

Network Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: November 25, 2015

A. Choudhary  
S. Shah  
Cisco Systems  
M. Jethanandani  
Ciena Corporation  
Y. Yan  
B. Liu  
Huawei Technologies  
N. Strahle  
Juniper Networks  
May 24, 2015

**YANG Model for Diffserv**  
**draft-asechoud-netmod-diffserv-model-02**

Abstract

This document describes a YANG model of Differentiated Services for configuration and operations.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 25, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|                             |                                    |                    |
|-----------------------------|------------------------------------|--------------------|
| <a href="#">1.</a>          | Introduction . . . . .             | <a href="#">2</a>  |
| <a href="#">2.</a>          | Terminology . . . . .              | <a href="#">3</a>  |
| <a href="#">3.</a>          | Diffserv Model Design . . . . .    | <a href="#">3</a>  |
| <a href="#">4.</a>          | Diffserv Model . . . . .           | <a href="#">4</a>  |
| <a href="#">5.</a>          | Diffserv Modules . . . . .         | <a href="#">9</a>  |
| <a href="#">5.1.</a>        | IETF-DIFFSERV-CLASSIFIER . . . . . | <a href="#">9</a>  |
| <a href="#">5.2.</a>        | IETF-DIFFSERV-POLICY . . . . .     | <a href="#">16</a> |
| <a href="#">5.3.</a>        | IETF-DIFFSERV-ACTION . . . . .     | <a href="#">19</a> |
| <a href="#">5.4.</a>        | IETF-DIFFSERV-TARGET . . . . .     | <a href="#">28</a> |
| <a href="#">6.</a>          | Security Considerations . . . . .  | <a href="#">34</a> |
| <a href="#">7.</a>          | Acknowledgement . . . . .          | <a href="#">34</a> |
| <a href="#">8.</a>          | References . . . . .               | <a href="#">34</a> |
| <a href="#">8.1.</a>        | Normative References . . . . .     | <a href="#">34</a> |
| <a href="#">8.2.</a>        | Informative References . . . . .   | <a href="#">35</a> |
| <a href="#">Appendix A.</a> | Open Items . . . . .               | <a href="#">35</a> |
|                             | Authors' Addresses . . . . .       | <a href="#">35</a> |

## [1.](#) Introduction

This document defines a YANG [[RFC6020](#)] data model for the configuration, state data of Differentiated Services. Any RPC or notification definition is not part of this document. As many vendors have different object constructs to represent the same data, it has been tried to design this model in a very flexible, extensible and generic way to fit into most of the vendor requirements. The model is based on Differentiated Services (Diffserv) architecture and various references have been made to already available standard architecture documents.

Diffserv is a preferred approach for network service providers to offer services to different customers based on their different kinds of network quality-of-service (QoS) objectives. The traffic streams are differentiated based on Differentiated Services Code Points (DSCP) carried in the IP header of each packet. The DSCP markings are applied by upstream node or by the edge router on entry to the Diffserv network.



## 2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [\[RFC2119\]](#).

## 3. Diffserv Model Design

Diffserv architecture [\[RFC3289\]](#) [\[RFC2475\]](#) describes network node packet classification function and packet conditioning functions.

The complex classification is done at the edge of network and non-edge network devices conditions appropriately marked aggregate traffic based on per-hop behavior rules. Accordingly, a Multi-Field classifier matches the different fields in a packet and a Behavior Aggregated Classifier matches on DS codepoint field of a packet.

Packets MAY be grouped when a logical set of rules are applied on different packet header fields. Also, packet grouping MAY be done based on different values or range of values of same packet header field. Packet grouping MAY also be done based on presence of some values or range of values of a packet field or absence of such values or ranges. This diffserv model is flexible enough to support such logical grouping of packets.

A classifier entry can be stored as an object and used across different interfaces for either of inbound or outbound traffic. Any modification or deletion of such object will in turn results in such changes to the classifier on the corresponding interfaces. A classifier entry contains one or more packet conditioning functions. A packet conditioning function is typically based on direction of traffic and may drop, mark or delay network packets. A set of such classifier entries with corresponding conditioning functions when arranged in order of priority represents a diffserv policy. A diffserv policy MAY contain one or more classifier entries. Actions are configured as inline as compared of classifier entry which can be stored as object or configured inline in a diffserv policy. This is mainly because actions generally contain more specific parameters like meter rate, or RED threshold. Any new classifier entry in a policy MAY be inserted before or after any other existing classifier-entry [\[RFC6020\]](#). Such policies is stored as an object and used across different network device interfaces.

A meter qualifies if the traffic arrival rate is based on agreed upon rate and variability. A meter is generically modeled as qualifying rate and variability defined as a token bucket. Single rate meter [\[RFC2697\]](#) can be defined as two such token buckets with first defining the rate and committed burst and excess burst for second



bucket. Similarly, two rates meter [[RFC2698](#)][RFC2859] can be defined as two such token buckets with first and second defining the committed rate and committed burst parameters and peak rate and peak burst respectively. Different Vendors can extend it to have other types of meters as well.

Metered traffic to each token bucket MAY either be marked or remarked appropriately of the diffserv codepoint packet field or even MAY be dropped. Classified packets through a classifier entry MAY directly be marked.

