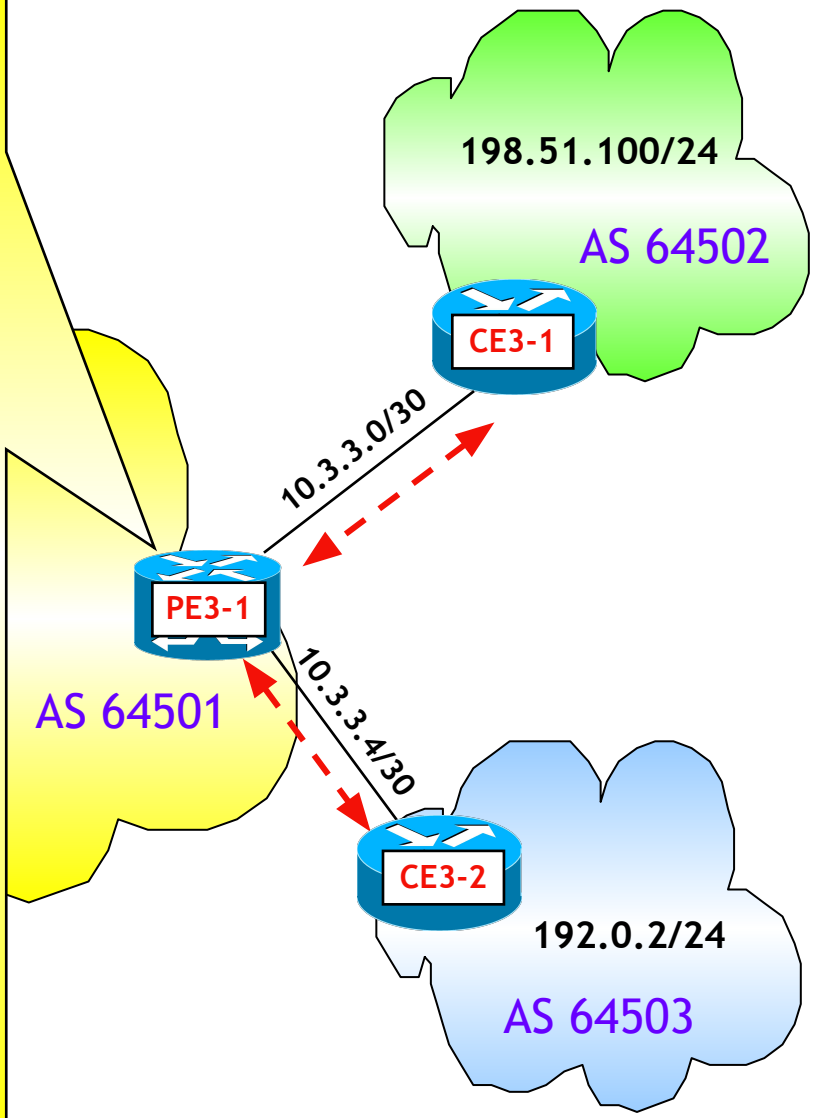
Esempio 1

```
prefix-set DENY-NET
  10.0.0.0/8 le 32,
  172.16.0.0/12 le 32,
  192.168.0.0/16 le 32,
  0.0.0.0/8 le 32,
  127.0.0.0/8 le 32,
  169.254.0.0/16 le 32,
  192.0.0.0/24 le 32,
  192.0.2.0/24 le 32,
  192.88.99.0/24 le 32,
  198.18.0.0/15 le 32,
  198.51.100.0/24 le 32,
  203.0.113.0/24 le 32,
  100.64.0.0/10 le 32,
  0.0.0.0/0 ge 25,
  0.0.0.0/0,
  224.0.0.0/3 le 32
end-set
!
route-policy NET-NOT-ALLOWED
  if destination in DENY-NET then
    drop
  else
    pass
  endif
end-policy
```

Esempio 2

REISS

```
route-policy SOLO-DEFAULT
 if destination in (0.0.0.0/0) then
     pass
 endif
end-policy
!
route-policy AS64502
 if destination in (198.51.100.0/24) then
     pass
 endif
end-policy
!
route-policy AS64503
 if destination in (192.0.2.0/24) then
     pass
 endif
end-policy
!
router bgp 64501
 address-family ipv4 unicast
 neighbor 10.3.3.2
  remote-as 64502
  address-family ipv4 unicast
   route-policy AS64502 in
   route-policy SOLO-DEFAULT out
 neighbor 10.3.3.6
  remote-as 64503
  address-family ipv4 unicast
   route-policy AS64503 in
   route-policy SOLO-DEFAULT out
```



Filtraggio degli annunci BGP

Definizione e motivazioni
Filtri basati sui prefissi
Filtri basati sulle *Community*
Applicazione dei filtri

REISS ROMOLI

Attributo *Community*: utilizzo

- Definire particolari politiche di filtraggio dei prefissi
  - Esempio: non annunciare i prefissi che hanno associato un determinato valore di *Community*
- Definire particolari politiche di routing per gruppi di prefissi omogenei
  - Esempio: assegnare diversi valori di *Local Preference* sulla base del valore dell'attributo *Community*
- Definire politiche di Qualità del Servizio sulla base di SLA definiti tra Clienti e ISP
  - Esempio: differenziare i livelli di Qualità del Servizio offerti ai Clienti (es. *Gold*, *Silver*, *Bronze*, *Best Effort*) sulla base del valore dell'attributo *Community*

Assegnazione via *route-map* e *route-policy*

- IOS e IOS XE

```
router(config)# route-map nome permit numero-sequenza
router(config-route-map)# condizioni
router(config-route-map)# set community valore [valore […]]
                                                    [additive]
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# route-policy nome-RP
RP/0/RP0/CPU0:router(config-rpl)# condizioni
RP/0/RP0/CPU0:router(config-rpl)# set community (valore [valore […]])
                                                    [additive]
```

- Regole fondamentali
  - È possibile specificare più valori di *Community*
  - I valori specificati sovrascrivono eventuali valori di C*ommunity* esistenti, a meno di utilizzare la parola chiave «**additive**», nel qual caso vengono aggiunti a quelli esistenti

Propagazione sulle sessioni BGP

- Di default gli attributi *Community* non vengono propagati sulle sessioni eBGP
  - NOTA: nell'IOS e IOS XE non vengono propagati nemmeno sulle sessioni iBGP
- L'eventuale propagazione va configurata manualmente
- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# neighbor indirizzo-IP-peer send-community
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp)# neighbor IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# send-community-ebgp
```

Condizioni basate sui valori di *Community*

- Nell'IOS e IOS XE, per le condizioni «*match*» delle *route-map* vengono utilizzate le *Community-list*, che consentono di definire un insieme di attributi *Community*
  - Possono essere di tipo standard o esteso
  - *Community-list* standard: permettono di definire un insieme ben definito di *Community*
  - *Community-list* estese: permettono di definire un insieme di C*ommunity* attraverso delle *RegExp*

- Nell'IOS XR, è possibile creare insiemi di valori di *Community* di tipo *inline set* o *named set*
  - È possibile definire un insieme di C*ommunity* o numericamente o attraverso delle *RegExp*

Configurazione di *Community-list*

- *Community-list* standard
  - Nelle *community-list* standard è possibile utilizzare la parola chiave «**internet**» per definire un qualsiasi valore di *community*

```
router(config)# ip community-list 1-99 permit|deny [valore […]]
```

- *Community-list* estese
  - Gli attributi *Community* associati ad un annuncio vengono convertiti in una stringa e quindi rappresentati attraverso *RegExp* come nel caso di filtri AS_PATH

```
router(config)# ip community-list 100-199 permit|deny regexp
```

- Applicazione alle condizioni «*match*» delle *route-map*

```
router(config)# route-map nome permit numero-sequenza
router(config-route-map)# match community numero
router(config-route-map)# < azioni >
```

Community-list: esempi

- *Match* su annunci che hanno due qualsiasi valori di *Community*

```
ip community-list 1 permit internet internet
```

- *Match* su annunci che hanno entrambi i valori di *Community* specificati

```
ip community-list 2 permit 64501:150 64501:200
```

- *Match* su annunci che contengono almeno un valore di *Community* di tipo 64501:1x (x=0,1,…,9)

```
ip community-list 100 permit _64501:1[0-9]_
```

- *Match* su annunci che contengono un qualsiasi insieme di *Community*

```
ip community-list 101 permit .*
```

Community-set (IOS XR)

- Nell'IOS XR è possibile creare insiemi di valori di *Community* utilizzando valori numerici, mnemonici e *RegExp*

- Quando si utilizzano valori numerici sono possibili rappresentazioni del tipo
  - *AS:numero*
  - *AS:*[*min..max*]
  - *AS:**

- Valori mnemonici
  - **internet:** indica qualsiasi valore di *Community*
  - **local-as:** indica la *Community well-known* «**NO_EXPORT_SUBCONFED**»
  - **no-advertise:** indica la *Community well-known* «**NO_ADVERTISE** »
  - **no-export:** indica la *Community well-known* «**NO_EXPORT** »

Community-set nelle *route-policy* (1/2)

- Tre tipi di *match*
  - «**is-empty**»: *match* soddisfatto se un annuncio non contiene alcun attributo *Community*
  - «**matches-any**»: *match* soddisfatto se un annuncio contiene almeno un valore di *Community* specificato nel *Community-set*
  - «**matches-every**»: *match* soddisfatto se un annuncio contiene tutti i valori di *Community* specificati nel *Community-set*

```
community-set PRIMARY
  ios-regex '64501:[12]..$'
end-set
!
route-policy AS-P
  if community matches-any PRIMARY then
    set local-preference 200
  endif
end-policy
```

Community-set nelle *route-policy* (2/2)

Annuncio eBGP

Prefisso: 10.25.0/24

- COMMUNITY = 64501:10
- COMMUNITY = 64501:20
- COMMUNITY = 64501:30

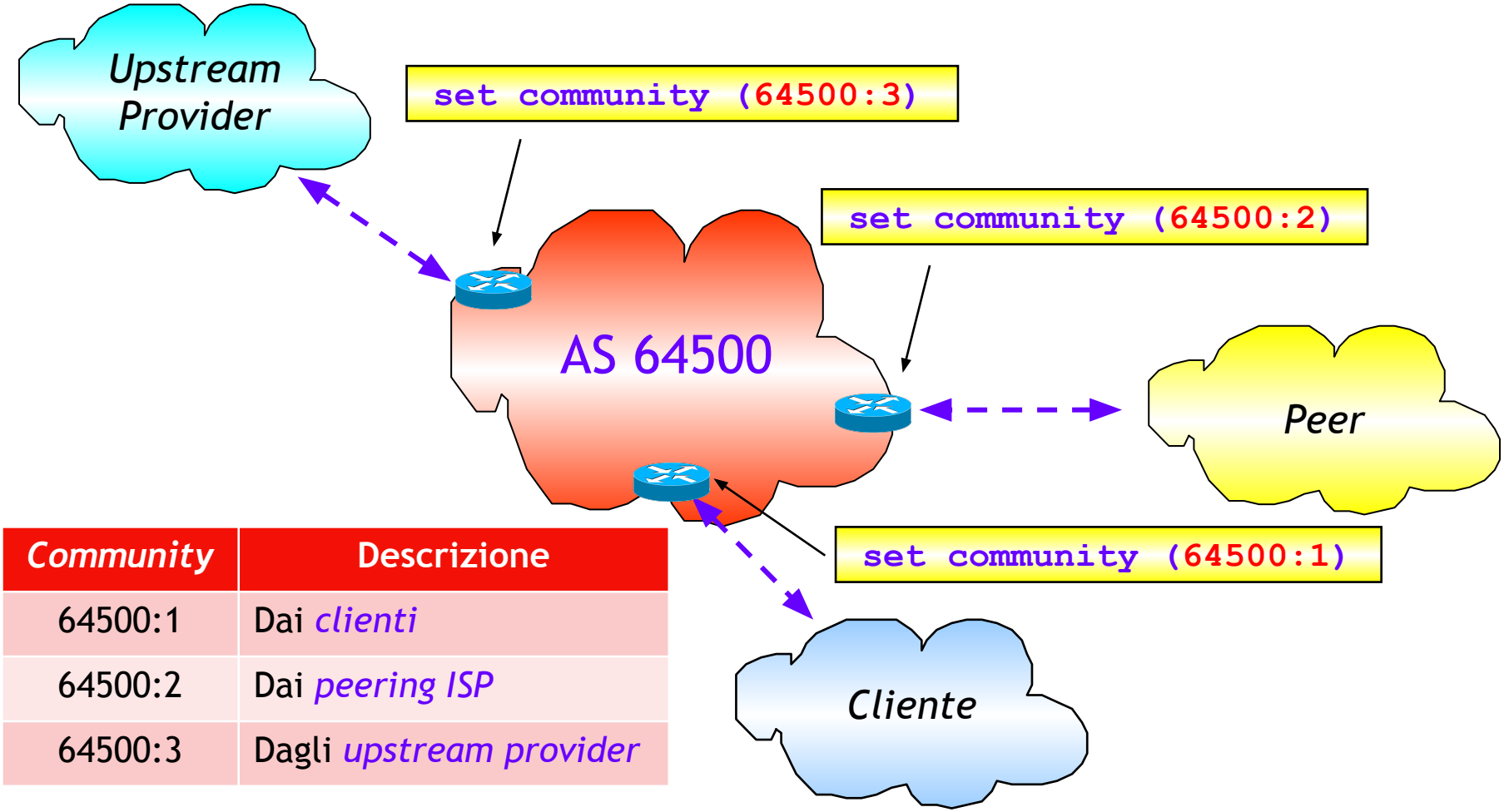
```
community-set TEST
  64501:10,
  64501:50
end-set
!
route-policy COMM-1
  if community matches-any TEST then
    pass
  endif
end-policy
!
route-policy COMM-2
  if community matches-every TEST then
    pass
  endif
end-policy
```

Pass

Drop

Esempio di utilizzo (1/2)

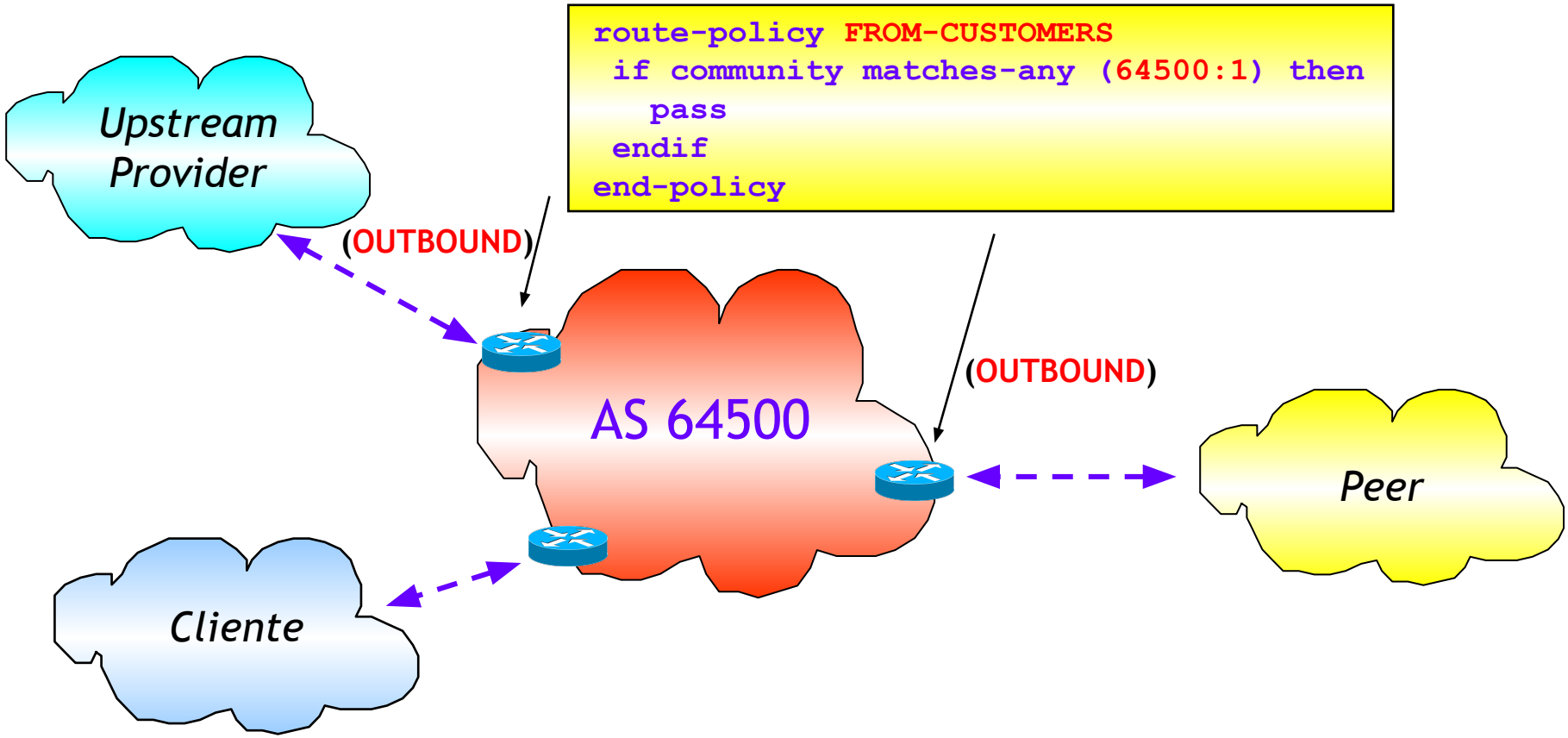
- Primo passo: associare diversi valori di *Community* ai diversi tipi di annunci



Community	Descrizione
64500:1	Dai *clienti*
64500:2	Dai *peering ISP*
64500:3	Dagli *upstream provider*

Esempio di utilizzo (2/2)

- Secondo passo: filtrare gli annunci



Filtraggio degli annunci BGP

Definizione e motivazioni
Filtri basati sui prefissi
Filtri basati sulle *Community*
Applicazione dei filtri

Applicazione dei filtri

- La modifica o l'aggiunta di un filtro si applica ai soli nuovi annunci BGP inviati/ricevuti

- Per poter applicare i filtri ai prefissi già presenti nella tabella BGP è necessario far elaborare quest'ultima dai filtri nuovi o modificati
  - Per applicare i filtri agli annunci *outbound*, occorre inviare nuovamente i messaggi BGP UPDATE ai *BGP peer*
  - Per applicare i filtri agli annunci *inbound*, occorre forzare i *BGP peer* a inviare di nuovo i messaggi BGP UPDATE

- Il meccanismo tradizionale è basato su un *reset* della sessione BGP
  - Il reset della sessione BGP implica una interruzione dell'inoltro dei pacchetti per un tempo dipendente dall'ampiezza della tabella BGP

Reset delle sessioni BGP

- IOS e IOS XE

```
router# clear ip bgp {*|indirizzo-IP-peer|nome-peer-group}
```

- IOS XR

```
RP/0/RP0/CPU0:router# clear bgp [indirizzo-IP-peer]
```

- Permette il reset di tutte le sessioni BGP (opzione *), di una singola, o di tutte le sessioni con i *BGP peer* di un *BGP peer group*
  - Causa la perdita di tutti i prefissi ricevuti dal/dai *BGP peer* e quindi una interruzione nell'inoltro del traffico
  - La nuova sessione viene ristabilita dopo 30-60 secondi dopodiché vengono scambiate di nuovo le tabelle BGP applicando così i nuovi filtri *inbound/outbound*
  - Può portare a tempi di interruzione del traffico molto lunghi quando le tabelle BGP sono molto grandi (es. *full internet routing table*)

BGP *soft reconfiguration*

- È un meccanismo che permette di applicare i filtri senza interrompere il flusso di traffico
  - È applicabile sia ai filtri in ingresso (*Inbound soft reconfiguration*) che a quelli in uscita (*Outbound soft reconfiguration*)
- *Outbound soft reconfiguration*
  - Si basa su un nuovo invio dell'intera tabella BGP ai *BGP peer*
- *Inbound soft reconfiguration*
  - Si basa sulla funzionalità di *Route Refresh*
  - È anche possibile abilitare la memorizzazione dei *BGP best-path* ricevuti dal *BGP peer* prima del passaggio nei filtri *inbound* (*Soft reconfiguration inbound*)
- NOTA: nell'IOS XR, ad ogni «**commit**» vengono applicati automaticamente i meccanismi di *BGP Soft reconfiguration*
  - Questo comportamento può essere disabilitato tramite il comando «**bgp auto-policy-soft-reset disable**»

Soft reconfiguration: applicazione

- IOS e IOS XE

```
router# clear ip bgp {*|indirizzo-IP-peer|nome-peer-group}
                                                [soft] [in|out]
```

- IOS XR

```
RP/0/RP0/CPU0:router# clear bgp ipv4 unicast [*|indirizzo-IP-peer]
                                                soft [in|out]
```

- L'opzione «**in**» applica i filtri *inbound* attivando la funzionalità di *route refresh* oppure utilizzando il *soft reconfiguration inbound*
- L'opzione «**out**» applica i filtri *outbound* facendo passare attraverso questi i *BGP best-path* inviati al/ai *BGP peer*
- NOTA: nell'IOS e IOS XE, la presenza o meno della parola chiave «**soft**» non produce alcuna differenza

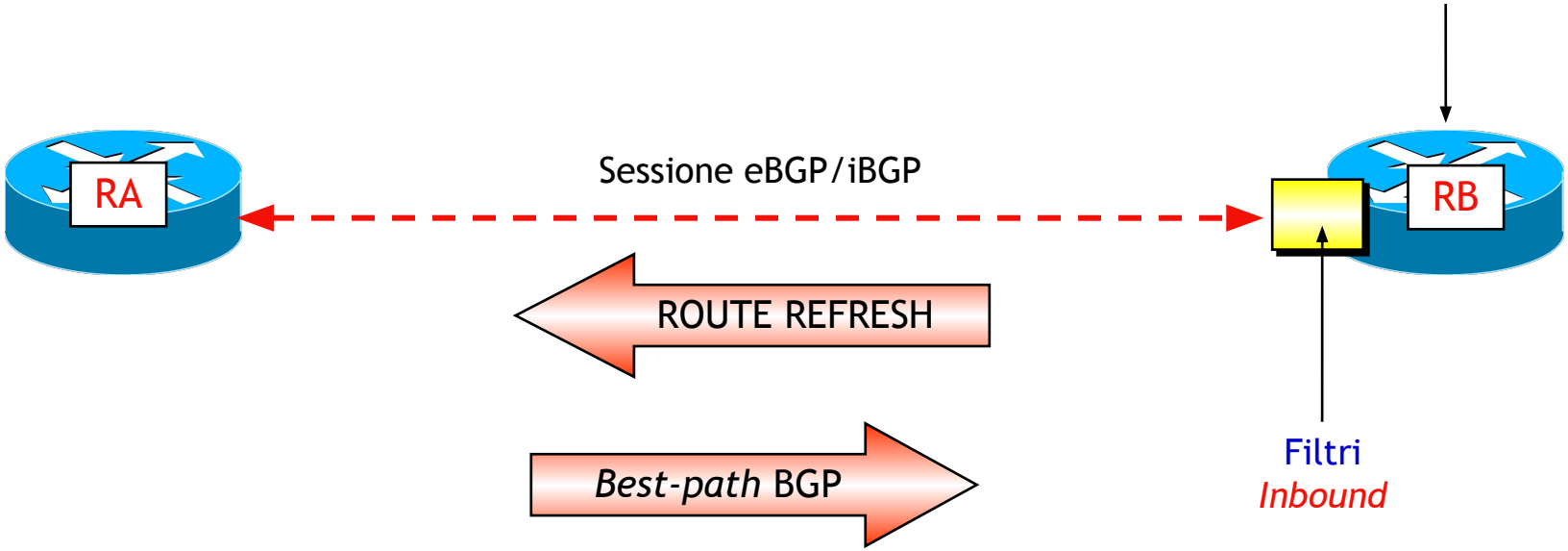
Route Refresh

- Il BGP *Route Refresh* è una *capability* opzionale del BGP che può essere negoziata attraverso il messaggio OPEN
  - *Capability code* = 2 ; *Capability Length* = 0
  - Nelle versioni più recenti delle varie versioni dell'IOS, la *BGP capability Route Refresh* viene negoziata automaticamente
    - Per verificare la negoziazione, utilizzare il comando «**show ip bgp neighbors** ...» nell'IOS/IOS XE e «**show bgp neighbors** ...» nell'IOS XR

- Funzionamento: permette ad un router di richiedere ad un *BGP peer* la ritrasmissione della sua Tabella BGP
  - La richiesta avviene attraverso l'invio al *BGP peer* del messaggio BGP ROUTE REFRESH (*Type* = 5)
  - L'invio del messaggio ROUTE REFRESH viene attivato da un comando di tipo «**clear ...**»

Route Refresh: funzionamento

```
! IOS e IOS XE
router# clear ip bgp {*|indirizzo-IP-peer|nome-peer-group} [soft] in
! IOS XR
RP/0/RP0/CPU0:router# clear bgp ipv4 unicast [*|indirizzo-IP-peer]
                                                          soft in
```



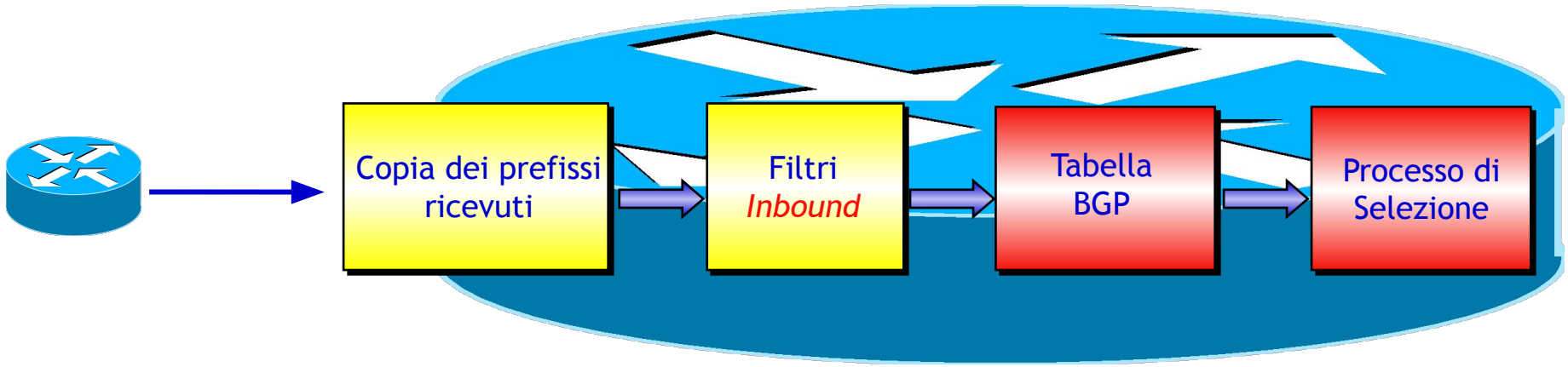
Soft reconfiguration inbound

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# neighbor indirizzo-IP-peer
                                    soft-reconfiguration inbound
```

