Policy Configuration Guide

Version 20.7.1, 15 July 2020

Registered Address	Support	Sales
26, Kingston Terrace, Princeton, New Jersey 08540, United States		
		+91 80 4850 5445
http://www.rtbrick.com	support@rtbrick.com	sales@rtbrick.com

Table of Contents

1. Overview

Routing Policies are the rules that allows you to control and modify the default behaviour of the routing protocols such as BGP and IS-IS.

A Routing Policy consist of different “terms”. This terms include “match” and “action” (with control) parts. The matched traffic with “match” field is behaved according to the “action” field. For more information, see the Building Blocks of Policy Configuration section.

To use a routing policy, firstly you need to generate it. After this, you can use this policy by enforcing them to the routes.

1.1. Supported Hardware

Routing Policy is supported on the following platforms:

- Broadcom’s Qumran Switch
- VPP based software forwarding platform

1.2. Guidelines

- The policy list names and policy names can contain alphanumeric characters and an underscore character. They must not include special characters like hyphen. For example, **BGP-EXPORT** is not supported, whereas **BGP_EXPORT** is supported. A valid name cannot start with a number but it can contain numbers and underscore (_) in the string. The length of the names should not exceed 54 characters.

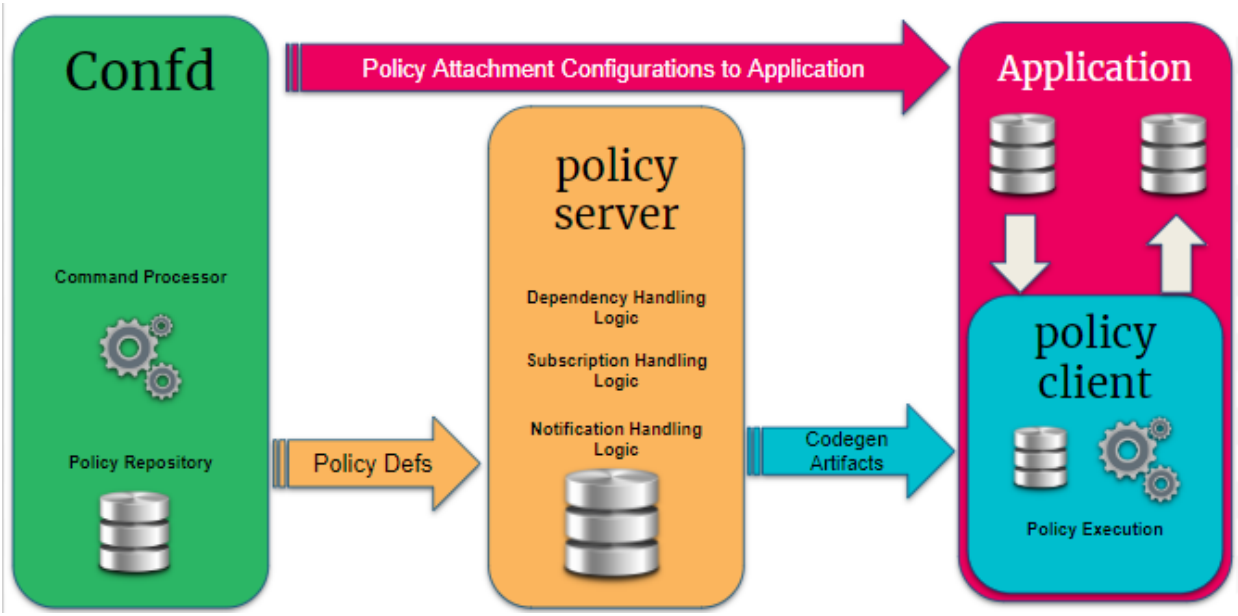
1.3. Limitations

- Configuring the raw hardware package filters through a generic representation model is not supported
- The following features are unavailable in the current policy implementation:
  - Conditional policies, that is, filtering based on conditions (that is, if a route is present in table x, then permit or deny)
  - Filtering based on Layer 2 constructs like MAC and ARP
  - Access Control Lists on generic criteria
  - Integration of subscriber policy-based routing
  - Policy Main is not supported in this release

1.4. Policy Components

In RtBrick Full Stack, the policy is divided into 4 sub-components:

- Policy Repository
- Command Processing Module
- Policy Server, the policy generation and relationship management component
- Policy Client, the policy enforcement component



1.4.1. Policy Repository

The policy repository contains all the tables that are related to policy and associated list of compare criteria

1.4.2. Command Processing Module

The command processing module is part of the Configuration daemon (confD), and that handles user interaction with the policy module. This is the back-end of the Command Line Interface (CLI) and JSON configuration that support the policy configurations.

This module maps the user-entered configuration into the back-end policy object, which is used by the execution engine (after verification) that ensures that the policy can be correctly executed. In the future implementation of policy, this will be extended to include dynamic criteria for permit and deny actions that is based on routes present in a specific table. This module relays the user intent and relays it via Publish Subscribe bus (PubSub bus) to policy server.

1.4.3. Policy Server

The Policy Server is a server component that manages all the policy rules in the various policy tables and also code generation of the policies.

The following are the functionalities of the policy server:

- Parses the objects in the policy tables, and it is an execution engine that generates the code to build the policy rules for evaluation, the relationship between various objects, and relays the intent to the evaluation engine.
- Maintains relationships between various policy constructs such as policy statements, rules, terms and lists.
- Tracks the attachment points so that when policies are modified, the appropriate clients are notified with the relevant new policies.
- Flattens the various relationships and generates a notification table that the clients subscribe to obtain notification based on specific interest groups.
- Uses the dependency table relationships to generate jobs to trigger code generation for various policy components.
- On code generation the policy server updates a notification table that maintains the mapping between the policy server has a notification table that maintains the mapping of the policy objects for which code is generated and the client interest groups. The notification table is a single point for the dissemination of information so that it can generate notifications for clients depending on their subscriptions for policy of interest.
- Policy server notification is generated towards the policy clients. A notification is received from the notification table with metadata information that notifies the client if this is a new version of the policy or the first version of the policy. The client uses this information to enforce the policy evaluation and to decide on the version of the policy rule to be used.