Packets can be always dropped if exceed agreed upon rates or it could be queued and then dropped based on any of various algorithms. Queue dropping is based on the threshold configured and can head-drop, tail-drop or dropped based on Active Queue Management algorithm like Random Early Detection (RED). Packets can be scheduled out based on priority with minimum-rate or WFQ with bandwidth sharing. Priority scheduler allow queue to use the entire capacity of the interface unless higher priority traffic is queued to be scheduled. If combination of EF [[RFC3246](#)] and multiple AF [[RFC3260](#)] classes of traffic needs to be scheduled, a combination of priority and WFQ scheduler SHOULD be used. Traffic can be shaped by defining a max rate and burst for a leaky bucket profile.

#### **4. Diffserv Model**

The model have four YANG modules. ietf-diffserv-classifier consists of classifier entries identified by a classifier entry name. Each such entry contains list of filter entries. When no filter entry is present in a classifier entry, it matches all traffic. Each filter entry represent any of the filter type [[RFC6991](#)] of a multi-field classifier which can be logically AND/OR with other filter types in the same classifier-entry. The model is flexible enough to take multiple values of the same filter type.



```

module: ietf-diffserv-classifier
  +--rw classifiers
    +--rw classifier-entry* [classifier-entry-name]
      +--rw classifier-entry-name          string
      +--rw classifier-entry-descr?        string
      +--rw classifier-entry-filter-operation? identityref
      +--rw filter-entry* [filter-type filter-logical-not]
        +--rw filter-type                  identityref
        +--rw filter-logical-not           boolean
        +--rw (filter-param)?
          +--:(dscp)
            | +--rw dscp-cfg* [dscp-min dscp-max]
            |   +--rw dscp-min    inet:dscp
            |   +--rw dscp-max    inet:dscp
          +--:(source-ip-address)
            | +--rw source-ip-address-cfg* [source-ip-addr]
            |   +--rw source-ip-addr    inet:ip-prefix
          +--:(destination-ip-address)
            | +--rw destination-ip-address-cfg*
            |   [destination-ip-addr]
            |   +--rw destination-ip-addr    inet:ip-prefix
          +--:(source-port)
            | +--rw source-port-cfg*
            |   [source-port-min source-port-max]
            |   +--rw source-port-min    inet:port-number
            |   +--rw source-port-max    inet:port-number
          +--:(destination-port)
            | +--rw destination-port-cfg*
            |   [destination-port-min destination-port-max]
            |   +--rw destination-port-min    inet:port-number
            |   +--rw destination-port-max    inet:port-number
          +--:(protocol)
            +--rw protocol-cfg* [protocol-min protocol-max]
            +--rw protocol-min    uint8
            +--rw protocol-max    uint8

```

An ietf-diffserv-policy module contains list of policy objects identified by a policy name which MUST be provided. Each policy object contains list of classifier-entries either configured inline or referred as an object. Each such classifier entry is augmented by set of actions. A policy object MAY contain a child-policy in each classifier-entry. A child policy MAY further classify the traffic and execute actions on classified packets.

```

module: ietf-diffserv-policy
  +--rw policies

```



```

+--rw policy-entry* [policy-name]
  +--rw policy-name      string
  +--rw policy-descr?    string
  +--rw classifier-entry* [classifier-entry-name]
    +--rw classifier-entry-name      string
    +--rw classifier-entry-inline?    boolean
    +--rw classifier-entry-filter-oper? identityref
  +--rw filter-entry* [filter-type filter-logical-not]
    {policy-inline-classifier-config}?
    | +--rw filter-type      identityref
    | +--rw filter-logical-not    boolean
    | +--rw (filter-param)?
    |   +--:(dscp)
    |   | +--rw dscp-cfg* [dscp-min dscp-max]
    |   |   +--rw dscp-min    inet:dscp
    |   |   +--rw dscp-max    inet:dscp
    |   +--:(source-ip-address)
    |   | +--rw source-ip-address-cfg* [source-ip-addr]
    |   |   +--rw source-ip-addr    inet:ip-prefix
    |   +--:(destination-ip-address)
    |   | +--rw destination-ip-address-cfg*
    |   |   [destination-ip-addr]
    |   |   +--rw destination-ip-addr    inet:ip-prefix
    |   +--:(source-port)
    |   | +--rw source-port-cfg*
    |   |   [source-port-min source-port-max]
    |   |   +--rw source-port-min    inet:port-number
    |   |   +--rw source-port-max    inet:port-number
    |   +--:(destination-port)
    |   | +--rw destination-port-cfg*
    |   |   [destination-port-min destination-port-max]
    |   |   +--rw destination-port-min    inet:port-number
    |   |   +--rw destination-port-max    inet:port-number
    |   +--:(protocol)
    |   | +--rw protocol-cfg* [protocol-min protocol-max]
    |   |   +--rw protocol-min    uint8
    |   |   +--rw protocol-max    uint8
  +--rw classifier-action-entry-cfg* [action-type]
    | +--rw action-type      identityref
    | +--rw (action-cfg-params)?
    |   +--:(marking)
    |   | +--rw action:marking-cfg
    |   |   +--rw action:dscp?    inet:dscp
    |   +--:(priority)
    |   | +--rw action:priority-cfg
    |   |   +--rw action:priority-level?    uint8
    |   +--:(meter)
    |   | +--rw action:meter-cfg

