Configuring Route Policy Manager

This chapter describes how to configure the Route Policy Manager on the Cisco NX-OS device.

This chapter includes the following sections:

About Route Policy Manager

Route Policy Manager supports route maps and IP prefix lists. These features are used for route redistribution. A prefix list contains one or more IPv4 or IPv6 network prefixes and the associated prefix length values. You can use a prefix list by itself in features such as Border Gateway Protocol (BGP) templates, route filtering, or redistribution of routes that are exchanged between routing domains.

Route maps can apply to both routes and IP packets. Route filtering and redistribution pass a route through a route map.

This section includes the following topics:

Prefix Lists

You can use prefix lists to permit or deny an address or range of addresses. Filtering by a prefix list involves matching the prefixes of routes or packets with the prefixes listed in the prefix list. An implicit deny is assumed if a given prefix does not match any entries in a prefix list.

You can configure multiple entries in a prefix list and permit or deny the prefixes that match the entry. Each entry has an associated sequence number that you can configure. If you do not configure a sequence number, Cisco NX-OS assigns a sequence number automatically. Cisco NX-OS evaluates prefix lists starting with the lowest sequence number. Cisco NX-OS processes the first successful match for a given prefix. Once a match occurs, Cisco NX-OS processes the permit or deny statement and does not evaluate the rest of the prefix list.

Note An empty prefix list permits all routes.

Prefix List Masks

Cisco NX-OS Release 7.0(3)I4(1) introduces masks for prefix lists. Masking uses the number 1 and the number 0 to specify how the software treats the corresponding IP address bits.

- A mask bit 0 means ignore the corresponding bit value.
- A mask bit 1 means check the corresponding bit value for an exact match.

You can use a prefix list to match the IP address in a route map, which in turn is used in routing protocols during redistribution. The IP address is matched against the prefix list where the bits corresponding to the mask bit 1 are the same as the subnet provided in the prefix list.

By carefully setting masks, you can select one or several IP addresses for permit or deny tests.

The prefix list mask allows noncontiguous bits in the mask. You can thus define a range of even- or odd-numbered IP addresses.

Route Maps

You can use route maps for route redistribution. Route map entries consist of a list of match and set criteria. The match criteria specify match conditions for incoming routes or packets, and the set criteria specify the action taken if the match criteria are met.

You can configure multiple entries in the same route map. These entries contain the same route map name and are differentiated by a sequence number.

You create a route map with one or more route map entries arranged by the sequence number under a unique route map name. The route map entry has the following parameters:

- Sequence number
- Permission—permit or deny
- Match criteria
- Set changes

By default, a route map processes routes or IP packets in a linear fashion, that is, starting from the lowest sequence number. You can configure the route map to process in a different order using the **continue** statement, which allows you to determine which route map entry to process next.

Match Criteria

You can use a variety of criteria to match a route or IP packet in a route map. Some criteria, such as BGP community lists, are applicable only to a specific routing protocol, while other criteria, such as the IP source or the destination address, can be used for any route or IP packet.

When Cisco NX-OS processes a route or packet through a route map, it compares the route or packet to each of the match statements configured. If the route or packet matches the configured criteria, Cisco NX-OS processes it based on the permit or deny configuration for that match entry in the route map and any set criteria configured.

The match categories and parameters are as follows:

- BGP parameters—Match based on AS numbers, AS-path, community attributes, or extended community attributes.
- Prefix lists—Match based on an address or range of addresses.
- Multicast parameters—Match based on a rendezvous point, groups, or sources.
- Other parameters—Match based on an IP next-hop address or packet length.

Set Changes

Once a route or packet matches an entry in a route map, the route or packet can be changed based on one or more configured set statements.

The set changes are as follows:

- BGP parameters—Change the AS-path, community, extended community, dampening, local preference, origin, or weight attributes.
- Metrics—Change the route-metric, the route-tag, or the route-type.
- Other parameters—Change the forwarding address or the IP next-hop address.

