Chapter 6
Merging Policies

This chapter provides information about merging policies on E-series routers. The chapter discusses the following topics:

Merging Policies Overview

Merging policies enables you to create multiple policy attachments at an attachment point, resulting in a merged policy that is created and attached at this interface. Executing more than one policy attachment command with the same attachment type at an interface triggers a policy merge through the CLI.

In Profile Configuration mode, policy interface commands for IP and L2TP allow attachments to be merged into any existing merge-capable attachment at an attachment point. Service Manager can request that multiple interface profiles be applied or removed at an interface as part of service activation or deactivation. Service Manager also specifies whether or not the attachments created from these interface profiles are persistent on subsequent reloads.

An interface and an attachment type identify an attachment point. The policies referenced by the component attachments merge into a new policy, which then attaches at the attachment point. The set of component policies are ordered alphabetically by name. This order determines how any merge conflicts are resolved, with the most recently executed command taking precedence.

With policy merging, a set of policies is combined to form a single new policy, which is a union of all the component policies. Classifier groups and policy rules from each component combine to create the merged policy as in the following example:

```
host1(config)#interface atm 5/0.1
host1(config-subif)#ip policy input p1 statistics enable merge
host1(config-subif)#ip policy input p2 statistics enable merge
host1(config-subif)#ip policy input p3 statistics enable merge
host1(config-subif)#ip policy output p4 statistics enable merge
host1(config-subif)#ip policy output p5 statistics enable merge
host1(config-subif)#exit
```

The example internally results in the following, where policies p1 + p2 + p3 = mpl_10 and policies p4 + p5 = mpl_11.

```
interface atm 5/0.1
ip policy input mpl_10 statistics enable merge
ip policy output mpl_11 statistics enable merge
exit
```

The classifier list referenced by the classifier group is neither split or merged. If a merged policy already exists for a set of component policies, then the merged policy is used for the attachment. An attachment enables a merged policy to have one or more attachments.

The CLI and the Service Manager applications are the only clients of policy management that can request merging of policy attachments. With policy merging, classifier groups and policy rules from each component policy combine into the merged policy.

Policy merging follows these rules:

- The Classifier list referenced by the classifier group cannot be split or merged.
- Policy merging combines classifier groups from all component policies into the merged policy. In the previous example, policies p1, p2, and p3 are the component policies and mpl_10 is the merged policy. The merge policy is created as if all CLI commands for each component policy are run in the context of the merged policy. The merged policy result is the sum of all commands executed in the respective component policies CLI context in a predetermined merge order.
- If a merged policy already exists for a set of component policies, the merged policy is used for the attachment instead of creating a new one. This functionality allows a merged policy to have one or more attachments. A merge policy is automatically deleted when the last reference is removed.

The following restrictions apply to policy merging:

- Classifier lists cannot be merged.
- Secure policies cannot be merged.
- Policies created using ascend-data-filters cannot be merged.
- Existing policy VSAs in RADIUS are not changed; attachments created by this method cannot be merged. Ascend data filter policies can be attached at input and output attachment points.
- SNMP support for polling statistics based on component policy attachments is not available.
- The merge policy naming convention is not configurable.

Resolving Policy Merge Conflicts

The set of component policies are first ordered by their name to form the final merged policy. For example, if the component policies sets contain cp_1, cp_3, cp_9, cp_2, the order in which these policies are merged is cp_1, cp_2, cp_3, and cp_9. The merge order is important for resolving merge conflicts.

Various conflicting combinations of component policies can result in a merged policy that is not a perfect union of the component policies. These conflicts are resolved as they currently are in policy CLI context, where, in any conflict, the most recently executed command takes precedence.

More than one component policy can contain the same classifier group. If the precedence does not match, the precedence of the classifier group defined in the last component policy becomes the final precedence for this classifier group in the merged policy, as in the following example:

```
host1(config)#ip policy-list p1
host1(config-policy)#classifier-group C1 precedence 90
host1(config-classifier-group)#forward
```

```
host1(config-classifier-group)#exit
host1(config)#ip policy-list p2
host1(config-policy)#classifier-group C1 precedence 100
host1(config-classifier-group)#forward
host1(config-classifier-group)#exit
host1(config)#ip policy-list p3
host1(config-policy)#classifier-group C1 precedence 130
host1(config-classifier-group)#forward
host1(config-classifier-group)#exit
```

If you combine p1, p2, and p3, you get the following with p1, p2, p3 as the merge order for the set of component policies.

```
ip policy-list mpl_10
classifier-group C1 precedence 130
forward
exit
```

For IP, the forward, filter, next-hop, and next-interface rules are mutually exclusive within a classifier group. For all other types, filter and forward rules are mutually exclusive.