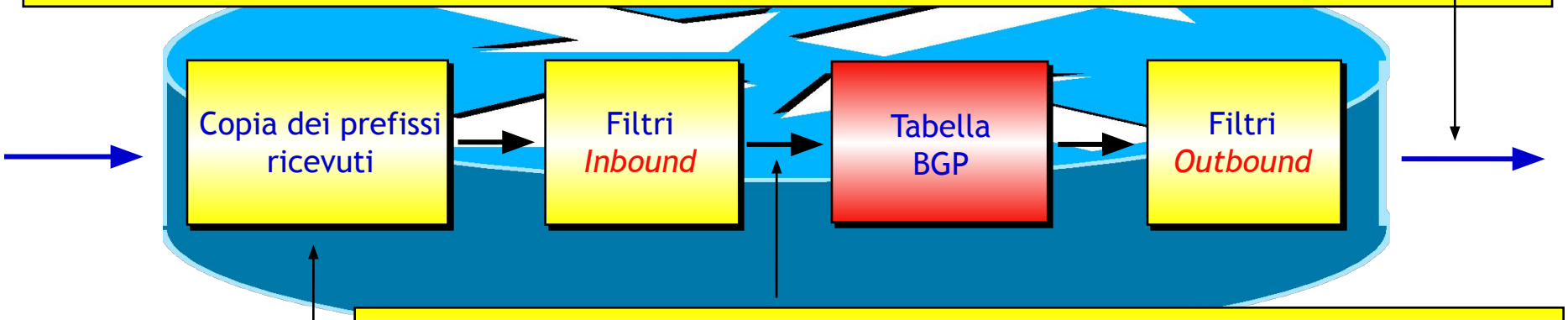
- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# soft-reconfiguration
                                        inbound [always]
```



Soft reconfiguration: monitoraggio

```
! IOS e IOS XE
router# show ip bgp neighbors IP-peer advertised-routes
! IOS XR
RP/0/RP0/CPU0:router# show bgp ipv4 unicast neighbors IP-peer advertised-routes
```




```
! IOS e IOS XE
router# show ip bgp neighbors IP-peer routes
! IOS XR
RP/0/RP0/CPU0:router# show bgp ipv4 unicast
                                        neighbors IP-peer routes
```

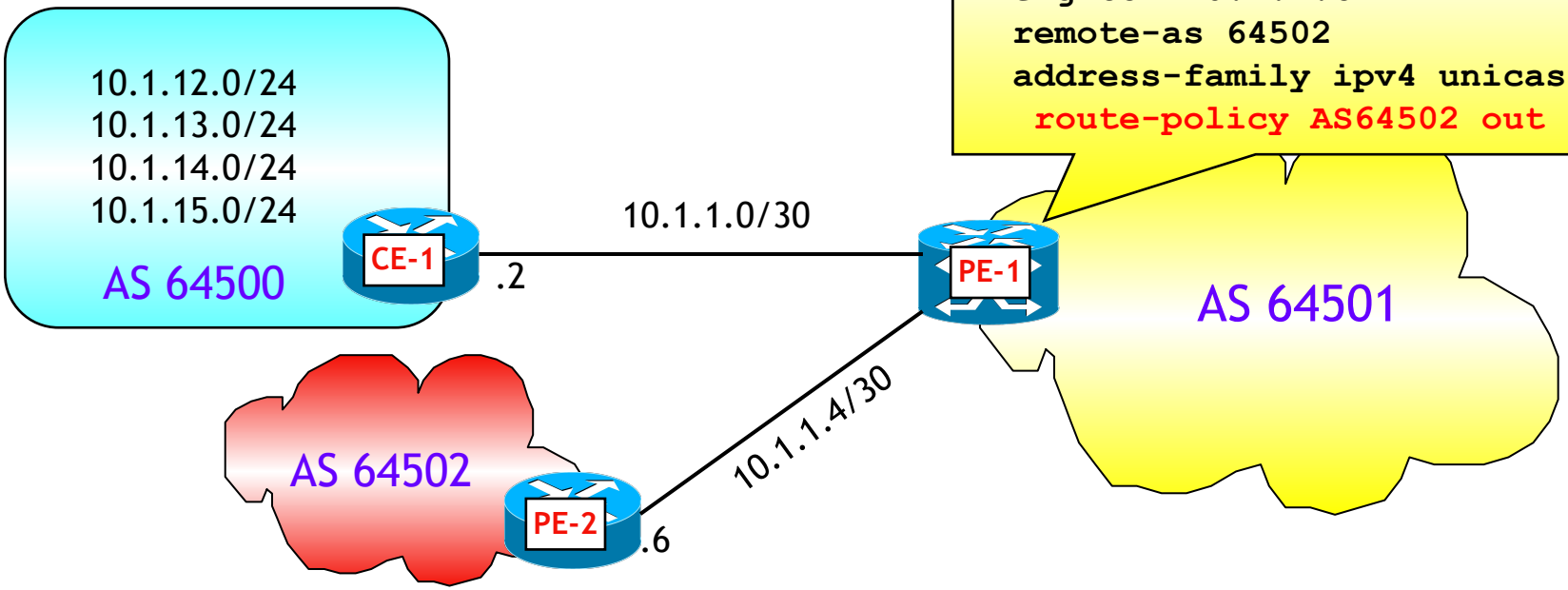
```
! IOS e IOS XE
router# show ip bgp neighbors IP-peer received-routes
! IOS XR
RP/0/RP0/CPU0:router# show bgp ipv4 unicast neighbors IP-peer received routes
```

Soft reconfiguration: esempio (1/2)

REISS ROMOLI

```
route-policy AS64500
 if destination in (10.1.12.0/23 eq 24) then
     pass
 endif
end-policy
!
route-policy AS64502
 if destination in (10.1.12.0/24) then
     pass
 endif
end-policy
```

```
router bgp 64501
 address-family ipv4 unicast
 neighbor 10.1.1.2
  remote-as 64500
  address-family ipv4 unicast
   soft-reconfiguration inbound
   route-policy AS64500 in
 neighbor 10.1.1.6
  remote-as 64502
  address-family ipv4 unicast
   route-policy AS64502 out
```



Soft reconfiguration: esempio (2/2)

- Tabella BGP di PE-1 dopo l'applicazione dei filtri *inbound*/*outbound*

```
RP/0/RP0/CPU0:PE-1# clear bgp ipv4 unicast 10.1.1.2 soft in
RP/0/RP0/CPU0:PE-1# clear bgp ipv4 unicast 10.1.1.6 soft out
RP/0/RP0/CPU0:PE-1#
RP/0/RP0/CPU0:PE-1# show bgp ipv4 unicast neighbors 10.1.1.2
                                                    received routes
. . .
Network            Next Hop            Metric LocPrf Weight Path
*> 10.1.12.0/24     10.1.1.2                 0             0 64500 i
*> 10.1.13.0/24     10.1.1.2                 0             0 64500 i
*  10.1.14.0/24     10.1.1.2                 0             0 64500 i
*  10.1.15.0/24     10.1.1.2                 0             0 64500 i

RP/0/RP0/CPU0:PE-1# show bgp ipv4 unicast neighbors 10.1.1.2 routes
. . .
*> 10.1.12.0/24     10.1.1.2                 0             0 64500 i
*> 10.1.13.0/24     10.1.1.2                 0             0 64500 i

RP/0/RP0/CPU0:PE-1# show bgp ipv4 unicast neighbors 10.1.1.6
                                                  advertised-routes
. . .
*> 10.1.12.0/24     10.1.1.2                 0             0 64500 i
```

Politiche di routing

Il processo di selezione nei router Cisco

Gestione del traffico *outbound* attraverso l'attributo *Local Preference*

Gestione del traffico *inbound* attraverso *AS_PATH prepending*

Gestione del traffico *inbound* attraverso l'attributo MED

Utilizzo dell'attributo *Community*

Validità degli annunci

- Il processo di selezione avviene solo sugli annunci considerati dal processo BGP come validi

- Un annuncio è considerato non valido se
  - Il *Next-Hop* è irraggiungibile
  - Il prefisso non è "sincronizzato" ed è abilitata la sincronizzazione
  - Il prefisso è filtrato da una politica *inbound*
  - Il prefisso è non utilizzabile a causa del meccanismo del *Route Flap Damping* (*)

(*) Il *Route Flap Damping* è un meccanismo di stabilità utile per mitigare l'effetto negativo dei *link flapping*

Sequenza di decisione modificata (1/3)

REISS ROMOLI

1. Preferire gli annunci con il più alto valore del parametro *Weight* (proprietario Cisco)
2. Preferire gli annunci con il più alto valore di *Local Preference*
3. Preferire gli annunci originati localmente dal router
   - Sono considerati annunci originati localmente quelli appresi dal processo BGP via comando «**network …**», redistribuzione (via comando «**redistribute …**») o aggregazione (via comando «**aggregate-address …**»)
   - Ordine di preferenza: «**network…**» → «**redistribute…**» → «**aggregate-address …**»
4. Preferire gli annunci con il minimo numero di elementi nell'attributo AS_PATH
   - Un AS_PATH di tipo AS_SET conta 1 elemento
     - Esempio: l'AS_PATH [**64502 64501 {64503 64500}**] ha lunghezza **3**
   - Eventuali segmenti di tipo AS_CONFED_SEQUENCE e AS_CONFED_SET non vengono contati
   - Il passo 4 può essere saltato via configurazione tramite il comando del processo BGP «**bgp bestpath as-path ignore**» (IOS, IOS XE, IOS XR)

(CONTINUA)

Sequenza di decisione modificata (2/3)

5. Preferire gli annunci con il minimo valore nell'attributo ORIGIN (0=IGP; 1=EGP; 2=INCOMPLETE)
6. Se gli annunci arrivano dallo stesso AS preferire quello con il minimo valore di MED
   - È possibile confrontare valori di MED provenienti da differenti AS abilitando il comando del processo BGP «**bgp always-compare-med**» (IOS/IOS XE) o «**bgp bestpath med always**» (IOS XR)
7. Preferire annunci provenienti da sessioni eBGP rispetto a quelli provenienti da sessioni iBGP
8. Preferire, tra gli annunci iBGP, quello che ha il *Next-Hop* più «vicino» secondo la metrica IGP
   - NOTA: nell'ipotesi che il processo di selezione non riesca a trovare un *best-path* fino a questo punto, è possibile inserire in tabella di routing più percorsi utilizzando il *BGP multipath*

(CONTINUA)

Sequenza di decisione modificata (3/3)

9. Preferire gli annunci ricevuti dal *BGP peer* con *BGP RID* più basso
   - Eccezione: se il *best-path* corrente proviene da un annuncio eBGP, non cambiare *best-path* anche se il BGP RID del nuovo *best-path* è più basso (ossia, preferire gli annunci "più vecchi")
   - È possibile comunque utilizzare sempre il BGP RID più basso configurando il comando del processo BGP «**bgp bestpath compare-routerid**» (IOS, IOS XE, IOS XR)
10. Preferire l'annuncio proveniente dal *BGP peer* con *Neighbor ID* più basso
    - Questo punto viene utilizzato solo nei (rari) casi in cui vi siano più sessioni BGP tra due router

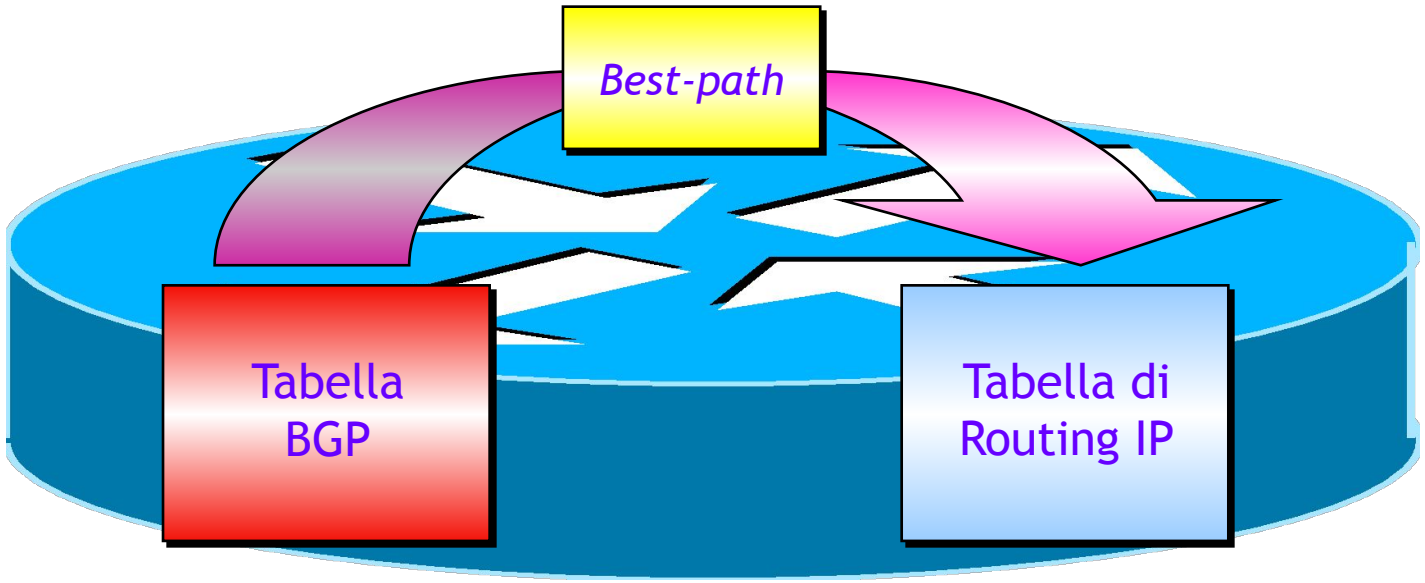
Criterio di confronto

- Quando esistono più annunci validi di un particolare prefisso, l'IOS Cisco crea una lista ordinata con ordinamento in senso inverso all'istante di ricezione
  - Il primo annuncio della lista è l'ultimo ricevuto (il più recente)
  - L'ultimo annuncio della lista è il primo ricevuto (il più datato)
  - NOTA: questo è anche l'ordine con cui gli annunci multipli appaiono nella visualizzazione della Tabella BGP
- Il processo BGP applica ogni punto del processo di selezione confrontando gli annunci sequenzialmente
  - Il primo annuncio della lista viene inizialmente scelto come *best-path*
  - Il *best-path* corrente viene quindi confrontato con l'annuncio successivo della lista
  - Il processo si ripete fino al completamento della lista
  - Il *best-path* risultante dall'ultimo confronto diventa quello definitivo

Dalla Tabella BGP alla Tabella di routing IP

REISS ROMOLI

- I percorsi risultanti come *best-path* dal processo di selezione sono candidati ad essere inseriti nella tabella di routing IP
  - L'inserimento effettivo dipende dalla Distanza Amministrativa
  - È possibile inserire più percorsi utilizzando il *BGP Multipath*



BGP Multipath

- Qualora il processo di selezione non riesca a trovare un unico *best-path* dopo i primi 8 punti è possibile installare nella Tabella di Routing IP più percorsi (*BGP Multipath*)
  - L'inserimento dei percorsi nella tabella di routing IP è sempre regolato dalla Distanza Amministrativa
- L'abilitazione del *BGP Multipath* non è automatica ma avviene attraverso un opportuno comando di configurazione
- NOTA IMPORTANTE: Anche nel caso in cui il *BGP Multipath* venga abilitato, il processo di selezione sceglie comunque un *best-path* utilizzando i punti 9 e 10
  - Ai *BGP peer* viene propagato il solo *best-path*

Abilitazione del *BGP Multipath*

REISS ROMOLI

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# maximum-paths [ibgp] numero-percorsi
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# maximum-paths {ebgp | ibgp}
                                                   numero-percorsi
```

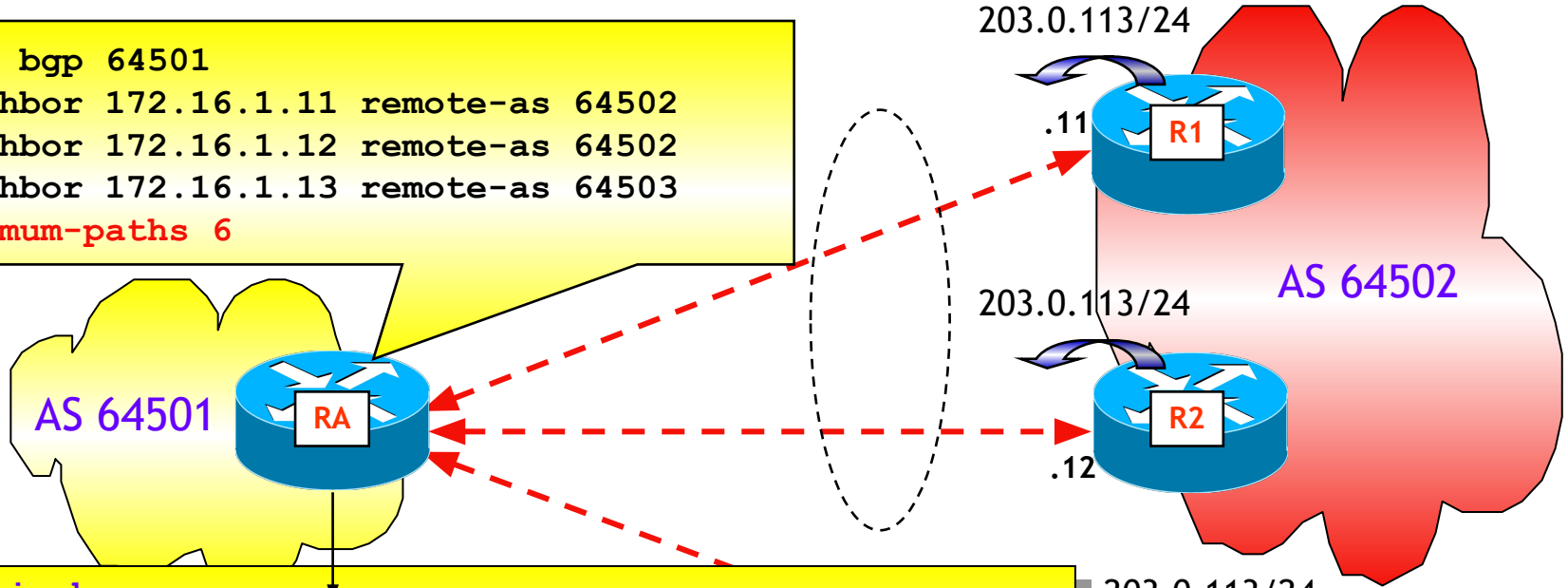
- Permette di abilitare il *BGP Multipath*
- L'opzione «**ibgp**» («**ebgp**») restringe la selezione ai soli percorsi iBGP (eBGP)
  - Nel comando IOS/IOS XE, l'assenza dell'opzione «**ibgp**» implica che la selezione è ristretta ai soli percorsi eBGP

BGP Multipath: regole di applicazione

- Percorsi eBGP
  - Vengono selezionati i soli percorsi provenienti dallo stesso AS da cui proviene il *Best-Path*
  - Qualora i percorsi selezionati superino il numero configurato, la scelta di quelli da inserire nella tabella di routing IP avviene attraverso i punti 9 e 10 del processo di selezione
- Percorsi iBGP
  - Vengono selezionati i soli percorsi che hanno attributi *Next-Hop* diversi e metrica IGP identica verso i *Next-Hop*
  - Qualora i percorsi selezionati superino il numero configurato, la scelta di quelli da inserire nella tabella di routing IP avviene attraverso i punti 9 e 10 del processo di selezione

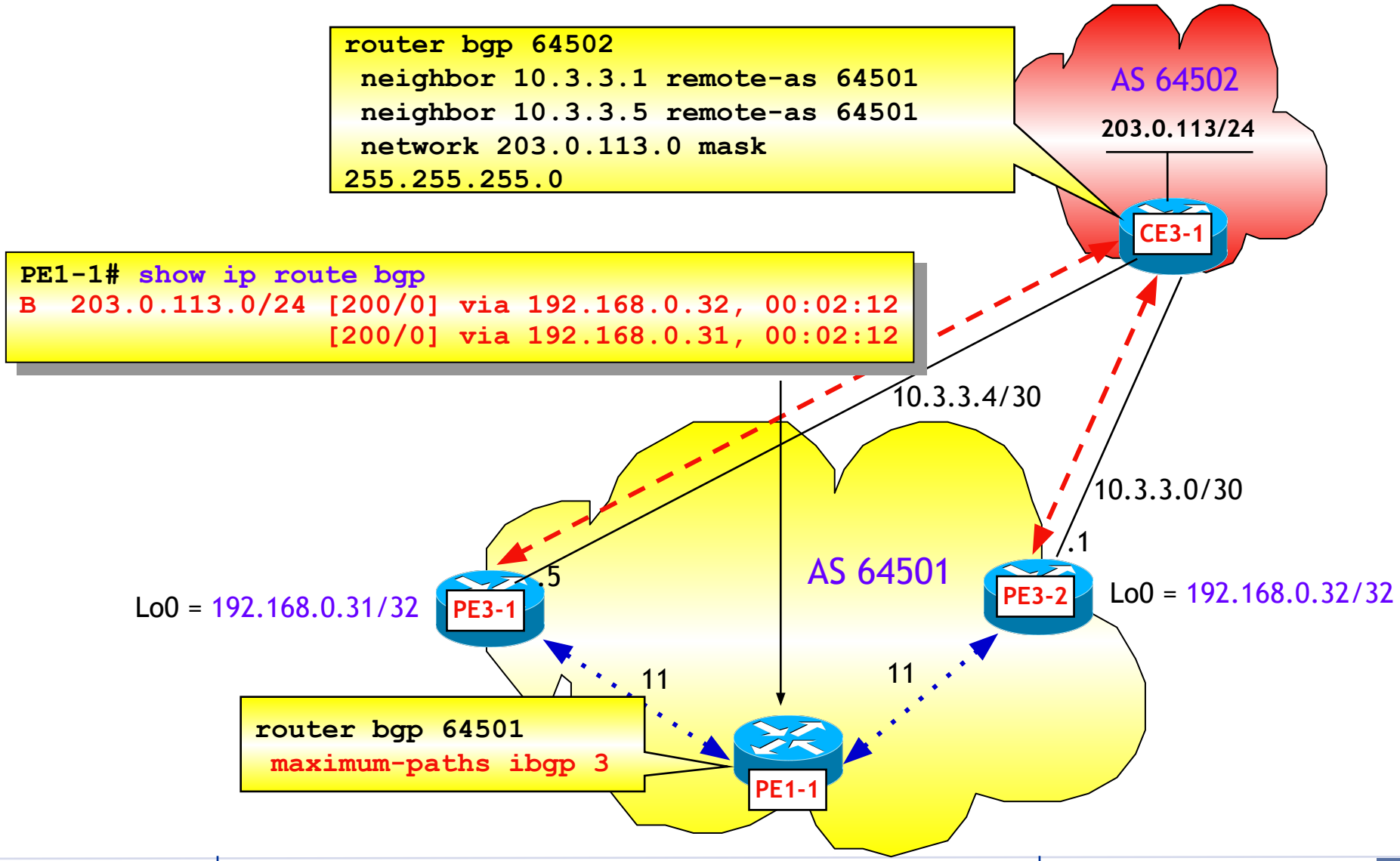
BGP Multipath: esempio 1

```
router bgp 64501
  neighbor 172.16.1.11 remote-as 64502
  neighbor 172.16.1.12 remote-as 64502
  neighbor 172.16.1.13 remote-as 64503
  maximum-paths 6
```




```
RA# show ip bgp
. . . < output omesso > . . .
   Network          Next Hop            Metric LocPrf Weight Path
*  203.0.113.0      172.16.1.13              0             0 64503 i
*                   172.16.1.12              0             0 64502 i
*>                  172.16.1.11              0             0 64502 i

RA# show ip route bgp
B    203.0.113.0/24   [20/0] via 172.16.1.12, 00:01:04
                      [20/0] via 172.16.1.11, 00:01:04
```

BGP Multipath: esempio 2



Politiche di routing

Il processo di selezione nei router Cisco

Gestione del traffico *outbound* attraverso l'attributo *Local Preference*

Gestione del traffico *inbound* attraverso *AS_PATH prepending*

Gestione del traffico *inbound* attraverso l'attributo MED

Utilizzo dell'attributo *Community*

Attributo *Local Preference* (1/2)

- L'attributo *Local Preference* permette, al pari del parametro *weight*, di scegliere il punto di uscita da un AS del traffico *outbound*
  - Può essere utilizzato in alternativa al parametro *weight*
- Differenze con il parametro *weight*
  - È un attributo standard (*well known discretionary*)
  - Viene propagato nelle sessioni iBGP ma non nelle sessioni eBGP (è quindi un valore locale all'AS)
  - Richiede configurazioni meno complesse
- Regola fondamentale: un router seleziona come punto di uscita dall'AS per un determinato prefisso quello che ha valore associato di *Local Preference* più elevato

Attributo *Local Preference* (2/2)

- Viene assegnato secondo due modalità
  - Per router: permette di stabilire un determinato valore per tutti i prefissi appresi da un router (anche localmente)
  - Via *route-map* (IOS e IOS XE) o *route-policy* (IOS XR)
- Valore di default: 100
- NOTA: ricordare che nei router Cisco il processo di selezione BGP considera al primo posto il parametro *weight* e solo immediatamente dopo l'attributo *Local Preference*