Access Lists

IP access lists can match the packet to a number of IP packet fields such as the following:

- Source or destination IPv4 or IPv6 address
- Protocol
- Precedence
- ToS

AS Numbers for BGP

You can configure a list of AS numbers to match against BGP peers. If a BGP peer matches an AS number in the list and matches the other BGP peer configuration, BGP creates a session. If the BGP peer does not match an AS number in the list, BGP ignores the peer. You can configure the AS numbers as a list, a range of AS numbers, or you can use an AS-path list to compare the AS numbers against a regular expression.

AS-Path Lists for BGP

You can configure an AS-path list to filter inbound or outbound BGP route updates. If the route update contains an AS-path attribute that matches an entry in the AS-path list, the router processes the route based on the permit or deny condition configured. You can configure AS-path lists within a route map.

You can configure multiple AS-path entries in an AS-path list by using the same AS-path list name. The router processes the first entry that matches.

Community Lists for BGP

You can filter BGP route updates based on the BGP community attribute by using community lists in a route map. You can match the community attribute based on a community list, and you can set the community attribute using a route map.

A community list contains one or more community attributes. If you configure more than one community attribute in the same community list entry, the BGP route must match all community attributes listed to be considered a match.

You can also configure multiple community attributes as individual entries in the community list by using the same community list name. In this case, the router processes the first community attribute that matches the BGP route, using the permit or deny configuration for that entry.

You can configure community attributes in the community list in one of the following formats:

- A named community attribute, such as **internet** or **no-export.**
- In *aa:nn* format, where the first two bytes represent the two-byte AS number and the last two bytes represent a user-defined network number.
- A regular expression.

Extended Community Lists for BGP

Extended community lists support 4-byte AS numbers. You can configure community attributes in the extended community list in one of the following formats:

- In *aa4:nn* format, where the first four bytes represent the four-byte AS number and the last two bytes represent a a user-defined network number.
- A regular expression.

Cisco NX-OS supports generic specific extended community lists, which provide similar functionality to regular community lists for four-byte AS numbers. You can configure generic specific extended community lists with the following properties:

- Transitive—BGP propagates the community attributes across autonomous systems.
- Nontransitive—BGP removes community attributes before propagating the route to another autonomous system.

Route Redistribution and Route Maps

You can use route maps to control the redistribution of routes between routing domains. Route maps match on the attributes of the routes to redistribute only those routes that pass the match criteria. The route map can also modify the route attributes during this redistribution using the set changes.

The router matches redistributed routes against each route map entry. If there are multiple match statements, the route must pass all of the match criteria. If a route passes the match criteria defined in a route map entry, the actions defined in the entry are executed. If the route does not match the criteria, the router compares the route against subsequent route map entries. Route processing continues until a match is made or the route is processed by all entries in the route map with no match. If the router processes the route against all entries in a route map with no match, the router accepts the route (inbound route maps) or forwards the route (outbound route maps).

Note When you redistribute BGP to IGP, iBGP is redistributed as well. To override this behavior, you must insert an additional deny statement into the route map.

Licensing Requirements for Route Policy Manager

The following table shows the licensing requirements for this feature:

Product	License Requirement
Cisco NX-OS	Route Policy Manager requires no license. Any feature not included in a license package is bundled with the nx-os image and is provided at no extra charge to you. For a complete explanation of the Cisco NX-OS licensing scheme, see the *Cisco NX-OS Licensing Guide*.

Guidelines and Limitations

Route Policy Manager has the following configuration guidelines and limitations:

- An empty route map denies all the routes.
- An empty prefix list permits all the routes.
- Without any match statement in a route-map entry, the permission (permit or deny) of the route-map entry decides the result for all the routes or packets.
- If referred policies (for example, prefix lists) within a match statement of a route-map entry return either a no-match or a deny-match, Cisco NX-OS fails the match statement and processes the next route-map entry.
- When you change a route map, Cisco NX-OS holds all the changes until you exit from the route-map configuration submode. Cisco NX-OS then sends all the changes to the protocol clients to take effect.
- Cisco recommends that you do not have both IPv4 and IPv6 match statements in the same route-map sequence. If both are required, they should be specified in different sequences in the same route-map.
- Because you can use a route map before you define it, verify that all your route maps exist when you finish a configuration change.
- You can view the route-map usage for redistribution and filtering. Each individual routing protocol provides a way to display these statistics.
- When you redistribute BGP to IGP, iBGP is redistributed as well. To override this behavior, you must insert an additional deny statement into the route map.
- Route Policy Manager does not support MAC lists.