1.4.4. Policy Client

Policy client is a shared library component that a client daemon like BGP, ISIS, OSPF etc links to. This is the component that performs policy enforcement. It performs the following tasks:

- Links with client daemons like BGP, ISIS, OSPF.
- Contains a listener that gets notifications on the availability of a new policy rule that is generated by the policy server.
- Evaluates the compiled rule and if there are any listeners/ interests, then notifies the components within the client daemon.
- Evaluates any policy configurations on the client daemon and invokes policy processing in response.

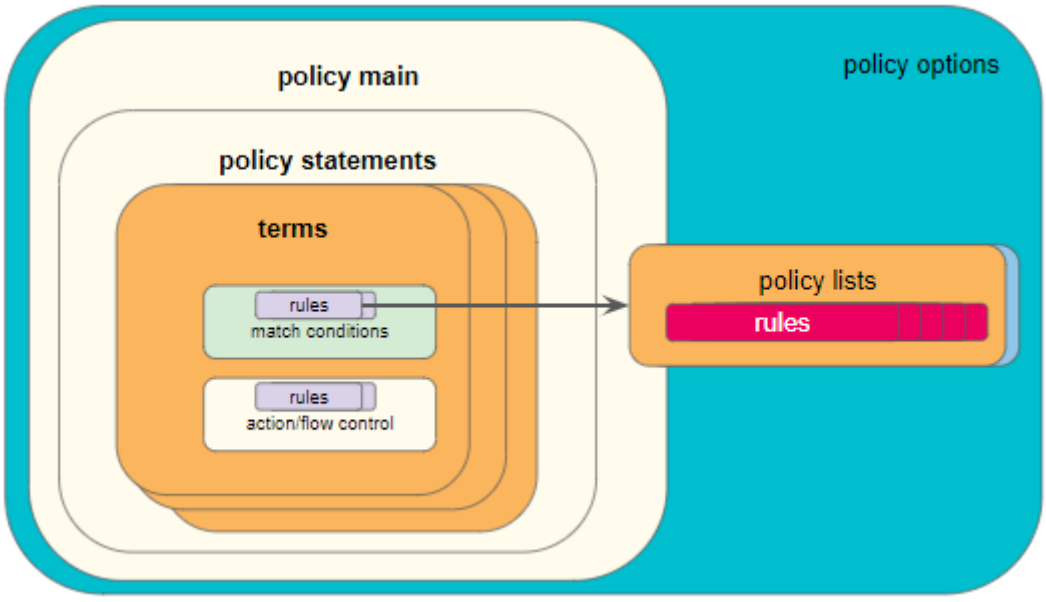
1.5. Support List Types

Following types of lists are supported:

- ipv6-prefix
- ipv4-address
- ipv6-address
- route-distinguisher
- community
- extended-community
- large-community
- as-path
- source
- sub-source
- cluster-list
- mpls-label
- mac-address

1.6. Building Blocks of Policy Configuration

The figure below shows the basic building blocks of the policy module. A policy is defined by a policy statement. A policy statement is a compound block of policy definition that consists of one or more policy terms.



A policy term is the smallest block to relay user policy intent and consists of rules for match and action blocks. Match blocks can either define single independent

elements like As-path, IP Prefix, IP addresses, Community, ext-community etc. or a list of these elements maintained in a different table.

Policy Options	Policy configuration mode
Policy Statement	Composed of one or more terms exercised in the order defined
Policy Term	Term has a name and ordinal. The terms are executed based on the ordinal in ascending order.
Match Conditions	Match criteria to define either a single or list of independent elements. This is an optional block in a policy term
Action	Action criteria to either perform an action or define flow control, that is, go to next term, accept, deny etc. This is an optional block in policy term with a default action deny
Policy Lists	Aggregation or list of items of various types that are used in various policy comparison blocks
Rules	Index inside a policy term that defines the ordering of match and/or action criteria

1.6.1. Statement

A policy statement name is a globally unique string that is used to identify the policy and also used by the application for attachment points.

1.6.2. Term

- A term name is unique string within the scope of a statement. This is used to name a match-condition and action.
- An ordinal must be unique number within the scope of a statement which determines the order of the term execution within a policy statement.
- If no terms exist or configured, and if the policy is used, then all routes/BDS objects will be denied.

1.6.3. Term Elements

- Match-conditions must be associated to the match-type, that is, and/or.
- Implicit sub-ordinal numbering, that is, the sub-ordinal value for match-conditions is 1 and the sub-ordinal value for action is 2.
- A match condition block in a term can exist without action block.
- An action block in a term can exist without match condition block.
- If no term elements exist or configured, and if the policy is used, then all routes/BDS objects will be denied.

1.6.4. Match Conditions

- The outcome of match-conditions block is deny by default.
- One or more matching rules make a match condition block; each matched routes/BDS objects are permitted by default.
- If a rule uses list match and if any one of the list entry matches to the attribute value, then the route is considered to be matched.
- If a list is defined and it is empty, then the route.bds object will be denied.
- Order of matching is based on the rule numbers.
- If match-type is or, then any one rule match will consider the route/BDS object as matched and permitted, otherwise it is denied.
- If match-type is and, then all rules match will consider the route/bds object as matched and permitted otherwise its denied.
- If match-conditions block results in a successful match, then corresponding action block is executed (resulting route/BDS object to be permitted).
- If match-conditions block results in a unsuccessful or there is no match, then corresponding action block is not executed instead next term is executed. If there are no more terms, then the policy execution will result in deny (resulting route/BDS object to be denied).