Assegnazione per router

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# bgp default local-preference valore
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# bgp default local-preference
                                                        valore
```

- Permette di variare il valore di default del *Local Preference* (che altrimenti è 100) per i prefissi appresi localmente e via sessioni eBGP

Assegnazione via *route-map* e *route-policy*

- IOS e IOS XE

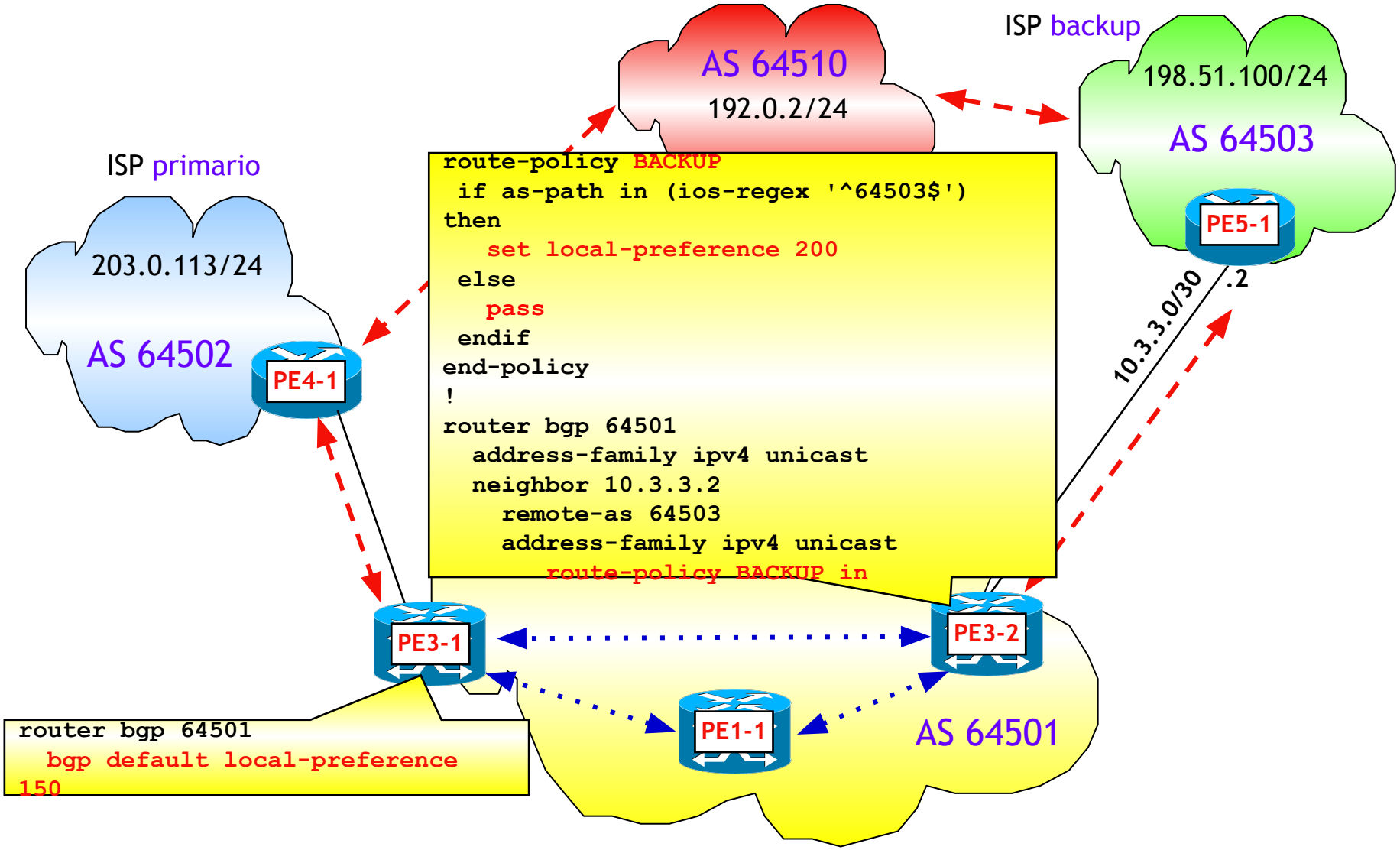
```
router(config)# route-map nome permit numero-sequenza
router(config-route-map)# condizioni
router(config-route-map)# set local-preference valore
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# route-policy nome-RP
RP/0/RP0/CPU0:router(config-rpl)# condizioni
RP/0/RP0/CPU0:router(config-rpl)# set local-preference valore
```

- Regole fondamentali
  - Le condizioni possono essere di qualsiasi tipo
  - Gli annunci che non soddisfano le condizioni vengono scartati
    - Nelle *route-map* per evitare lo scarto aggiungere una riga finale senza condizioni *match*
    - Nelle *route-policy* per evitare lo scarto aggiungere una azione «**pass**»

Esempio (1/2)


AS 64510
192.0.2/24
ISP backup
198.51.100/24
AS 64503
PE5-1
ISP primario
203.0.113/24
AS 64502
PE4-1
10.3.3.0/30
.2
route-policy BACKUP
if as-path in (ios-regex '^64503$')
then
set local-preference 200
else
pass
endif
end-policy
!
router bgp 64501
address-family ipv4 unicast
neighbor 10.3.3.2
remote-as 64503
address-family ipv4 unicast
route-policy BACKUP in
PE3-1
PE3-2
PE1-1
AS 64501
router bgp 64501
bgp default local-preference 150

Esempio (2/2)

```
RP/0/RP0/CPU0:PE1-1# show bgp
. . .
    Network          Next Hop          Metric LocPrf Weight Path
*>i203.0.113.0/24    192.168.0.31           0    150      0 64502 i
*>i198.51.100.0/24   192.168.0.32           0    200      0 64503 i
*>i192.0.2.0/24      192.168.0.31           0    150      0 64502 64510 i



RP/0/RP0/CPU0:PE1-1# show bgp 192.0.2.0/24
. . .
Paths: (1 available, best #1)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  64502 64510
    192.168.0.31 (metric 71) from 192.168.0.31 (192.168.0.31)
      Origin IGP, metric 0, localpref 150, valid, internal, best
```

Politiche di routing

Il processo di selezione nei router Cisco

Gestione del traffico *outbound* attraverso l'attributo *Local Preference*

Gestione del traffico *inbound* attraverso *AS_PATH prepending*

Gestione del traffico *inbound* attraverso l'attributo MED

Utilizzo dell'attributo *Community*

REISS ROMOLI

AS_PATH prepending

- Il meccanismo di *AS_PATH prepending* permette di forzare il punto di ingresso in un AS del traffico *inbound*
- È basato sul fatto che il processo di selezione BGP sceglie per un prefisso, a parità di parametro *weight* e attributo *Local Preference*, il percorso che attraversa il minimo numero di AS
- Funzionamento: associare ad un annuncio su una sessione eBGP un AS_PATH più lungo rispetto al default
- Viene configurato attraverso una *route-map* (IOS e IOX XE) o una *route-policy* (IOS XR)

Configurazione (1/2)

- IOS e IOS XE

```
router(config)# route-map nome permit numero-sequenza
router(config-route-map)#  condizioni
router(config-route-map)# set as-path prepend [stringa | last-as
                                                numero-volte]
```

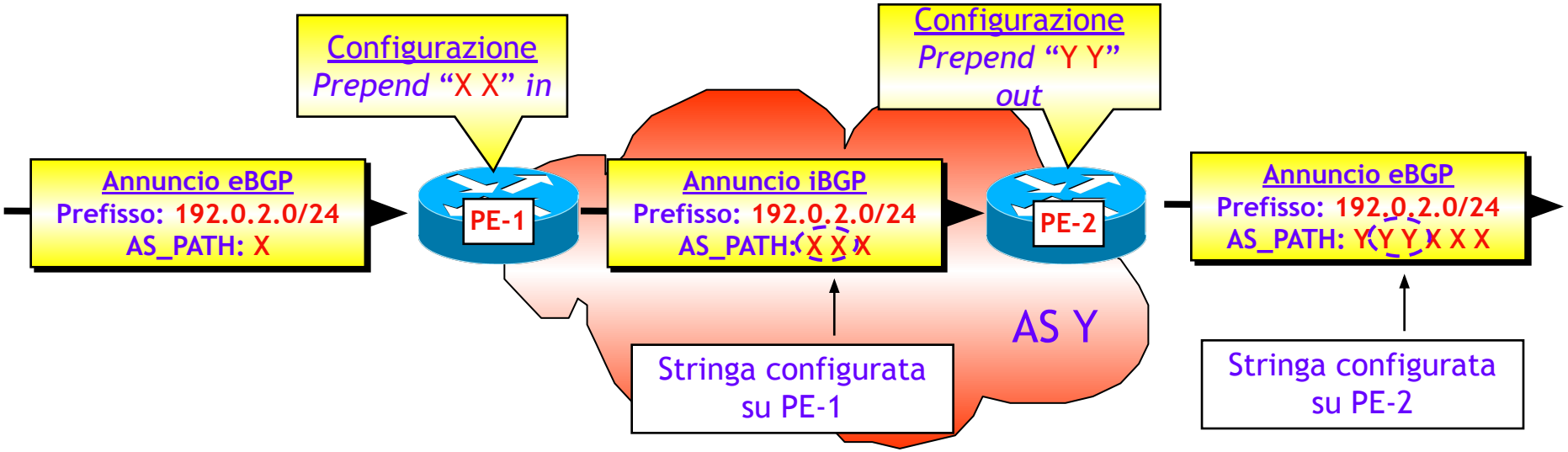
- IOS XR

```
RP/0/RP0/CPU0:router(config)# route-policy nome-RP
RP/0/RP0/CPU0:router(config-rpl)# condizioni
RP/0/RP0/CPU0:router(config-rpl)# prepend_as-path {AS | most-recent}
                                                  [numero-volte]
```

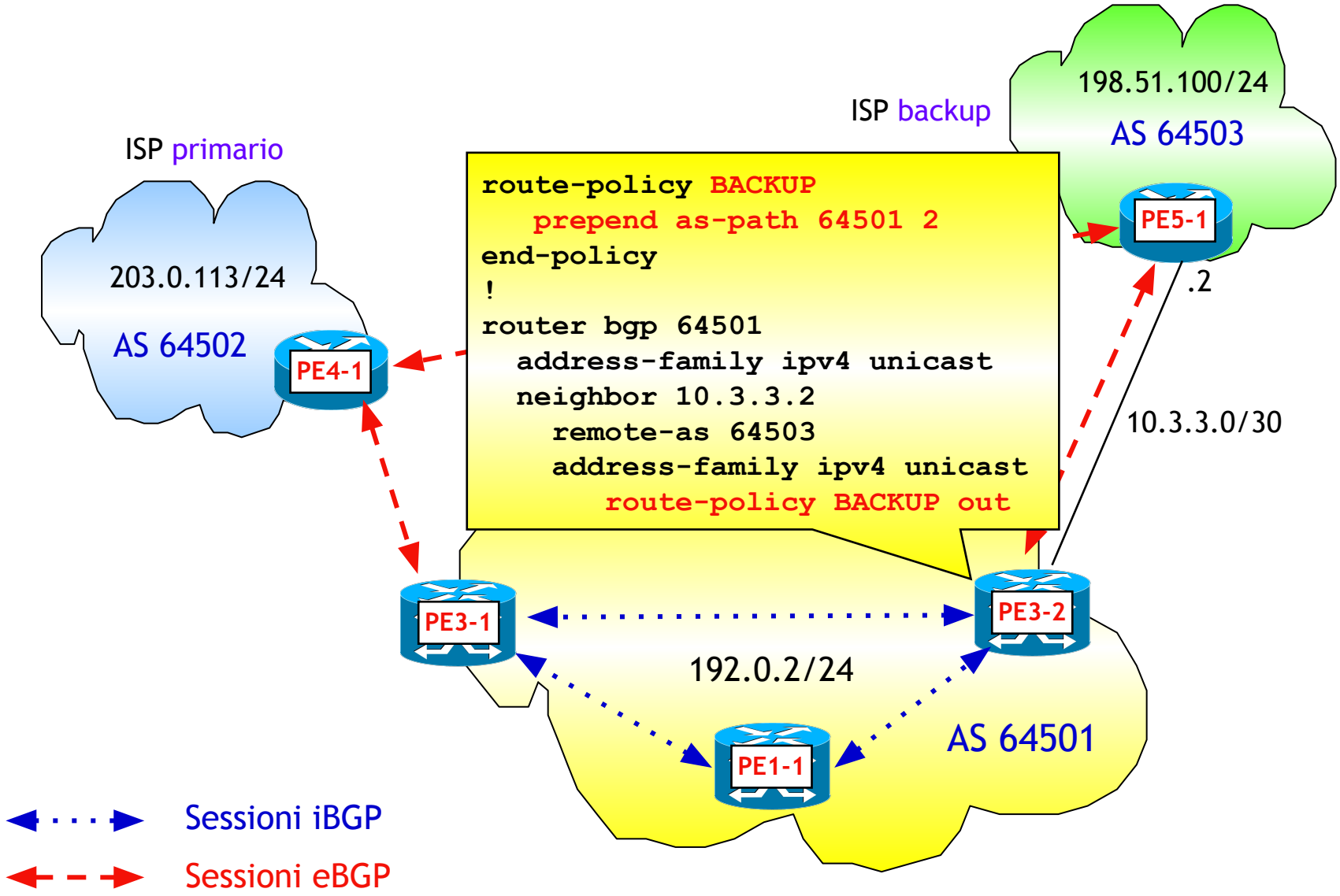
- Regole fondamentali
  - Le *route-map* e *route-policy* possono essere applicate ad annunci entranti o uscenti e hanno effetto solo quando applicate a sessioni eBGP
  - Gli annunci che non soddisfano le condizioni vengono scartati

Configurazione (2/2)

- Il router aggiunge alla lista AS_PATH una stringa pari a quella configurata nelle azioni «**set as-path prepend ...**» o «**prepend as-path ...**»
  - Una volta aggiunta la stringa configurata, l'elaborazione dell'AS_PATH avviene secondo le regole usuali



Esempio (1/2)


ISP primario
203.0.113/24
AS 64502
PE4-1
ISP backup
198.51.100/24
AS 64503
PE5-1
.2
10.3.3.0/30
route-policy BACKUP
prepend as-path 64501 2
end-policy
!
router bgp 64501
address-family ipv4 unicast
neighbor 10.3.3.2
remote-as 64503
address-family ipv4 unicast
route-policy BACKUP out
PE3-1
PE3-2
192.0.2/24
PE1-1
AS 64501
Sessioni iBGP
Sessioni eBGP

Esempio (2/2)

Best-path per il prefisso 192.0.2.0/24 (AS_PATH più corto)

```
RP/0/RP0/CPU0:PE5-1# show bgp
. . .
    Network          Next Hop  Metric LocPrf Weight   Path
*> 198.51.100.0/24   0.0.0.0                 32768   i
*  192.0.2.0/24      10.3.3.1         0          0   64501 64501 64501 i
*> 192.0.2.0/24      10.3.3.13        0          0   64502 64501 i
*> 203.0.113.0/20    10.3.3.13        0          0   64502 i
```

- NOTA: la visualizzazione del risultato dell'applicazione del meccanismo di *AS_PATH prepending* può essere effettuata solo sul router che riceve l'annuncio

Quanti AS aggiungere?

- Scenario *primario/backup*
  - Utilizzare un numero sufficientemente elevato di AS sugli annunci inviati sul collegamento di *backup* per assicurarsi che la lista AS_PATH degli annunci inviati sul collegamento primario sia più breve
  - Attenzione: un numero troppo elevato di AS fa sprecare memoria sui router che ricevono gli annunci
  - Regola empirica: provare con un numero crescente di AS fino ad ottenere che sul collegamento di *backup* non passi traffico *inbound* e quindi aggiungere qualche AS in più (2-3) per sicurezza

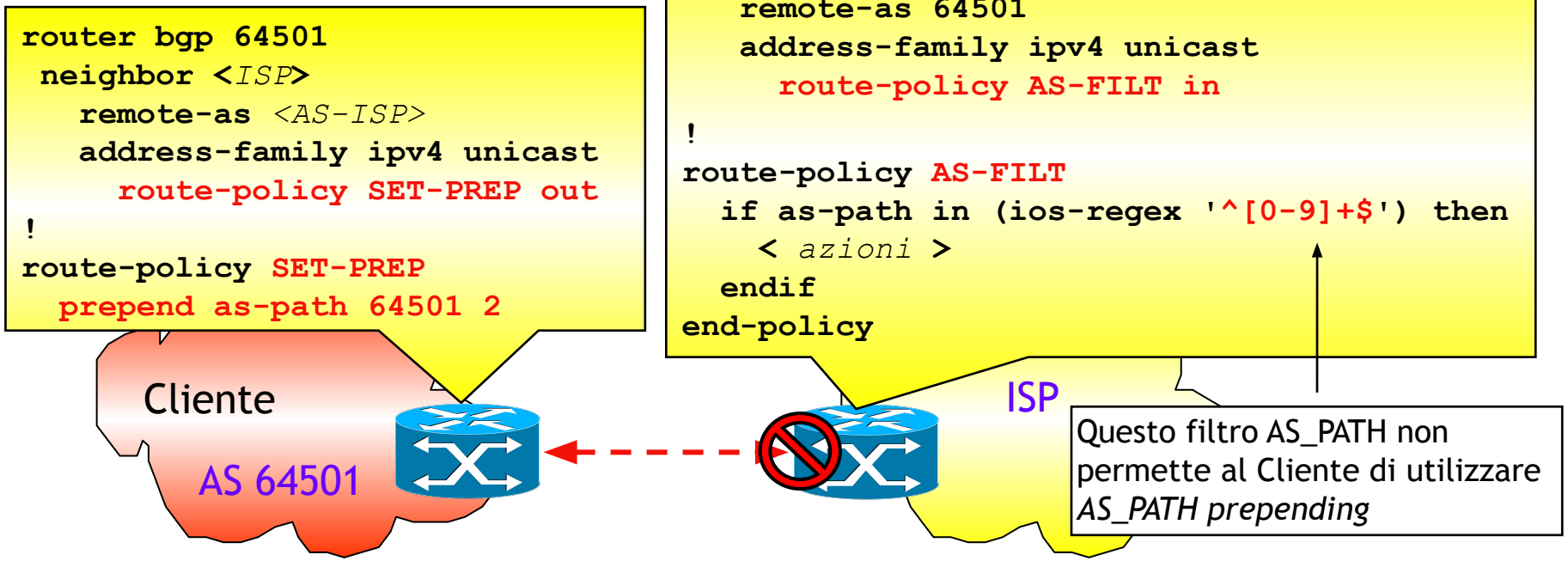
- Scenario *load balancing*
  - Poiché la distribuzione del traffico *inbound* non è nota a priori, è necessario andare per tentativi monitorando continuamente il traffico *inbound* sui collegamenti ed incrementando il numero di AS aggiunti quando necessario

AS_PATH prepending e filtri AS_PATH

- Gli ISP solitamente utilizzano filtri AS_PATH per controllare gli annunci BGP ricevuti dai Clienti
- Problema: il filtro AS_PATH configurato lato ISP potrebbe impedire di accettare annunci BGP che hanno utilizzato il meccanismo di *AS_PATH prepending*

```
router bgp 64501
 neighbor <ISP>
   remote-as <AS-ISP>
   address-family ipv4 unicast
     route-policy SET-PREP out
!
route-policy SET-PREP
  prepend as-path 64501 2
```

```
router bgp <AS-ISP>
 neighbor <Cliente>
   remote-as 64501
   address-family ipv4 unicast
     route-policy AS-FILT in
!
route-policy AS-FILT
  if as-path in (ios-regex '^[0-9]+$') then
    < azioni >
  endif
end-policy
```



Politiche di routing

Il processo di selezione nei router Cisco

Gestione del traffico *outbound* attraverso l'attributo *Local Preference*

Gestione del traffico *inbound* attraverso *AS_PATH prepending*

Gestione del traffico *inbound* attraverso l'attributo MED

Utilizzo dell'attributo *Community*

Attributo MED

- È un attributo *Optional Non Transitive* che specifica una metrica utilizzabile per forzare il punto di ingresso del traffico *inbound* in un AS
- Regole fondamentali nell'utilizzo del MED
  - Il valore MED viene assegnato su base configurazione (default = 0) da router che emettono un annuncio
  - Un router che riceve un annuncio con un determinato valore di MED non propaga il valore all'esterno dell'AS
    - Quando l'annuncio viene propagato ad altri AS il MED viene posto al valore 0 (a meno che non venga ridefinito su base configurazione)
  - Di default un router confronta i valori di MED solo di annunci provenienti dallo stesso AS
    - Questo comportamento può essere variato su base configurazione utilizzando il comando «**bgp always-compare-med**» (IOS/IOS XE) o «**bgp bestpath med always**» (IOS XR)
  - Un router seleziona come punto di ingresso nell'AS per un determinato prefisso, quello che ha valore associato di MED più basso

Attributo MED e redistribuzione

- Nella redistribuzione da qualsiasi protocollo di routing in BGP, l'assegnazione del MED può essere effettuata in tre modi:

  1. Non assegnare manualmente alcun valore di MED: di default viene assegnato un MED pari al valore di *costo totale verso il prefisso redistribuito, presente nella Tabella di Routing IP*

  2. Assegnare un valore di MED utilizzando l'opzione «**metric**» nel comando «**redistribute …**»

  3. Utilizzare il comando «**default-metric** *metrica*», che permette di definire manualmente un valore di MED *per tutti i prefissi appresi via redistribuzione*

- NOTA: l'opzione 2 ha sempre precedenza sull'opzione 3

Configurazione

- IOS e IOS XE

```
router(config)# route-map nome permit numero-sequenza
router(config-route-map)#  condizioni
router(config-route-map)# set metric valore-MED
```

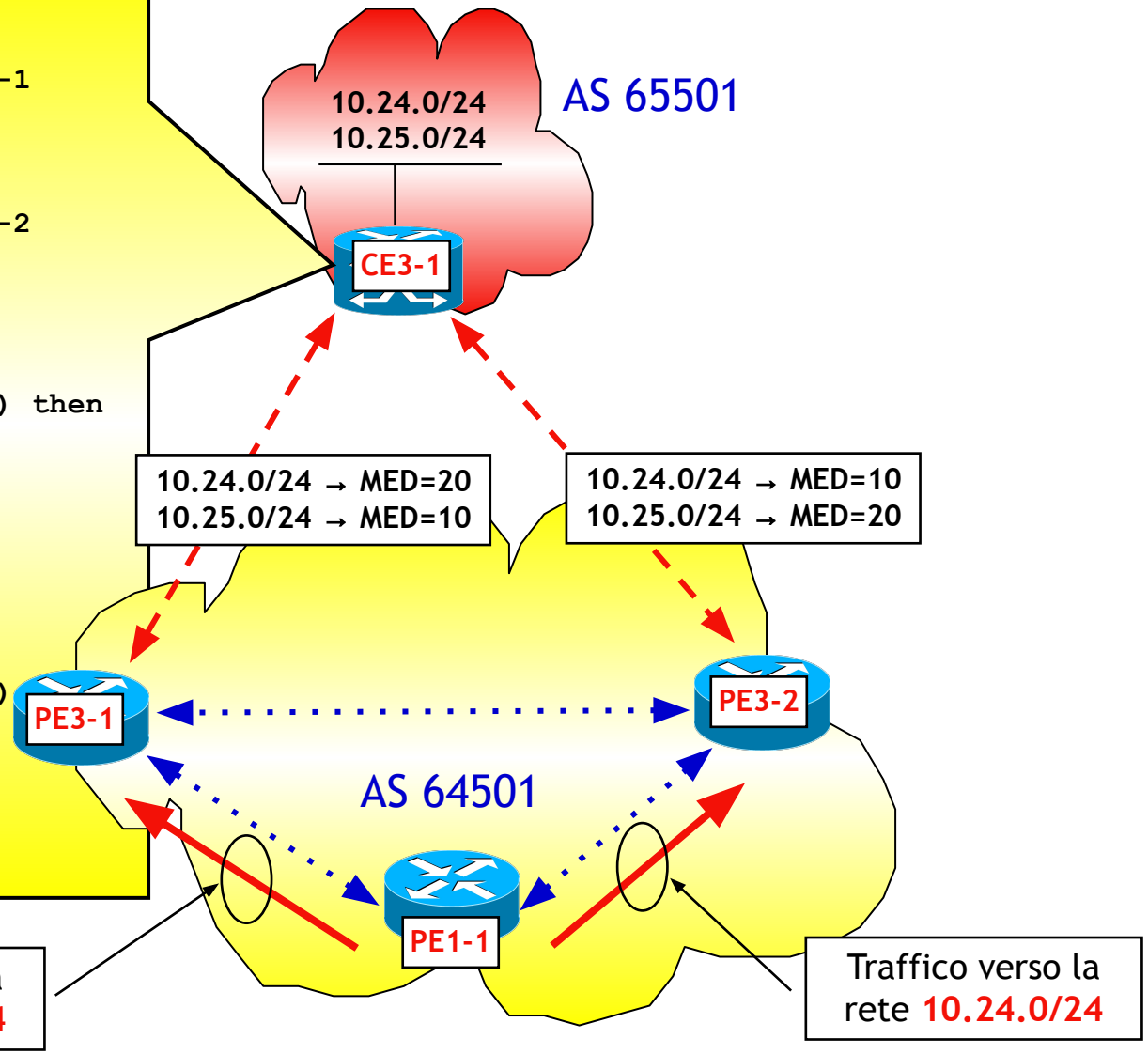
- IOS XR

```
RP/0/RP0/CPU0:router(config)# route-policy nome-RP
RP/0/RP0/CPU0:router(config-rpl)# condizioni
RP/0/RP0/CPU0:router(config-rpl)# set med {[+ | -] valore-MED |
                                           igp-cost | max-reachable}
```

- Regole fondamentali
  - Le *route-map* e *route-policy* possono essere applicate ad annunci entranti o uscenti e hanno effetto solo quando applicate a sessioni eBGP
  - Gli annunci che non soddisfano le condizioni vengono scartati

Esempio (1/2)

```
router bgp 65501
 ...
 neighbor 10.3.3.1
   description SESSIONE VERSO PE3-1
   address-family ipv4 unicast
     route-policy SET-MED1 out
 neighbor 10.3.3.5
   description SESSIONE VERSO PE3-2
   address-family ipv4 unicast
     route-policy SET-MED2 out
!
route-policy SET-MED1 permit 10
  if destination in (10.24.0.0/24) then
    set med 20
  else
    set med 10
  endif
end-policy
!
route-policy SET-MED2 permit 10
  if destination in (10.25.0.0/24)
    set med 20
  else
    set med 10
  endif
end-policy
```



Esempio (2/2)

```
RP/0/RP0/CPU0:PE1-1# show bgp
. . .

   Network          Next Hop            Metric LocPrf Weight Path
*>i10.24.0.0/24     192.168.0.32             10             0 65501 i
* i                 192.168.0.31             20             0 65501 i
* i10.25.0.0/24     192.168.0.32             20             0 65501 i
*>i                 192.168.0.31             10             0 65501 i


RP/0/RP0/CPU0:PE1-1# show bgp 10.24.0.0/24
. . .