```



```

|      |      +--rw action:meter-list* [meter-id]
|      |      |      +--rw action:meter-id          uint16
|      |      |      +--rw action:meter-rate?       uint64
|      |      |      +--rw (burst-type)?
|      |      |      |      +--:(size)
|      |      |      |      |      +--rw action:burst-size?      uint64
|      |      |      |      +--:(interval)
|      |      |      |      |      +--rw action:burst-interval?  uint64
|      |      |      +--rw action:color
|      |      |      |      +--rw action:classifier-entry-name?
|      |      |      |      string
|      |      |      |      +--rw action:classifier-entry-descr?
|      |      |      |      string
|      |      |      |      +--rw action:
|      |      |      |      classifier-entry-filter-operation?
|      |      |      |      identityref
|      |      |      +--rw action:succeed-action
|      |      |      |      +--rw action:meter-action-type?
|      |      |      |      identityref
|      |      |      |      +--rw action:next-meter-id?      uint16
|      |      |      |      +--rw (val)?
|      |      |      |      |      +--:(meter-action-mark)
|      |      |      |      |      |      +--rw action:dscp?   inet:dscp
|      |      |      |      |      +--:(meter-action-drop)
|      |      |      |      |      |      +--rw action:drop-action? empty
|      |      |      +--rw action:fail-action
|      |      |      |      +--rw action:meter-action-type?
|      |      |      |      identityref
|      |      |      |      +--rw action:next-meter-id?      uint16
|      |      |      |      +--rw (val)?
|      |      |      |      |      +--:(meter-action-mark)
|      |      |      |      |      |      +--rw action:dscp?   inet:dscp
|      |      |      |      |      +--:(meter-action-drop)
|      |      |      |      |      |      +--rw action:drop-action? empty
|      |      +--:(min-rate)
|      |      |      +--rw action:min-rate-cfg
|      |      |      |      +--rw action:min-rate?      uint64
|      |      +--:(max-rate)
|      |      |      +--rw action:max-rate-cfg
|      |      |      |      +--rw action:absolute-rate?   uint64
|      |      |      |      +--rw (burst-type)?
|      |      |      |      |      +--:(size)
|      |      |      |      |      |      +--rw action:burst-size?  uint64
|      |      |      |      |      +--:(interval)
|      |      |      |      |      |      +--rw action:burst-interval? uint64
|      |      +--:(algorithmic-drop)
|      |      |      +--rw (drop-algorithm)?
|      |      |      |      +--:(always-drop)

```



```

|         | +--rw action:drop-cfg
|         |   +--rw action:drop-action?   empty
|         +---:(tail-drop)
|         |   +--rw action:tail-drop-cfg
|         |   +--rw action:qlimit-dscp-thresh*
|         |       [dscp-min dscp-max]
|         |       +--rw action:dscp-min      inet:dscp
|         |       +--rw action:dscp-max      inet:dscp
|         |       +--rw action:threshold
|         |       +--rw (threshold-type)?
|         |       +---:(size)
|         |       |   +--rw action:threshold-size?
|         |       |       uint64
|         |       +---:(interval)
|         |       +--rw action:
|         |           threshold-interval?
|         |           uint64
|         +---:(random-detect)
|         +--rw action:random-detect-cfg
|         |   {aqm-red-support}?
|         +--rw action:exp-weighting-const? uint32
|         +--rw action:red-min-thresh
|         |   +--rw action:threshold
|         |   +--rw (threshold-type)?
|         |   +---:(size)
|         |   |   +--rw action:threshold-size?
|         |   |       uint64
|         |   +---:(interval)
|         |   +--rw action:
|         |       threshold-interval?
|         |       uint64
|         +--rw action:red-max-thresh
|         |   +--rw action:threshold
|         |   +--rw (threshold-type)?
|         |   +---:(size)
|         |   |   +--rw action:threshold-size?
|         |   |       uint64
|         |   +---:(interval)
|         |   +--rw action:
|         |       threshold-interval?
|         |       uint64
|         +--rw action:mark-probability? uint32
+--rw child-policy? leafref {hierarchical-policy-support}?

```

ietf-diffserv-action module contains set of diffserv actions which are augmented to ietf-diffserv-policy module and to ietf-diffserv-target module. Marking sets Diffserv codepoint value in the



classified packet. Color-aware and Color-blind meters can be configured. Action counters are defined as grouping and are currently not augmented to any diffserv module.

ietf-diffserv-target module contains reference of diffserv-policy for either direction of network traffic and is augmented to ietf-interfaces [[RFC7223](#)] module.

```
module: ietf-diffserv-target
augment /if:interfaces/if:interface:
  +--rw diffserv-target-entry* [direction policy-name]
    +--rw direction                identityref
    +--rw policy-name              string
    +--ro diffserv-target-classifier-statistics*
      [classifier-entry-name parent-path]
      +--ro classifier-entry-name    string
      +--ro parent-path              string
      +--ro classifier-entry-statistics
      | +--ro classified-pkts?      uint64
      | +--ro classified-bytes?    uint64
      | +--ro classified-rate?     uint64
      +--ro meter-statistics* [meter-id]
      | +--ro meter-id              uint16
      | +--ro meter-succeed-pkts?   uint64
      | +--ro meter-succeed-bytes?  uint64
      | +--ro meter-failed-pkts?    uint64
      | +--ro meter-failed-bytes?   uint64
      +--ro queuing-statistics
      | +--ro output-pkts?          uint64
      | +--ro output-bytes?         uint64
      | +--ro queue-size-pkts?     uint64
      | +--ro queue-size-bytes?    uint64
      | +--ro drop-pkts?           uint64
      | +--ro drop-bytes?          uint64
      +--ro red-stats
      | +--ro early-drop-pkts?     uint64
      | +--ro early-drop-bytes?    uint64
```