1.6.5. Actions in Policy

Action	Description
action goto-next-term	If next term exists, then next term is executed and the policy result is decided based on execution result.
action return-deny	Stops policy execution and returns result as deny (resulting route/BDS object to be denied)
action return-permit	Stop policy execution and return result as permit (resulting route/BDS object to be permitted)
operation delete-attribute	Deletes the attribute from the route/BDS object, that is, clearing all the info for that specific attribute in the object
operation <operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type

1.7. Policy Match Options, Compare Types, and Operations

Policy Match Options	Operation Types Supported	Compare Types Supported
route ipv4-prefix	overwrite	regex-match exact greater greater-or-exact prefix-length-exact prefix-length-greater prefix-length-greater-or-exact
route ipv6-prefix	overwrite	regex-match exact greater greater-or-exact prefix-length-exact prefix-length-greater prefix-length-greater-or-exact
route distinguisher	overwrite	regex-match exact
route community	append prepend overwrite	regex-match exact exists
route extended-community	append prepend overwrite	regex-match exact exists
route large-community	append prepend overwrite	regex-match exact exists
route as-path	append prepend overwrite	regex-match exact exists
route cluster-list	append prepend overwrite	regex-match exact exists
route source	overwrite	regex-match exact
route sub-source	overwrite	regex-match exact
route originator-identifier	overwrite	regex-match exact
route peer-router-id	overwrite	regex-match exact

Policy Match Options	Operation Types Supported	Compare Types Supported
route ipv4-nexthop	overwrite	regex-match exact
route ipv6-nexthop	overwrite	regex-match exact
route label	overwrite	regex-match exact
route peer-ipv4	overwrite	regex-match exact
route peer-ipv6	overwrite	regex-match exact

1.7.1. Policy Compare Types

Policy Compare Types	Description
regex-match	An attribute can be matched using a standard Linux egrep regular expression. Example: "label": "label-op:push,label:206,bos:1" In this example, the label is a 64bit number, which has label value, bos, and operation encoding. A regex is used to match the string which is displayed in the table dump, that is, label-op:push,label:206,bos:1 not the 64bit value. The same is applicable to an array type attribute. A regex can be written to the string which is visible in the table dump output.
exact	Value configured in the command must be same as application attribute value
exists	This is applicable only for array type attribute; an exist match is the one where value configured in the command must exist in the application attribute value which is an array.
lesser	The application attribute value must be lesser than the value configured in the command

Policy Compare Types	Description
lesser-or-exact	The application attribute value must be lesser than or exact value configured in the command
greater	The application attribute value must be greater than the value configured in the command
greater-or-exact	The application attribute value must be greater than or exact value configured in the command
prefix-length-exact	The application attribute value whose prefix length must be lesser than or exact value configured in the command
prefix-length-greater	The application attribute value whose prefix length must be greater than or exact value configured in the command

1.7.2. Policy Operation Types

Policy Operation Types	Description
Add	The application attribute value will be added with the value configured in the command
Subtract	The application attribute value will be subtracted with the value configured in the command. If the result of the subtraction results in a number less than 0, the value "0" is used.
Multiply	The application attribute value will be multiplied with the value configured in the command
Divide	The application attribute value will be divided with the value configured in the command
Overwrite	The application attribute value will be overwritten with the value configured in the command