Paths: (2 available, best #2)
  Not advertised to any peer
  Path #1: Received by speaker 0
  Not advertised to any peer
  65501
    192.168.0.31 (metric 71) from 192.168.0.31 (192.168.0.31)
      Origin IGP, metric 20, localpref 100, valid, internal
  Path #2: Received by speaker 0
  Not advertised to any peer
  65501
    192.168.0.32 (metric 71) from 192.168.0.32 (192.168.0.32)
      Origin IGP, metric 10, localpref 100, valid, internal, best
```

MED

Politiche di routing

Il processo di selezione nei router Cisco

Gestione del traffico *outbound* attraverso l'attributo *Local Preference*

Gestione del traffico *inbound* attraverso *AS_PATH prepending*

Gestione del traffico *inbound* attraverso l'attributo MED

Utilizzo dell'attributo *Community*

REISS ROMOLI

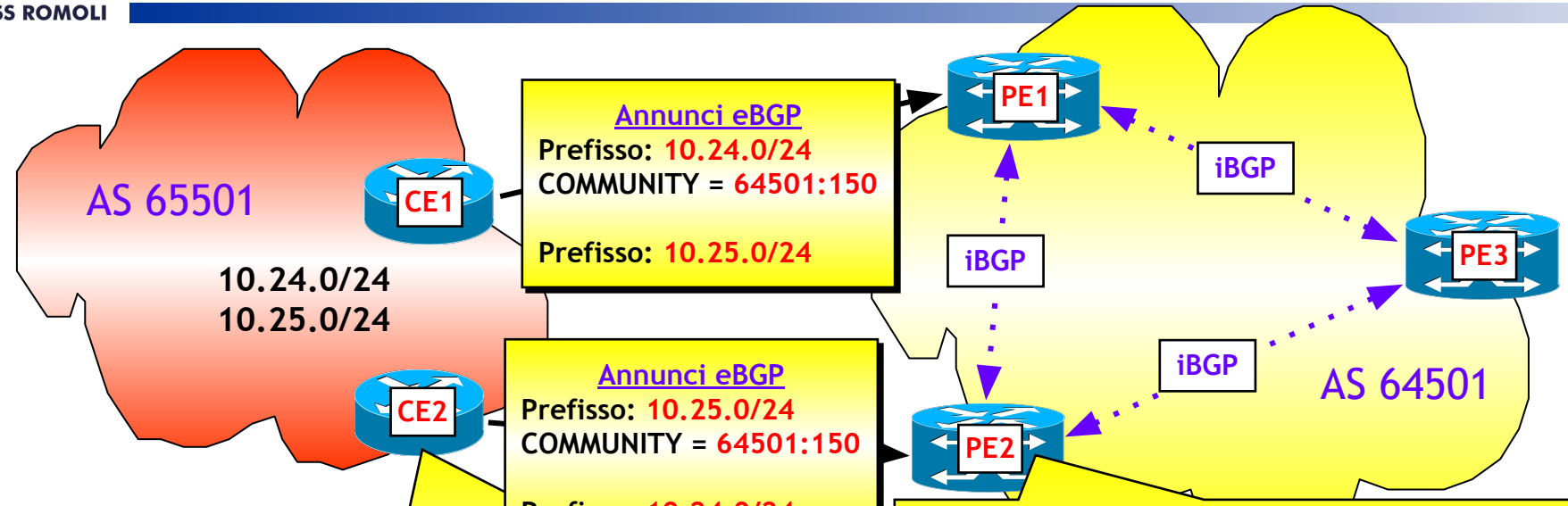
Attributo *Community*: utilizzo

- Definire particolari politiche di filtraggio dei prefissi
  - Esempio: non annunciare i prefissi che hanno associato un determinato valore di *Community*
- Definire particolari politiche di routing per gruppi di prefissi omogenei
  - Esempio: assegnare diversi valori di *Local Preference* sulla base del valore dell'attributo *Community*
- Definire politiche di Qualità del Servizio sulla base di SLA definiti tra Clienti e ISP
  - Esempio: differenziare i livelli di Qualità del Servizio offerti ai Clienti (es. *Gold*, *Silver*, *Bronze*, *Best Effort*) sulla base del valore dell'attributo *Community*

Esempio di utilizzo

Scopo	Community
Assegna *Local Preference* = 150	64501:150
Assegna *Local Preference* = 200	64501:200
Effettua un *AS_PATH prepending* di un AS	64501:11
Effettua un *AS_PATH prepending* di due AS	64501:22
Assegna *IP Precedence* = 0	64501:0
Assegna *IP Precedence* = 5	64501:5

Esempio



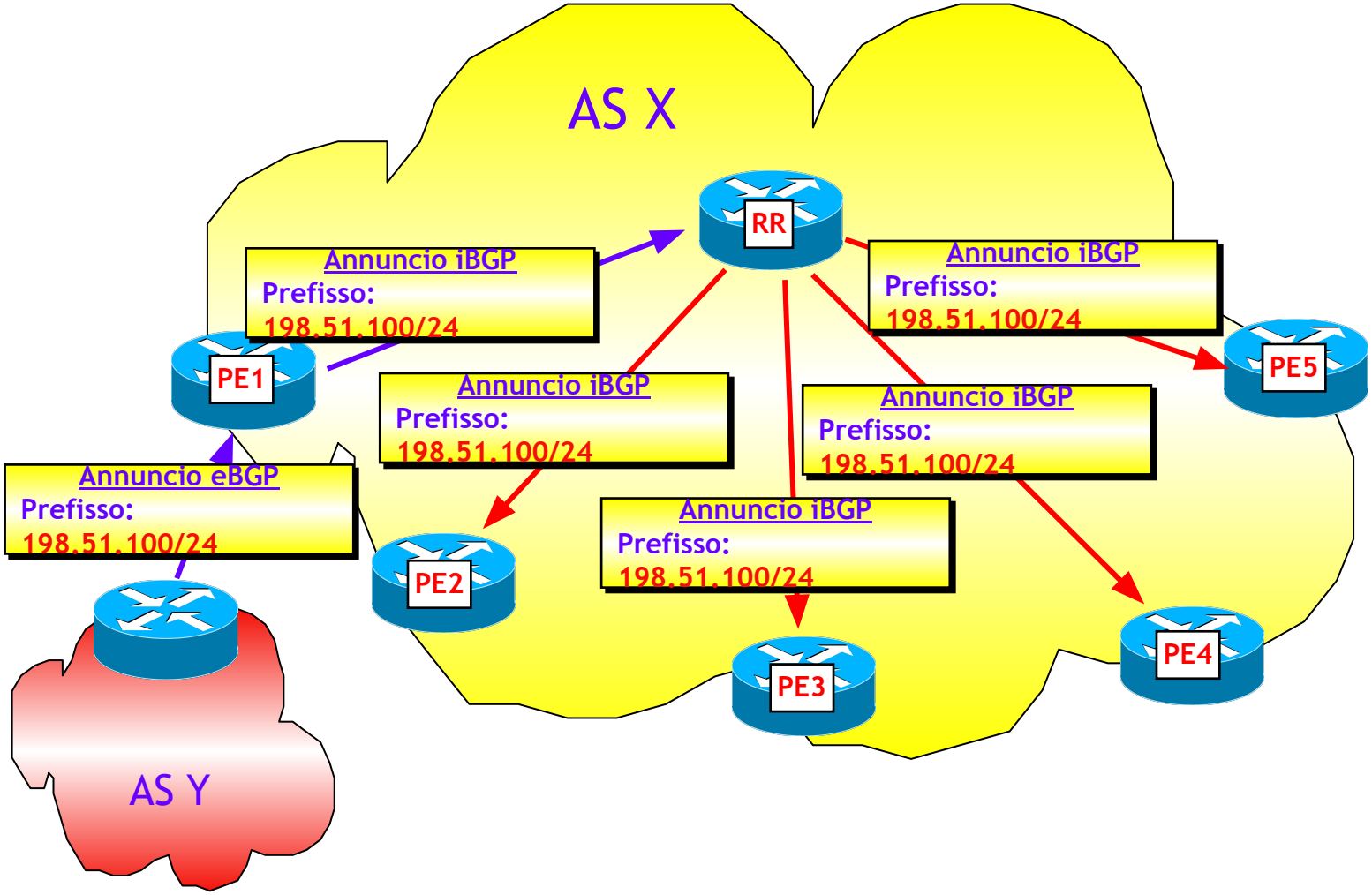

```
router bgp 65501
 neighbor 10.3.3.1 remote-as 64501
 neighbor 10.3.3.1 route-map SETLP out
 neighbor 10.3.3.1 send-community
!
ip prefix-list P25 permit 10.25.0.0/24
!
route-map SETLP permit 10
 match ip address prefix-list P25
 set community (64501:150)
route-map SETLP permit 20
```

```
community-set PRIMARY
 64501:150
end-set
!
route-policy SETLP
  if community matches-any PRIMARY then
    set local-preference 150
  else
    pass
  endif
end-policy
!
router bgp 64501
  neighbor 10.3.3.2 remote-as 65501
    address-family ipv4 unicast
      route-policy SETLP in
```

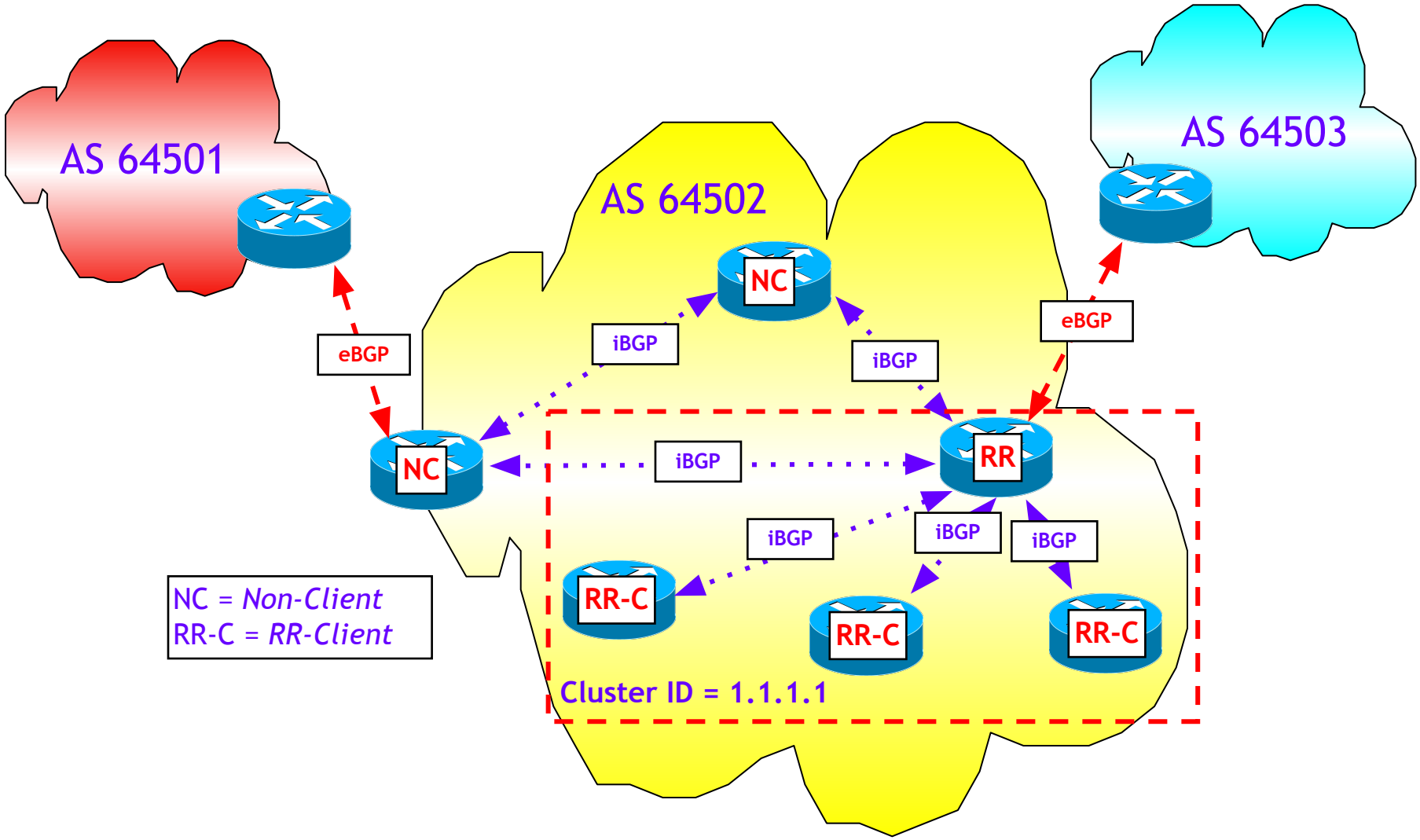
Il BGP nelle reti dei *Service Provider*

Route Reflector (cenni)

Il concetto chiave ...



Nomenclatura



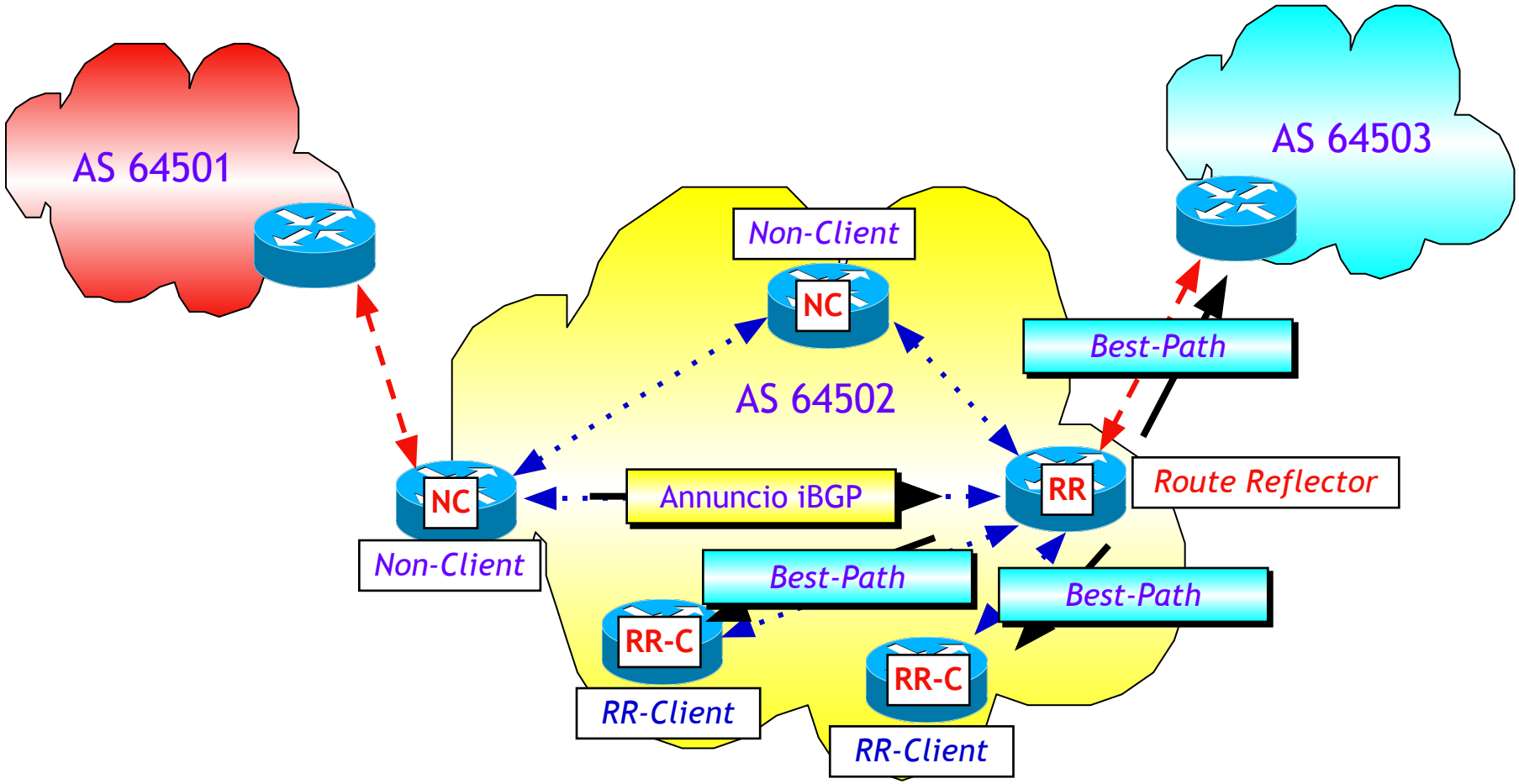
REISS ROMOLI

Regole fondamentali (1/3)

- Regola N. 1: un RR riflette solo il percorso migliore
- Regola N. 2: un RR riflette sempre gli annunci agli *eBGP peer*
- Regola N. 3: un *RR-client* segue la classica regola dell'*iBGP split-horizon*
- Regola N. 4: un RR propaga gli annunci ricevuti sulle sessioni eBGP a tutte le altre sessioni BGP
- La propagazione degli annunci a *iBGP peer* segue delle regole (vedi prossime regole 5 e 6) dipendenti dalla provenienza dell'annuncio

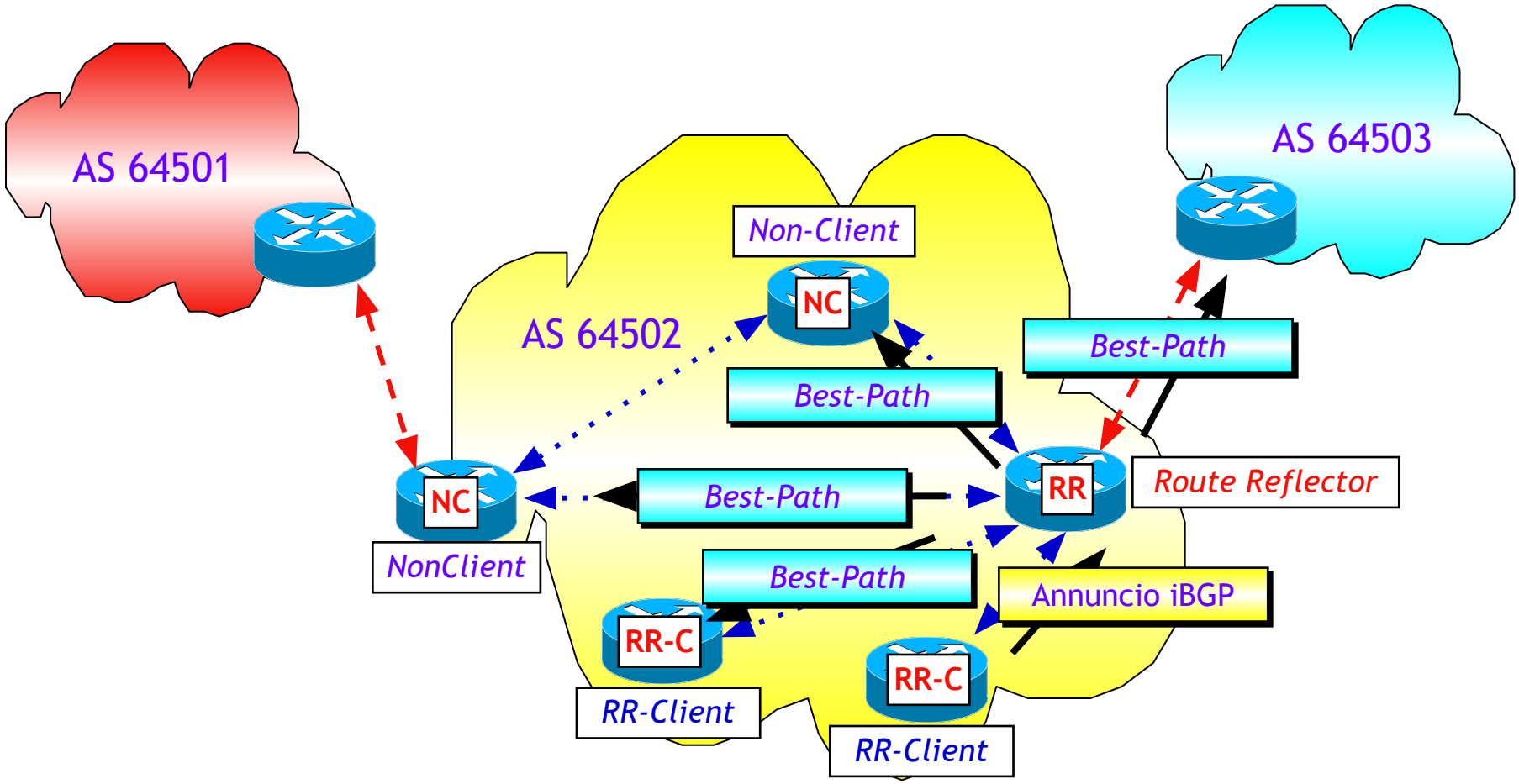
Regole fondamentali (2/3)

- Regola N. 5: un RR propaga gli annunci ricevuti sulle sessioni iBGP con *Non-client* a tutti gli *RR-client* e a tutte le sessioni eBGP



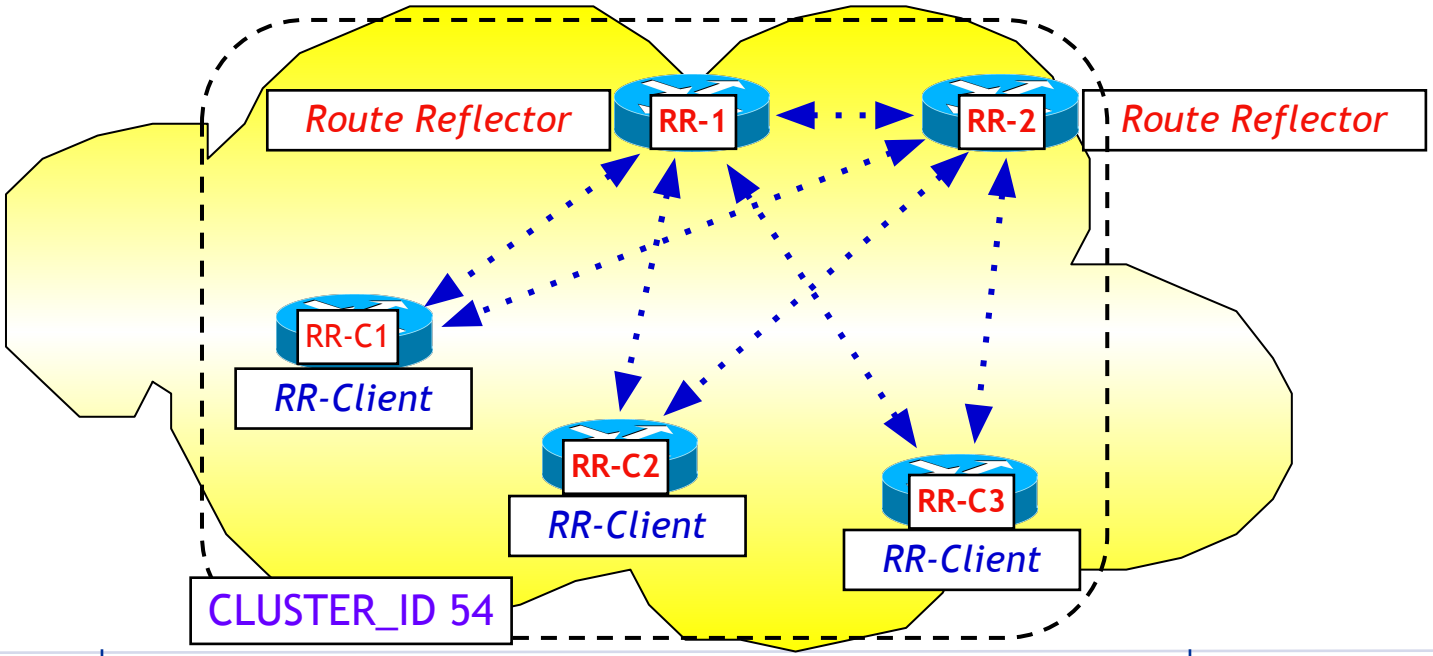
Regole fondamentali (3/3)

- Regola N. 6: un RR propaga gli annunci ricevuti su una sessione iBGP con un *RR-client* a tutti gli altri *RR-client* e a tutti i *Non-client* (iBGP ed eBGP)



Configurazioni *fault-tolerant* di RR

- Un singolo RR costituisce un *single point of failure* per cui nelle reti vengono implementate configurazioni *fault-tolerant* di RR
- Un gruppo di RR *fault-tolerant* e i loro *RR-client* formano un *cluster*
  - Un *cluster* è identificato da un valore di 4 *byte* detto CLUSTER_ID
  - Un *cluster* al minimo è formato da un RR e dai suoi *RR-client*
  - Di default ciascun RR e i suoi *RR-client* formano un *cluster* il cui CLUSTER_ID è il BGP RID del RR



Prevenzione dei *loop*

- Nelle configurazioni *fault-tolerant* di RR è possibile la formazione di *routing information loop* e *forwarding loop*
- La prevenzione dei loop in un *cluster* con più RR si basa su due nuovi attributi BGP
  - ORIGINATOR_ID
  - CLUSTER_LIST
- Nei router Cisco l'introduzione di questi due attributi ha comportato una variazione del processo di selezione
  - Nel punto 9, l'attributo ORIGINATOR_ID, se presente, sostituisce nella scelta il BGP RID
  - Il punto 10 diventa il punto 11
  - Il nuovo punto 10 diventa il seguente: "scegliere l'annuncio con il CLUSTER_LIST più breve"

Configurazione dei RR

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# bgp cluster-id cluster-ID
router(config-router)# neighbor indirizzo-IP-peer
                        route-reflector-client
```

- IOS XR

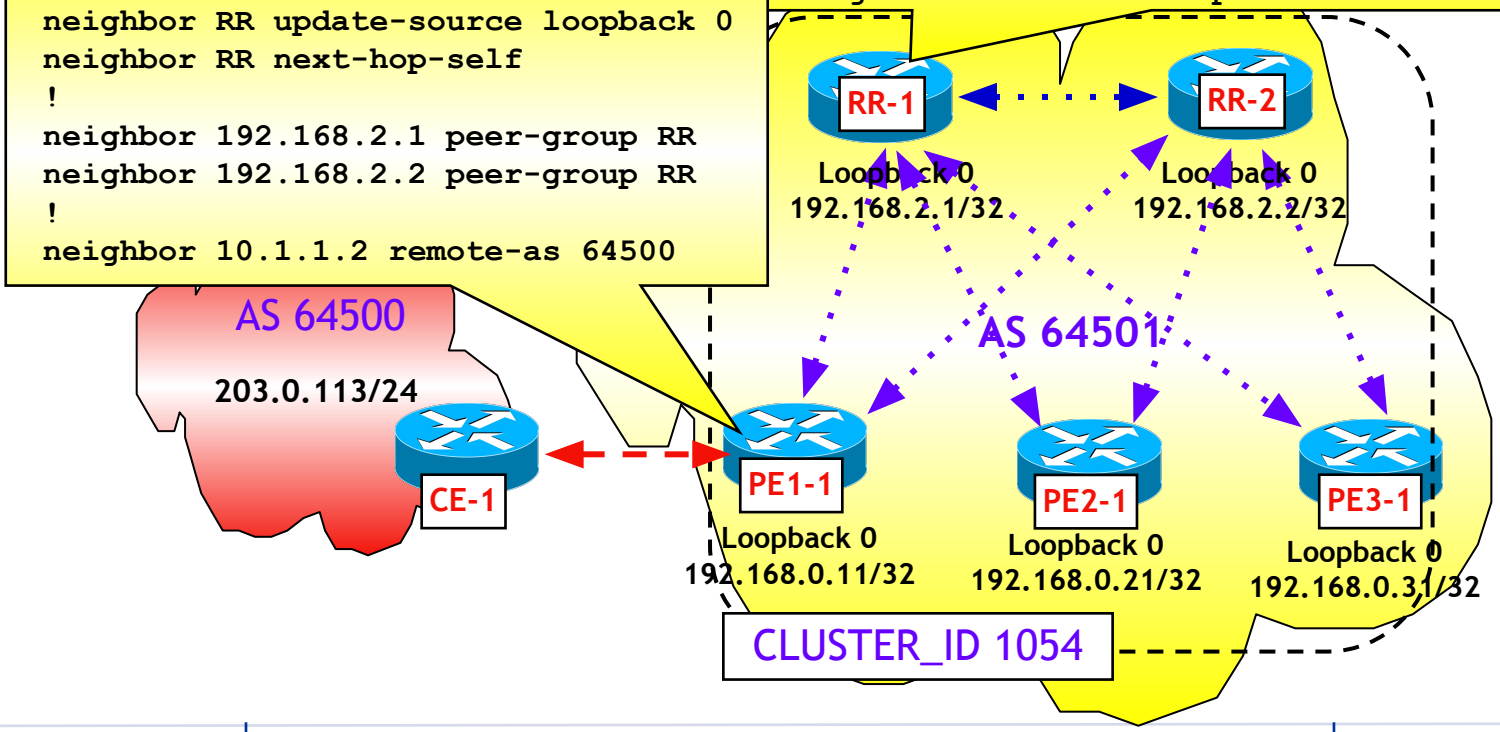
```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# bgp cluster-id cluster-ID
RP/0/RP0/CPU0:router(config-bgp)# neighbor indirizzo-IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as AS-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-reflector-client
```