## 5. Diffserv Modules

### 5.1. IETF-DIFFSERV-CLASSIFIER

```
<CODE BEGINS>file "ietf-diffserv-classifier@2014-09-07.yang"
module ietf-diffserv-classifier {
  yang-version 1;
```



```
namespace "urn:ietf:params:xml:ns:yang:ietf-diffserv-classifier";
prefix classifier;

import ietf-inet-types {
  prefix inet;
}

organization "IETF NETMOD (Netmod Working Group) Working Group";
contact
  "WG Web:   <http://tools.ietf.org/wg/netmod/>
   WG List:  <mailto:netmod@ietf.org>

   WG Chair: Jurgen Schonwalder
              <mailto:j.schoenwaelder@jacobs-university.de>

   WG Chair: Tom Nadeau
              <mailto:tnadeau@lucidvision.com>

   Editor:   Aseem Choudhary
              <mailto:asechoud@cisco.com>

   Editor:   Shitanshu Shah
              <mailto:svshah@cisco.com>";
description
  "This module contains a collection of YANG definitions for
   configuring diffserv specification implementations.

   Copyright (c) 2014 IETF Trust and the persons identified as
   authors of the code.  All rights reserved.

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Simplified BSD License
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (http://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC XXXX; see
   the RFC itself for full legal notices.";

revision 2015-04-07 {
  description
    "Latest revision of diffserv based classifier";
  reference "RFC XXXX";
}

feature policy-inline-classifier-config {
  description
```



```
    " This feature allows classifier configuration
      directly under policy.";
  }

  identity filter-type {
    description
      " This is identity of base filter-type";
  }

  identity dscp {
    base filter-type;
    description
      "DSCP filter-type";
  }

  identity source-ip-address {
    base filter-type;
    description
      "source-ip-address filter-type";
  }

  identity destination-ip-address {
    base filter-type;
    description
      "destination-ip-address filter-type";
  }

  identity source-port {
    base filter-type;
    description
      "source-port filter-type";
  }

  identity destination-port {
    base filter-type;
    description
      "destination-port filter-type";
  }

  identity protocol {
    base filter-type;
    description
      "protocol filter-type";
  }

  identity classifier-entry-filter-operation-type {
    description
      "Classifier entry filter logical operation";
```



```
}

identity match-any-filter {
  base classifier-entry-filter-operation-type;
  description
    "Classifier entry filter logical OR operation";
}

identity match-all-filter {
  base classifier-entry-filter-operation-type;
  description
    "Classifier entry filter logical AND operation";
}

grouping filters {
  description
    "Filters in a Classifier entry";
  leaf filter-type {
    type identityref {
      base filter-type;
    }
    description
      "This leaf defines type of the filter";
  }
  leaf filter-logical-not {
    type boolean;
    description
      "
        This is logical-not operator for a filter. When true, it
        indicates filter looks for absence of a pattern defined
        by the filter
      ";
  }
}

choice filter-param {
  description
    "Choice of filter types";
  case dscp {
    list dscp-cfg {
      key "dscp-min dscp-max";
      description
        "list of dscp ranges";
      leaf dscp-min {
        type inet:dscp;
        description
          "Minimum value of dscp range";
      }
      leaf dscp-max {
        type inet:dscp;
      }
    }
  }
}
```



```
        description
            "maximum value of dscp range";
    }
}
description
    "Filter containing list of dscp ranges";
}
case source-ip-address {
    list source-ip-address-cfg {
        key "source-ip-addr";
        description
            "list of source ip address";
        leaf source-ip-addr {
            type inet:ip-prefix;
            description
                "source ip prefix";
        }
    }
}
description
    "Filter containing list of source ip addresses";
}
case destination-ip-address {
    list destination-ip-address-cfg {
        key "destination-ip-addr";
        description
            "list of destination ip address";
        leaf destination-ip-addr {
            type inet:ip-prefix;
            description
                "destination ip prefix";
        }
    }
}
description
    "Filter containing list of destination ip address";
}
case source-port {
    list source-port-cfg {
        key "source-port-min source-port-max";
        description
            "list of ranges of source port";
        leaf source-port-min {
            type inet:port-number;
            description
                "minimum value of source port range";
        }
        leaf source-port-max {
            type inet:port-number;
            description
```



```
        "maximum value of source port range";
    }
}
description
    "Filter containing list of source-port ranges";
}
case destination-port {
    list destination-port-cfg {
        key "destination-port-min destination-port-max";
        description
            "list of ranges of destination port";
        leaf destination-port-min {
            type inet:port-number;
            description
                "minimum value of destination port range";
        }
        leaf destination-port-max {
            type inet:port-number;
            description
                "maximum value of destination port range";
        }
    }
}
description
    "Filter containing list of destination-port ranges";
}
case protocol {
    list protocol-cfg {
        key "protocol-min protocol-max";
        description
            "list of ranges of protocol values";
        leaf protocol-min {
            type uint8 {
                range "0..255";
            }
            description
                "minimum value of protocol range";
        }
        leaf protocol-max {
            type uint8 {
                range "0..255";
            }
            description
                "maximum value of protocol range";
        }
    }
}
description
    "Filter Type Protocol";
}
```



```
    }
  }

  grouping classifier-entry-generic-attr {
    description
      "Classifier attributes";
    leaf classifier-entry-name {
      type string;
      description
        "Diffserv classifier name";
    }
    leaf classifier-entry-descr {
      type string;
      description
        "Description of the class template";
    }
    leaf classifier-entry-filter-operation {
      type identityref {
        base classifier-entry-filter-operation-type;
      }
      default "match-any-filter";
      description
        "Filters are applicable as any or all filters";
    }
  }
}

grouping classifier-entry-inline-attr {
  description
    "Classifier inline attributes";
  leaf classifier-entry-inline {
    type boolean;
    default "false";
    description
      "Indication of inline classifier entry";
  }
  leaf classifier-entry-filter-oper {
    type identityref {
      base classifier-entry-filter-operation-type;
    }
    default "match-any-filter";
    description
      "Filters are applicable as any or all filters";
  }
  list filter-entry {
    if-feature policy-inline-classifier-config;
    must "classifier-entry-inline == true" {
      description
        "For inline filter configuration, inline attribute
```



```
        must be true";
    }
    key "filter-type filter-logical-not";
    uses filters;
    description
        "Filters configured inline in a policy";
}
}

container classifiers {
    description
        "list of classifier entry";
    list classifier-entry {
        key "classifier-entry-name";
        description
            "classifier entry template";
        uses classifier-entry-generic-attr;
        list filter-entry {
            key "filter-type filter-logical-not";
            uses filters;
            description
                "Filter configuration";
        }
    }
}
}
}
<CODE ENDS>
```