1.8. Structure of Policy Statements

1.8.1. Syntax and Structure

The following example shows the syntax and structure of the policy options and statements.

```
edit policy-options
  edit policy-statement <policy-name>
    edit term <term-name-1> ordinal <term-ordinal-value-1>
      edit match-conditions match-type or
        edit rules
```

```
          set rule <rule-number> route ipv4-prefix match exact value <value>
          set rule <rule-number> route ipv4-prefix match exact value <value>
          set rule <rule-number> route ipv6-prefix match exact value <value>
          set rule <rule-number> route distinguisher match exact value
<value>
          set rule <rule-number> route community match exact value <value>
          set rule <rule-number> route extended-community match exact value
<value>
          set rule <rule-number> route large-community match exact value
<value>
          set rule <rule-number> route as-path match exact value <value>
          set rule <rule-number> route cluster-list match exact value <value>
          set rule <rule-number> route source match exact value <value>
          set rule <rule-number> route sub-source match exact value <value>
          set rule <rule-number> route originator-identifier match exact
value <value>
          set rule <rule-number> route peer-router-id match exact value
<value>
          set rule <rule-number> route ipv4-nexthop match exact value <value>
          set rule <rule-number> route ipv6-nexthop match exact value <value>
          set rule <rule-number> route label match exact value <value>
          set rule <rule-number> route peer-ipv4 match exact value <value>
          set rule <rule-number> route peer-ipv6 match exact value <value>
        exit
      exit
      edit action
        edit rules
          set rule <rule-number> route ipv4-prefix operation overwrite value
<value>
          set rule <rule-number> route ipv6-prefix operation overwrite value
<value>
          set rule <rule-number> route distinguisher operation overwrite
value <value>
          set rule <rule-number> route community operation overwrite value
<value>
          set rule <rule-number> route extended-community operation overwrite
value <value>
          set rule <rule-number> route large-community operation overwrite
value <value>
          set rule <rule-number> route as-path operation overwrite value
<value>
          set rule <rule-number> route cluster-list operation overwrite value
<value>
          set rule <rule-number> route source operation overwrite value
<value>
          set rule <rule-number> route sub-source operation overwrite value
<value>
          set rule <rule-number> route originator-identifier operation
overwrite value <value>
          set rule <rule-number> route peer-router-id operation overwrite
value <value>
          set rule <rule-number> route ipv4-nexthop operation overwrite value
<value>
          set rule <rule-number> route ipv6-nexthop operation overwrite value
<value>
          set rule <rule-number> route label operation overwrite value
<value>
          set rule <rule-number> route peer-ipv4 operation overwrite value
```

```
<value>
          set rule <rule-number> route peer-ipv6 operation overwrite value
<value>
          set rule <rule-number-n+1> action goto-next-term
        exit
      exit
    exit
    edit term <term-name-2> ordinal <term-ordinal-value-2>
      edit match-conditions match-type or
        edit rules
          set rule <rule-number> route ipv4-prefix match exact value <value>
          set rule <rule-number> route ipv4-prefix match exact value <value>
          set rule <rule-number> route ipv6-prefix match exact value <value>
          set rule <rule-number> route distinguisher match exact value
<value>
          set rule <rule-number> route community match exact value <value>
          set rule <rule-number> route extended-community match exact value
<value>
          set rule <rule-number> route large-community match exact value
<value>
          set rule <rule-number> route as-path match exact value <value>
          set rule <rule-number> route cluster-list match exact value <value>
          set rule <rule-number> route source match exact value <value>
          set rule <rule-number> route sub-source match exact value <value>
          set rule <rule-number> route originator-identifier match exact
value <value>
          set rule <rule-number> route peer-router-id match exact value
<value>
          set rule <rule-number> route ipv4-nexthop match exact value <value>
          set rule <rule-number> route ipv6-nexthop match exact value <value>
          set rule <rule-number> route label match exact value <value>
          set rule <rule-number> route peer-ipv4 match exact value <value>
          set rule <rule-number> route peer-ipv6 match exact value <value>
        exit
      exit
      edit action
        edit rules
          set rule <rule-number> route ipv4-prefix operation overwrite value
<value>
          set rule <rule-number> route ipv4-prefix operation overwrite value
<value>
          set rule <rule-number> route ipv6-prefix operation overwrite value
<value>
          set rule <rule-number> route distinguisher operation overwrite
value <value>
          set rule <rule-number> route community operation overwrite value
<value>
          set rule <rule-number> route extended-community operation overwrite
value <value>
          set rule <rule-number> route large-community operation overwrite
value <value>
          set rule <rule-number> route as-path operation overwrite value
<value>
          set rule <rule-number> route cluster-list operation overwrite value
<value>
          set rule <rule-number> route source operation overwrite value
<value>
          set rule <rule-number> route sub-source operation overwrite value
```

```
<value>
          set rule <rule-number> route originator-identifier operation
overwrite value <value>
          set rule <rule-number> route peer-router-id operation overwrite
value <value>
          set rule <rule-number> route ipv4-nexthop operation overwrite value
<value>
          set rule <rule-number> route ipv6-nexthop operation overwrite value
<value>
          set rule <rule-number> route label operation overwrite value
<value>
          set rule <rule-number> route peer-ipv4 operation overwrite value
<value>
          set rule <rule-number> route peer-ipv6 operation overwrite value
<value>
          set rule <rule-number-n+1> action return-permit
        exit
      exit
    exit
    edit term <term-name-3> ordinal <term-ordinal-value-3>
      edit match-conditions match-type and
        edit rules
          set rule <rule-number> route ipv4-prefix match-list-name <list-
name>
          set rule <rule-number> route ipv6-prefix match-list-name <list-
name>
          set rule <rule-number> route distinguisher match-list-name <list-
name>
          set rule <rule-number> route community match-list-name <list-name>
          set rule <rule-number> route extended-community match-list-name
<list-name>
          set rule <rule-number> route large-community match-list-name <list-
name>
          set rule <rule-number> route as-path match-list-name <list-name>
          set rule <rule-number> route cluster-list match-list-name <list-
name>
          set rule <rule-number> route source match-list-name <list-name>
          set rule <rule-number> route sub-source match-list-name <list-name>
          set rule <rule-number> route originator-identifier match-list-name
<list-name>
          set rule <rule-number> route peer-router-id match-list-name <list-
name>
          set rule <rule-number> route ipv4-nexthop match-list-name <list-
name>
          set rule <rule-number> route ipv6-nexthop match-list-name <list-
name>
          set rule <rule-number> route label match-list-name <list-name>
          set rule <rule-number> route peer-ipv4 match-list-name <list-name>
          set rule <rule-number> route peer-ipv6 match-list-name <list-name>
        exit
      exit
      edit action
        edit rules
          set rule <rule-number-n+1> action return-deny
        exit
      exit
    exit
  exit
```

```
edit policy-list <list-name> type ipv4-prefix
  edit entries
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
  exit
exit
edit policy-list <list-name> type ipv4-prefix
  edit entries
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
  exit
exit
edit policy-list <list-name> type ipv6-prefix
  edit entries
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
  exit
exit
edit policy-list <list-name> type ipv4-address
  edit entries
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
  exit
exit
edit policy-list <list-name> type ipv6-address
  edit entries
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
  exit
exit
edit policy-list <list-name> type route-distinguisher
  edit entries
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
  exit
exit
edit policy-list <list-name> type community
  edit entries
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
  exit
exit
edit policy-list <list-name> type extended-community
  edit entries
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
    set ordinal <rule-number> value <value>
  exit
exit
edit policy-list <list-name> type large-community
  edit entries
```

```
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
    exit
  exit
  edit policy-list <list-name> type as-path
    edit entries
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
    exit
  exit
  edit policy-list <list-name> type source
    edit entries
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
    exit
  exit
  edit policy-list <list-name> type sub-source
    edit entries
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
    exit
  exit
  edit policy-list <list-name> type cluster-list
    edit entries
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
    exit
  exit
  edit policy-list <list-name> type mpls-label
    edit entries
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
    exit
  exit
  edit policy-list <list-name> type mac-address
    edit entries
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
      set ordinal <rule-number> value <value>
    exit
  exit
exit
```