Default Settings

Table 15-1 lists the default settings for Route Policy Manager.

Table 15-1 Default Route Policy Manager Parameters

Parameters	Default
Route Policy Manager	Enabled
Administrative distance	115

Configuring Route Policy Manager

This section includes the following topics:

Note If you are familiar with the Cisco IOS CLI, be aware that the Cisco NX-OS commands for this feature might differ from the Cisco IOS commands that you would use.

Configuring IP Prefix Lists

IP prefix lists match the IP packet or route against a list of prefixes and prefix lengths. You can create an IP prefix list for IPv4 and create an IPv6 prefix list for IPv6.

You can configure the prefix list entry to match the prefix length exactly or to match any prefix with a length that matches the configured range of prefix lengths.

Use the **ge** and **lt** keywords to create a range of possible prefix lengths. The incoming packet or route matches the prefix list if the prefix matches and if the prefix length is greater than or equal to the **ge** keyword value (if configured) and less than or equal to the **lt** keyword value (if configured).

Use the **mask** keyword to define a range of possible contiguous or non-contiguous routes to be compared to the prefix address.

SUMMARY STEPS

1. **configure terminal**
2. {**ip** | **ipv6**} **prefix-list** *name* **description** *string*
3. **ip prefix-list** *name* [**seq** *number*] [{**permit** | **deny**} *prefix* {[**eq** *prefix-length*] | [**ge** *prefix-length*] [**le** *prefix-length*]}] [**mask** *mask*]

   or

   ipv6 prefix-list *name* [**seq** *number*] [{**permit** | **deny**} *prefix* **{[eq** *prefix-length*] | [**ge** *prefix-length*] [**le** *prefix-length*]}] [**mask** *mask*]
4. (Optional) **show** {**ip** | **ipv6**} **prefix-list** *name*
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	{**ip** | **ipv6**} **prefix-list** *name* **description** *string* **Example**: switch(config)# ip prefix-list AllowPrefix description allows engineering server	(Optional) Adds an information string about the prefix list.
Step 3	**ip prefix-list** *name* [**seq** *number*] [{**permit** | **deny**} *prefix* {[**eq** *prefix-length*] | [**ge** *prefix-length*] [**le** *prefix-length*]}] [**mask** *mask*] **Example:** switch(config)# ip prefix-list AllowPrefix seq 10 permit 192.0.2.0/24 eq 24 **Example:** switch(config)# ip prefix-list even permit 0.0.0.0/32 mask 0.0.0.1	Creates an IPv4 prefix list or adds a prefix to an existing prefix list. The prefix length is matched as follows: • eq—Matches the exact *prefix length*. • ge—Matches a prefix length that is equal to or greater than the configured *prefix length*. • le—Matches a prefix length that is equal to or less than the configured *prefix length*. • mask—Specifies the bits of a prefix address in a prefix list that are compared to the bits of the prefix address used in routing protocols during redistribution.
	ipv6 prefix-list *name* [**seq** *number*] [{**permit** | **deny**} *prefix* {[**eq** *prefix-length*] | [**ge** *prefix-length*] [**le** *prefix-length*]}] [**mask** *mask*] **Example:** switch(config)# ipv6 prefix-list AllowIPv6Prefix seq 10 permit 2001:0DB8:: le 32	Creates an IPv6 prefix list or adds a prefix to an existing prefix list. The prefix length is configured as follows: • eq—Matches the exact *prefix length*. • ge—Matches a prefix length that is equal to or greater than the configured *prefix length*. • le—Matches a prefix length that is equal to or less than the configured *prefix length*. • mask—Specifies the bits of a prefix address in a prefix list that are compared to the bits of the prefix address used in routing protocols during redistribution.
Step 4	**show** {**ip** | **ipv6**} **prefix-list** *name* **Example**: switch(config)# show ip prefix-list AllowPrefix	(Optional) Displays information about prefix lists.
Step 5	**copy running-config startup-config** **Example:** switch# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to create an IPv4 prefix list with two entries and apply the prefix list to a BGP neighbor:

```
switch# configure terminal
switch(config)# ip prefix-list allowprefix seq 10 permit 192.0.2.0/24 eq 24
switch(config)# ip prefix-list allowprefix seq 20 permit 209.165.201.0/27 eq 27
switch(config)# router bgp 65535
switch(config-router)# neighbor 192.0.2.1/16 remote-as 65534
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# prefix-list allowprefix in
```