## **5.2. IETF-DIFFSERV-POLICY**

```
<CODE BEGINS>file "ietf-diffserv-policy@2015-04-07.yang"
module ietf-diffserv-policy {
    yang-version 1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-diffserv-policy";
    prefix policy;

    import ietf-diffserv-classifier {
        prefix classifier;
    }

    organization "IETF NETMOD (Netmod Working Group) Working Group";
    contact
        "WG Web:  <http://tools.ietf.org/wg/netmod/>
        WG List:  <mailto:netmod@ietf.org>

        WG Chair: Jurgen Schonwalder
```



<mailto:j.schoenwaelder@jacobs-university.de>

WG Chair: Tom Nadeau

<mailto:tnadeau@lucidvision.com>

Editor: Aseem Choudhary

<mailto:asechoud@cisco.com>

Editor: Shitanshu Shah

<mailto:svshah@cisco.com>";

description

"This module contains a collection of YANG definitions for configuring diffserv specification implementations.

Copyright (c) 2014 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in [Section 4.c](#) of the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>).

This version of this YANG module is part of RFC XXXX; see the RFC itself for full legal notices.";

revision 2015-04-07 {

description

"Latest revision of diffserv policy";

reference "RFC XXXX";

}

feature hierarchial-policy-support {

description

" This feature allows hierarchial policy to be configured";

}

grouping policy-generic-attr {

description

"Policy Attributes";

leaf policy-name {

type string;

description

"Diffserv policy name";

}

leaf policy-descr {

type string;



```
        description
            "Diffserv policy description";
    }
}

identity action-type {
    description
        "This base identity type defines action-types";
}

grouping classifier-action-entry-cfg {
    description
        "List of Configuration of classifier & associated actions";
    list classifier-action-entry-cfg {
        key "action-type";
        ordered-by user;
        description
            "Configuration of classifier & associated actions";
        leaf action-type {
            type identityref {
                base action-type;
            }
            description
                "This defines action type ";
        }
        choice action-cfg-params {
            description
                "Choice of action types";
        }
    }
}

container policies {
    description
        "list of policy templates";
    list policy-entry {
        key "policy-name";
        description
            "policy template";
        uses policy-generic-attr;
        list classifier-entry {
            key "classifier-entry-name";
            ordered-by user;
            description
                "Classifier entry configuration in a policy";
            leaf classifier-entry-name {
                type string;
                description

```



```
        "Diffserv classifier entry name";
    }
    uses classifier:classifier-entry-inline-attr;
    uses classifier-action-entry-cfg;
    leaf child-policy {
        if-feature hierarchial-policy-support;
        type leafref {
            path "/policies/policy-entry/policy-name";
        }
        description
            "Child Policy in the hierarchial configuration";
    }
}
}
}
}
}
<CODE ENDS>
```

### **5.3. IETF-DIFFSERV-ACTION**

```
<CODE BEGINS>file "ietf-diffserv-action@2015-04-07.yang"
module ietf-diffserv-action {
    namespace "urn:ietf:params:xml:ns:yang:ietf-diffserv-action";
    prefix action;

    import ietf-inet-types {
        prefix inet;
    }
    import ietf-diffserv-classifier {
        prefix classifier;
    }
    import ietf-diffserv-policy {
        prefix policy;
    }

    organization "IETF NETMOD (Netmod Working Group) Working Group";
    contact
        "WG Web:  <http://tools.ietf.org/wg/netmod/>
        WG List:  <mailto:netmod@ietf.org>

        WG Chair: Jurgen Schonwalder
                  <mailto:j.schoenwaelder@jacobs-university.de>

        WG Chair: Tom Nadeau
                  <mailto:tnadeau@lucidvision.com>
```



Editor: Aseem Choudhary  
<mailto:asechoud@cisco.com>

Editor: Shitanshu Shah  
<mailto:svshah@cisco.com>";

description

"This module contains a collection of YANG definitions for configuring diffserv specification implementations.