1.8.2. Sample Configuration for Policy

```
edit policy-options
  edit policy-statement p1
    edit term t1 ordinal 1
      edit match-conditions match-type or
        edit rules
          set rule 3 route ipv4-prefix match-regex value [0-9]{1,3}.[0-
9]{1,3}.[0-9]{1,3}.(15)/[0-9]{1,2}
        exit
      exit
      edit action
        edit rules
          set rule 1 route local-preference operation overwrite value 123
          set rule 3 action goto-next-term
        exit
      exit
    exit
    edit term t2 ordinal 2
      edit match-conditions match-type or
        edit rules
          set rule 1 route ipv4-prefix match-list-name l1
        exit
      exit
      edit action
        edit rules
          set rule 1  route med operation overwrite value 321
        exit
      exit
    exit
    edit term t3 ordinal 3
      edit match-conditions match-type or
        edit rules
          set rule 3 route ipv4-prefix match-regex value [0-9]{1,3}.[0-
9]{1,3}.[0-9]{1,3}.(16)/[0-9]{1,2}
        exit
      exit
      edit action
        edit rules
          set rule 4 route local-preference operation delete-attribute
          set rule 5 action return-deny
        exit
      exit
    exit
  exit
  edit policy-list l1 type ipv4-prefix
    edit entries
      set ordinal 1 value 200.0.0.1/32
      set ordinal 2 value 200.0.0.3/32
      set ordinal 3 value 200.0.0.5/32
      set ordinal 4 value 200.0.0.7/32
      set ordinal 5 value 200.0.0.9/32
    exit
  exit
exit
```

1.9. Tables and Subscriptions

The table below shows the various tables and their sharing across various policy components.

Confd	global.policy.list global.policy.list.entries global.policy.rules global.policy.statement global.policy.term global.policy.term.elements global.policy.mapping.list global.policy.mapping.rules	Policy Statement is composed of one or more policy terms. Each term has a match action criteria. In the match and action criteria either a single element or a list of elements are compared and actions are taken. The actions include accept, deny, flow-control etc.
policy.server	global.policy.dependency global.<bds_name>.policy.subscription global.<bds_name>.policy.notification	Policy Server subscribes to all the tables from confd and creates tables that track policy-entry and dependency and notifies clients after code generation.
policy.client	global.<bds_name>.policy.shared.object.cache global.<bds_name>.policy.subscription global.<bds_name>.policy.context	Subscribes to code generation notifications, application context and maintains cache of subscribed .so

1.10. Using Policy with BGP

RtBrick supports attaching a BGP routing policy at two levels:

- Peer group address-family level
- Instance address-family level

In each case, you can apply the policy as an import or export policy and filter. As expected, import filters determine which routing updates are accepted and export filters determine which routes are advertised to other peers.

For more information, see the *RBFS BGP Configuration Guide*.

1.11. Using the Policy Test Feature

You can use the policy test feature to test a policy before attaching it to a BGP peer group or an instance.

Perform the following tasks:

1. Identify the table that you want to input to the policy.

```
ubuntu@leaf1:~$ rtb bgp.appd.1 show datastore table dump default.bgp.routing-
table.ipv4.vpn-unicast | grep prefix
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.0.3/32
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.0.4/32
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.101.0/24
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.102.0/24
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.103.0/24
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.51.0/24
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.52.0/24
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.53.0/24
```

2. Run the 'policy-test-run' command on the table that you identified in step-1.

```
ubuntu@leaf1:~$ rtb bgp.appd.1 policy-test-run BGP_EXPORT on
default.bgp.routing-table.ipv4.vpn-unicast
```

The test feature creates two result tables. The result tables have .policy.permit and .policy.deny appended to the name of the input table.

The result tables show which routes are permitted and denied:

```
ubuntu@leaf1:~$ rtb bgp.appd.1 show datastore table dump default.bgp.routing-
table.ipv4.vpn-unicast.policy.permit | grep prefix
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.0.3/32
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.0.4/32
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.101.0/24
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.102.0/24
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.103.0/24
```

```
ubuntu@leaf1:~$ rtb bgp.appd.1 show datastore table dump default.bgp.routing-
table.ipv4.vpn-unicast.policy.deny | grep prefix
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.51.0/24
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.52.0/24
  attribute: prefix4 (3), type: ipv4prefix (13), length: 5, value:
192.168.53.0/24
```

1.11.1. Clearing the Result Tables

You can clear the result tables using the 'policy-test-clear' command:

```
ubuntu@leaf1:~$ rtb bgp.appd.1 policy-test-clear BGP_EXPORT on
default.bgp.routing-table.ipv4.vpn-unicast
```

2. Policy Configuration Commands

This section presents a list of commands used for configuring policies. For information about the match options, compare operations and types, refer to Policy Match Options, Compare Types, and Operations.

2.1. Policy list rules configuration

2.1.1. ordinal value

Use this command to execute in an order

Syntax

set ordinal <ordinal> **value** <value>

Command arguments

<ordinal>	it gives order of execution of rules
<value>	Mention proper value

Command modes

```
[policy_options.policy_list.entries]
```

Example

```
edit policy-list l1 type ipv4-prefix
edit entries
set ordinal 1 value 200.0.0.1/32
set ordinal 2 value 200.0.0.3/32
set ordinal 3 value 200.0.0.5/32
set ordinal 4 value 200.0.0.7/32
set ordinal 5 value 200.0.0.9/32
exit
exit
```

To delete the operation that you performed, enter the following command:

delete ordinal <ordinal>

Command arguments

<ordinal>	it gives order of deleting the executed rules

Example

```
edit policy-list l1 type ipv4-prefix
edit entries
delete ordinal 1
delete ordinal 2
delete ordinal 3
delete ordinal 4
delete ordinal 5
exit
exit
```

2.2. Policy rules match list options

2.2.1. ipv4-prefix match-list

Use this command to set the rule to match condition for ipv4 prefix list

Syntax

set rule <rule> **route ipv4-prefix match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv4-prefix match-list-name l1
exit
exit
exit
exit
```

2.2.2. ipv6-prefix match-list

Use this command to set the rule to match condition for ipv6 prefix list

Syntax

set rule <rule> **route ipv6-prefix match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv6-prefix match-list-name l1
exit
exit
exit
exit
```

2.2.3. distinguisher match-list

Use this command to set the rule to match condition for distinguisher list

Syntax

set rule <rule> **route distinguisher match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route distinguisher match-list-name l1
exit
exit
exit
exit
```

2.2.4. community match-list

Use this command to set the rule to match condition for community list

Syntax

set rule <rule> **route community match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route community match-list-name l1
exit
exit
exit
exit
```