A conflict arises when more than one component policy has the same classifier group and when the rule sets defined in these classifier groups conflict. To resolve the merge conflict, the last command entered replaces any previous conflicting commands for a classifier group, as in the following example:

```
host1(config)#ip policy-list p1
host1(config-policy)#classifier-group C1 precedence 90
host1(config-classifier-group)#forward
host1(config-classifier-group)#exit
host1(config)#ip policy-list p2
host1(config-policy)#classifier-group C1 precedence 90
host1(config-classifier-group)#next-hop 1.1.1.1
host1(config-classifier-group)#exit
host1(config)#ip policy-list p3
host1(config-policy)#classifier-group C1 precedence 90
host1(config-classifier-group)#filter
host1(config-classifier-group)#exit
```

Combining p1 and p2 internally results in:

```
ip policy-list mpl_20
classifier-group C1 precedence 90
next-hop 1.1.1.1
exit
```

Combining p2 and p3 internally results in:

```
ip policy-list mpl_21
classifier-group C1 precedence 90
filter
exit
```

Combining p1, p2, and p3 internally results in:

ip policy-list mpl_22

classifier-group C1 precedence 90
filter
exit

If you have the same policy rule with different parameters, the parameter of the last rule entered with the same type is used, with the exception of IP forward rule, to resolve the conflict, as in the following example:

host1(config)#**ip policy-list p1**
host1(config-policy)#**classifier-group C1 precedence 90**
host1(config-classifier-group)#**color red**
host1(config-classifier-group)#**exit**
host1(config)#**ip policy-list p2**
host1(config-policy)#**classifier-group C1 precedence 90**
host1(config-classifier-group)#**color yellow**
host1(config-classifier-group)#**exit**

Combining p1 and p2 internally results in:

ip policy-list mpl_20
classifier-group C1 precedence 90
color yellow
exit

With the IP policy forward rule, when more forward rules are added to an existing classifier group, the list of forward rules is created. This is also true during merging, as in the following example:

host1(config)#**ip policy-list p1**
host1(config-policy)#**classifier-group C1 precedence 90**
host1(config-classifier-group)#**forward next-hop 1.1.1.1**
host1(config-classifier-group)#**exit**
host1(config)#**ip policy-list p2**
host1(config-policy)#**classifier-group C1 precedence 90**
host1(config-classifier-group)#**forward next-interface atm 5/0.1**
host1(config-classifier-group)#**exit**
host1(config)#**ip policy-list p2**
host1(config-policy)#**classifier-group C1 precedence 90**
host1(config-classifier-group)#**forward next-interface fastEthernet 4/0.1 next-hop 1.1.1.2**
host1(config-classifier-group)#**exit**

Combining p1, p2, and p3, internally results in the following:

ip policy-list mpl_10
classifier-group C1 precedence 90
forward next-hop 1.1.1.1
forward next-interface atm 5/0.1
forward next-interface fastEthernet 4/0.1 next-hop 1.1.1.2
exit

Policy management enables multiple policy attachments at the same attachment point, which results in a merged policy that is created and attached at the specified attachment point. The logical OR of the **statistics** and **baseline** keywords of all attachments are used as the **statistics** and **baseline** keyword for the merged policy attachment, as in the following example:

```
host1(config)#interface atm 5/0.1
host1(config-subif)#ip policy input p1 statistics enable baseline enable merge
host1(config-subif)#ip policy input p2 merge
host1(config-subif)#ip policy input p3 statistics enable merge
host1(config-subif)#exit
```

Results in the following:

```
interface atm 5/0.1
ip policy input mpl_5 statistics enable baseline enable merge
exit
```

Merged Policy Naming Conventions

Merged policies are dynamically created. The naming convention is mpl_*hex_of internally_generated_policy ID*, such as mpl_10. If the newly generated name already exists, then a sequence number is appended to the new name to make it unique. The sequence number starts at 1 and increments until the name is unique, such as mpl_10_2.

Reference Counting for Merged Policies

The reference counts in all containers referenced within a merged policy are incremented by the number of times they are referenced within the merged policy. Also, the reference counts of all component policies of a merged policy are incremented because of the association of the component policies with the merged policy. This means you cannot delete a component policy while a merged policy is still associated with it.

Persistent Configuration Differences for Merged Policies Through Service Manager

Service Manager can specify whether a component policy attachment is nonvolatile. If the interface where the component policy is attached is volatile, then policy management makes the attachment volatile even when the Service Manager specifies otherwise. A nonvolatile interface can have both volatile and nonvolatile component policy attachments. The merged policy that is created is the merge of all component policies attached at a given attachment point regardless of their volatility. The merged policy and its attachments are always volatile and reconstructed on each reload operation.

Policy Attachment Sequence at Login Through Service Manager

During a user login, you can specify policy attachments through Service Manager, RADIUS, and Interface Profile. The order that is used to select the policy attachment source is Service Manager, RADIUS, and Interface Profile.

For example, if you configure Ingress-Policy-Name VSA for a user in RADIUS and also have a profile with an input policy reference applied to this user's interface column, when the user logs in, the RADIUS VSA is selected as the source for the input policy attachment. If you also have service profiles applied to the user's interface column, the service profiles override both RADIUS VSA and the policy name specified in the interface profile.

NOTE: Policy merging is not supported with ascend data filter policies.

Policy management does not reselect the source if the policy attachment fails for the selected source. If the policy attachment via service profiles fails, policy management does not reselect RADIUS VSA as the next source. This means the interface does not have any input policy attachment.