This example shows how to create an IPv4 prefix list with a match mask for all /24 odd IP addresses:

```
switch# configure terminal
switch(config)# ip prefix-list list1 seq 7 permit 22.1.1.0/24 mask 255.255.1.0
switch(config)# show route-map test
route-map test, permit, sequence 7
  Match clauses:
    ip address prefix-lists: list1
  Set clauses:
    extcommunity COST:igp:10:20
switch(config)# show ip prefix-list list1
ip prefix-list list1: 1 entries
   seq 7 permit 22.1.1.0/24 mask 255.255.1.0
```

This example shows how to create an IPv4 prefix list that matches all subnets of 21.1.0.0/16 where the subnet prefix is 17 or greater. Due to the **mask** option, only those incoming prefixes where the first bit in the third octet is unset (even) will be matched.

```
switch# configure terminal
switch(config)# ip prefix-list list1 seq 10 permit 21.1.0.0/16 ge 17 mask 255.255.1.0
```

Configuring AS-Path Lists

You can specify an AS-path list filter on both inbound and outbound BGP routes. Each filter is an access list based on regular expressions. If the regular expression matches the representation of the AS-path attribute of the route as an ASCII string, the permit or deny condition applies.

SUMMARY STEPS

1. **configure terminal**
2. **ip as-path access-list** *name* {**deny** | **permit**} *expression*
3. (Optional) **show** {**ip** | **ipv6**} **as-path list** *name*
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

	Command	Purpose
Step 2	**ip as-path access-list** *name* {**deny** | **permit**} *expression* **Example:** switch(config)# ip as-path access-list Allow40 permit 40	Creates a BGP AS-path list using a regular expression.
Step 3	**show** {**ip** | **ipv6**} **as-path-access-list** *name* **Example**: switch(config)# show ip as-path-access-list Allow40	(Optional) Displays information about as-path access lists.
Step 4	**copy running-config startup-config** **Example:** switch# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to create an AS-path list with two entries and apply the AS path list to a BGP neighbor:

```
switch# configure terminal
switch(config)# ip as-path access-list AllowAS permit 64510
switch(config)# ip as-path access-list AllowAS permit 64496
switch(config)# copy running-config startup-config
switch(config)# router bgp 65535:20
switch(config-router)# neighbor 192.0.2.1/16 remote-as 65535:20
switch(config-router-neighbor)# address-family ipv4 unicast
switch(config-router-neighbor-af)# filter-list AllowAS in
```

Configuring Community Lists

You can use community lists to filter BGP routes based on the community attribute. The community number consists of a 4-byte value in the *aa:nn* format. The first two bytes represent the autonomous system number, and the last two bytes represent a user-defined network number.

When you configure multiple values in the same community list statement, all community values must match to satisfy the community list filter. When you configure multiple values in separate community list statements, the first list that matches a condition is processed.

Use community lists in a match statement to filter BGP routes based on the community attribute.

SUMMARY STEPS

1. **configure terminal**
2. **ip community-list standard** *list-name* {**deny** | **permit**} [*community-list*] [**internet**] [**local-AS**] [**no-advertise**] [**no-export**]