- NOTE sul valore di CLUSTER_ID
  - Il valore di default del CLUSTER_ID coincide con il proprio BGP RID
  - È richiesto solo in presenza di configurazioni *fault-tolerant* di RR

Esempio (1/3)

```
router bgp 64501
 bgp cluster-id 1054
 neighbor RR-CLIENT peer-group
 neighbor RR-CLIENT remote-as 64501
 neighbor RR-CLIENT update-source loopback 0
 neighbor RR-CLIENT route-reflector-client
 neighbor 192.168.0.11 peer-group RR-CLIENT
 neighbor 192.168.0.21 peer-group RR-CLIENT
 neighbor 192.168.0.31 peer-group RR-CLIENT
 !
 neighbor 192.168.2.2 remote-as 64501
 neighbor 192.168.2.2 update-source loopback0
```

```
router bgp 64501
 neighbor RR peer-group
 neighbor RR remote-as 64501
 neighbor RR update-source loopback 0
 neighbor RR next-hop-self
 !
 neighbor 192.168.2.1 peer-group RR
 neighbor 192.168.2.2 peer-group RR
 !
 neighbor 10.1.1.2 remote-as 64500
```



Esempio (2/3)

```
RR-1# show bgp ipv4 unicast neighbors 192.168.0.11
BGP neighbor is 192.168.0.11,  remote AS 64501, internal link
  BGP version 4, remote router ID 192.168.0.11
  BGP state = Established, up for 00:31:24
  Last read 00:00:24, hold time is 180, keepalive interval is 60
   seconds
  Neighbor capabilities:
    Route refresh: advertised and received(old & new)
    Address family IPv4 Unicast: advertised and received
  Message statistics:
    . . .
 For address family: IPv4 Unicast
  BGP table version 6, neighbor version 6
  Index 1, Offset 0, Mask 0x2
  Route-Reflector Client
  . . .
```

Esempio (3/3)

- Prefisso ricevuto dal *RR-client* PE1-1 visto nella tabella BGP di RR-1

```
RR-1# show bgp ipv4 unicast 203.0.113.0
BGP routing table entry for 203.0.113.0/24, version 6
Paths: (1 available, best #1, table Default-IP-Routing-Table)
  Advertised to non peer-group peer:
  192.168.2.2 192.168.0.21 192.168.0.31
  64500, (Received from a RR-client)
    192.168.0.11 (metric 85) from 192.168.0.11 (192.168.0.11)
      Origin IGP, metric 0, localpref 100, valid, internal, best
```

- Prefisso "riflesso" dai RR visto nella tabella BGP di PE3-1

```
PE3-1# show bgp ipv4 unicast 203.0.113.0
BGP routing table entry for 203.0.113.0/24, version 9
Paths: (2 available, best #1, table Default-IP-Routing-Table)
  Not advertised to any peer
  64500
    192.168.0.11 (metric 85) from 192.168.2.1 (192.168.2.1)
      Origin IGP, metric 0, localpref 100, valid, internal, best
      Originator: 192.168.0.11, Cluster list: 0.0.4.30
  64500
    192.168.0.11 (metric 85) from 192.168.2.2 (192.168.2.2)
      Origin IGP, metric 0, localpref 100, valid, internal
      Originator: 192.168.0.11, Cluster list: 0.0.4.30
```

RR-1

RR-2

Aspetti di sicurezza

Problemi e soluzioni

Contromisure

Remote-Triggered Black-Hole Filtering (*RTBH*)

RPKI e *ROA*

Problemi di sicurezza

- PREMESSA: Il BGP, essendo il protocollo più importante delle reti IP, è quello su cui si focalizza maggiormente l'attenzione degli «*Hacker*»

- Il BGP è soggetto ad attacchi esterni di vario tipo, classificabili a grandi linee come
  - Attacchi a livello di sessione
    - Attacchi alla connessione TCP (es. *TCP reset, SYN flooding*)
  - Attacchi *Denial of Service* (*DoS*)
    - Saturazione delle risorse di un router
    - *BGP spoofing*
    - *Prefix Hijacking*
    - *Route flapping*
  - *Comportamento «non etico»*
    - *Traffic/Bandwidth stealing*

- Per ciascuno di questi attacchi sono state individuate delle contromisure e altre sono allo studio

Attacchi a livello di sessione

- Alcuni importanti tipi di attacchi a livello di sessione
  - Deduzione di informazioni confidenziali
    - Una eventuale intercettazione dei messaggi BGP (in particolare dei messaggi UPDATE) permette di svelare il contenuto degli accordi tra ISP e le politiche di routing adottate
  - Attacco all'integrità dei messaggi
    - Si potrebbe ad esempio inserire messaggi BGP falsi tra due *BGP peer*, con lo scopo di alterare le informazioni di routing, eliminare dei messaggi, come ad esempio i KEEPALIVE, con lo scopo di abbattere la sessione BGP, intercettare e modificare "al volo" un messaggio, alterandone il contenuto.
  - Terminazione non voluta di una sessione
    - Invio di un segmento *TCP reset*, introduzione di eventi errati che comportino transizioni della macchina a stati finiti del BGP che portano la sessione negli stati *Idle* o *Active*, blocco di messaggi KEEPALIVE, creazione di messaggi NOTIFICATION falsi, ecc.
  - Attacchi tipici al protocollo TCP (es. *Syn flooding*, *TCP reset*, *Session Hijacking*/*Desynchronization*, ecc.)

- Conseguenza degli attacchi a livello di sessione è la manipolazione della Tabella BGP

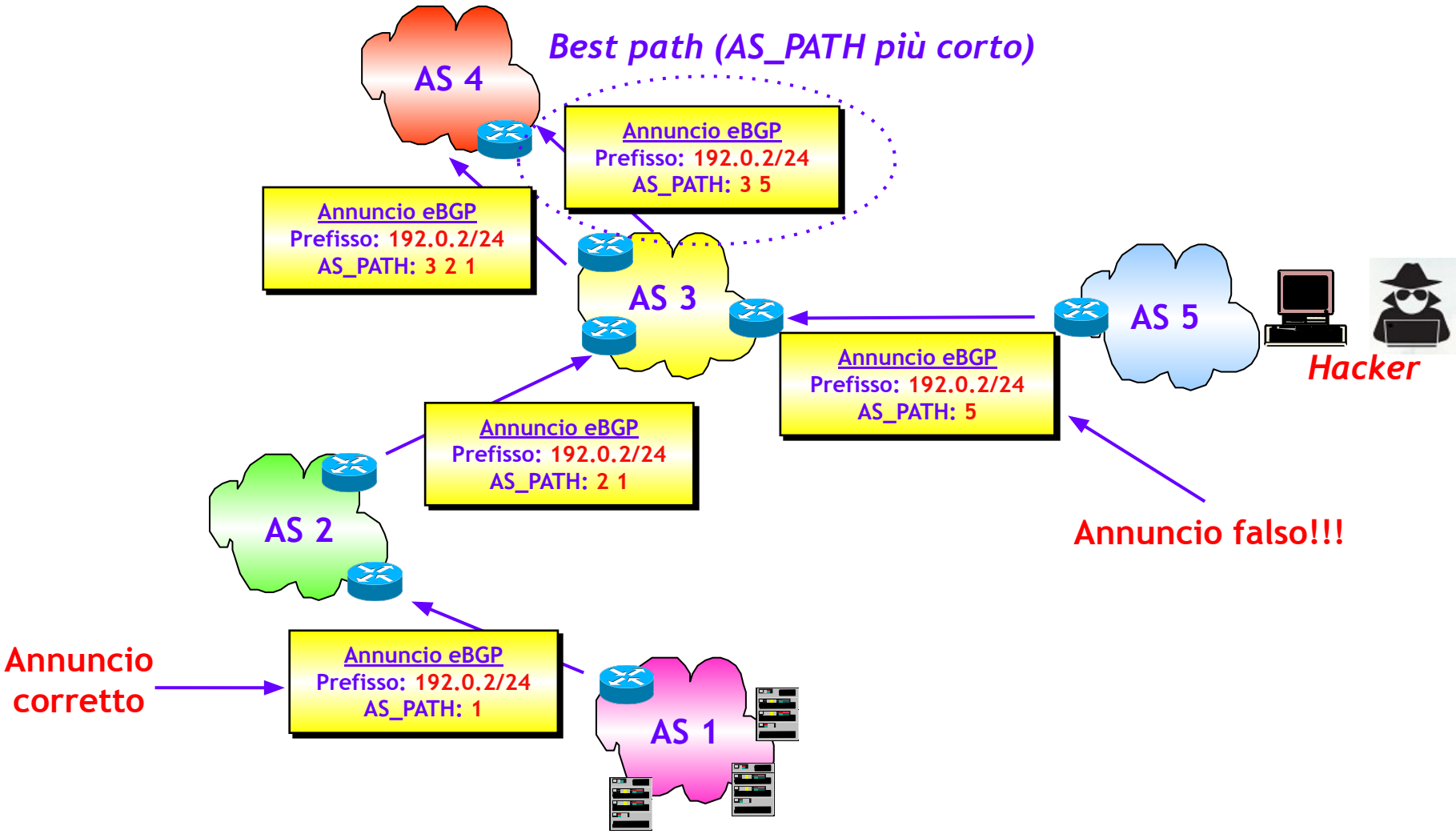
REISS ROMOLI

Attacchi *Denial of Service* (*DoS*)

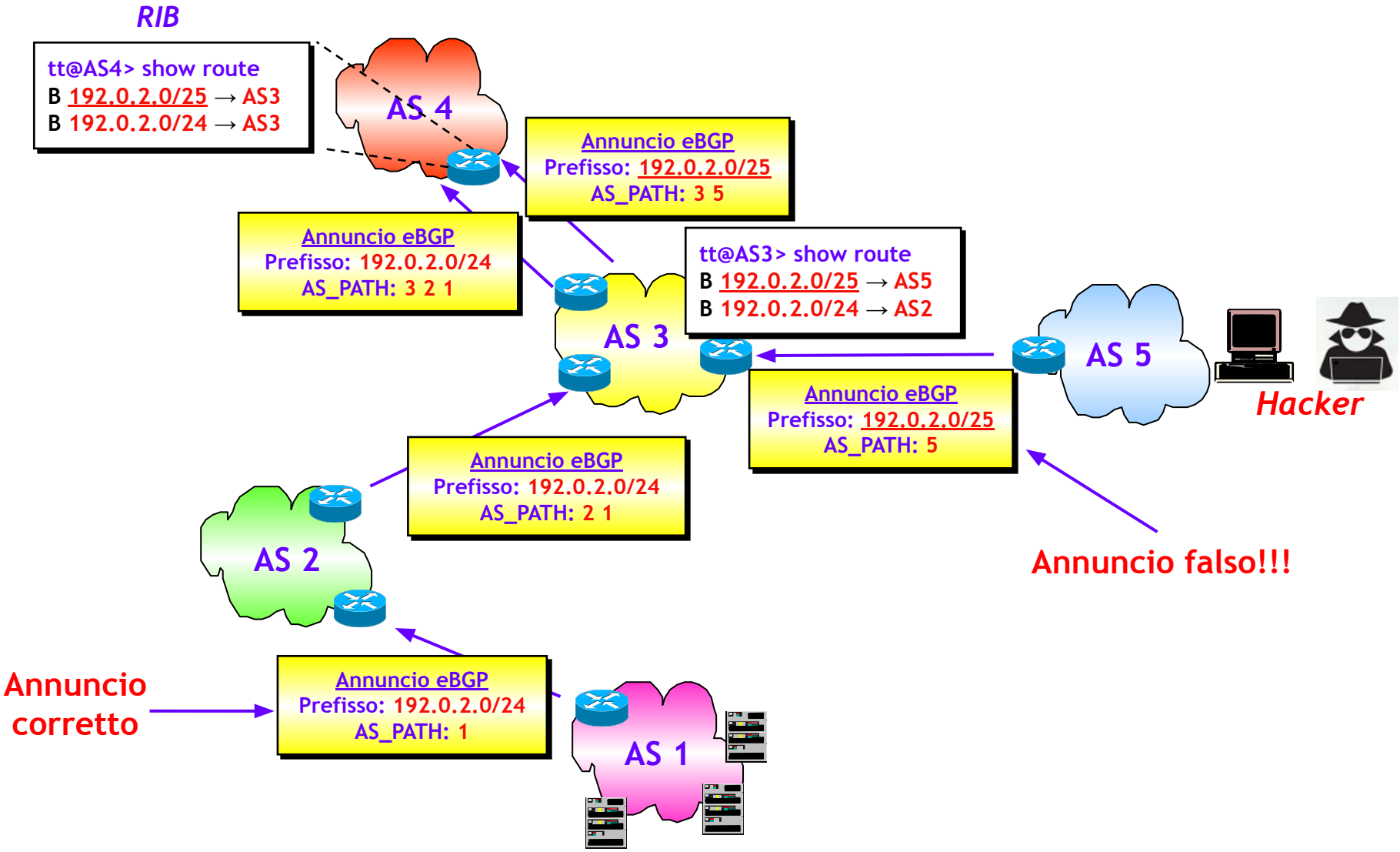
- Attacchi tendenti a negare il funzionamento del servizio

- Alcuni esempi
  - *Prefix Hijacking*
    - Attacco tendente a dirottare il traffico diretto a un determinato prefisso verso il proprio router, per poi scartarlo
  - Utilizzo del *Route Flap Damping*
    - Simulando dei *route flap*, è possibile mantenere congelato un annuncio per molto tempo, alterando il corretto instradamento del traffico
  - Saturazione delle risorse di memoria e/o CPU di un router
    - Elevate quantità di annunci BGP, *Syn flooding*

Prefix Hijacking (1/2)



Prefix Hijacking (2/2)



Aspetti di sicurezza

Problemi e soluzioni

Contromisure

Remote-Triggered Black-Hole Filtering (*RTBH*)

RPKI e *ROA*

Tipi di contromisure

Contromisure	Attacchi alla sessione	*Prefix Hijacking*	Attacchi (D)DoS
Autenticazione messaggi	Si	No	No
Gestione sicura del TTL	Si	No	Si
Limitazione del numero di prefissi	No	No	Si
RTBH e BGP FlowSpec	No	No	Si
RPKI + ROA	No	Si	No

REISS ROMOLI

Autenticazione dei messaggi

- La vecchia RFC 1771 prevedeva la possibilità di negoziare con il messaggio di OPEN, l'autenticazione dei messaggi
  - Non era specificato il particolare algoritmo di autenticazione ma solo delle linee guida
  - Era previsto l'utilizzo del campo *Marker* nell'intestazione comune dei messaggi BGP per il trasporto della firma digitale
- La RFC 4271 consiglia di utilizzare direttamente le funzionalità di autenticazione del TCP
  - Autenticazione via TCP MD5 (RFC 2385)
  - Non è richiesta alcuna negoziazione

Autenticazione MD5: configurazione

- IOS e IOS XE

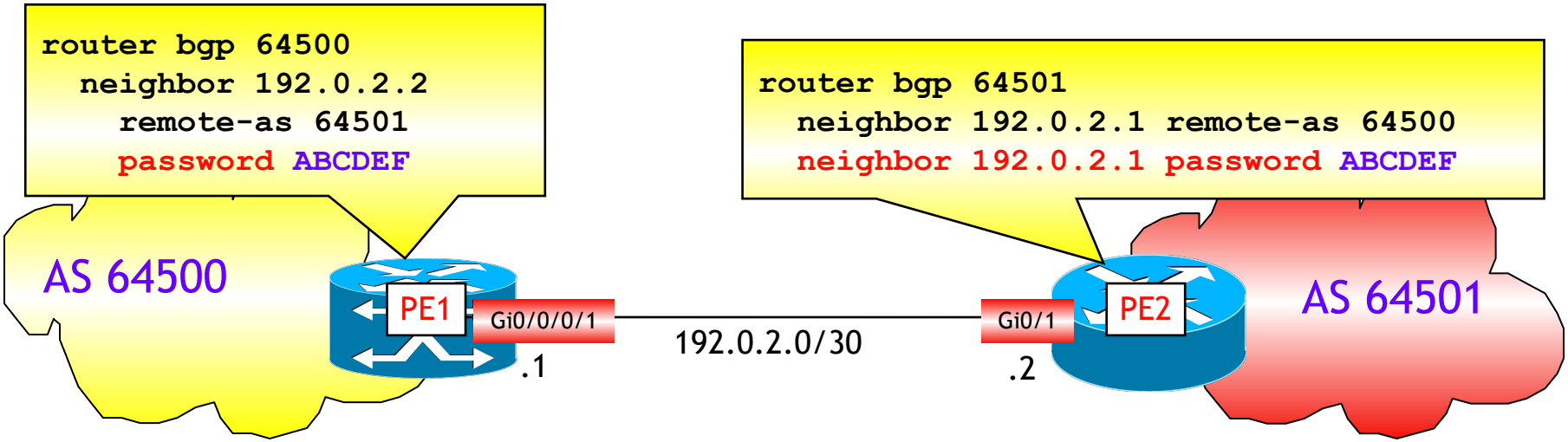
```
router(config)# router bgp numero-AS
router(config-router)# neighbor indirizzo-IP-peer
                      password password
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor indirizzo-IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# password password
```

- IMPORTANTE: la *password* deve coincidere su entrambi i *BGP peer*

Autenticazione MD5: esempio



- Se la *password* venisse configurata solo da un lato si otterrebero messaggi di errore del tipo

```
6d22h: %TCP-6-BADAUTH: No MD5 digest from 192.0.2.1(11004)
   to 192.0.2.2(179)
```

- Se le *password* configurate fossero diverse si otterrebbero messaggi di errore del tipo

```
6d22h: %TCP-6-BADAUTH: Invalid MD5 digest from 192.0.2.1(11002) to
   192.0.2.2 (179)
```

Limitazione del numero di prefissi ricevuti

- Un *BGP peer* mal configurato o intenzionalmente può inviare all'altro *BGP peer* un numero molto elevato di annunci di diversi prefissi fino a saturarne la memoria
  - È un classico attacco di tipo *DoS*
  - Sono stati riportati vari incidenti di questo tipo nell'Internet
- La contromisura consiste nel limitare il numero di prefissi che un *BGP speaker* può ricevere da un determinato *BGP peer*
  - Il superamento del limite comporta lo scarto degli annunci successivi e (opzionale) può provocare un *restart* della sessione BGP

Configurazione

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# neighbor indirizzo-IP-peer
                      maximum-prefix valore [soglia-warning]
                      [warning-only | restart intervallo]
```

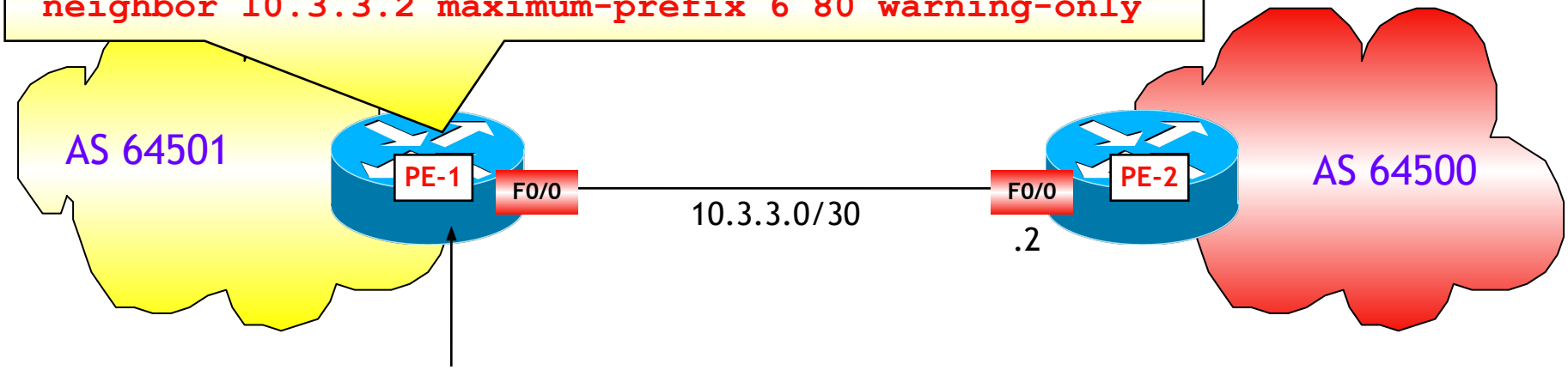
- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor indirizzo-IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# remote-as AS-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# maximum-prefix valore
                                        [soglia-warning]
                                        [warning-only | restart intervallo]
```

- *Default*: al superamento del numero massimo di prefissi ammessi la sessione BGP viene abbattuta

Esempio (1/2)

```
router bgp 64501
 neighbor 10.3.3.2 remote-as 64500
 neighbor 10.3.3.2 maximum-prefix 6 80 warning-only
```




```
PE-1# show bgp ipv4 unicast neighbors 10.3.3.2
BGP neighbor is 10.3.3.2,  remote AS 64500, external link
  BGP version 4, remote router ID 10.3.99.1
  BGP state = Established, up for 4d20h
  . . . (output omesso)
  Maximum prefixes allowed 6 (warning-only)
  Threshold for warning message 80%
  Number of NLRIs in the update sent: max 1, min 0
  . . . (output omesso)
```

Esempio (2/2)

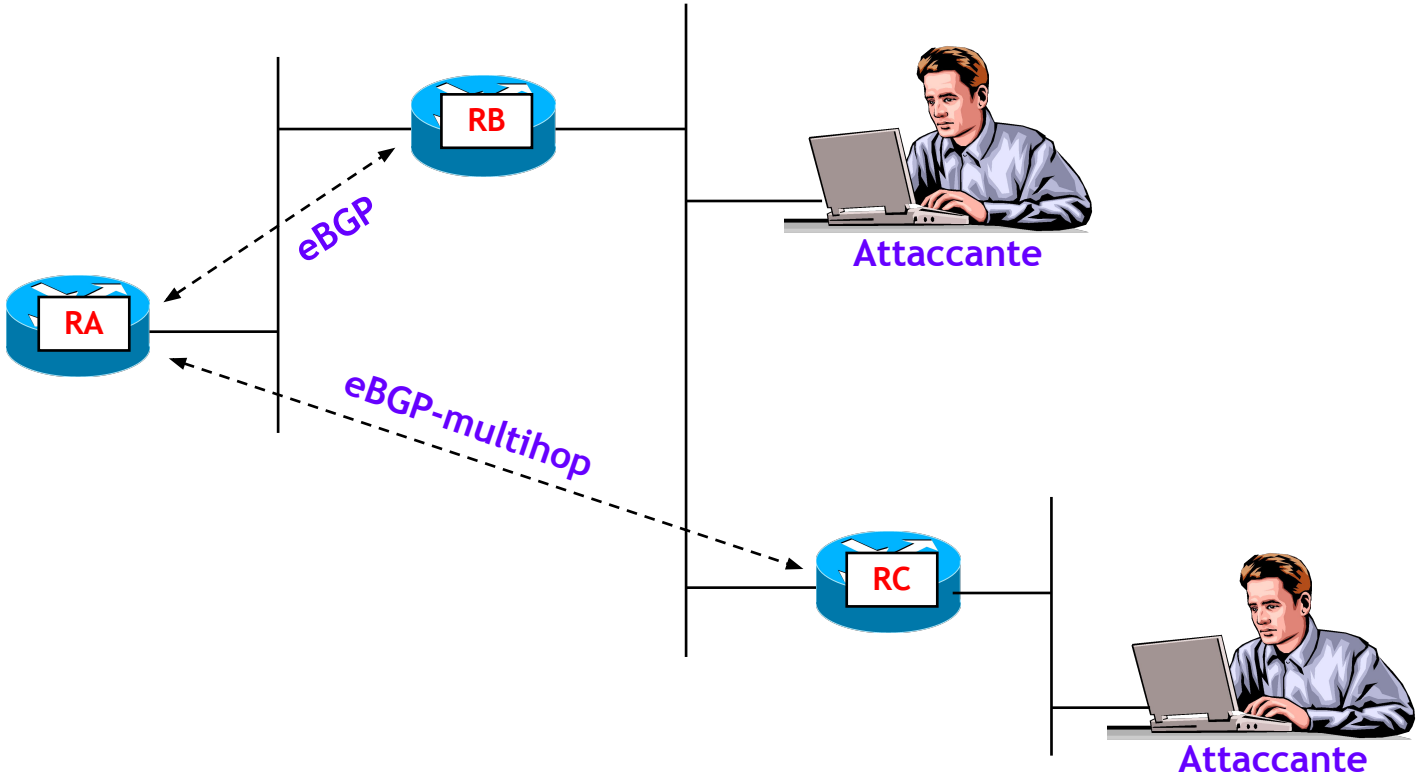
- Messaggio visualizzato al superamento della soglia di *warning* e al superamento del numero massimo di prefissi ammessi

```
PE-1#
4d21h: %BGP-4-MAXPFX: No. of prefix received from 10.3.3.2 (afi 0) reaches 5,
   max 6
4d21h: %BGP-3-MAXPFXEXCEED: No. of prefix received from 10.3.3.2 (afi 0): 7
   exceed limit 6
```

- Senza l'opzione «**warning-only**» la sessione BGP viene abbattuta e inviato al *BGP peer* un messaggio NOTIFICATION

```
PE-1#
4d21h: %BGP-5-ADJCHANGE: neighbor 10.3.3.2 Down BGP Notification sent
4d21h: %BGP-3-NOTIFICATION: sent to neighbor 10.3.3.2 3/1 (update malformed)

PE-1# show bgp ipv4 unicast summary
. . .
Neighbor    V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
10.3.3.2    4 64500    7075    7077        0    0    0 00:00:33 Idle (PfxCt)
```

Gestione sicura del TTL



- Variare la gestione del TTL da parte di RA, sulle sessioni eBGP *multi-hop*: un pacchetto IP della sessione BGP è accettato se e solo se il suo *TTL* è pari a (255-*N*+1), dove *N* è il numero di «*hop*» di distanza tra i *BGP peer*
  - La regola vale, come caso particolare, per le sessioni eBGP ordinarie (*N* = 1)

Configurazione

- IOS e IOS XE (*N* = numero di «*hop*»)

```
router(config)# router bgp numero-AS
router(config-router)# neighbor IP-peer ttl-security hops N
```