Policy Attachment Rules for Merged Policies

The attributes of a policy attachment are as follows:

- Policy name—Name of policy to be attached.
- Attachment type—Type of attachment.
- Statistics enable/disable—Enable or disable statistics for the attachment.
- Baseline enable/disable—Enable or disable baselining for the attachment.
- Merge or Replace—Allow an attachment to become merge-capable and merge with any other attachments that are merge-capable. If the **merge** keyword is not specified, then it replaces any existing attachments with the new attachment. Merging always preserves statistics.
- Preserve—Preserve statistics from earlier attachment when replacing an attachment. This keyword is mutually exclusive with **merge** keyword.

Various possibilities result from a policy attachment at an interface due to the presence or absence of these keywords. The same rules apply while attaching policies based on interface profiles provided by Service Manager except as noted.

Attachments made through Interface Configuration mode follow these rules:

- If an attachment is issued with the **merge** keyword specified:
  - Any existing attachment of the same type at the interface without the **merge** keyword is replaced by the new attachment, which then becomes merge-capable.
  - An attachment is merged with any existing attachments of the same type that have the **merge** keyword set. If a merged policy already exists for the set of component policies, then this merged policy is used or a new merged policy is created dynamically and attached. The statistics for common classifier groups are preserved when replacing the existing merged attachment.
- If an attachment is issued when no **merge** or **preserve** keyword is set, then it replaces all other attachments with the same type at the interface. This attachment is not merge-capable for future use and statistics from previous attachments are not preserved.
- If an attachment is issued when the **merge** keyword is not set, but the **preserve** keyword is set, it replaces all other attachments with the same type at the interface. This attachment is not merge-capable for future use. Statistics from existing attachments are preserved for all the common classifier-groups.
- You cannot have multiple attachments of the same policy on a single attachment point. Only Service Manager executes multiple attachments of the same policy at the same attachment point.
- A detachment based on the policy name removes all attachments for that policy at the specified attachment point in a single command regardless of creation source. A detachment based on attachment type detaches all attachments at that attachment point regardless of creation source. Service Manager can delete only one attachment at a time through service deactivation.
- The **statistics** and **baseline** keywords for the merged policy attachment are computed as a logical OR for all attachments at the specified attachment point.
- If you delete an attachment:
  - The merged policy is recomputed with the remaining attachments of the same type that have the **merge** keyword set. The statistics for common classifier groups are preserved when replacing the existing merged attachment.
  - The **statistics** and **baseline** keywords for the merged policy attachment are recomputed to be a logical OR of all remaining attachments at the specified attachment point.

Error Conditions for Merged Policies

Most errors, such as mismatched interface types while merging attachments, are caught during configuration. If merging fails, the attachment at the given interface is not modified.

You can modify component policies manually. Although you might want to do this for debugging purposes, we highly discourage you doing this because it can affect synchronization with the Service Manager application. You cannot manually attach a final merged policy to any interfaces. Instead, attach the set of component policies that constitute this merged policy. If you want to modify the final merged policy, use existing policy merging or component policy modification to achieve this.

Merging Policies Configuration

In the following example IP policy p1 and IP policy p2 are attached at interface atm5/0.1 as input attachments. Subsequently, policy p3 is attached at the same point. Then policies p1 and p2 are attached as output at atm 5/0.2.

1. Create IP policy p1.

```
host1(config)#ip classifier-list C1 tcp host 1.1.1.1 any eq 80
host1(config)#ip classifier-list C2 icmp any any 8 0
host1(config)#ip policy-list p1
host1(config-policy)#classifier-group C1 precedence 90
host1(config-policy-classifier-group)#forward next-hop 10.1.1.1
host1(config-policy-classifier-group)#exit
host1(config-policy)#classifier-group C2 precedence 10
host1(config-policy-classifier-group)#filter
host1(config-policy-classifier-group)#exit
```

2. Create IP policy p2.

```
host1(config)#ip classifier-list C1 tcp host 1.1.1.1 any eq 80
host1(config)#ip classifier-list C3 ip any host 2.2.2.2
host1(config)#ip policy-list p2
host1(config-policy)#classifier-group C1 precedence 90
host1(config-policy-classifier-group)#forward next-hop 20.1.1.1
host1(config-policy-classifier-group)#exit
host1(config-policy)#classifier-group C3 precedence 10
host1(config-policy-classifier-group)#filter
host1(config-policy-classifier-group)#exit
host1(config-policy)#classifier-group * precedence 1000
host1(config-policy-classifier-group)#forward
host1(config-policy-classifier-group)#exit
```

3. Attach IP policy p1 as input at interface atm5/0.1.

```
host1(config)#Interface atm 5/0.1
host1(config-subif)#ip policy input p1 statistics enable merge
host1(config-subif)#exit
```

4. Attach IP policy p2 as input at interface atm 5/0.1. A merged policy is created.

```
host1(config)#Interface atm 5/0.1
host1(config-subif)#ip policy input p2 statistics enable merge
host1(config-subif)#exit
```