   or

   ip community-list expanded *list-name* {**deny** | **permit**} *expression*
3. (Optional) **show ip community-list** *name*
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose
Step 1	**configure terminal** **Example:** switch# configure terminal switch(config)#	Enters global configuration mode.
Step 2	**ip community-list standard** *list-name* {**deny** | **permit**} [*community-list*] [**internet**] [**local-AS**] [**no-advertise**] [**no-export**] **Example:** switch(config)# ip community-list standard BGPCommunity permit no-advertise 65535:20	Creates a standard BGP community list. The *list-name* can be any case-sensitive, alphanumeric string up to 63 characters. The *community-list* can be one or more communities in the *aa:nn* format.
	ip community-list expanded *list-name* {**deny** | **permit**} *expression* **Example:** switch(config)# ip community-list expanded BGPComplex deny 50000:[0-9][0-9]_	Creates an expanded BGP community list using a regular expression.
Step 3	**show ip community-list** *name* **Example**: switch(config)# show ip community-list BGPCommunity	(Optional) Displays information about community lists.
Step 4	**copy running-config startup-config** **Example:** switch# copy running-config startup-config	(Optional) Saves this configuration change.

This example shows how to create a community list with two entries:

```
switch# configure terminal
switch(config)# ip community-list standard BGPCommunity permit no-advertise 65535:20
switch(config)# ip community-list standard BGPCommunity permit local-AS no-export
switch(config)# copy running-config startup-config
```

Configuring Extended Community Lists

You can use extended community lists to filter BGP routes based on the community attribute. The community number consists of a 6-byte value in the *aa4:nn* format. The first four bytes represent the autonomous system number, and the last two bytes represent a user-defined network number.

When you configure multiple values in the same extended community list statement, all extended community values must match to satisfy the extended community list filter. When you configure multiple values in separate extended community list statements, the first list that matches a condition is processed.

Use extended community lists in a match statement to filter BGP routes based on the extended community attribute.

SUMMARY STEPS

1. **configure terminal**
2. **ip extcommunity-list standard** *list-name* {**deny** | **permit**} **4bytegeneric** {**transitive** | **non-transitive**} *aa4:nn*

   or

   ip extcommunity-list expanded *list-name* {**deny** | **permit**} *expression*
3. (Optional) **show ip extcommunity-list** *name*
4. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**ip extcommunity-list standard** *list-name* {**deny** | **permit**} **4bytegeneric** {**transitive** | **nontransitive**} *community1* [*community2...*] **Example:** `switch(config)# ip extcommunity-list standard BGPExtCommunity permit 4bytegeneric transitive 65535:20`	Creates a standard BGP extended community list. The *community* can be one or more extended communities in the *aa4:nn* format.
	ip extcommunity-list expanded *list-name* {**deny** | **permit**} *expression* **Example:** `switch(config)# ip extcommunity-list expanded BGPExtComplex deny 1.5:[0-9][0-9]_`	Creates an expanded BGP extended community list using a regular expression.
Step 3	**show ip community-list** *name* **Example**: `switch(config)# show ip community-list BGPCommunity`	(Optional) Displays information about extended community lists.
Step 4	**copy running-config startup-config** **Example:** `switch# copy running-config startup-config`	(Optional) Saves this configuration change.

This example shows how to create a generic specific extended community list:

```
switch# configure terminal
switch(config)# ip extcommunity-list standard test1 permit 4bytegeneric transitive
65535:40 65535:60
switch(config)# copy running-config startup-config
```

Configuring Route Maps

You can use route maps for route redistribution or route filtering. Route maps can contain multiple match criteria and multiple set criteria.

Configuring a route map for BGP triggers an automatic soft clear or refresh of BGP neighbor sessions.

SUMMARY STEPS

1. **configure terminal**
2. **route-map** *map-name* [**permit** | **deny**] [*seq*]
3. (Optional) **continue** *seq*
4. (Optional) **exit**
5. (Optional) **copy running-config startup-config**

DETAILED STEPS

	Command	Purpose

Step 1	**configure terminal**
	
	Example:
	`switch# configure terminal`
	`switch(config)#`
		Enters global configuration mode.

Step 2	**route-map** *map-name* [**permit** | **deny**] [*seq*]
	
	Example:
	`switch(config)# route-map Testmap permit 10`
	`switch(config-route-map)#`
		Creates a route map or enters route-map configuration mode for an existing route map. Use *seq* to order the entries in a route map.