Copyright (c) 2014 IETF Trust and the persons identified as authors of the code. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, is permitted pursuant to, and subject to the license terms contained in, the Simplified BSD License set forth in [Section 4.c](#) of the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>).

This version of this YANG module is part of RFC XXXX; see the RFC itself for full legal notices.";

revision 2015-04-07 {

description

"Latest revision for diffserv actions";  
reference "RFC XXXX";

}

feature hierarchial-policy-support {

description

" This feature allows hierarchial policy to be configured";

}

feature aqm-red-support {

description

" This feature allows AQM RED to be configured";

}

grouping dscp-range {

description

"dscp range definition";

leaf dscp-min {

type inet:dscp;

description

"Minimum of dscp range";

}

leaf dscp-max {

type inet:dscp;



```
        description
            "Maximum of dscp range";
    }
}

grouping burst {
    description
        "burst size or interval configuration";
    choice burst-type {
        case size {
            leaf burst-size {
                type uint64;
                units "bytes";
                description
                    "burst size";
            }
        }
        case interval {
            leaf burst-interval {
                type uint64;
                units "microsecond";
                description
                    "burst interval";
            }
        }
    }
    description
        "Choice of burst type";
}

grouping threshold {
    description
        "Threshold Parameters";
    container threshold {
        description
            "threshold";
        choice threshold-type {
            case size {
                leaf threshold-size {
                    type uint64;
                    units "bytes";
                    description
                        "Threshold size";
                }
            }
            case interval {
                leaf threshold-interval {
                    type uint64;
                }
            }
        }
    }
}
```



```
        units "microsecond";
        description
            "Threshold interval";
    }
}
description
    "Choice of threshold type";
}
}

identity marking {
    base policy:action-type;
    description
        "marking action type";
}

identity meter {
    base policy:action-type;
    description
        "meter action type";
}

identity priority {
    base policy:action-type;
    description
        "priority action type";
}

identity min-rate {
    base policy:action-type;
    description
        "min-rate action type";
}

identity max-rate {
    base policy:action-type;
    description
        "max-rate action type";
}

identity algorithmic-drop {
    base policy:action-type;
    description
        "algorithmic-drop action type";
}

identity drop-type {
```



```
    description
      "drop algorithm";
  }

  identity always-drop {
    base drop-type;
    description
      "always drop algorithm";
  }

  identity tail-drop {
    base drop-type;
    description
      "tail drop algorithm";
  }

  identity random-detect {
    base drop-type;
    description
      "random detect algorithm";
  }

  identity meter-action-type {
    description
      "action type in a meter";
  }

  identity meter-action-drop {
    base meter-action-type;
    description
      "drop action type in a meter";
  }

  identity meter-action-set {
    base meter-action-type;
    description
      "mark action type in a meter";
  }

  grouping drop {
    leaf drop-action {
      type empty;
      description
        "always drop algorithm";
    }
    description
      "the drop action";
  }
```



```
grouping queue-limit {
  list qlimit-dscp-thresh {
    key "dscp-min dscp-max";
    uses dscp-range;
    uses threshold;
    description
      "the queue limit per dscp range";
  }
  description
    "the queue limit beyond which queue will not hold any packet";
}

grouping meter-action-params {
  leaf meter-action-type {
    type identityref {
      base meter-action-type;
    }
    description
      "meter action type";
  }
  leaf next-meter-id {
    type uint16;
    description
      "next meter identifier";
  }
  choice val {
    case meter-action-mark {
      uses marking;
      description
        "meter action: mark";
    }
    case meter-action-drop {
      description
        "meter action: drop";
      uses drop;
    }
  }
  description
    " meter action based on choice of meter action type";
}
description
  "meter action parameters";
}

grouping meter {
  leaf meter-id {
    type uint16;
    description
      "meter identifier";
  }
}
```



```
    }
    leaf meter-rate {
        type uint64;
        units "bits-per-second";
        description
            "meter rate";
    }
    uses burst;
    container color {
        uses classifier:classifier-entry-generic-attr;
        description
            "color aware & color blind attributes container";
    }
    container succeed-action {
        uses meter-action-params;
        description
            "confirm action";
    }
    container fail-action {
        uses meter-action-params;
        description
            "exceed action";
    }
    description
        "meter attributes";
}

grouping priority {
    leaf priority-level {
        type uint8;
        description
            "priority level";
    }
    description
        "priority attributes";
}

grouping min-rate {
    leaf min-rate {
        type uint64;
        units "bits-per-second";
        description
            "minimum rate";
    }
    description
        "minimum rate grouping";
}
```



```
grouping marking {
  leaf dscp {
    type inet:dscp;
    description
      "dscp marking";
  }
  description
    "marking grouping";
}

grouping max-rate {
  leaf absolute-rate {
    type uint64;
    units "bits-per-second";
    description
      "rate in bits per second";
  }
  uses burst;
  description
    "maximum rate attributes";
}

grouping red-threshold {
  container red-min-thresh {
    uses threshold;
    description
      "Minimum threshold";
  }
  container red-max-thresh {
    uses threshold;
    description
      "Maximum threshold";
  }
  leaf mark-probability {
    type uint32 {
      range "1..1000";
    }
    description
      "Mark probability";
  }
  description
    "RED threshold attributes";
}

grouping randomdetect {
  leaf exp-weighting-const {
    type uint32;
    description
```



```
        "Exponential weighting constant factor for red profile ";
    }
    uses red-threshold;
    description
        "Random detect attributes";
}

augment "/policy:policies/policy:policy-entry" +
    "/policy:classifier-entry" +
    "/policy:classifier-action-entry-cfg" +
    "/policy:action-cfg-params" {
    case marking {
        container marking-cfg {
            uses marking;
            description
                "Marking configuration container";
        }
    }
    case priority {
        container priority-cfg {
            uses priority;
            description
                "priority attributes container";
        }
    }
    case meter {
        container meter-cfg {
            list meter-list {
                key "meter-id";
                uses meter;
                description
                    "Meter configuration";
            }
            description
                "Meter list configuration container";
        }
    }
    case min-rate {
        container min-rate-cfg {
            uses min-rate;
            description
                "min guaranteed bandwidth";
        }
    }
    case max-rate {
        container max-rate-cfg {
            uses max-rate;
            description
```



```
        "maximum rate attributes";
    }
}
case algorithmic-drop {
    choice drop-algorithm {
        case always-drop {
            container drop-cfg {
                uses drop;
                description
                    "Always Drop configuration container";
            }
        }
        case tail-drop {
            container tail-drop-cfg {
                uses queuelimit;
                description
                    "Tail Drop configuration container";
            }
        }
        case random-detect {
            container random-detect-cfg {
                if-feature aqm-red-support;
                uses randomdetect;
                description
                    "Random Detect configuration container";
            }
        }
    }
    description
        "Choice of Drop Algorithm";
}
}
description
    "Augment the actions to policy entry";
}
}
<CODE ENDS>
```