5. Display the policy lists.

```
host1#show policy-list

                                  Policy Table
                                  ------ -----
IP Policy p1
   Administrative state: enable
   Reference count:      1
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)

   Referenced by interfaces:
      None

   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_5

IP Policy p2
   Administrative state: enable
   Reference count:      1
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 20.1.1.1, order 100, rule 3 (active)
   Classifier control list: *, precedence 1000
      forward

   Referenced by interfaces:
      None

   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_5

IP Policy mpl_5
   Administrative state: enable
   Reference count:      1
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 90
```

```
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)
           next-hop 20.1.1.1, order 100, rule 3 (reachable)
   Classifier control list: *, precedence 1000
      forward

   Referenced by interfaces:
      ATM5/0.1  input policy, statistics enabled, virtual-router default

   Referenced by profiles:
      None

   Component policies:
      p1
      p2
```

6. Show configuration.

```
host1#show conf

! Configuration script being generated on TUE APR 26 2005 17:33:01 UTC
! Juniper Edge Routing Switch ERX-1440
! Version: 9.9.9 development-4.0 (April 4, 2005  15:39)
! Copyright (c) 1999-2005 Juniper Networks, Inc.  All rights reserved.
!
! Commands displayed are limited to those available at privilege level 15
!
…
interface atm 5/0.1
 ip policy input p1 statistics enabled merge
 ip policy input p2 statistics enabled merge
 exit
…
…
ip policy-list p1
 classifier-group C2 precedence 10
  filter
 classifier-group C1 precedence 90
  forward next-hop 10.1.1.1
!
ip policy-list p2
 classifier-group C3 precedence 10
  filter
 classifier-group C1 precedence 90
  forward next-hop 20.1.1.1
 classifier-group * precedence 1000
  forward
!
…
…
! End of generated configuration script.
```

7. Display interface statistics.

```
host1#show ip interface atm 5/0.1

ATM5/0.1 line protocol Atm1483 is up, ip is up
  Network Protocols: IP
  Internet address is 99.99.99.2/255.255.255.0
  Broadcast address is 255.255.255.255
```

```
Operational MTU = 9180  Administrative MTU = 0
Operational speed = 155520000  Administrative speed = 0
Discontinuity Time = 721112
Router advertisement = disabled
Proxy Arp = disabled
Network Address Translation is disabled
TCP MSS Adjustment = disabled
Administrative debounce-time = disabled
Operational debounce-time    = disabled
Access routing = disabled
Multipath mode = hashed
Auto Configure = disabled
Auto Detect = disabled
Inactivity Timer = disabled

In Received Packets 0, Bytes 0
  Unicast Packets 0, Bytes 0
  Multicast Packets 0, Bytes 0
In Policed Packets 0, Bytes 0
In Error Packets 0
In Invalid Source Address Packets 0
In Discarded Packets 0
Out Forwarded Packets 0, Bytes 0
  Unicast Packets 0, Bytes 0
  Multicast Routed Packets 0, Bytes 0
Out Scheduler Dropped Packets 0, Bytes 0
Out Policed Packets 0, Bytes 0
Out Discarded Packets 0

IP policy input mpl_5
  classifier-group C2 entry 1
    0 packets, 0 bytes
    filter
  classifier-group C3 entry 1
    0 packets, 0 bytes
    filter
  classifier-group C1 entry 1
    0 packets, 0 bytes
    forward
  classifier-group *
    0 packets, 0 bytes
    forward
queue 0: traffic class best-effort, bound to ip ATM5/0.1
  Queue length 0 bytes
  Forwarded packets 0, bytes 0
  Dropped committed packets 0, bytes 0
  Dropped conformed packets 0, bytes 0
  Dropped exceeded packets 0, bytes 0
```

8. Attach IP policy p1 at atm 5/0.2 as output.

```
host1(config)#interface atm 5/0.2
host1(config-subif)#ip policy output p1 statistics enable merge
host1(config-subif)#exit
```

9. Attach IP policy p2 at atm 5/0.2 as output. Merge policy mpl_5 is now attached.

```
host1(config)#interface atm 5/0.2
host1(config-subif)#ip policy output p2 merge
host1(config-subif)#exit
```

10. Display policies to verify that mpl_5 is created.

```
host1#show policy-list

                                    Policy Table
                                    ------ -----
IP Policy p1
   Administrative state: enable
   Reference count:      1
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)

   Referenced by interfaces:
      None

   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_5

IP Policy p2
   Administrative state: enable
   Reference count:      1
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 20.1.1.1, order 100, rule 3 (active)
   Classifier control list: *, precedence 1000
      forward

   Referenced by interfaces:
      None

   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_5

IP Policy mpl_5
   Administrative state: enable
   Reference count:      2
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)
           next-hop 20.1.1.1, order 100, rule 3 (reachable)
   Classifier control list: *, precedence 1000
      forward
```

```
   Referenced by interfaces:
      ATM5/0.1  input policy, statistics enabled, virtual-router default
      ATM5/0.2  output policy, statistics enabled, virtual-router default

   Referenced by profiles:
      None

   Component policies:
      p1
      p2
```