Step 3	**continue** *seq* **Example:** `switch(config-route-map)# continue 10`	(Optional) Determines what sequence statement to process next in the route map. Used only for filtering and redistribution.
Step 4	**exit** **Example:** `switch(config-route-map)# exit`	(Optional) Exits route-map configuration mode.
Step 5	**copy running-config startup-config** **Example:** `switch(config)# copy running-config startup-config`	(Optional) Saves this configuration change.

You can configure the following optional match parameters for route maps in route-map configuration mode:

Note The **default-information originate** command ignores **match** statements in the optional route map.

Command	Purpose
match as-path *name* [*name...*] **Example:** `switch(config-route-map)# match as-path Allow40`	Matches against one or more AS-path lists. Create the AS-path list with the **ip as-path access-list** command.
match as-number {*number* [*,number...*] | **as-path-list** *name* [*name...*]} **Example:** `switch(config-route-map)# match as-number 33,50-60`	Matches against one or more AS numbers or AS-path lists. Create the AS-path list with the **ip as-path access-list** command. The number range is from 1 to 65535. The AS-path list name can be any case-sensitive, alphanumeric string up to 63 characters.
match community *name* [*name...*][***exact-match***] **Example:** `switch(config-route-map)# match community BGPCommunity`	Matches against one or more community lists. Create the community list with the **ip community-list** command.
match extcommunity *name* [*name...*][***exact-match***] **Example:** `switch(config-route-map)# match extcommunity BGPextCommunity`	Matches against one or more extended community lists. Create the community list with the **ip extcommunity-list** command.
match interface *interface-type number* [*interface-type number...* **Example:** `switch(config-route-map)# match interface e 1/2`	Matches any routes that have their next hop out one of the configured interfaces. Use **?** to find a list of supported interface types.
match ip address prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ip address prefix-list AllowPrefix`	Matches against one or more IPv4 prefix lists. Use the **ip prefix-list** command to create the prefix list.
match ipv6 address prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ip address prefix-list AllowIPv6Prefix`	Matches against one or more IPv6 prefix lists. Use the **ipv6 prefix-list** command to create the prefix list.
match ip multicast [**source** *ipsource*] [[**group** *ipgroup*] [*rp iprp*]] **Example:** `switch(config-route-map)# match ip multicast rp 192.0.2.1`	Matches an IPv4 multicast packet based on the multicast source, group, or rendezvous point.

Command	Purpose
match ipv6 multicast [**source** *ipsource*] [[**group** *ipgroup*] [*rp iprp*]] **Example:** `switch(config-route-map)# match ip multicast source 2001:0DB8::1`	Matches an IPv6 multicast packet based on the multicast source, group, or rendezvous point.
match ip next-hop prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ip next-hop prefix-list AllowPrefix`	Matches the IPv4 next-hop address of a route to one or more IP prefix lists. Use the **ip prefix-list** command to create the prefix list.
match ipv6 next-hop prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ipv6 next-hop prefix-list AllowIPv6Prefix`	Matches the IPv6 next-hop address of a route to one or more IP prefix lists. Use the **ipv6 prefix-list** command to create the prefix list.
match ip route-source prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ip route-source prefix-list AllowPrefix`	Matches the IPv4 route source address of a route to one or more IP prefix lists. Use the **ip prefix-list** command to create the prefix list.
match ipv6 route-source prefix-list *name* [*name...*] **Example:** `switch(config-route-map)# match ipv6 route-source prefix-list AllowIPv6Prefix`	Matches the IPv6 route-source address of a route to one or more IP prefix lists. Use the **ipv6 prefix-list** command to create the prefix list.
match metric *value* [**+-** *deviation.*] [*value..*] **Example:** `switch(config-route-map)# match metric 50 + 10`	Matches the route metric against one or more metric values or value ranges. Use +- *deviation* argument to set a metric range. The route map matches any route metric that falls within the range: *value* - *deviation* to *value* + *deviation*.
match ospf-area *area-id* **Example:** `switch(config-route-map)# match ospf-area 1`	Matches the OSPFv2 or OSPFv3 area ID. The *area-id* range is from 0 to 4294967295.