2.2.5. extended-community match-list

Use this command to set the rule to match condition for extended-community list

Syntax

set rule <rule> **route extended-community match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route extended-community match-list-name l1
exit
exit
exit
exit
```

2.2.6. large-community match-list

Use this command to set the rule to match condition for large-community list

Syntax

set rule <rule> **route large-community match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route large-community match-list-name l1
exit
exit
exit
exit
```

2.2.7. as-path match-list

Use this command to set the rule to match condition for as-path list

Syntax

set rule <rule> **route as-path match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route as-path match-list-name l1
exit
exit
exit
exit
```

2.2.8. cluster-list match-list

Use this command to set the rule to match condition for cluster-list

Syntax

set rule <rule> **route cluster-list match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy

<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route cluster-list match-list-name l1
exit
exit
exit
exit
```

2.2.9. source match-list

Use this command to set the rule to match condition for source list

Syntax

set rule <rule> **route source match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route source match-list-name l1
exit
exit
exit
exit
```

2.2.10. sub-source match-list

Use this command to set the rule to match condition for sub-source list

Syntax

set rule <rule> **route sub-source match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route sub-source match-list-name l1
exit
exit
exit
exit
```

2.2.11. originator-identifier match-list

Use this command to set the rule to match condition for originator-identifier list

Syntax

set rule <rule> **route originator-identifier match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route originator-identifier match-list-name l1
exit
exit
exit
exit
```

2.2.12. peer-router-id match-list

Use this command to set the rule to match condition for peer_router-id list

Syntax

set rule <rule> **route peer-router-id match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy

<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route peer-router-id match-list-name l1
exit
exit
exit
exit
```

2.2.13. ipv4-nexthop match-list

Use this command to set the rule to match condition for ipv4-nexthop list

Syntax

set rule <rule> **route ipv4-nexthop match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv4-nexthop match-list-name l1
exit
exit
exit
exit
```

2.2.14. ipv6-nexthop match-list

Use this command to set the rule to match condition for ipv6-nexthop list

Syntax

set rule <rule> **route ipv6-nexthop match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv6-nexthop match-list-name l1
exit
exit
exit
exit
```

2.2.15. label match-list

Use this command to set the rule to match condition for label list

Syntax

set rule <rule> **route label match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route label match-list-name l1
exit
exit
exit
exit
```

2.2.16. peer-ipv4 match-list

Use this command to set the rule to match condition for peer-ipv4 list

Syntax

set rule <rule> **route peer-ipv4 match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy

<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route peer-ipv4 match-list-name l1
exit
exit
exit
exit
```

2.2.17. peer-ipv6 match-list

Use this command to set the rule to match condition for peer-ipv6 list

Syntax

set rule <rule> **route peer-ipv6 match-list-name** <list-name>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<list-name>	name of the list of prefix or attributes which is going to be used for setting up the policy

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route peer-ipv6 match-list-name l1
exit
exit
exit
exit
```

2.3. Policy rules delete for match

2.3.1. delete rule

Use this command to delete the rule set for policy statement

Syntax

delete rule <rule>

Command arguments

<rule>	rule which are going to delete either for match condition

Command modes

```
[policy_options.statement.term.match.rules]
[policy_options.statement.term.action.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
delete rule 1
exit
exit
exit
exit
```

2.4. Policy rules match rules options

2.4.1. ipv4-prefix match-type

Use this command to setup a rule to match-value type for route ipv4-prefix

Syntax

set rule <rule> **route ipv4-prefix match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid IPv4 address, for example 2.2.2.4/24

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv4-prefix match prefix-length-exact value 2.2.2.4/24
exit
exit
exit
exit
exit
```

2.4.2. ipv6-prefix match-type

Use this command to setup a rule to match-value type for route ipv6-prefix

Syntax

set rule <rule> **route ipv6-prefix match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid IPv6 address, for example 2001:db8:3c4d:15::/64

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv6-prefix match exact value 2001:db8:3c4d:15::/64
exit
exit
exit
exit
exit
```

2.4.3. distinguisher match-type

Use this command to setup a rule to match-value type for route distinguisher

Syntax

set rule <rule> **route distinguisher match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy

<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route distinguisher value, for example 192.168.1.1:65002

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route distinguisher match exact value 192.168.1.1:65002
exit
exit
exit
exit
exit
```

2.4.4. community match-type

Use this command to setup a rule to match-value type for route community

Syntax

set rule <rule> **route community match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route community value, for example 7018:5000

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route community match exact value 7018:5000
exit
exit
exit
exit
exit
```

2.4.5. extended-community match-type

Use this command to setup a rule to match-value type for route extended-community

Syntax

set rule <rule> **route extended-community match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route extended-community value, for example 192.168.0.0:5000

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route extended-community match exact value 192.168.0.0:5000
exit
exit
exit
exit
exit
```

2.4.6. large-community match-type

Use this command to setup a rule to match-value type for route larger-community

Syntax

set rule <rule> **route large-community match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route large-community value, for example 2914:65400:5000

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route large-community match exact value 2914:65400:5000
exit
exit
exit
exit
exit
```

2.4.7. as-path match-type

Use this command to setup a rule to match-value type for route as-path

Syntax

set rule <rule> **route as-path match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route as-path value, for example 65001

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route as-path match exact value 65001
exit
exit
exit
exit
exit
```

2.4.8. cluster-list match-type

Use this command to setup a rule to match-value type for route cluster-list

Syntax

set rule <rule> **route cluster-list match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid IPv4 address, for example 10.10.10.2

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route cluster-list match exact value 192.168.1.5
exit
exit
exit
exit
exit
```

2.4.9. origin match-type

Use this command to setup a rule to match-value type for route origin

Syntax

set rule <rule> **route origin match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route origin value, for example IGP, EGP etc

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route origin match exact value IGP
exit
exit
exit
exit
exit
```

2.4.10. med match-type

Use this command to setup a rule to match-value type for route MED

Syntax

set rule <rule> **route med match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route med value, for example 100,200 etc