11. Create and attach IP policy p3 at atm 5/0.1. A new merge policy mpl_7 is created, which is a combination of p1, p2, and p3. The previous merge policy attachment is removed.

```
host1(config)#ip classifier-list C4 udp host 1.1.1.1 any eq 900
host1(config)#ip policy-list p3
host1(config-policy)#classifier-group C4 precedence 900
host1(config-policy-classifier-group)#color red
host1(config-policy-classifier-group)#exit
host1(config-policy)#classifier-group C1 precedence 80
host1(config-policy-classifier-group)#color yellow
host1(config-policy-classifier-group)#exit
host1(config-policy)#exit
host1(config)#interface atm 5/0.1
host1(config-subif)#ip policy input p3 statistics enable merge
host1(config-subif)#exit
```

12. Display policies to verify that mpl_5 and mpl_7 have been created.

```
host1#show policy-list

                                  Policy Table
                                  ------ -----
IP Policy p1
   Administrative state: enable
   Reference count:      2
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)

   Referenced by interfaces:
      None

   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_5
      mpl_7

IP Policy p2
   Administrative state: enable
   Reference count:      2
   Classifier control list: C3, precedence 10
      filter
```

```
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 20.1.1.1, order 100, rule 3 (active)
   Classifier control list: *, precedence 1000
      forward

   Referenced by interfaces:
      None

   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_5
      mpl_7

IP Policy p3
   Administrative state: enable
   Reference count:      1
   Classifier control list: C1, precedence 80
      color yellow
   Classifier control list: C4, precedence 900
      color red

   Referenced by interfaces:
      None

   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_7

IP Policy mpl_5
   Administrative state: enable
   Reference count:      1
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)
           next-hop 20.1.1.1, order 100, rule 3 (reachable)
   Classifier control list: *, precedence 1000
      forward

   Referenced by interfaces:
      ATM5/0.2  output policy, statistics enabled, virtual-router default

   Referenced by profiles:
      None

   Component policies:
      p1
      p2

IP Policy mpl_7
   Administrative state: enable
```

```
   Reference count:      1
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 80
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)
           next-hop 20.1.1.1, order 100, rule 3 (reachable)
      color yellow
   Classifier control list: C4, precedence 900
      color red
   Classifier control list: *, precedence 1000
      forward

   Referenced by interfaces:
      ATM5/0.1  input policy, statistics enabled, virtual-router default

   Referenced by profiles:
      None

   Component policies:
      p1
      p2
      p3
```

13. Detach p2 from atm 5/0.1. A new merge policy mpl_8 is created, which is a combination of p1 and p3. The previous merge policy mpl_7 is detached and, because this policy has no attachments, it is deleted.

```
host1(config)#interface atm 5/0.1
host1(config-subif)#no ip policy input p2
host1(config-subif)#exit
```

14. Display policies to verify that the mpl_7 is removed and the new merge policy mpl_8 is created.

```
host1#show policy-list

                                  Policy Table
                                  ------ -----
IP Policy p1
   Administrative state: enable
   Reference count:      2
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)

   Referenced by interfaces:
      None

   Referenced by profiles:
      None
```

```
   Referenced by merge policies:
      mpl_5
      mpl_8

IP Policy p2
   Administrative state: enable
   Reference count:      1
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 20.1.1.1, order 100, rule 3 (active)
   Classifier control list: *, precedence 1000
      forward

   Referenced by interfaces:
      None

   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_5

IP Policy p3
   Administrative state: enable
   Reference count:      1
   Classifier control list: C1, precedence 80
      color yellow
   Classifier control list: C4, precedence 900
      color red

   Referenced by interfaces:
      None

   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_8

IP Policy mpl_5
   Administrative state: enable
   Reference count:      1
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)
           next-hop 20.1.1.1, order 100, rule 3 (reachable)
   Classifier control list: *, precedence 1000
      forward

   Referenced by interfaces:
      ATM5/0.2  output policy, statistics enabled, virtual-router default
```

```
   Referenced by profiles:
      None

   Component policies:
      p1
      p2

IP Policy mpl_8
   Administrative state: enable
   Reference count:      1
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C1, precedence 80
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)
           next-hop 20.1.1.1, order 100, rule 3 (reachable)
      color yellow
   Classifier control list: C4, precedence 900
      color red

   Referenced by interfaces:
      ATM5/0.1  input policy, statistics enabled, virtual-router default

   Referenced by profiles:
      None

   Component policies:
      p1
      p3
```

15. Detach p1 from atm 5/0.1. Merge policy mpl_8 is detached and deleted, and only p3 is attached to this interface.

```
host1(config)#interface atm 5/0.1
host1(config-subif)#no ip policy input p1
host1(config-subif)#exit
```

16. Display policies to verify that p3 is attached to atm 5/0.1 and mpl_8 is removed.

```
host1#show policy-list

                                  Policy Table
                                  ------ -----
IP Policy p1
   Administrative state: enable
   Reference count:      1
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)

   Referenced by interfaces:
      None
```

```
   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_5

IP Policy p2
   Administrative state: enable
   Reference count:      1
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 20.1.1.1, order 100, rule 3 (active)
   Classifier control list: *, precedence 1000
      forward

   Referenced by interfaces:
      None

   Referenced by profiles:
      None

   Referenced by merge policies:
      mpl_5

IP Policy p3
   Administrative state: enable
   Reference count:      1
   Classifier control list: C1, precedence 80
      color yellow
   Classifier control list: C4, precedence 900
      color red

   Referenced by interfaces:
      ATM5/0.1  input policy, statistics disabled, virtual-router default

   Referenced by profiles:
      None

   Referenced by merge policies:
      None

IP Policy mpl_5
   Administrative state: enable
   Reference count:      1
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)
           next-hop 20.1.1.1, order 100, rule 3 (reachable)
   Classifier control list: *, precedence 1000
      forward
```

```
   Referenced by interfaces:
      ATM5/0.2  output policy, statistics enabled, virtual-router default

   Referenced by profiles:
      None

   Component policies:
      p1
      p2
```