#### **5.4. IETF-DIFFSERV-TARGET**

```
<CODE BEGINS>file "ietf-diffserv-target@2015-04-07.yang"
module ietf-diffserv-target {
    yang-version 1;
    namespace "urn:ietf:params:xml:ns:yang:ietf-diffserv-target";
    prefix target;

    import ietf-interfaces {
```



```
    prefix if;
  }

organization "IETF NETMOD (Netmod Working Group) Working Group";
contact
  "WG Web:   <http://tools.ietf.org/wg/netmod/>
   WG List:  <mailto:netmod@ietf.org>

   WG Chair: Jurgen Schonwalder
             <mailto:j.schoenwaelder@jacobs-university.de>

   WG Chair: Tom Nadeau
             <mailto:tnadeau@lucidvision.com>

   Editor:   Aseem Choudhary
             <mailto:asechoud@cisco.com>

   Editor:   Shitanshu Shah
             <mailto:svshah@cisco.com>";
description
  "This module contains a collection of YANG definitions for
   configuring diffserv specification implementations.

   Copyright (c) 2014 IETF Trust and the persons identified as
   authors of the code.  All rights reserved.

   Redistribution and use in source and binary forms, with or
   without modification, is permitted pursuant to, and subject
   to the license terms contained in, the Simplified BSD License
   set forth in Section 4.c of the IETF Trust's Legal Provisions
   Relating to IETF Documents
   (http://trustee.ietf.org/license-info).

   This version of this YANG module is part of RFC XXXX; see
   the RFC itself for full legal notices.";

revision 2015-04-07 {
  description
    "Latest revision diffserv based policy applied to a target";
  reference "RFC XXXX";
}

identity direction {
  description
    "This is identity of traffic direction";
}

identity inbound {
```



```
    base direction;
    description
        "Direction of traffic coming into the network entry";
}

identity outbound {
    base direction;
    description
        "Direction of traffic going out of the network entry";
}

feature target-inline-policy-config {
    description
        "This feature allows the policy configuration
        directly under a target.";
}

grouping red-stats {
    description
        "RED Counters";
    leaf early-drop-pkts {
        type uint64;
        description
            "Early drop packets ";
    }
    leaf early-drop-bytes {
        type uint64;
        description
            "Early drop bytes ";
    }
}

grouping classifier-entry-stats {
    description
        "Classifier Counters";
    container classifier-entry-statistics {
        config false;
        description
            "
            This group defines the classifier filter statistics of
            each classifier entry

            ";
    }
    leaf classified-pkts {
        type uint64;
        description
            " Number of total packets which filtered
            to the classifier-entry";
    }
}
```



```
    }
    leaf classified-bytes {
      type uint64;
      description
        " Number of total bytes which filtered
        to the classifier-entry";
    }
    leaf classified-rate {
      type uint64;
      units "bits-per-second";
      description
        " Rate of average data flow through the
        classifier-entry";
    }
  }
}

grouping queuing-stats {
  description
    "Queuing Counters";
  container queuing-statistics {
    description
      "queue related statistics ";
    leaf output-pkts {
      type uint64;
      description
        "Number of packets transmitted from queue ";
    }
    leaf output-bytes {
      type uint64;
      description
        "Number of bytes transmitted from queue ";
    }
    leaf queue-size-pkts {
      type uint64;
      description
        "Number of packets currently buffered ";
    }
    leaf queue-size-bytes {
      type uint64;
      description
        "Number of bytes currently buffered ";
    }
    leaf drop-pkts {
      type uint64;
      description
        "Total number of packets dropped ";
    }
  }
}
```



```
    leaf drop-bytes {
      type uint64;
      description
        "Total number of bytes dropped ";
    }
    container red-stats {
      uses red-stats;
      description
        "Container for RED statistics";
    }
  }
}

grouping meter-stats {
  description
    "Metering Counters";
  list meter-statistics {
    key "meter-id";
    description
      "Meter statistics";
    leaf meter-id {
      type uint16;
      description
        "Meter Identifier";
    }
    leaf meter-succeed-pkts {
      type uint64;
      description
        "Number of packets which succeed the meter";
    }
    leaf meter-succeed-bytes {
      type uint64;
      description
        "Bytes of packets which succeed the meter";
    }
    leaf meter-failed-pkts {
      type uint64;
      description
        "Number of packets which failed the meter";
    }
    leaf meter-failed-bytes {
      type uint64;
      description
        "Bytes of packets which failed the meter";
    }
  }
}
```



```
augment "/if:interfaces/if:interface" {
  description
    "Augments Diffserv Target Entry to Interface module";
  list diffserv-target-entry {
    key "direction policy-name";
    description
      "policy target for inbound or outbound direction";
    leaf direction {
      type identityref {
        base direction;
      }
      description
        "Direction fo the traffic flow either inbound or outbound";
    }
    leaf policy-name {
      type string;
      description
        "Policy entry name";
    }
    list diffserv-target-classifier-statistics {
      key "classifier-entry-name parent-path";
      config false;
      description
        "Statistics for each Classifier Entry in a Policy";
      leaf classifier-entry-name {
        type string;
        description
          "Classifier Entry Name";
      }
      leaf parent-path {
        type string;
        description
          "Path of the Classifier Entry in a hierarchial policy ";
      }
      uses classifier-entry-stats;
      uses meter-stats;
      uses queuing-stats;
    }
  }
}
```

