

Chapter 7

Classifying Packets Based on Services

On Adaptive Services (AS) PICs, there is an additional method of classifying traffic flows based on applications, such as stateful firewalls and network address translation (NAT).

This method allows you to configure a rule-based service that provides DiffServ code point (DSCP) marking and forwarding-class assignments for traffic transiting the AS PIC. The service enables you to specify matching by application, application set, source, destination address, and match direction, and uses a similar structure to other rule-based services such as stateful firewall. The service actions allow you to associate the DSCP alias or value, forwarding-class name, system log activity, or a preconfigured application profile with the matched packet flows.

To configure class-of-service (CoS) features on the Adaptive Services PIC, include the `cos` statements at the `[edit services]` hierarchy level:

```
cos {
  application-profile profile-name {
    sip-text {
      dscp (alias | bits);
      forwarding-class class-name;
    }
    sip-video {
      dscp (alias | bits);
      forwarding-class class-name;
    }
    sip-voice {
      dscp (alias | bits);
      forwarding-class class-name;
    }
  }
  rule rule-name {
    match-direction (input | output | input-output);
    term term-name {
      from {
        applications [ application-names ];
        application-sets [ set-names ];
        destination-address address;
        source-address address;
      }
    }
  }
}
```

```

then {
  application-profile profile-name;
  dscp (alias | bits);
  forwarding-class class-name;
  syslog;
  (reflexive | reverse) {
    application-profile profile-name;
    dscp (alias | bits);
    forwarding-class class-name;
    syslog;
  }
}
}
}
rule-set rule-set-name {
  [ rule rule-names ];
}
}

```

This chapter contains the following sections:

- Configuring the Class-of-Service Rule Set on page 78
- Configuring Class-of-Service Rule Content on page 79
- Output Packet Rewriting on page 83
- Example: Configuring Class-of-Service Properties on page 83

Configuring the Class-of-Service Rule Set

The `rule-set` statement defines a collection of CoS rules that determine what actions the router software performs on packets in the data stream. You define each rule by specifying a rule name and configuring terms. You then specify the order of the rules by including the `rule-set` statement at the `[edit services cos]` hierarchy level:

```

rule-set rule-set-name {
  rule rule-name1;
  rule rule-name2;
  rule rule-name3;
  ...
}

```

The router software processes the rules in the order in which you specify them in the configuration. If a term in a rule matches the packet, the router performs the corresponding action and the rule processing stops. If no term in a rule matches the packet, processing continues to the next rule in the rule set. If none of the rules match the packet, the packet is dropped by default.

Configuring Class-of-Service Rule Content

To configure a CoS rule, include the rule *rule-name* statement at the [edit services cos] hierarchy level:

```
rule rule-name {
  match-direction (input | output | input-output);
  term term-name {
    from {
      applications [ application-names ];
      application-sets [ set-names ];
      destination-address address;
      source-address address;
    }
    then {
      application-profile profile-name;
      dscp (alias | bits);
      forwarding-class class-name;
      syslog;
      (reflexive | reverse) {
        application-profile profile-name;
        dscp (alias | bits);
        forwarding-class class-name;
        syslog;
      }
    }
  }
}
```

Each CoS rule consists of a set of terms, similar to a filter configured at the [edit firewall] hierarchy level. A term consists of the following:

- **from** statement—Specifies the match conditions and applications that are included and excluded.
- **then** statement—Specifies the actions and action modifiers to be performed by the router software.

In addition, each rule must include a **match-direction** statement that specifies the direction in which the rule match is applied. To configure where the match is applied, include the **match-direction** statement at the [edit services cos rule *rule-name*] hierarchy level:

```
match-direction (input | output | input-output);
```

If you configure **match-direction input-output**, bidirectional rule creation is allowed.

The match direction is used with respect to the traffic flow through the AS PIC. When a packet is sent to the AS PIC, direction information is carried along with it.

With an interface service set, packet direction is determined by whether a packet is entering or leaving the interface on which the service set is applied.

With a next-hop service set, packet direction is determined by the interface used to route the packet to the AS PIC. If the inside interface is used to route the packet, the packet direction is **input**. If the outside interface is used to direct the packet to the AS PIC, the packet direction is **output**. For more information on inside and outside interfaces, see the *JUNOS Services Interfaces Configuration Guide*.

On the AS PIC, a flow lookup is performed. If no flow is found, rule processing is performed. All rules in the service set are considered. During rule processing, the packet direction is compared against rule directions. Only rules with direction information that matches the packet direction are considered.

The following sections describe CoS rule content in more detail:

- Configuring Class-of-Service Match Conditions on page 80
- Configuring Class-of-Service Actions on page 81

Configuring Class-of-Service Match Conditions

To configure CoS match conditions, include the **from** statement at the [edit services cos rule *rule-name* term *term-name*] hierarchy level:

```

from {
  applications [ application-names ];
  application-sets [ set-names ];
  destination-address address;
  source-address address;
}

```

You can use either the source address or the destination address as a match condition, in the same way that you would configure a firewall filter; for more information, see the *JUNOS Policy Framework Configuration Guide*.

If you omit the **from** term, the router accepts all traffic and the default protocol handlers take effect:

- User Datagram Protocol (UDP), Transmission Control Protocol (TCP), and Internet Control Message Protocol (ICMP) create a bidirectional flow with a predicted reverse flow.
- IP creates a unidirectional flow.