17. Detach p3 from atm 5/0.1.

```
host1(config)#interface atm 5/0.1
host1(config-subif)#no ip policy input p3
host1(config-subif)#exit
```

18. Detach p1 from atm 5/0.2. Merge policy mpl_5 is detached and deleted and only p2 is now attached.

```
host1(config)#interface atm 5/0.2
host1(config-subif)#no ip policy output p1
host1(config-subif)#exit
```

19. Detach p2 from atm 5/0.2.

```
host1(config)#interface atm 5/0.2
host1(config-subif)#no ip policy output p2
host1(config-subif)#exit
```

20. Display policies to verify that no merge policies exist and that all other policies have a 0 reference count because they are not attached anywhere.

```
host1#show policy-list

                                  Policy Table
                                  ------ -----
IP Policy p1
   Administrative state: enable
   Reference count:      0
   Classifier control list: C2, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 10.1.1.1, order 100, rule 2 (active)

IP Policy p2
   Administrative state: enable
   Reference count:      0
   Classifier control list: C3, precedence 10
      filter
   Classifier control list: C1, precedence 90
      forward
         Virtual-router: default
         List:
           next-hop 20.1.1.1, order 100, rule 3 (active)
   Classifier control list: *, precedence 1000
      forward
```

```
IP Policy p3
   Administrative state: enable
   Reference count:      0
   Classifier control list: C1, precedence 80
      color yellow
   Classifier control list: C4, precedence 900
      color red
```

Related Topics

- Merging Policies Overview on page 87
- *Chapter 9, Monitoring Policy Management*

Parent Group Merge Algorithm

The parent group merge algorithm enables the system to merge policies that contain references to parent groups and create an internal parent group for each internal parent group in a component policy in the final merged policy. There is a one-to-one correspondence between an internal parent group in the merged policy and an internal parent group in a component policy.

NOTE: The naive parent group merging algorithm is not compatible with this parent group merge algorithm. If you have service definitions that used the naive parent group algorithm, you need to modify those service definitions to work with this algorithm.

- If there is no existing internal parent group with the same name in the merged policy, the system creates a corresponding internal parent group with the same name.
- If an internal parent group with the same name already exists, the system uses a name built by appending an internally generated sequence number to the name of the internal parent group in the component policy.
- If the length of the name exceeds the maximum length allowed, the policy merge fails.
- If a classifier group in a component policy refers to an internal parent group, the same classifier group in the merged policy corresponds to the internal parent group in the merged policy.
- If a classifier group in a component policy refers to an external parent group, the same classifier group in the merged policy refers to the same external parent group.
- If there is a conflict where two or more component policies contain the same classifier group referring to an internal parent group in a corresponding component policy or to an external parent group, then last one is used.

In the following example, component policies P1 and P2 create the merged policy mpl_88000001.

```
host1#show policy-list P1

                                  Policy Table
                                  ------ -----
IP Policy P1
   Administrative state: enable
   Reference count:      1
   Classifier control list: *, precedence 100, parent-group Z
    forward
   Classifier control list: A, precedence 100, parent-group X
forward
   Classifier control list: B, precedence 100, parent-group X
    forward
   Classifier control list: C, precedence 100, external parent-group EPG1
parameter foo
    forward
   Classifier control list: D, precedence 100, external parent-group EPG1
parameter foo
    forward

   Parent group: X, parent-group Z
      rate-limit-profile R1
   Parent group: Z
      rate-limit-profile R2

host1#show policy-list P2

                                  Policy Table
                                  ------ -----
IP Policy P2
   Administrative state: enable
   Reference count:      1
   Classifier control list: B, precedence 100, parent-group X
    forward
   Classifier control list: C, precedence 100, parent-group Y
       forward
   Classifier control list: D, precedence 100, external parent-group EPG2
parameter abcd
    forward

   Parent group: X, parent-group Y
      rate-limit-profile R3
   Parent group: Y
      rate-limit-profile R4

host1#show policy-list mpl_88000001

                                  Policy Table
                                  ------ -----

IP Policy mpl_88000001
Administrative state: enable
Reference count:      1
   Classifier control list: *, precedence 100, parent-group Z
    forward
   Classifier control list: A, precedence 100, parent-group X
    forward
   Classifier control list: B, precedence 100, parent-group X_1
    forward
```

```
   Classifier control list: C, precedence 100, parent-group Y
forward
   Classifier control list: D, precedence 100, external parent-group EPG2
parameter abcd
    forward

   Parent group: X, parent-group Z
      rate-limit-profile R1
   Parent group: Z
      rate-limit-profile R2
   Parent group: X_1, parent-group P2_Y
      rate-limit-profile R3
   Parent group: Y
      rate-limit-profile R4

   Referenced by interfaces:
      ATM5/0.1  input policy, statistics enabled, virtual-router default

   Referenced by profiles:
      None

   Component policies:
      P1
      P2
```

Overlapping Classification for IP Input Policy

IP auxiliary input policy can be used with IP input policy to provide overlapping classification. Two policies, each with a set of independent rules and actions, run in sequence so that each policy can independently produce a set of actions in sequence. A packet that matches both the input policies and auxiliary input policies is subject to both sets of policy actions.