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route med match exact 100
exit
exit
exit
exit
exit
```

2.4.11. local-preference match-type

Use this command to setup a rule to match-value type for local-preference route

Syntax

set rule <rule> **route local-preference match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route local-preference value, for example 100, 200 etc

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route local-preference match exact 100
exit
exit
exit
exit
exit
```

2.4.12. preference match-type

Use this command to setup a rule to match-value type for preference route

Syntax

set rule <rule> **route preference match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route preference value, for example 2, 100, 200 etc

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route preference match exact 100
exit
exit
exit
exit
exit
```

2.4.13. source match-type

Use this command to setup a rule to match-value type for source route

Syntax

set rule <rule> **route source match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route source value, for example BGP

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route source match exact bgp
exit
exit
exit
exit
exit
```

2.4.14. sub-source match-type

Use this command to setup a rule to match-value type for sub-source route

Syntax

set rule <rule> **route sub-source match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route sub-source value, for example 100

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route sub-source match exact 200
exit
exit
exit
exit
exit
```

2.4.15. originator-identifier match-type

Use this command to setup a rule to match-value type for originator-identifier route

Syntax

set rule <rule> **route originator-identifier match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid IPv4 address, for example 20.20.20.4

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route originator-id match exact 192.168.1.7
exit
exit
exit
exit
exit
```

2.4.16. peer-router-id match-type

Use this command to setup a rule to match-value type for peer-router-id

Syntax

set rule <rule> **route peer-router-id match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid IPv4 address, for example 20.20.20.4

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route peer-router-id match exact 5.1.168.192
exit
exit
exit
exit
exit
```

2.4.17. ipv4-nexthop match-type

Use this command to setup a rule to match-value type for ipv4-nexthop route

Syntax

set rule <rule> **route ipv4-nexthop match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid IPv4 address, for example 20.20.20.4

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv4-nexthop match exact 129.121.76.192
exit
exit
exit
exit
exit
```

2.4.18. ipv6-nexthop match-type

Use this command to setup a rule to match-value type for ipv6-nexthop route

Syntax

set rule <rule> **route ipv6-nexthop match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid IPv6 address, for example 17f0:949f:6a53:898f:8369:beb9:cd89:5ced

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv6-nexthop match exact
17f0:949f:6a53:898f:8369:beb9:cd89:5ced
exit
exit
exit
exit
exit
```

2.4.19. receive-path-identifier match-type

Use this command to setup a rule to match-value type for receive-path-identifier route

Syntax

set rule <rule> **route receive-path-identifier match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route receive-path-identifier value, for example 1885178186

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route receive-path-id match exact 1885178186
exit
exit
exit
exit
exit
```

2.4.20. send-path-identifier match-type

Use this command to setup a rule to match-value type for send-path-identifier route

Syntax

set rule <rule> **route send-path-identifier match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route send-path-identifier value, for example 1885178186

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route send-path-id match exact 1885178186
exit
exit
exit
exit
exit
```

2.4.21. label match-type

Use this command to setup a rule to match-value type for label route

Syntax

set rule <rule> **route label match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid mpls-lable, for example label-op:push,label:20001,bos-op:compare,bos:1

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route label match exact label-op:push,label:20001,bos-
op:compare,bos:1
exit
exit
exit
exit
exit
```

2.4.22. igp-metric match-type

Use this command to setup a rule to match-value type for igp-metric route

Syntax

set rule <rule> **route igp-metric match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid route igp-metric value, for example 1885178186

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route igp-metric match exact 1885178186
exit
exit
exit
exit
exit
```

2.4.23. peer-ipv4 match-type

Use this command to setup a rule to match-value type for peer-ipv4 route

Syntax

set rule <rule> **route peer-ipv4 match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid IPv4 address, for example 129.121.76.192

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route peer-ipv4 match exact value 129.121.76.192
exit
exit
exit
exit
exit
```

2.4.24. peer-ipv6 match-type

Use this command to setup a rule to match-value type for peer ipv6-route

Syntax

set rule <rule> **route peer-ipv6 match** <match-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<match-type>	is condition of "and/or" ('and' means set of rules to be satisfied & 'or' means any one rule is satisfied policy should get applied)
<attribute-value>	Specify a valid IPv6 address, for example 17f0:949f:6a53:898f:8369:beb9:cd89:5ced

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route peer-ipv6 match exact value
17f0:949f:6a53:898f:8369:beb9:cd89:5ced
exit
exit
exit
exit
exit
```

2.5. Policy rules action-rules options

2.5.1. ipv4-prefix operation

Use this command to perform the operation on configured ipv4-prefix rule

Syntax

set rule <rule> **route ipv4-prefix operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid IPv4 address, for example 20.20.20.3/24

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv4-prefix operation overwrite value 1.1.1.1/24
exit
exit
exit
exit
exit
```

2.5.2. ipv6-prefix operation

Use this command to perform the operation on configured ipv6-prefix rule

Syntax

set rule <rule> **route ipv6-prefix operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid IPv6 address, for example 2001:db8:3c4d:15::/64

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv6-prefix operation overwrite value 2001:db8:3c4d:15::/64
exit
exit
exit
exit
exit
```

2.5.3. distinguisher operation

Use this command to perform the operation on configured distinguisher rule

Syntax

set rule <rule> **route distinguisher operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route distinguisher operation overwrite value 192.168.1.4:65002
exit
exit
exit
exit
exit
```

2.5.4. community operation

Use this command to perform the operation on configured community rule

Syntax

set rule <rule> **route community operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route community value, for example 7018:5000

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route community operation overwrite value 7018:5000
exit
exit
exit
exit
exit
```

2.5.5. extended-community operation

Use this command to perform the operation on configured extended-community rule

Syntax

set rule <rule> **route extended-community operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route extended-community value, for example 192.168.0.0:5000

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route extended-community operation overwrite value
192.168.0.0:5000
exit
exit
exit
exit
exit
```