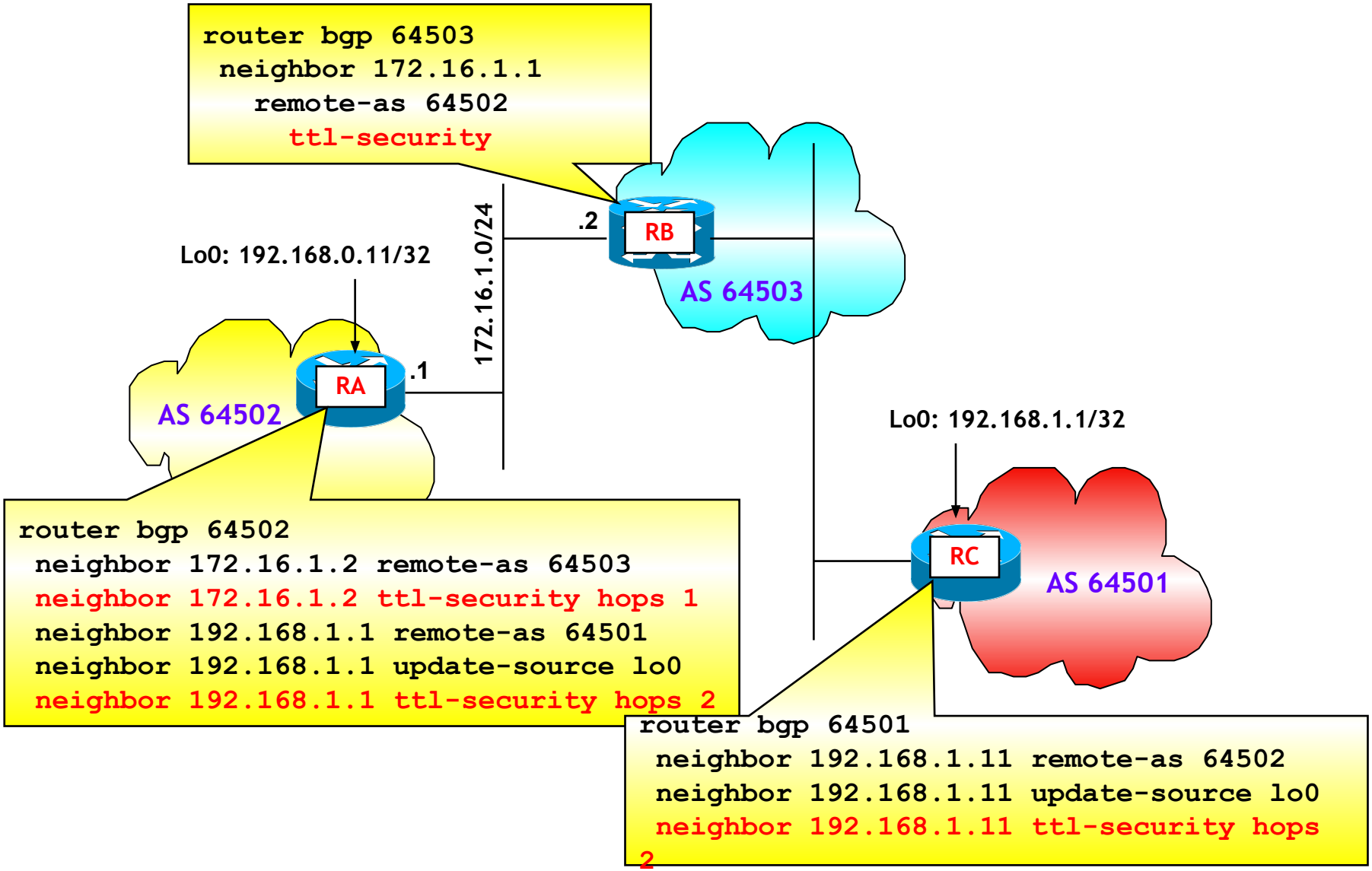
  - REGOLA: il BGP accetta messaggi con il $TTL \geq 255 - N + 1$
  - NOTA: questo comando è incompatibile con il comando «**neighbor … ebgp-multihop**»

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor indirizzo-IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# ttl-security
```

  - NOTA: l'IOS XR non supporta il *multi hop TTL security*

Esempio


router bgp 64503
neighbor 172.16.1.1
remote-as 64502
ttl-security
.2
RB
AS 64503
Lo0: 192.168.0.11/32
172.16.1.0/24
.1
RA
AS 64502
Lo0: 192.168.1.1/32
RC
AS 64501
router bgp 64502
neighbor 172.16.1.2 remote-as 64503
neighbor 172.16.1.2 ttl-security hops 1
neighbor 192.168.1.1 remote-as 64501
neighbor 192.168.1.1 update-source lo0
neighbor 192.168.1.1 ttl-security hops 2
router bgp 64501
neighbor 192.168.1.11 remote-as 64502
neighbor 192.168.1.11 update-source lo0
neighbor 192.168.1.11 ttl-security hops 2

REISS ROMOLI

Limitazione della lunghezza dell'AS_PATH

- Gli annunci BGP nell'Internet hanno tipicamente una lunghezza media dell'AS_PATH di pochi AS
  - Secondo uno studio RIPE NCC la lunghezza media di un AS_PATH nell'Internet è di 4,3 per l'Internet IPv4 e 3,6 per l'Internet IPv6
- Si sono verificati casi di AS_PATH abnormi a causa di *AS_PATH prepending* eccessivi
  - Esempio (reale): l'AS 197158 (Est Europa) ha annunciato il prefisso 95.47.142.0/23 con il seguente AS_PATH: «… 3255 197158» eseguendo un *AS_PATH prepending* di 23 volte il proprio numero di AS
- Problemi
  - Un utente malintenzionato potrebbe intercettare o manipolare il traffico verso questo prefisso, utilizzando ad esempio un ISP di dubbia morale che consente annunci dello stesso prefisso con un AS_PATH del tipo "ASxxxx ASyyyy 197158"
  - Inoltre in passato questo ha causato dei *down* delle sessioni BGP nelle piattaforme Cisco a causa di un bug software

Configurazione

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# bgp maxas-limit valore
```

- IOS XR

```
RP/0/RP0/CPU0:PE4(config)#route-policy nome-RP
RP/0/RP0/CPU0:PE4(config-rpl)#  if as-path length le valore then
RP/0/RP0/CPU0:PE4(config-rpl-if)#    pass
RP/0/RP0/CPU0:PE4(config-rpl-if)#  endif
RP/0/RP0/CPU0:PE4(config-rpl)#end-policy
!
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor indirizzo-IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# route-policy nome-RP in
```

Esempio

- Accettare dal BGP peer 172.20.1.1 dell'AS 65541 solo annunci che hanno AS_PATH di lunghezza al massimo 2
  - IPOTESI: router Cisco con IOS XR

```
route-policy MAX_AS_PATH_LENGTH
  if as-path length le 2 then
    pass
  endif
end-policy
!
router bgp 64501
  neighbor 172.20.1.1
    remote-as 65541
    address-family ipv4 unicast
      route-policy MAX_AS_PATH_LENGTH in
```

Aspetti di sicurezza

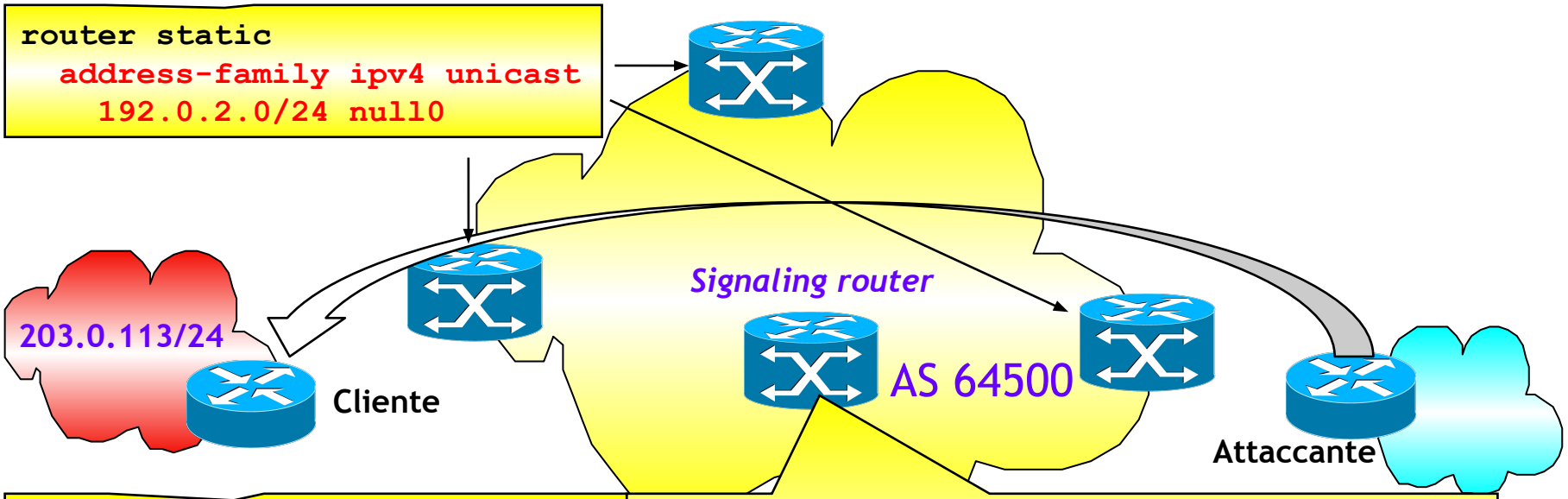
Problemi e soluzioni
Contromisure
Remote-Triggered Black-Hole Filtering (RTBH)
RPKI e ROA

REISS ROMOLI

Scopo e funzionamento

- Quando un Cliente è sotto un attacco DoS (o DDoS), il traffico diretto al Cliente può causare danni «collaterali» alla rete dell'ISP
  - Quando l'attacco viene rilevato, questo traffico dovrebbe essere scartato ai bordi della rete dell'ISP per evitare di sovraccaricarne gli apparati
- Funzionamento del RTBH
  - Un router viene scelto (via configurazione) come «*signaling router*»
  - Il «*signaling router*» segnala via BGP ai router di «*edge*» che il traffico generato dall'attacco DoS deve essere scartato
- Due versioni
  - *Destination-based RTBH*: il traffico verso l'indirizzo IP del Cliente oggetto di attacco viene scartato dai router di *Edge*
  - *Source-based RTBH*: il traffico proveniente dall'indirizzo IP dell'attaccante viene scartato dai router di *Edge*
    - Richiede l'utilizzo del controllo *uRPF* (*unicast Reverse Path Forwarding*)

Destination-based RTBH



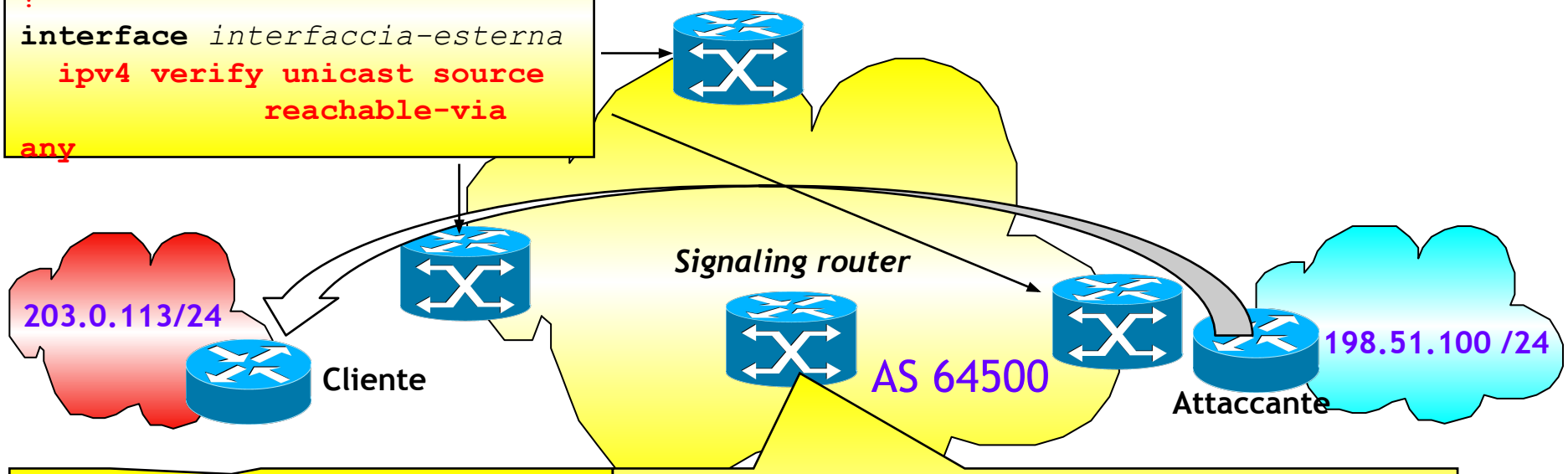

```
route-policy DB-RTBH
  if tag is 100 then
    set next-hop 192.0.2.1
    set community (no-export)
    set local-preference 5000
  endif
end-policy
```

```
router bgp 64500
  address-family ipv4 unicast
   redistribute static route-policy DB-RTBH
!
! DOPO AVER RILEVATO L'ATTACCO
!
router static
  address-family ipv4 unicast
    203.0.113.0/24 null0 tag 100
```

Source-based RTBH

REISS ROMOLI

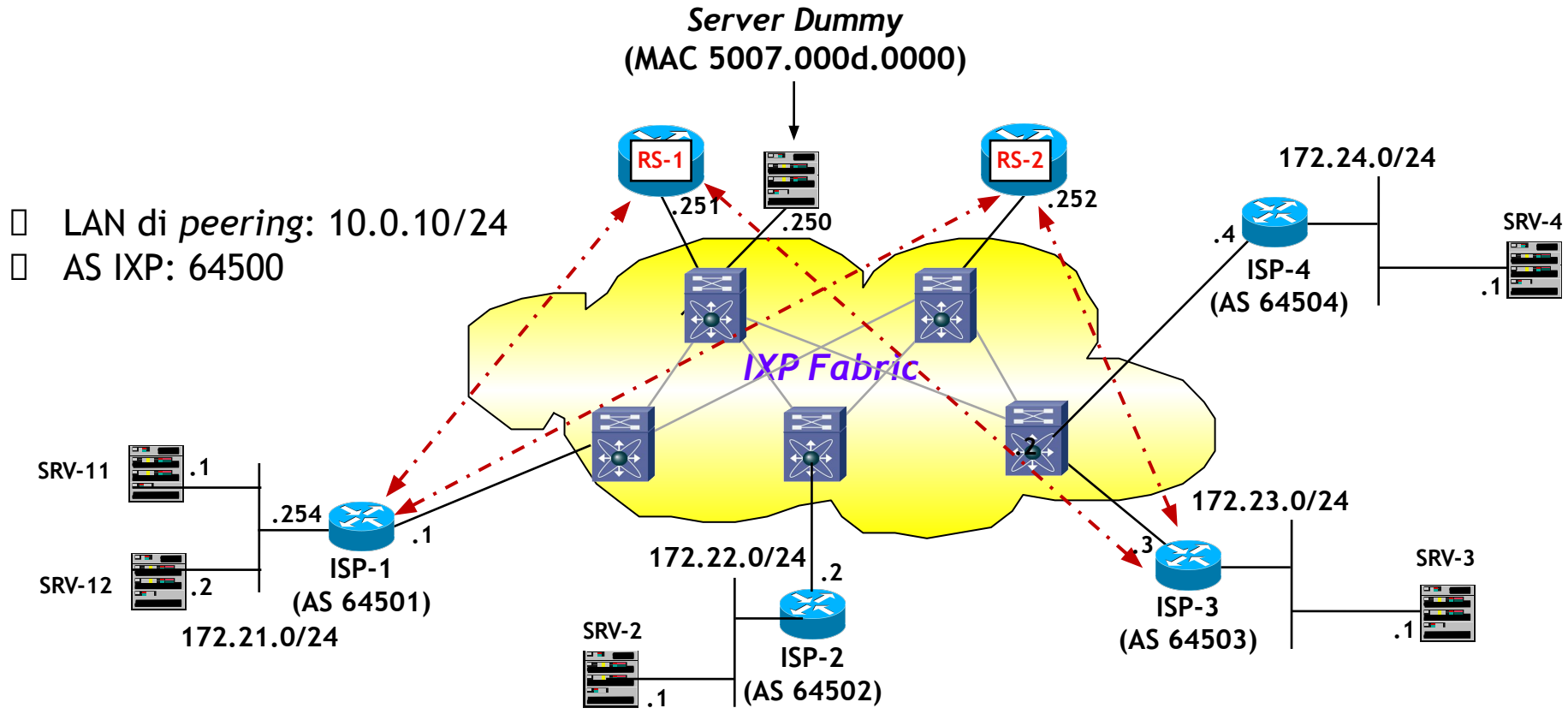
```
router static
  address-family ipv4 unicast
    192.0.2.0/24 null0
!
interface interfaccia-esterna
  ipv4 verify unicast source
              reachable-via
any
```




```
route-policy SB-RTBH
  if tag is 100 then
    set next-hop 192.0.2.1
    set community (no-export)
    set local-preference 5000
  endif
end-policy
```

```
router bgp 64500
  address-family ipv4 unicast
   redistribute static route-policy SB-RTBH
!
! DOPO AVER RILEVATO L'ATTACCO
!
router static
  address-family ipv4 unicast
    198.51.100.0/24 null0 tag 100
```

Case Study: *RTBH* in un *IXP* (1/6)



- **IPOTESI**: il server 172.21.0.1 dell'ISP-1 subisce un attacco DDOS
- **OBIETTIVO**: bloccare tutto il traffico diretto a 172.21.0.1 ai bordi della *IXP Fabric*

Case Study: *RTBH* in un *IXP* (2/6)

REISS ROMOLI

```
hostname ISP-1
!
router bgp 64501
 bgp router-id 192.168.0.1
 no bgp enforce-first-as
 network 172.21.0.0 mask 255.255.255.0
 redistribute static route-map RTBH
 neighbor RS peer-group
 neighbor RS remote-as 64500
 neighbor RS description "SESSIONE CON RS-1/2"
 neighbor RS send-community
 neighbor RS route-map ONLY-LOCAL out
 neighbor 10.0.10.251 peer-group RS
 neighbor 10.0.10.252 peer-group RS
!
ip prefix-list IP-RTBH seq 5 permit 0.0.0.0/0 ge 32
ip as-path access-list 1 permit ^$
!
route-map RTBH permit 10
 match ip address prefix-list IP-RTBH
 set community 65535:666  ! COMMUNITY RISERVATA PER APPLICAZIONI RTBH
!
route-map ONLY-LOCAL permit 10
 match as-path 1
```

Case Study: RTBH in un IXP (3/6)

```
hostname RS-1
!
router bgp 64500
 bgp router-id 172.20.0.1
 neighbor RS-C peer-group
 neighbor 10.0.10.1 remote-as 64501
 neighbor 10.0.10.1 peer-group RS-C
 neighbor 10.0.10.2 remote-as 64502
 neighbor 10.0.10.2 peer-group RS-C
 neighbor 10.0.10.3 remote-as 64503
 neighbor 10.0.10.3 peer-group RS-C

 !
 address-family ipv4
  neighbor RS-C route-server-client
  neighbor RS-C route-map RTBH out
  neighbor 10.0.10.1 activate
  neighbor 10.0.10.2 activate
  neighbor 10.0.10.3 activate
 exit-address-family
```

```
ip prefix-list HOST-ROUTES seq 5
                    permit 0.0.0.0/0 ge
32
!
ip community-list 66 permit 65535:666
!
route-map RTBH permit 10
 description *** RTBH ROUTES OUT ***
 match ip address prefix-list
HOST-ROUTES
 match community 66
 set community no-export
 set ip next-hop 10.0.10.250
!
route-map RTBH permit 20
```

Indirizzo IP del *Dummy Server*

Case Study: RTBH in un *IXP* (4/6)

- Per bloccare il traffico verso l'attaccato sul bordo della *IXP Fabric* si utilizza una «MAC ACL»
  - Con la «MAC ACL» si blocca tutto il traffico con MAC destinazione il MAC del *server dummy*

```
mac access-list extended RTBH
 deny  any host 5007.000d.0000
 permit any any

interface tipo numero
 mac access-group RTBH in
```

Indirizzo MAC del *server dummy*

- NOTA: la «MAC ACL» va configurata su tutti gli switch dove sono attestati gli ISP e quindi va applicata su tutte le interfacce dove sono attestati gli ISP

Case Study: *RTBH* in un *IXP* (5/6)

- Appena rilevato il *target* dell'attacco (= 172.21.0.1) ...

```
hostname ISP-1
!
ip route 172.21.0.1 255.255.255.255 Null0
```

```
RS-1# show ip bgp 172.21.0.1/32
BGP routing table entry for 172.21.0.1/32, version 7
Paths: (1 available, best #1, table default)
Advertised to update-groups:
     2
  Refresh Epoch 2
  64501
    10.0.10.1 from 10.0.10.1 (192.168.0.1)
      Origin incomplete, metric 0, localpref 100, valid, external, best
      Community: 65535:666
      rx pathid: 0, tx pathid: 0x0
```

Case Study: *RTBH* in un *IXP* (6/6)

- Verifica
  - NOTA: il server SRV-3 è simulato da un router

Istante di ricezione dell'annuncio della *host route* 172.21.0.1/32 su ISP-3

```
SRV-3# ping 172.21.0.1 rep 500
Type escape sequence to abort.
Sending 1000, 100-byte ICMP Echos to 101.1.0.1, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!U.
.....................................................................
Success rate is 66 percent (138/208), round-trip min/avg/max = 3/17/26 ms
```

- Analisi *wireshark* di un pacchetto ICMP *Echo Request* inviato da SRV-3

```
> Ethernet II, Src: aa:bb:cc:00:67:00 (aa:bb:cc:00:67:00), Dst: 50:07:00:0d:00:00 (50:07:00:0d:00:00)
> Internet Protocol Version 4, Src: 172.23.0.1, Dst: 172.21.0.1
v Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x74ef [correct]
    [Checksum Status: Good]
    Identifier (BE): 3 (0x0003)
    Identifier (LE): 768 (0x0300)
    Sequence number (BE): 2 (0x0002)
    Sequence number (LE): 512 (0x0200)
  > [No response seen]
  > Data (72 bytes)
```

Indirizzo MAC del *server dummy*

Aspetti di sicurezza

Problemi e soluzioni
Contromisure
Remote-Triggered Black-Hole Filtering (*RTBH*)
RPKI e ROA

RPKI: cosa è?

- È una architettura standard basata su Certificati Digitali che consente a un ISP di verificare se un annuncio ricevuto è corretto
  - Per corretto si intende che l'AS che origina l'annuncio è autorizzato a farlo, ossia che sta annunciando prefissi che gli sono stati assegnati dai *RIR*
- I Certificati Digitali sono contenuti in *RPKI repository* accessibili pubblicamente e gestiti dai 5 *RIR* mondiali
  - La gerarchia dei certificati digitali segue la gerarchia di assegnazione degli indirizzi IP: *IANA→RIR→NIR/LIR*
- RPKI utilizza il formato dei certificati digitali X.509, con l'estensione definita dalla RFC 3779 - *X.509 Extensions for IP Addresses and AS Identifiers*
  - Per la cifratura e la firma digitale di oggetti, utilizza lo standard definito nella RFC 3852 "Cryptographic Message Syntax".

Route Origin Authorization (ROA)

- Un *ROA* è un oggetto corredato di un certificato digitale, che fornisce un mezzo per verificare se un AS è autorizzato ad annunciare un determinato prefisso IP

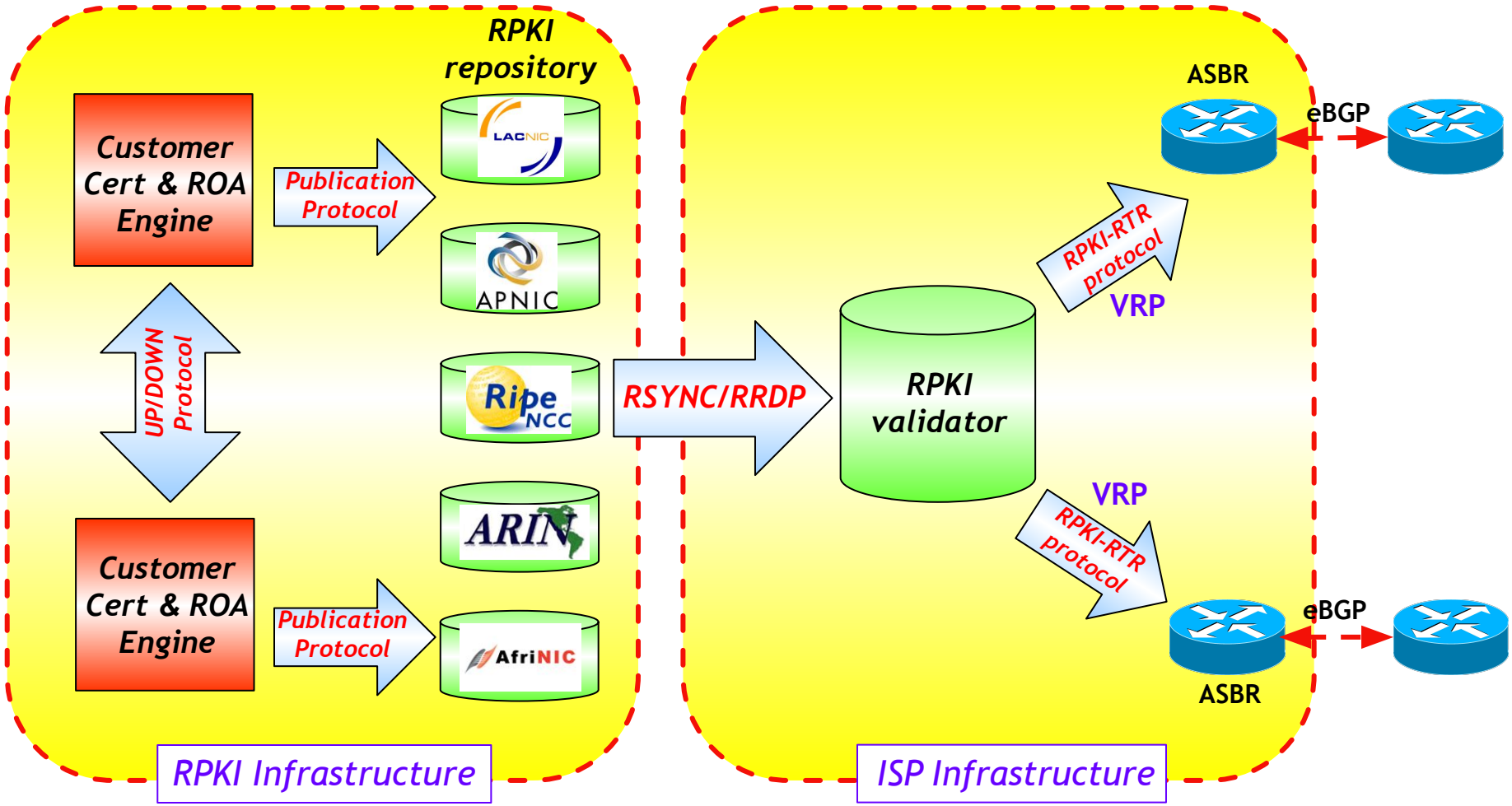
- Formato dei *ROA*

ROA IPv4 *ROA* IPv6

Prefisso originato	172.16.0.0/16	2001:db8:1::/48
Lunghezza massima della Maschera	24	48
AS Origine	64500	64500
Firma Digitale	qç!r5@eX!%89?@cv!	sdg@!dr34@?1QWx!@a

- NOTA: la RFC 9319, CONSIGLIA un valore di Lunghezza massima della Maschera pari a quello della maschera del prefisso IP