E-series routers allow four input and two output policies per IP interface:

- One secure input policy
- Three nonsecure input policies
- One secure output policy
- One nonsecure output policy

Each classifier-group has a set of associated actions that is taken if it is the highest priority match. The system performs only one set of actions per policy attachment. By using an input and secondary-input policy, you can have overlapping classification with multiple policy actions on ingress. Overlapping classification on egress is not supported.

An additional policy attachment point enables overlapping classification within the input classification stage, between the input and secondary-input stages. There are five attachment points for IP policies that are executed in series:

- input
- secondary-input

- secure-input
- output
- secure-output

An explicit filter action, a forward action with a null next-interface, or a rate-limit action can cause an immediate packet discard at any stage. Other actions, such are marking and coloring can be done at each stage, with the last of each of these actions taking precedence over the others.

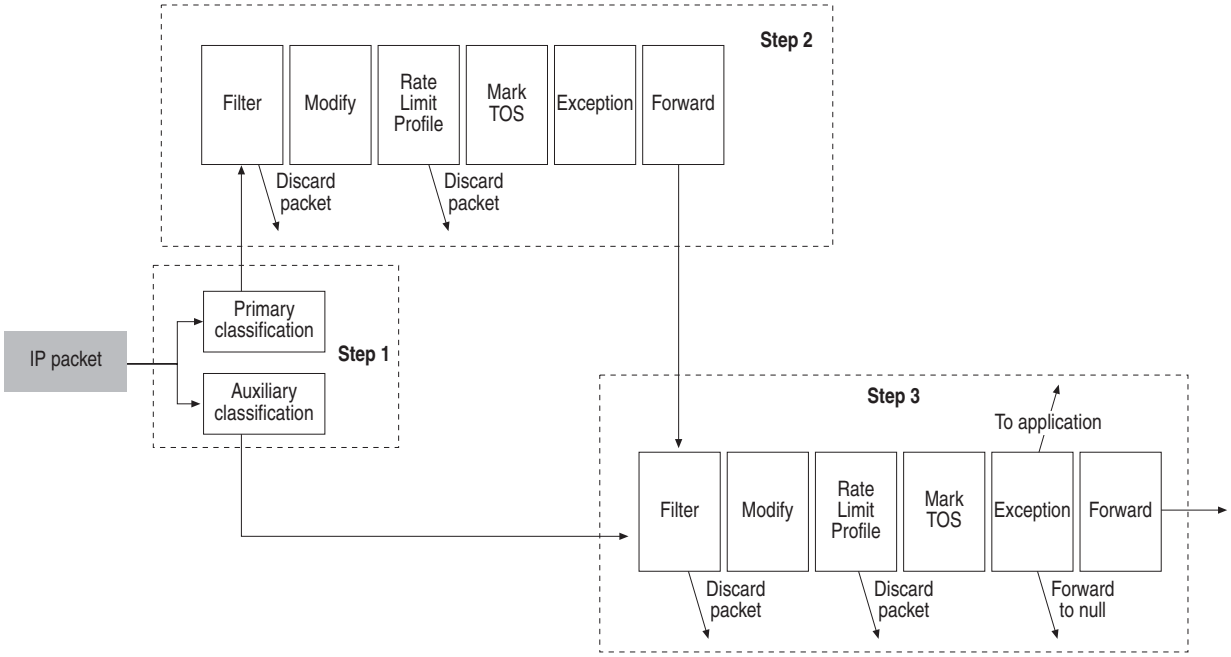
For example, unique policies can be attached at each stage, all of which mark the IP TOS field differently. The packet then exits the router with the TOS value that was set in the output policy stage. However, if TOS is also used as a classification (input) term for each of these policies, three different TOS values are presented to the classifier:

- Original TOS received
- TOS modified by the input policy
- TOS value modified by the secondary-input policy

Figure 7 shows the input policy stage after the addition of the auxiliary substage. It is divided into three steps:

1. Apply classification for both substages.
2. Perform policy actions (if any) for the primary attachment.
3. Perform policy actions (if any) for the auxiliary attachment.

Figure 7: Input Policy with Primary Stage and Auxiliary Substage



The order of policy action execution for each attachment is:

1. Filter
2. Modify (includes setting of color, traffic class, user packet class) and Log
3. Rate limit profile/color
4. Mark TOS
5. Exception
6. Forward

Starting Policy Processing

Input and auxiliary-input classification operations, specified by the details of each policy, are performed in parallel. Classifier inputs for both policies are determined concurrently using the initial values of the classification terms. Policy attachments within a stage cannot communicate between the input and auxiliary-input classification operations. For example, any changes made by the input attachment to traffic-class, color, TOS, or user packet class are not visible in the auxiliary-input policy classification. If this communication is needed, it can only be done between different policy stages, rather than within a single stage.