Command	Purpose
match route-type *route-type* **Example:** `switch(config-route-map)# match route-type level 1 level 2`	Matches against a type of route. The *route-type* can be one or more of the following: • external—The external route (BGP, EIGRP, and OSPF type 1 or 2) • inter-area—The OSPF inter-area route • internal—The internal route (including the OSPF intra- or inter-area) • intra-area—The OSPF intra-area route • level-1—The IS-IS level 1 route • level-2—The IS-IS level 2 route • local—The locally generated route • nssa-external—The NSSA external route (OSPF type 1 or 2). • type-1—The OSPF external type 1 route • type-2—The OSPF external type 2 route
match tag *tagid* [*tagid...*] **Example:** `switch(config-route-map)# match tag 2`	Matches a route against one or more tags for filtering or redistribution.
match vlan *vlan-id* [*vlan-range*] **Example:** `switch(config-route-map)# match vlan 3, 5-10`	Matches against a VLAN.

You can configure the following optional set parameters for route maps in route-map configuration mode:

Command	Purpose
set as-path {**tag** | **prepend** {**last-as** *number* | *as-1* [*as-2...*]}} **Example:** `switch(config-route-map)# set as-path prepend 10 100 110`	Modifies an AS-path attribute for a BGP route. You can prepend the configured *number* of last AS numbers or a string of particular AS-path values (*as-1 as-2...as-n*).
set comm-list *name* **delete** **Example:** `switch(config-route-map)# set comm-list BGPCommunity delete`	Removes communities from the community attribute of an inbound or outbound BGP route update. Use the **ip community-list** command to create the community list.

Command	Purpose
set community {**none** | **additive** | **local-AS** | **no-advertise** | **no-export** | *community-1* [community-2...]} **Example:** switch(config-route-map)# set community local-AS	Sets the community attribute for a BGP route update. **Note** When you use both the **set community** and **set comm-list delete** commands in the same sequence of a route map attribute, the deletion operation is performed before the set operation. **Note** Use the **send-community** command in BGP neighbor address-family configuration mode to propagate BGP community attributes to BGP peers.
set dampening *halflife reuse suppress duration* **Example:** switch(config-route-map)# set dampening 30 1500 10000 120	Sets the following BGP route dampening parameters: • *halflife*—The range is from 1 to 45 minutes. The default is 15. • *reuse*—The range is from is 1 to 20000 seconds. The default is 750. • *suppress*—The range is from is 1 to 20000. The default is 2000. • *duration*—The range is from is 1 to 255 minutes. The default is 60.
set distance *value* **Example:** switch(config-route-map)# set distance 150	Sets the administrative distance of routes for OSPFv2 or OSPFv3. The range is from 1 to 255.
set extcomm-list *name* **delete** **Example:** switch(config-route-map)# set extcomm-list BGPextCommunity delete	Removes communities from the extended community attribute of an inbound or outbound BGP route update. Use the **ip extcommunity-list** command to create the extended community list.
set extcommunity 4byteas-generic {**transitive** | **nontransitive**} {**none** | **additive**] *community-1* [community-2...]} **Example:** switch(config-route-map)# set extcommunity generic transitive 1.0:30	Sets the extended community attribute for a BGP route update. **Note** When you use both the **set extcommunity** and **set extcomm-list delete** commands in the same sequence of a route map attribute, the deletion operation is performed before the set operation. **Note** Use the **send-community** command in BGP neighbor address-family configuration mode to propagate BGP extended community attributes to BGP peers.