Architettura *RPKI*

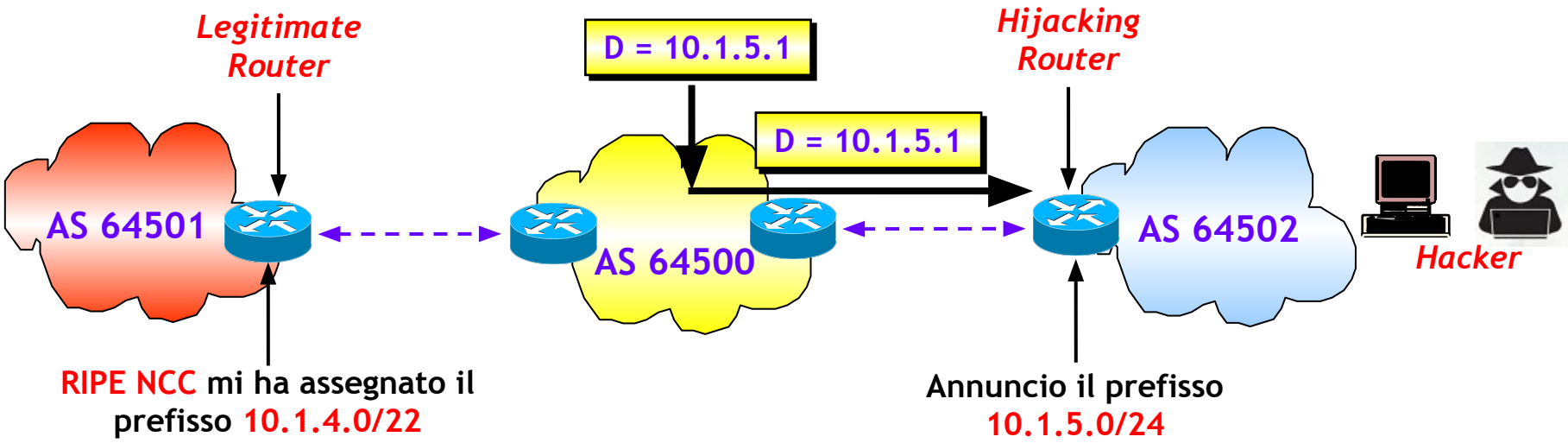


VRP = *Validated ROA Payload*

RPKI e Prefix Hijacking (1/2)

Senza RPKI

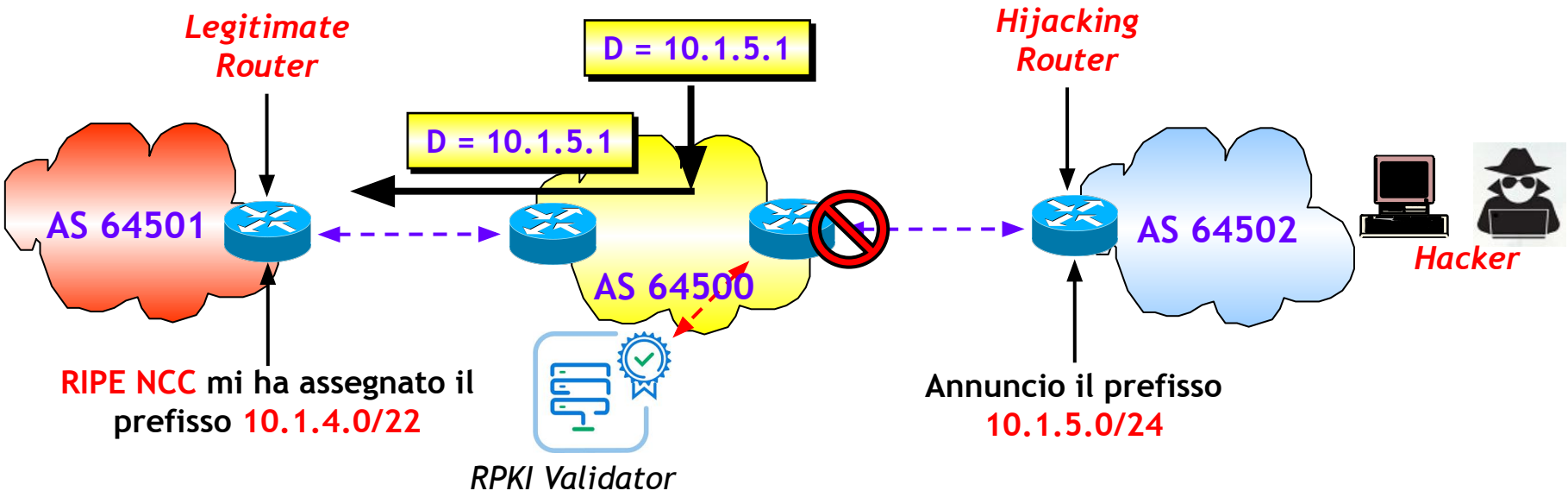
- IPOTESI: un router non legittimato (*Hijacking Router*) annuncia un prefisso più specifico di un prefisso annunciato legittimamente da un altro AS
- CONSEGUENZA 1: il traffico diretto a Host del prefisso più specifico annunciato non legittimamente (10.1.5.0/24), finisce su un altro AS
- CONSEGUENZA 2: gli Host dell'AS 64501 con indirizzo IP appartenente al prefisso 10.1.5.0/24 non ricevono più traffico



RPKI e Prefix Hijacking (2/2)

Con RPKI

- IPOTESI: un router non legittimato (*Hijacking Router*) annuncia un prefisso più specifico di un prefisso annunciato legittimamente da un altro AS
- L'ISP verifica attraverso il suo *RPKI Validator* che l'AS che ha annunciato il prefisso non è autorizzato a farlo e quindi dichiara non valido l'annuncio
- CONSEGUENZA: gli Host dell'AS 64501 con indirizzo IP appartenente al prefisso 10.1.5.0/24 ricevono regolarmente il traffico



Validazione di un annuncio (1/2)

- Si supponga di voler validare un annuncio BGP delle *address-family IPv4/IPv6 unicast* con
  - ***NLRI*** = ***Pfx/Mask***
  - ***AS Origine*** = ***AS-X***

- I possibili risultati del processo di ***validazione*** sono i seguenti
  - ***Valid***: nella *RPKI Table* esiste un *VRP* = <*Pfx-VRP/Mask-VRP*; *Max-Maschera*; *AS-VRP*> con *AS-X* = *AS-VRP*, *Pfx/Mask* è un prefisso più specifico di *Pfx-VRP/Mask-VRP* e *Mask* ≤ *Max-Maschera*
  - ***Invalid***: nella *RPKI Table* esiste un *VRP* = <*Pfx-VRP/Mask-VRP*; *Max-Maschera*; *AS-VRP*>, *Pfx/Mask* è un prefisso più specifico di *Pfx-VRP/Mask-VRP* , ma o *AS-X* ≠ *AS-VRP* e/o *Mask* > *Max-Maschera*
  - ***NotFound***: nella *RPKI Table* non esiste alcun *VRP* per cui *Pfx/Mask* è un prefisso più specifico di *Pfx-VRP/Mask-VRP*

Validazione di un annuncio (2/2)

- Esempio di validazione di prefissi IPv4

VRP	=	6762	79.140.80.0/20	24
		AS Origine	Prefisso	Max maschera

Valid	AS 6762	79.140.80.0/21
Valid	AS 6762	79.140.92.0/22
Invalid	AS 2914	79.140.84.0/22
Invalid	AS 6762	79.140.91.0/25
NotFound	AS 1299	79.140.0.0/16

ROA con AS Origine = 0

- AS = 0 è un numero di AS speciale
  - Non è assegnato ad alcun *Autonomous System*
  - Non è autorizzato ad annunciare prefissi
- In *RPKI* può essere utilizzato per proteggere uno spazio di indirizzamento che non dovrebbe essere visto nella *Default Free Zone*
  - Ad esempio le *network* di *peering* negli *IXP*
- NOTA: come *best-practice* il parametro *MaxLength* va impostato uguale alla dimensione del prefisso

Aspetti di configurazione (1/2)

- La configurazione richiede tre parametri
  - Indirizzo IP del *RPKI Validator*
  - Porta TCP da utilizzare
  - Periodo delle *query* verso il *RPKI Validator* (*refresh time*)

- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# bgp rpki server tcp IP-RPKI-Validator
                        port porta-RPKI-Validator refresh secondi
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# rpki server IP-RPKI-Validator
RP/0/RP0/CPU0:router(config-bgp-rpki-server)# transport tcp
                                              port porta-RPKI-Validator
RP/0/RP0/CPU0:router(config-bgp-rpki-server)# refresh-time secondi
```

Aspetti di configurazione (2/2)

- Con *RPKI* abilitato il comportamento di *default* dei router Cisco è di non propagare lo stato di validazione *RPKI* sulle sessioni iBGP
- Per consentirne la propagazione utilizzare i comandi seguenti
  - Le informazioni sullo stato di validazione vengono propagate attraverso una *BGP Extended Community*
- IOS e IOS XE

```
router(config)# router bgp numero-AS
router(config-router)# address-family ipv4 unicast
router(config-router-af)# neighbor IP-iBGP-peer announce rpki state
router(config-router-af)# neighbor IP-iBGP-peer send-community
                                                        extended
```

- IOS XR

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# bgp origin-as validation
                                                    signal ibgp
```

RPKI e processo di selezione BGP (IOS XE)

- Di default, gli annunci con stato di validazione *RPKI* ***Invalid*** non partecipano al processo di selezione BGP

- Per consentirne l'utilizzo eseguire il comando seguente

```
router(config)# router bgp numero-AS
router(config-router)# address-family ipv4 unicast
router(config-router-af)# bgp bestpath prefix-validate allow-invalid
```

- NOTA 1: l'utilizzo degli annunci dichiarati ***Invalid*** è comunque sconsigliato dalle *best practice*
- NOTA 2: in presenza di più annunci dello stesso prefisso, di cui almeno uno dichiarato ***Valid***, l'annuncio dichiarato ***Valid*** ha comunque la precedenza e viene eletto *best-path* indipendentemente dalle metriche BGP

RPKI e processo di selezione BGP (IOS XR)

- Per abilitare l'utilizzo degli stati di validazione nel processo di selezione BGP eseguire il comando seguente

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# bgp bestpath origin-as use validity
```

- Per consentire l'utilizzo degli annunci ***Invalid*** nel processo di selezione eseguire il comando seguente
  - Valgono comunque le considerazione delle due note nella diapositiva precedente

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# address-family ipv4 unicast
RP/0/RP0/CPU0:router(config-bgp-af)# bgp bestpath origin-as
                                              allow invalid
```

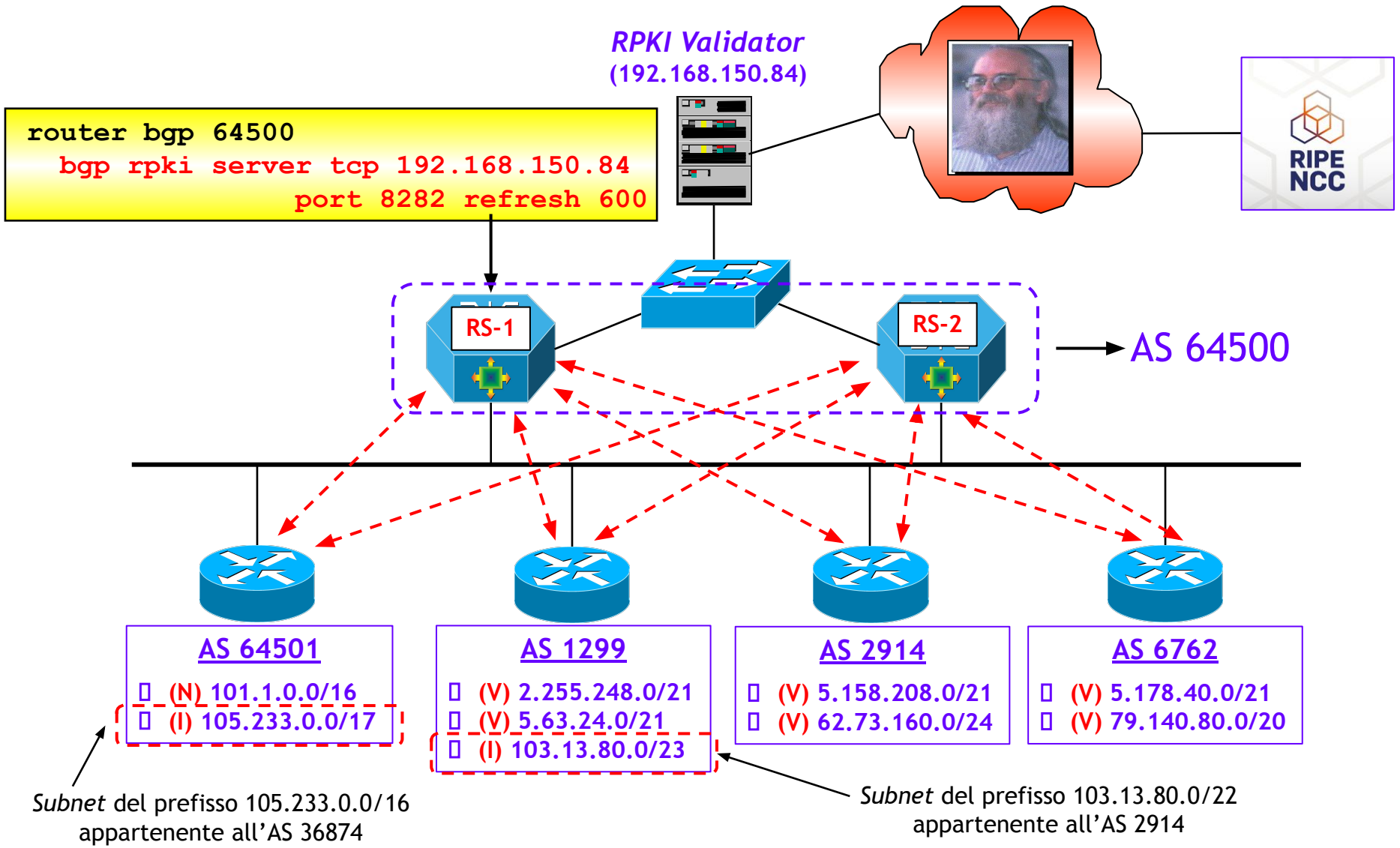
Azioni sugli annunci

- In funzione dei risultati del processo di validazione di un annuncio è possibile definire particolari azioni
- Esempio (IOS e IOS XE)
  - Assegna LP = 200 agli annunci ***Valid***
  - Assegna LP = 100 agli annunci ***NotFound***
  - Assegna LP = 50 agli annunci ***Invalid***

```
route-map RPKI permit 10
 match rpki invalid
 set local-preference 50
!
route-map RPKI permit 20
 match rpki not-found
 set local-preference 100
!
route-map RPKI permit 30
 match rpki valid
 set local-preference 200
```

```
router bgp 64500
  bgp bestpath prefix-validate allow-invalid
  neighbor 10.1.1.1 route-map RPKI in
```

Case Study (1/4)



Case Study (2/4)

- Verifica della connessione TCP con il *RPKI Validator*

```
RS-1# show bgp ipv4 unicast rpki servers
BGP SOVC neighbor is 192.168.150.84/8282 connected to port 8282
Flags 64, Refresh time is 60, Serial number is 7, Session ID is 53667
InQ has 0 messages, OutQ has 0 messages, formatted msg 33
Session IO flags 3, Session flags 4008
 Neighbor Statistics:
  Prefixes 106876
  Connection attempts: 488
  Connection failures: 482
  Errors sent: 0
  Errors received: 4

Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Disabled, Mininum incoming TTL 0, Outgoing TTL 255
Local host: 192.168.150.1, Local port: 15600
Foreign host: 192.168.150.84, Foreign port: 8282
Connection tableid (VRF): 0
Maximum output segment queue size: 50

. . . < resto dell'output omesso > . . .
```

Case Study (3/4)

- Elenco dei *ROA* IPv4/IPv6 nel router RS-1 scaricati dal *RPKI Validator* (utilizzando il protocollo *RPKI-to-Router*)

```
RS-1# show bgp ipv4 unicast rpki table
85094 BGP sovc network entries using 13615040 bytes of memory
91633 BGP sovc record entries using 2932256 bytes of memory

Network                 Maxlen  Origin-AS  Source  Neighbor
1.0.0.0/24              24      13335      0       192.168.150.84/8282
1.1.1.0/24              24      13335      0       192.168.150.84/8282
1.9.0.0/16              24      4788       0       192.168.150.84/8282
1.9.12.0/24             24      65037      0       192.168.150.84/8282
. . . < resto dell'output omesso > . . .
```

```
RS-1# show bgp ipv6 unicast rpki table
14037 BGP sovc network entries using 2582808 bytes of memory
15251 BGP sovc record entries using 488032 bytes of memory

Network                 Maxlen  Origin-AS  Source  Neighbor
2001:200::/32           32      2500       0       192.168.150.84/8282
2001:200:136::/48       48      9367       0       192.168.150.84/8282
2001:200:900::/40       40      7660       0       192.168.150.84/8282
2001:200:8000::/35      35      4690       0       192.168.150.84/8282
. . . < resto dell'output omesso > . . .
```

Case Study (4/4)

- Contenuto della Tabella BGP (IPv4 unicast) su RS-1

```
RS-1# show bgp ipv4 unicast
   BGP table version is 27, local router ID is 172.20.0.1
Status codes: s suppressed, d damped, h history, * valid, > best,
i - internal, r RIB-failure, S Stale, m multipath, b backup-path,
f RT-Filter, x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

     Network          Next Hop            Metric LocPrf Weight Path
V*>  2.255.248.0/21   10.0.10.99               0             0 1299 i
V*>  5.63.24.0/21     10.0.10.99               0             0 1299 i
V*>  5.158.208.0/21   10.0.10.29               0             0 2914 i
V*>  5.178.40.0/21    10.0.10.62               0             0 6762 i
V*>  62.73.160.0/24   10.0.10.29               0             0 2914 i
V*>  79.140.80.0/20   10.0.10.62               0             0 6762 i
N*>  101.1.0.0/16     10.0.10.11               0             0 101 i
I*   103.13.80.0/23   10.0.10.99               0             0 1299 i
I*   105.233.0.0/17   10.0.10.11               0             0 64501 i
```

APPENDICE

Scalabilità della configurazione

BGP peer groups
BGP peer templates
BGP configuration templates

Definizione e caratteristiche (1/2)

- Un *BGP peer group* è un insieme di *BGP peer* che condividono una identica politica di routing *outbound*
  - Funzionalità disponibile solo nell'IOS e IOS XE
- Vantaggi
  - Riduzione consistente del carico sulla CPU
  - Riduzione consistente del numero di righe di configurazione
- Due tipi di *BGP peer group*
  - Statici: i *BGP peer* appartenti al *BGP peer-group* sono definiti manualmente su base configurazione
  - Dinamici: i *BGP peer* appartenti al *BGP peer-group* sono definiti automaticamente dal router e non vi è alcun bisogno di configurazione manuale
    - Noti anche come *BGP Update Groups*

REISS ROMOLI

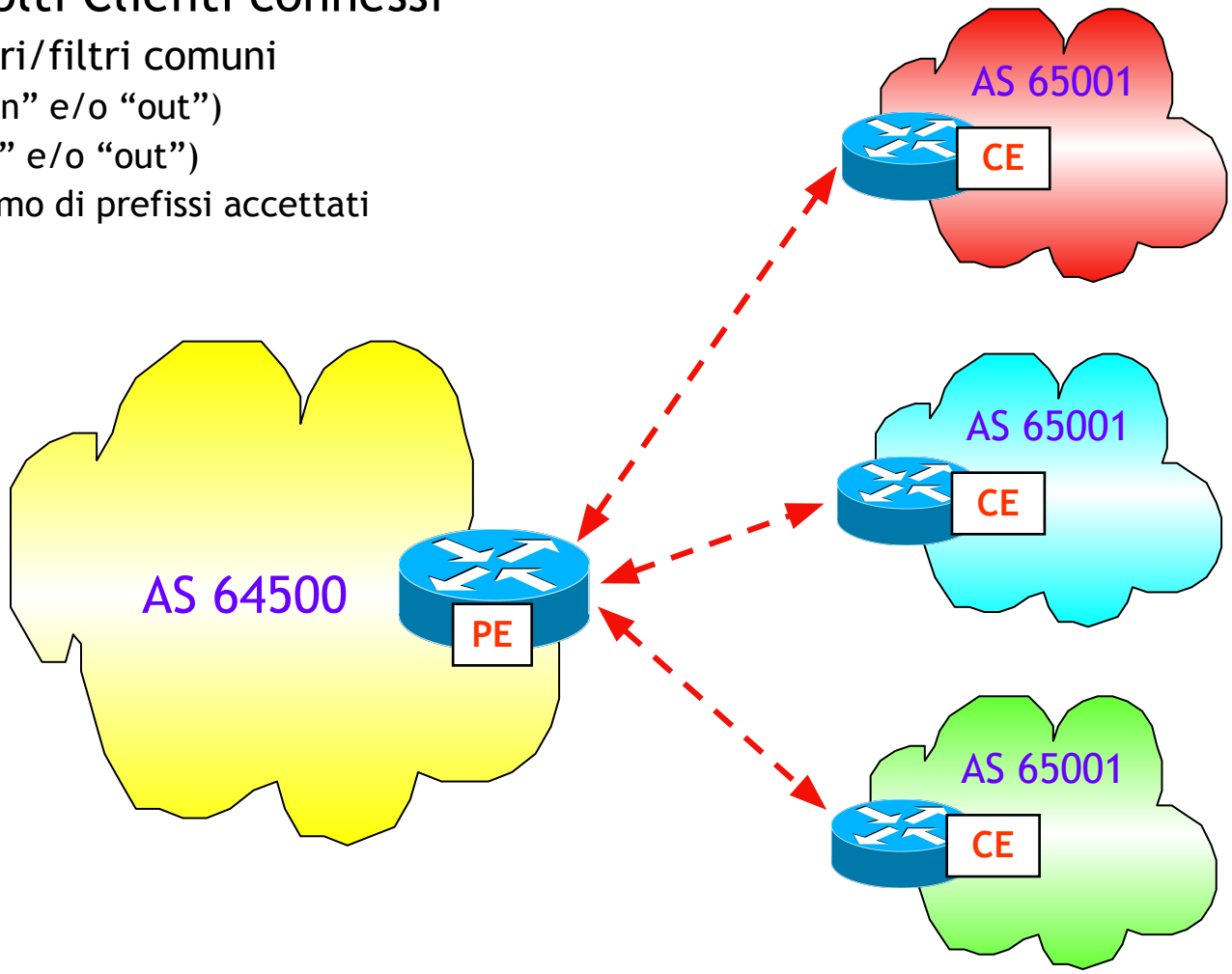
Definizione e caratteristiche (2/2)

- I *BGP peer group* hanno delle limitazioni legate al tipo di messaggio BGP UPDATE creato per le sessioni
  - I parametri per *BGP peer* per gli annunci *outbound* degli appartenenti al *BGP peer group* non possono essere modificati
    - NOTA: è possibile sovrascrivere i singoli parametri per gli annunci *inbound* degli appartenenti al *BGP peer group* tramite configurazione specifica del *neighbor*
  - Un *BGP peer group* non può contenere contemporaneamente *iBGP* ed *eBGP peer*

- I parametri comuni configurabili includono
  - Filtri (*filter-list*, *distribute-list*, *route-map*)
  - *Weight*
  - *Password* MD5
  - Sessioni *eBGP-multihop*
  - Indirizzo IP sorgente per le sessioni TCP
  - Propagazione degli attributi C*ommunity*

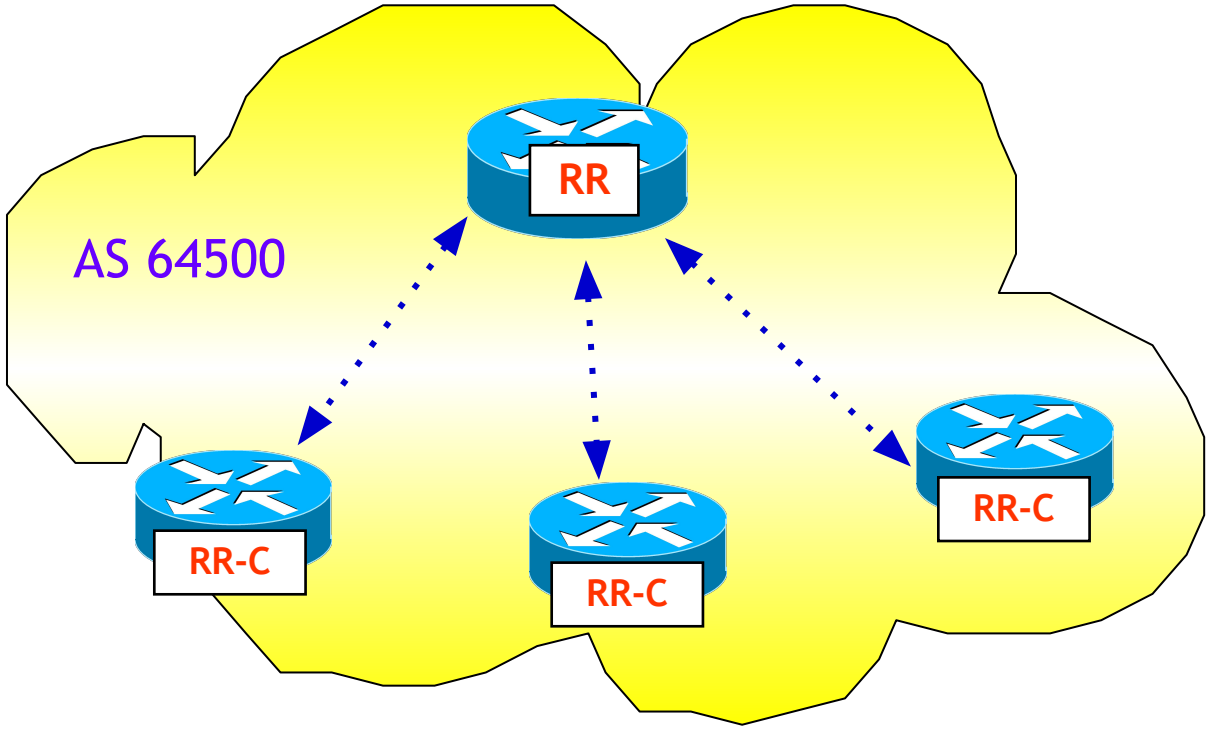
Scenari di applicazione (1/3)

- Router PE con molti Clienti connessi
  - Possibili parametri/filtri comuni
    - *Route-map* ("in" e/o "out")
    - *Filter-list* ("in" e/o "out")
    - Numero massimo di prefissi accettati
    - . . .



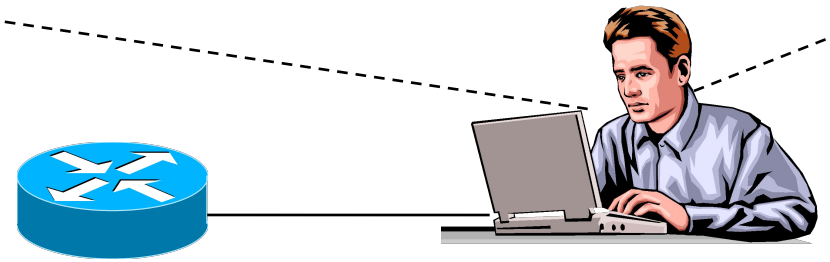
Scenari di applicazione (2/3)

- *Route Reflector* con molti *RR-client*
  - Possibili parametri/filtri comuni
    - Numero AS
    - IP sorgente della connessione TCP
    - *Password* MD5
    - . . .