2.5.6. large-community operation

Use this command to perform the operation on configured large-community rule

Syntax

set rule <rule> **route large-community operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route large-community value, for example 2914:65400:5000

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route large-community operation overwrite value 2914:65400:5000
exit
exit
exit
exit
exit
```

2.5.7. as-path operation

Use this command to perform the operation on configured as-path rule

Syntax

set rule <rule> **route as-path operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route as-path value, for example 65001

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route as-path operation overwrite value 65002
exit
exit
exit
exit
exit
```

2.5.8. cluster-list operation

Use this command to perform the operation on configured cluster-list rule

Syntax

set rule <rule> **route cluster-list operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid IPv4 address, for example 52.10.100.250

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route cluster-list operation overwrite value 52.10.100.250
exit
exit
exit
exit
exit
```

2.5.9. origin operation

Use this command to perform the operation on configured origin rule

Syntax

set rule <rule> **route origin operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route origin value, for example IGP, EGP etc

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route origin operation overwrite value 123
exit
exit
exit
exit
exit
```

2.5.10. med operation

Use this command to perform the operation on configured MED rule

Syntax

set rule <rule> **route med operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route med value, for example 100,200 etc

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route med operation overwrite value 123
exit
exit
exit
exit
exit
```

2.5.11. local-preference operation

Use this command to perform the operation on configured local-preference rule

Syntax

set rule <rule> **route local-preference operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route local-preference value, for example 100, 200 etc

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route local-preference operation overwrite value 123
exit
exit
exit
exit
exit
```

2.5.12. preference operation

Use this command to perform the operation on configured preference rule

Syntax

set rule <rule> **route preference operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route preference value, for example 2, 100, 200 etc

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route preference operation overwrite value 123
exit
exit
exit
exit
exit
```

2.5.13. source operation

Use this command to perform the operation on configured source rule

Syntax

set rule <rule> **route source operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route source value, for example BGP

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route source operation overwrite value bgp
exit
exit
exit
exit
exit
```

2.5.14. sub-source operation

Use this command to perform the operation on configured sub-source rule

Syntax

set rule <rule> **route sub-source operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route sub-source value, for example 100

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route sub-source operation overwrite value 303243
exit
exit
exit
exit
exit
```

2.5.15. originator-identifier operation

Use this command to perform the operation on configured originator-identifier rule

Syntax

set rule <rule> **route originator-identifier operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid IPv4 address, for example 129.121.76.192

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route originator-identifier operation overwrite value 192.168.2.8
exit
exit
exit
exit
exit
```

2.5.16. peer-router-id operation

Use this command to perform the operation on configured peer-router-id rule

Syntax

set rule <rule> **route peer-router-id operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid IPv4 address, for example 129.121.76.192

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route peer-router-id operation overwrite value 5.1.168.192
exit
exit
exit
exit
exit
```

2.5.17. ipv4-nexthop operation

Use this command to perform the operation on configured ipv4-nexthop rule

Syntax

set rule <rule> **route ipv4-nexthop operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid IPv4 address, for example 129.121.76.192

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv4-nexthop operation overwrite value 10.10.10.2
exit
exit
exit
exit
exit
```

2.5.18. ipv6-nexthop operation

Use this command to perform the operation on configured ipv6-nexthop rule

Syntax

set rule <rule> **route ipv6-nexthop operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid IPv6 address, for example 17f0:949f:6a53:898f:8369:beb9:cd89:5ced

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route ipv6-nexthop operation overwrite value
17f0:949f:6a53:898f:8369:beb9:cd89:5ced
exit
exit
exit
exit
exit
```

2.5.19. receive-path-identifier operation

Use this command to perform the operation on configured receive-path-id rule

Syntax

set rule <rule> **route receive-path-identifier operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route receive-path-identifier value, for example 1885178186

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route receive-path-identifier operation overwrite value 1885178186
exit
exit
exit
exit
exit
```

2.5.20. send-path-identifier operation

Use this command to perform the operation on configured send-path-id rule

Syntax

set rule <rule> **route send-path-identifier operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route send-path-identifier value, for example 1885178186

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route send-path-identifier operation overwrite value 1885178186
exit
exit
exit
exit
exit
```

2.5.21. label operation

Use this command to perform the operation on configured label rule

Syntax

set rule <rule> **route label operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid mpls-lable, for example label-op:push,label:20001,bosop:compare,bos:1

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route lable operation overwrite value label-op
exit
exit
exit
exit
exit
```

2.5.22. igp-metric operation

Use this command to perform the operation on configured igp-metric rule

Syntax

set rule <rule> **route igp-metric operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid route igp-metric value, for example 1885178186

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route igp-metric operation overwrite value 12323
exit
exit
exit
exit
exit
```

2.5.23. peer-ipv4 operation

Use this command to perform the operation on configured peer-ipv4 rule

Syntax

set rule <rule> **route peer-ipv4 operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid IPv4 address, for example 129.121.76.192

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route peer-ipv4 operation overwrite value 26.0.1.1
exit
exit
exit
exit
exit
```

2.5.24. peer-ipv6 operation

Use this command to perform the operation on configured peer-ipv6 rule

Syntax

set rule <rule> **route peer-ipv6 operation** <operation-type> **value** <attribute-value>

Command arguments

<rule>	rule which are going to set either for match condition or action for policy
<operation-type>	An operation is performed on that attribute in the route/BDS object based on the operation-type
<attribute-value>	Specify a valid IPv6 address, for example 17f0:949f:6a53:898f:8369:beb9:cd89:5ced

Command modes

```
[policy_options.statement.term.match.rules]
```

Example

```
edit policy-options
edit policy-statement p1
edit term t1 ordinal 1
edit match-conditions match-type or
edit rules
set rule 1 route peer-ipv6 operation overwrite value
17f0:949f:6a53:898f:8369:beb9:cd89:5ced
exit
exit
exit
exit
exit
```