The results of the input policy actions are passed forward to the auxiliary-input policy action processing. This means that a color-aware rate limit profile action in the auxiliary substage recognizes any change in color caused by primary policy actions.

Processing the Classifier Result

The classifier result of the input policy attachment is processed and a set of actions is identified. When you configure filter, it is the first action taken and immediately discards the packet. This is followed by any modification, such as mark or logging. If a rate limit profile is configured, the packet is dropped or colored. If the packet is not dropped, it is sent to the exception path (if configured). If the packet is not exceptioned, any configured forward action is saved in the packet for use later (unless overridden in Step 3). (See Figure 7 on page 111.)

Some information generated by the action processing in Step 2 is forwarded to Step 3, where it may affect the action processing for the auxiliary-input attachment. This information can include color, exception information, and forwarding information. The color can affect a rate-limit in the auxiliary-input attachment. Step 3 acts on the exception and forwarding information, if it is not overridden by similar actions from the auxiliary-input attachment.

The transmit information (transmit conditional, transmit unconditional, transmit final) generated with hierarchical policies does not carry forward from input to auxiliary-input action processing.

Processing the Auxiliary-Input Policy Attachment

If the packet is not filtered or exceptioned in policy Step 2, the classifier result of the auxiliary policy attachment is processed and a set of actions identified. The packet can be filtered or exceptioned at this time. These operations, if configured, are performed regardless of whether a forward action was performed in Step 2. If the packet is not discarded, either by a filter action or a rate limit, it can be exceptioned (if configured). If the packet is not filtered, rate-limited, or exceptioned, any configured forward action is applied and overrides any forward action from Step 2. If no forward action is configured, any forward action from Step 2 applies.

Policy Actions

The set of actions in the following list specified by the input and auxiliary-input policy attachments are executed in the order: input, auxiliary-input.

- Color packet action—Explicitly sets the packet color. Each policy attachment can set the color and the final value persists. A rate limit profile action can also set the color, which overrides the value of the color packet action.

- Mark action—Each attachment can set the TIP TOS, TOS precedence, and DS fields. The cumulative result of all configured mark actions determines the resulting value of these fields.

- Mirror action—Executes in the order: secure input policy follows secondary input policy, secure output policy follows output policy. Mirror is the only supported action for secure policies.

- Rate-limit profile action—Can be specified by any nonsecure input policy attachment. This enables the application of multiple rate limits either within a policy stage or across policy stages. These rate limits run serially; if the rate limit imposed in the primary substage causes the packet to drop, the auxiliary rate limit does not run and the associated token buckets are not affected. If you configure more than a single rate limit per interface, it significantly impacts forwarding performance. Attaching two policies with rate limit profiles in the same policy stage is equivalent to having two policies attached in the same order, but in separate stages.

- Traffic class action—If both the input and auxiliary-input attachments need this action, the value configured in the auxiliary policy overwrites that of the primary policy.

- User packet class action—Can be set twice per stage, with the second value overriding the first.

- The filter, next-hop, forward interface, and forward next-hop actions—Mutually exclusive within a classifier group. However, two policies in series can result in conflicting actions, which are resolved using the following precedence rules:

    - The filter action has highest priority. A filter action in input or auxiliary-input policy always prevails.

    - The exception action takes precedence over forward actions.

    - If multiple exception actions are required by the policy attachments, the last one takes precedence.

    - If forward operations are required by both input and auxiliary-input policy attachments, the auxiliary-input forward action takes precedence.

In Table 11, the filter action for the input policy takes precedence over the others so that if a filter action is configured for either policy, the packet is filtered. If neither policy has a filter action, but both policies specify a forward action, the action specified by the auxiliary policy takes precedence. If only one policy specifies a forwarding action, that action is executed. The next-hop rule is inoperative for auxiliary-input policies, just as it is for secondary input policies. This policy rule has been superseded (but not replaced) by the forward next-hop rule, which is operative for auxiliary-input policies.

Table 11: Filter, Forward, and Exception Action Resolution for Input and Auxiliary-Input Policies

Input Action	Secondary Input Action					
	None	Exception	Filter	Next-hop	Forward Interface	Forward Next-hop
None	None	Exception Auxiliary	Filter	None	Forward Interface Auxiliary	Forward Next-hop Auxiliary
Exception	Exception Primary	Exception Auxiliary	Filter	Exception	Exception Primary	Exception Primary
Filter	Filter	Filter	Filter	Filter	Filter	Filter
Next-hop	Next-hop Primary	Exception Auxiliary	Filter	Next-hop Primary	Forward Interface Auxiliary	Forward Next-hop Auxiliary
Fwd Interface	Forward Interface Primary	Exception Auxiliary	Filter	Forward Interface Primary	Forward Interface Auxiliary	Forward Next-hop Auxiliary
Fwd next-hop	Forward Next-hop Primary	Exception Auxiliary	Filter	Forward Next-hop Primary	Forward Interface Auxiliary	Forward Next-hop Auxiliary