Configurazione

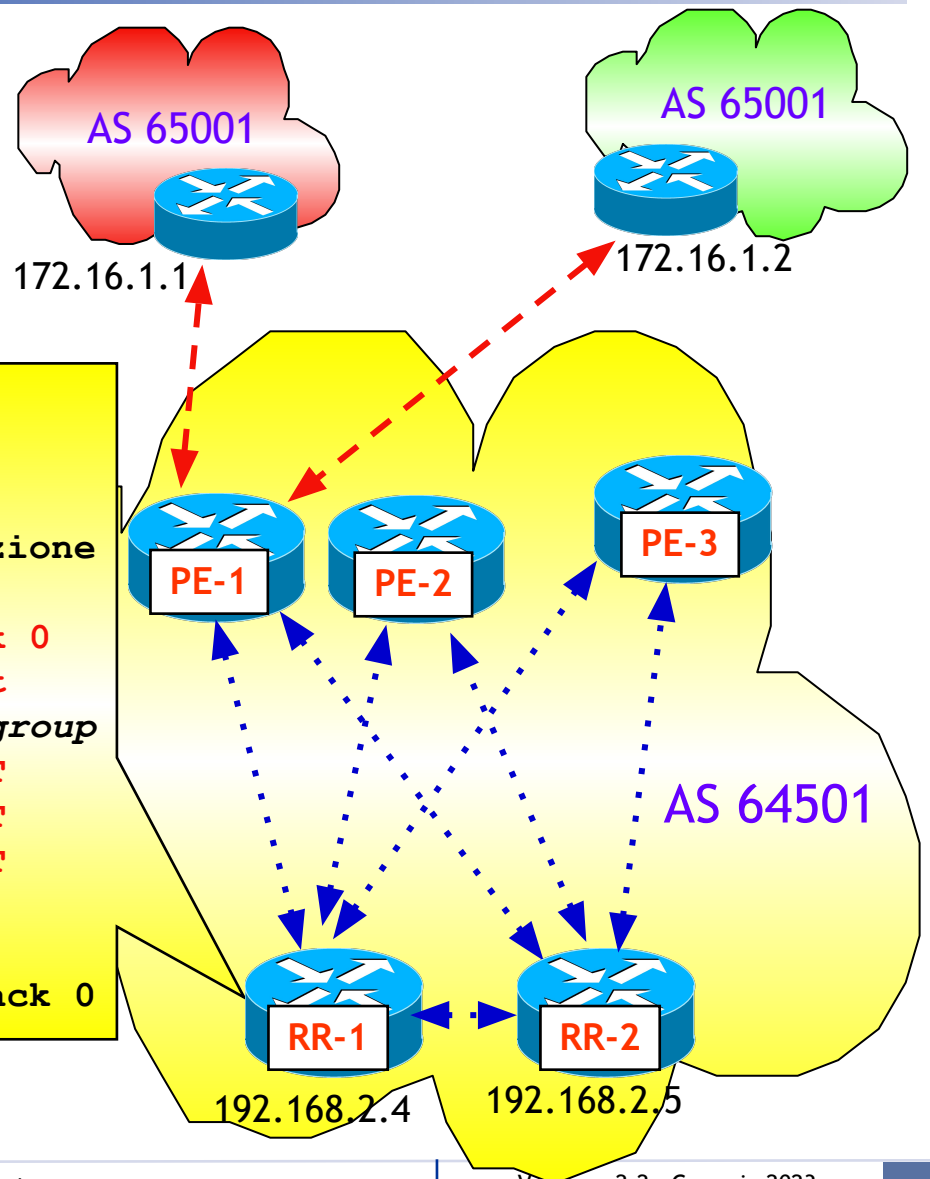
```
router(config)# router bgp numero-AS
router(config-router)# neighbor nome-peer-group peer-group
router(config-router)# neighbor nome-peer-group <parametro/filtro>
...
router(config-router)# neighbor nome-peer-group <parametro/filtro>
router(config-router)# neighbor indirizzo-IP peer-group nome-peer-group
...
router(config-router)# neighbor indirizzo-IP peer-group nome-peer-group
```



- La configurazione di un *BGP peer group* nell'IOS e IOS XE richiede i seguenti passi:
  1. Assegnare un nome al *BGP peer group*
  2. Definire il *template* di configurazione (parametri, filtri, ecc.)
  3. Definire i *neighbor* appartenenti al *BGP peer group*

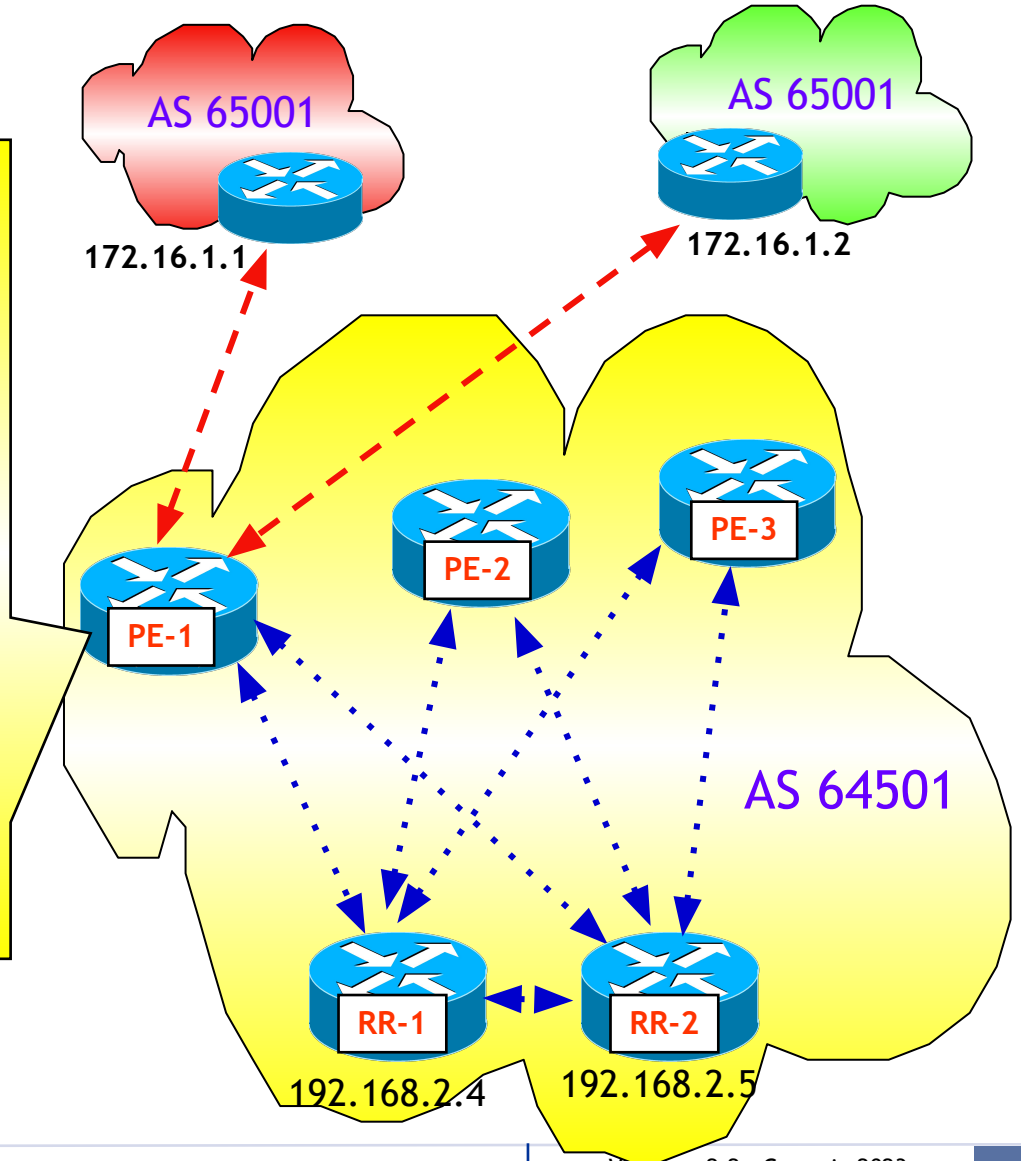
Esempio (1/4)

```
router bgp 64501
! Assegnazione del nome al BGP peer-group
 neighbor RR-CLIENT peer-group
! Definizione del "template" di configurazione
 neighbor RR-CLIENT remote-as 64501
 neighbor RR-CLIENT update-source loopback 0
 neighbor RR-CLIENT route-reflector-client
 ! Definizione dei BGP peer del BGP peer-group
 neighbor 192.168.0.1 peer-group RR-CLIENT
 neighbor 192.168.0.2 peer-group RR-CLIENT
 neighbor 192.168.0.3 peer-group RR-CLIENT
 ! Sessione iBGP verso RR-2
 neighbor 192.168.2.5 remote-as 64501
 neighbor 192.168.2.5 update-source loopback 0
```



Esempio (2/4)

```
router bgp 64501
 neighbor RR peer-group
 neighbor RR remote-as 64501
 neighbor RR update-source loopback 0
 neighbor RR next-hop-self
 !
 neighbor 192.168.2.4 peer-group RR
 neighbor 192.168.2.5 peer-group RR
 !
 neighbor EBGP peer-group
 neighbor EBGP remote-as 65001
 neighbor EBGP route-map SETMED out
 neighbor EBGP filter-list 1 out
 neighbor EBGP filter-list 2 in
 !
 neighbor 172.16.1.1 peer-group EBGP
 neighbor 172.16.1.2 peer-group EBGP
```



Esempio (3/4)

```
PE-1# show bgp ipv4 unicast peer-group
BGP peer-group is RR,  remote AS 64501
  BGP version 4
  Default minimum time between advertisement runs is 5 seconds
 For address family: IPv4 Unicast
  BGP neighbor is RR, peer-group internal, members:
  192.168.2.4 192.168.2.5
  Index 1, Offset 0, Mask 0x2
  NEXT_HOP is always this router
  Update messages formatted 3, replicated 3
  Number of NLRIs in the update sent: max 0, min 0

BGP peer-group is EBGP
  BGP version 4
  Default minimum time between advertisement runs is 30 seconds
 For address family: IPv4 Unicast
  BGP neighbor is EBGP, peer-group external, members:
  172.16.1.1 172.16.1.2
  Index 2, Offset 0, Mask 0x4
  Incoming update AS path filter list is 2
  Outgoing update AS path filter list is 1
  Route map for outgoing advertisements is SETMED
  Update messages formatted 0, replicated 0
  Number of NLRIs in the update sent: max 0, min 0
```

Esempio (4/4)

```
RR-1# show bgp ipv4 unicast peer-group RR-CLIENT summary
BGP router identifier 192.168.2.4, local AS number 64501
BGP table version is 1, main routing table version 1

Neighbor        V     AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd

192.168.0.1     4  64501       5       5        1    0    0 00:01:47        0
192.168.0.2     4  64501       6       6        1    0    0 00:01:20        0
192.168.0.3     4  64501      12       8        1    0    0 00:00:31        0



RR-1# show bgp ipv4 unicast peer-group RR-CLIENT
BGP peer-group is RR-CLIENT,  remote AS 64501
  BGP version 4
  Default minimum time between advertisement runs is 5 seconds

 For address family: IPv4 Unicast
  BGP neighbor is RR-CLIENT, peer-group internal, members:
     192.168.0.1    192.168.0.2    192.168.0.3
  Index 1, Offset 0, Mask 0x2
  Route-Reflector Client
  Update messages formatted 0, replicated 0
  Number of NLRIs in the update sent: max 0, min 0
```

BGP Update Groups

- I *BGP Update Group* disaccoppiano la configurazione dei *BGP neighbor* dalla generazione dei messaggi BGP UPDATE
  - L'individuazione dei *BGP peer-group* avviene automaticamente
  - Non hanno bisogno di configurazione

- Poiché la generazione dei messaggi BGP UPDATE viene ottimizzata automaticamente, rendono obsoleti i *BGP peer group* classici
  - È possibile quindi pensare a tecniche di configurazione più flessibili che abbiano meno vincoli dei *BGP peer group*

- Disponibili in tutte le varie versioni dell'IOS (IOS, IOS XE e IOS XR)
  - Introdotti a partire dalla versione IOS 12.0(24)S

Scalabilità della configurazione

BGP peer groups
BGP peer templates
BGP configuration templates

Definizione

- I *BGP peer template* sono blocchi di comandi comuni che possono essere applicati a *BGP neighbor* che condividono le stesse politiche
  - Sono riutilizzabili e supportano caratteristiche di «ereditarietà» ossia la possibilità di creare *Peer Template* più generali, da inserire in *Peer Template* più specifici, creando di fatto dei *Peer Template* nidificati (gerarchici)

- Due tipi di *Peer Template*
  - *Peer Session Template:* utilizzati per creare *Template* di configurazione che raggruppano tutti i comandi a livello di sessione
  - *Peer Policy Template:* utilizzati per creare *Template* di configurazione che raggruppano tutti i comandi relativi alle politiche di routing

Ereditarietà e precedenza

- Regole di ereditarietà
  - Un *peer session template* può ereditare la configurazione di un altro *peer session template*
  - Un *peer policy template* può ereditare la configurazione di un altro *peer policy template*
  - Un *BGP neighbor* può ereditare la configurazione di un *peer policy template* e/o *peer session template*
- Regola di precedenza
  - I comandi impartiti a livello più specifico hanno ragione sugli stessi comandi dati a livello più generale

Configurazione

REISS ROMOLI

Creazione *peer session template*

```
router(config)# router bgp numero-AS
router(config-router)# template peer-session nome-peer-session-template
router(config-router-stmp)# comandi di sessione
router(config-router-stmp)# <inherit peer-session nome-peer-session-template>
router(config-router-stmp)# exit-peer-session
!
router(config-router)# template peer-policy nome-peer-policy-template
router(config-router-ptmp)# comandi politiche di routing
router(config-router-ptmp)# <inherit peer-policy nome-peer-policy-template
                                                    numero-sequenza>
...
router(config-router-ptmp)# exit-peer-policy
!
router(config-router)# neighbor indirizzo-IP-peer
                          inherit peer-session nome-peer-session-template

router(config-router)# neighbor indirizzo-IP-peer
                        inherit peer-policy nome-peer-policy-template
```

Creazione *peer policy template*

Applicazione *peer templates*

Esempio (1/4)

			AS 64501	AS 64502	AS 64503	AS 64504	AS 64505
Session	Versione BGP e *Timer*				X		
	Sessione eBGP standard				X		X
	Sessione eBGP *Multi-hop*		X	X		X	
Policy	*Outbound*	Solo *default route*	X	X			
		default route + prefissi locali AS 64500			X	X	
		Full Internet Routing Table					X
	Inbound	Accettare dai Clienti al massimo 5 prefissi			X		

Esempio (2/4)

- Definizione dei *peer session template*

```
router bgp 64500
 template peer-session PARAM-COMUNI
   version 4
   timers 30 90
 exit-peer-session
!
 template peer-session EBGP-MULTIHOP
   ebgp-multihop 2
   update-source Loopback0
   inherit peer-session PARAM-COMUNI
 exit-peer-session
```

Esempio (3/4)

- Definizione dei *peer policy template*

```
 template peer-policy POLITICHE-INBOUND
   filter-list 10 in
   maximum-prefix 5
 exit-peer-policy
!
 template peer-policy DEFAULT-ONLY
   default-originate
   prefix-list SOLODEFAULT out
   inherit peer-policy POLITICHE-INBOUND 10
 exit-peer-policy
!
 template peer-policy DEFAULT-AND-LOC
   filter-list 20 out
   default-originate
   inherit peer-policy POLITICHE-INBOUND 10
 exit-peer-policy
!
 template peer-policy FRT
   prefix-list ANY out
   inherit peer-policy POLITICHE-INBOUND 10
 exit-peer-policy
```

Esempio (4/4)

- Applicazione dei *peer session template* e *peer policy template* ai *BGP peer*

```
 neighbor 1.1.1.1 remote-as 64501
 neighbor 1.1.1.1 inherit peer-session EBGP-MULTIHOP
 neighbor 1.1.1.1 inherit peer-policy DEFAULT-ONLY
!
 neighbor 2.2.2.2 remote-as 64502
 neighbor 2.2.2.2 inherit peer-session EBGP-MULTIHOP
 neighbor 2.2.2.2 inherit peer-policy DEFAULT-ONLY
!
 neighbor 3.3.3.3 remote-as 64503
 neighbor 3.3.3.3 inherit peer-session PARAM-COMUNI
 neighbor 3.3.3.3 inherit peer-policy DEFAULT-AND-LOC
!
 neighbor 4.4.4.4 remote-as 64504
 neighbor 4.4.4.4 inherit peer-session EBGP-MULTIHOP
 neighbor 4.4.4.4 inherit peer-policy DEFAULT-AND-LOC
!
 neighbor 5.5.5.5 remote-as 64505
 neighbor 5.5.5.5 inherit peer-session PARAM-COMUNI
 neighbor 5.5.5.5 inherit peer-policy FIRT
```

Scalabilità della configurazione

BGP peer groups
BGP peer templates
BGP configuration templates

Definizione

- I *BGP configuration template* sono basati su tre tipi di *template* di configurazione
  - *Address-family group*: raggruppano comandi specifici di una determinata *address-family*
  - *Session group*: raggruppano comandi indipendenti dalle *address-family*
  - *Neighbor group*: raggruppano comandi dipendenti e indipendenti dalle *address-family*, comuni a uno o più *BGP peer*

- Stile di configurazione disponibile solo nell'IOS XR

Ereditarietà e precedenza

- Regole di ereditarietà per configurazioni indipendenti dalle *address-family*
  - Un *session group* può ereditare la configurazione di un altro *session group*
  - Un *neighbor group* può ereditare la configurazione di *session group e/o altri neighbor group*
  - Un *neighbor* può ereditare la configurazione di un *session group e/o neighbor group*

- Regole di ereditarietà per configurazioni dipendenti dalle *address-family*
  - Un *address-family group* può ereditare la configurazione di un altro *address-family group*
  - Un *neighbor group* può ereditare la configurazione di un *address-family group* e/o *altri neighbor group*
  - Un *neighbor* può ereditare la configurazione di un *address-family group* e/o *neighbor group*

Configurazione

- Configurazione di un *address-family group*

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# af-group nome-af-group address-family
                          {ipv4 | ipv6 | vpnv4 | vpnv6 } {unicast | multicast}
RP/0/RP0/CPU0:router(config-bgp-afgrp)# . . .
```

- Configurazione di un *session group*

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# session-group nome-session-group
RP/0/RP0/CPU0:router(config-bgp-sngrp)# . . .
```

- Configurazione di un *neighbor group*

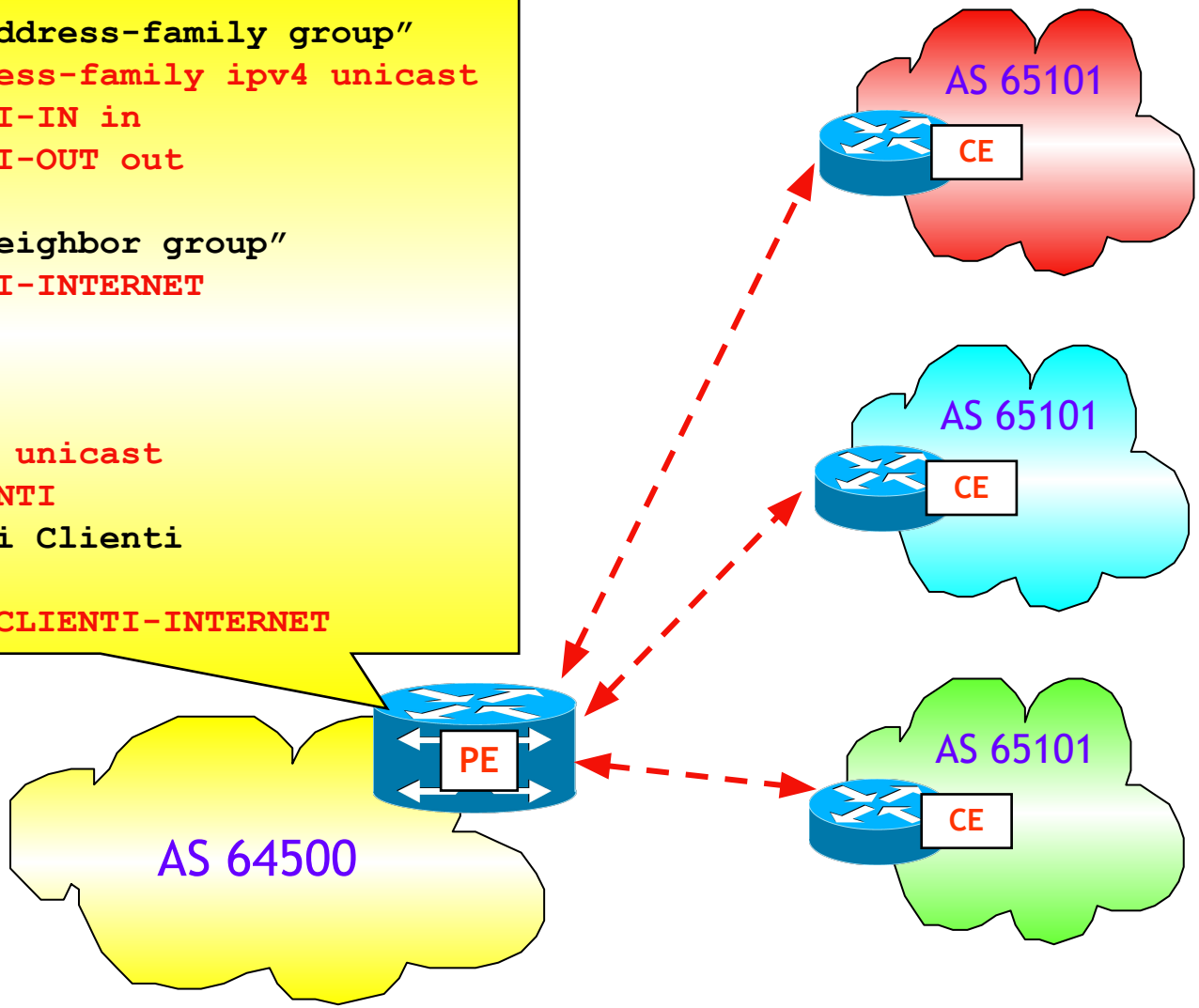
```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor-group nome-neighbor-group
RP/0/RP0/CPU0:router(config-bgp-nbrgrp)# . . .
```

- Applicazione di un *address-family/session/neighbor group*

```
RP/0/RP0/CPU0:router(config)# router bgp numero-AS
RP/0/RP0/CPU0:router(config-bgp)# neighbor indirizzo-IP-peer
RP/0/RP0/CPU0:router(config-bgp-nbr)# use {session-group | neighbor-group} nome
RP/0/RP0/CPU0:router(config-bgp-nbr)# address-family ...
RP/0/RP0/CPU0:router(config-bgp-nbr-af)# use af-group nome-af-group
```

Esempio (1/3)

```
router bgp 64500
! Definizione di un "address-family group"
 af-group CLIENTI address-family ipv4 unicast
   route-policy CLIENTI-IN in
   route-policy CLIENTI-OUT out
   maximum-prefix 5
! Definizione di un "neighbor group"
 neighbor-group CLIENTI-INTERNET
   remote-as 65101
   password SSGRR
   ttl-security
   address-family ipv4 unicast
     use af-group CLIENTI
! Sessioni eBGP verso i Clienti
 neighbor 10.1.1.2
   use neighbor-group CLIENTI-INTERNET
```



Esempio (2/3)

```
RP/0/RP0/CPU0:PE# show bgp af-group CLIENTI configuration
af-group CLIENTI address-family ipv4 unicast
  maximum-prefix 5 75             []
  policy CLIENTI-IN in            []
  policy CLIENTI-OUT out          []
```

- Verifica della configurazione dell'*address-family group* CLIENTI

```
RP/0/RP0/CPU0:PE# show bgp af-group CLIENTI users
IPv4 unicast: 10.1.1.2 n:CLIENTI-INTERNET
```

- Visualizza i *neighbor*, *neighbor group* e *address-family group* che ereditano la configurazione dell'*address-family group* CLIENTI

```
RP/0/RP0/CPU0:PE# show bgp neighbor-group CLIENTI-INTERNET configuration
neighbor-group CLIENTI-INTERNET
  password encrypted 05383528137E   []
  remote-as 65101                 []
  ttl-security                    []
  address-family ipv4 unicast
    maximum-prefix 5 75           [a:CLIENTI]
    policy CLIENTI-IN in          [a:CLIENTI]
    policy CLIENTI-OUT out        [a:CLIENTI]
```

- Verifica della configurazione del *neighbor group* CLIENTI-INTERNET

Esempio (3/3)

```
RP/0/RP0/CPU0:PE# show bgp neighbor-group CLIENTI-INTERNET users
Session:        10.1.1.2
IPv4 unicast: 10.1.1.2
```

- Visualizza i *neighbor* e *neighbor group* che ereditano la configurazione del *neighbor group* CLIENTI-INTERNET

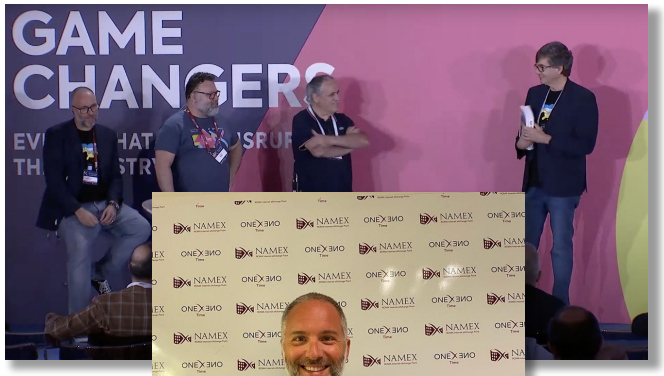
```
RP/0/RP0/CPU0:PE# show bgp neighbors 10.1.1.2 configuration
neighbor 10.1.1.2
  password encrypted 05383528137E   [n:CLIENTI-INTERNET]
  remote-as 65101             [n:CLIENTI-INTERNET]
  ttl-security                [n:CLIENTI-INTERNET]
  address-family ipv4 unicast  [n:CLIENTI-INTERNET]
    maximum-prefix 5 75        [n:CLIENTI-INTERNET a:CLIENTI]
    policy CLIENTI-IN in       [n:CLIENTI-INTERNET a:CLIENTI]
    policy CLIENTI-OUT out     [n:CLIENTI-INTERNET a:CLIENTI]
```

- Visualizza l'effettiva configurazione del «**neighbor 10.1.1.2**», inclusi i gruppi da dove i comandi sono stati ereditati

Ultima Diapositiva (finalmente …)

REISS ROMOLI

Per saperne di più …



https://book.reissromoli.com/

Per uno sconto del 10% utilizza
il codice promozionale **bgp-2022**