You can also include application protocol definitions that you have configured at the [edit applications] hierarchy level; for more information, see the *JUNOS Services Interfaces Configuration Guide*.

- To apply one or more specific application protocol definitions, include the `applications` statement at the [edit services cos rule *rule-name* term *term-name* from] hierarchy level.
- To apply one or more sets of application protocol definitions you have defined, include the `application-sets` statement at the [edit services cos rule *rule-name* term *term-name* from] hierarchy level.



NOTE: If you include a statement that specifies application protocols, the router derives port and protocol information from the corresponding configuration at the [edit applications] hierarchy level; you cannot specify these properties as match conditions.

Configuring Class-of-Service Actions

To configure CoS actions, include the `then` statement at the [edit services cos rule *rule-name* term *term-name*] hierarchy level:

```

then {
  application-profile profile-name;
  dscp (alias | bits);
  forwarding-class class-name;
  syslog;
  (reflexive | reverse) {
    application-profile profile-name;
    dscp (alias | bits);
    forwarding-class class-name;
    syslog;
  }
}

```

The principal CoS actions are as follows:

- `dscp`—Marks the packet with the specified DiffServ code point (DSCP) value or alias.
- `forwarding-class`—Assigns the packet to the specified forwarding class.

You can optionally set the configuration to record information in the system logging facility by including the `syslog` statement at the [edit services cos rule *rule-name* term *term-name* then] hierarchy level. This statement overrides any `syslog` setting included in the service set or interface default configuration.

For information about some additional CoS actions, see the following sections:

- Configuring Application Profiles on page 82
- Configuring Reflexive and Reverse CoS Actions on page 82

Configuring Application Profiles

You can optionally define one or more application profiles for inclusion in CoS actions. To configure, include the `application-profile` statement at the `[edit services cos]` hierarchy level:

```
application-profile profile-name {
  sip-text {
    dscp (alias | bits);
    forwarding-class class-name;
  }
  sip-video {
    dscp (alias | bits);
    forwarding-class class-name;
  }
  sip-voice {
    dscp (alias | bits);
    forwarding-class class-name;
  }
}
```

The `application-profile` statement includes three fixed components: `sip-text`, `sip-video`, and `sip-voice`. You can set the appropriate `dscp` and `forwarding-class` values for each component within the application profile.

You can apply the application profile to a CoS configuration by including it at the `[edit services cos rule rule-name term term-name then]` hierarchy level.

Configuring Reflexive and Reverse CoS Actions

It is important to understand that CoS services are unidirectional. It might be necessary to specify different treatments for flows in opposite directions.

Regardless of whether a packet matches the input, output, or input-output direction, flows in both directions are created. The difference is that a forward, reverse, or forward-and-reverse CoS action is associated with each flow. You should bear in mind that the flow in the opposite direction might end up having a CoS action associated with it, which you have not specifically configured.

To control the direction in which service is applied, separate from the direction in which the rule match is applied, you can configure the `(reflexive | reverse)` statement at the `[edit services cos rule rule-name term term-name then]` hierarchy level:

```
(reflexive | reverse) {
  application-profile profile-name;
  dscp (alias | bits);
  forwarding-class class-name;
  syslog;
}
```

The two actions are mutually exclusive. If nothing is specified, data flows inherit the CoS behavior of the forward control flow.

- `reflexive` causes the equivalent reverse CoS action to be applied to flows in the opposite direction.
- `reverse` allows you to define the CoS behavior for flows in the reverse direction.

Output Packet Rewriting

On M-series routers, you can configure rewrite rules to change packet header information and attach it to an output interface. Because these rules can possibly overwrite the DSCP marking configured on the AS PIC, it is important to create system-wide configurations carefully.

For example, knowing that the AS PIC can mark packets with any ToS or DSCP value and the output interface is restricted to only eight DSCP values, rewrite rules on the output interface condense the mapping from 64 to 8 values with overall loss of granularity. In this case, you have the following options:

- Remove rewrite rules in the output interface.
- Configure the output interface to include the most important mappings.

Example: Configuring Class-of-Service Properties

The following example show a CoS configuration containing two rules, one for input matching on a specified application set and the other for output matching on a specified source address:

```

services {
  cos {
    application-profile cosprofile {
      ftp {
        data {
          dscp af11;
          forwarding-class 1;
        }
      }
    }
    application-profile cosrevprofile {
      ftp {
        data {
          dscp af22;
        }
      }
    }
  }
  rule cosrule {
    match-direction input;
    term costerm {
      from {
        source-address {
          any-unicast;
        }
      }
      applications junos-ftp;
    }
  }
}

```

```

        then {
            dscp af33;
            forwarding-class 3;
            application-profile cosprofile;
            reverse {
                dscp af43;
                application-profile cosrevprofile;
            }
        }
    }
}
stateful-firewall {
    rule r1 {
        match-direction input;
        term t1 {
            from {
                application-sets junos-algs-outbound;
            }
            then {
                accept;
            }
        }
        term t2 {
            then {
                accept;
            }
        }
    }
}
service-set test {
    stateful-firewall-rules r1;
    cos-rules cosrule;
    interface-service {
        service-interface sp-1/3/0;
    }
}
}

```

Verifying Your Configuration

In addition to `show class-of-service` commands, you can issue the following operational mode commands to verify your configuration:

- `show services cos statistics diffserv`
- `show services cos statistics forwarding-class`