Command	Purpose
set extcommunity cost *community-id1 cost* [**igp** | **pre-bestpath**] [*community-id2...*]} **Example:** switch(config-route-map)# set extcommunity cost 33 1.0:30	Sets the cost community attribute for a BGP route update. This attribute allows you to customize the BGP best-path selection process for a local autonomous system or confederation. The *community-id* range is from 0 to 255. The *cost* range is from 0 to 4294967295. The path with the lowest cost is preferred. For paths with equal cost, the path with the lowest community ID is preferred. The **igp** keyword compares the cost after the IGP cost comparison. The **pre-bestpath** keyword compares before all other steps in the bestpath algorithm.
set extcommunity rt *community-1* [**additive**] [community-2...]} **Example:** switch(config-route-map)# set extcommunity rt 1.0:30	Sets the extended community route target attribute for a BGP route update. The *community* value can be a 2-byte AS number:4-byte network number, a 4-byte AS number:2-byte network number, or an IP address:2-byte network number. Use the **additive** keyword to add a route target to an existing extended community route target attribute.
set forwarding-address **Example:** switch(config-route-map)# set forwarding-address	Sets the forwarding address for OSPF.
set ip next-hop unchanged **Example:** switch(config-route-map)# set ip next-hop unchanged	Specifies an unchanged next-hop IP address. This command is required for BGP IPv6-over-IPv4 peering.
set level {**backbone** | **level-1** | **level-1-2** | **level-2**} **Example:** switch(config-route-map)# set level backbone	Sets what area to import routes to for IS-IS. The options for IS-IS are level-1, level-1-2, or level-2. The default is level-1.
set local-preference *value* **Example:** switch(config-route-map)# set local-preference 4000	Sets the BGP local preference value. The range is from 0 to 4294967295.
set metric [**+** | **-**]*bandwidth-metric* **Example:** switch(config-route-map)# set metric +100	Adds or subtracts from the existing metric value. The metric is in Kb/s. The range is from 0 to 4294967295.

Command	Purpose
set metric *bandwidth* [*delay reliability load mtu*] **Example:** `switch(config-route-map)# set metric 33 44 100 200 1500`	Sets the route metric values. Metrics are as follows: • *metric0*—Bandwidth in Kb/s. The range is from 0 to 4294967295. • *metric1*—Delay in 10-microsecond units. • *metric2*—Reliability. The range is from 0 to 255 (100 percent reliable). • *metric3*—Loading. The range is from 1 to 255 (100 percent loaded). • *metric4*—MTU of the path. The range is from 1 to 16777215.
set metric-type {**external** | **internal** | **type-1** | **type-2**} **Example:** `switch(config-route-map)# set metric-type internal`	Sets the metric type for the destination routing protocol. The options are as follows: external—IS-IS external metric internal— IGP metric as the MED for BGP type-1—OSPF external type 1 metric type-2—OSPF external type 2 metric
set nssa-only **Example:** `switch(config-route-map)# set nssa-only`	Sets Type-7 LSA generated on ASBR with no P bit set. This prevents Type-7 to Type-5 LSA translation in OSPF.
set origin {**egp** *as-number* | **igp** | **incomplete**} **Example:** `switch(config-route-map)# set origin incomplete`	Sets the BGP origin attribute. The EGP *as-number* range is from 0 to 65535.
set weight *count* **Example:** `switch(config-route-map)# set weight 33`	Sets the weight for the BGP route. The range is from 0 to 65535.

The **set metric-type internal** command affects an outgoing policy and an eBGP neighbor only. If you configure both the **metric** and **metric-type internal** commands in the same BGP peer outgoing policy, Cisco NX-OS ignores the **metric-type internal** command.

Verifying the Route Policy Manager Configuration

To display route policy manager configuration information, perform one of the following tasks:

Command	Purpose
show ip community-list [*name*]	Displays information about a community list.
show ip extcommunity-list [*name*]	Displays information about an extended community list.

Command	Purpose
show [**ip** | **ipv6**] **prefix-list** [*name*]	Displays information about an IPv4 or IPv6 prefix list.
show route-map [*name*]	Displays information about a route map.

Configuration Examples for Route Policy Manager

This example shows how to use an address family to configure Route Policy Manager so that any unicast and multicast routes from neighbor 209.0.2.1 are accepted if they match prefix-list AllowPrefix:

```
router bgp 64496

neighbor 172.16.0.1 remote-as 64497
  address-family ipv4 unicast
    route-map filterBGP in

route-map filterBGP
 match ip address prefix-list AllowPrefix

ip prefix-list AllowPrefix 10 permit 192.0.2.0/24
ip prefix-list AllowPrefix 20 permit 172.16.201.0/27
```

Related Topics

The following topics can give more information on Route Policy Manager:

- Chapter 9, “Configuring Basic BGP”