<CODE ENDS>



## **6. Security Considerations**

## **7. Acknowledgement**

The editor of this document wishes to thank Fred Baker for over-viewing the document and provide useful comments, Andrew Mao for the guidance and support, Fred Yip and Aleksandr Zhdankin for helpful suggestions and contributions.

## **8. References**

### **8.1. Normative References**

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC2697] Heinanen, J. and R. Guerin, "A Single Rate Three Color Marker", [RFC 2697](#), September 1999.
- [RFC2698] Heinanen, J. and R. Guerin, "A Two Rate Three Color Marker", [RFC 2698](#), September 1999.
- [RFC2859] Fang, W., Seddigh, N., and B. Nandy, "A Time Sliding Window Three Colour Marker (TSWTCM)", [RFC 2859](#), June 2000.
- [RFC3246] Davie, B., Charny, A., Bennet, J., Benson, K., Le Boudec, J., Courtney, W., Davari, S., Firoiu, V., and D. Stiliadis, "An Expedited Forwarding PHB (Per-Hop Behavior)", [RFC 3246](#), March 2002.
- [RFC3260] Grossman, D., "New Terminology and Clarifications for Diffserv", [RFC 3260](#), April 2002.
- [RFC3289] Baker, F., Chan, K., and A. Smith, "Management Information Base for the Differentiated Services Architecture", [RFC 3289](#), May 2002.
- [RFC6020] Bjorklund, M., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", [RFC 6020](#), October 2010.
- [RFC6991] Schoenwaelder, J., "Common YANG Data Types", [RFC 6991](#), July 2013.
- [RFC7223] Bjorklund, M., "A YANG Data Model for Interface Management", [RFC 7223](#), May 2014.



## **8.2. Informative References**

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", [RFC 2475](#), December 1998.

## **Appendix A. Open Items**

The current model represents hierarchical QoS alike with the non-leaf and leaf nodes, in a scheduling hierarchy, without any restrictions of actions, such as AQM, that should not be allowed at non-leaf nodes. This is to be addressed in subsequent revisions.

### Authors' Addresses

Aseem Choudhary  
Cisco Systems  
170 W. Tasman Drive  
San Jose, CA 95134  
US

Email: asechoud@cisco.com

Shitanshu Shah  
Cisco Systems  
170 W. Tasman Drive  
San Jose, CA 95134  
US

Email: svshah@cisco.com

Mahesh Jethanandani  
Ciena Corporation  
3939 North 1st Street  
San Jose, CA 95134  
US

Email: mjethanandani@gmail.com



Gang Yan  
Huawei Technologies  
Huawei Bld., No. 156 Beiqing Rd  
Beijing 100095  
China

Email: yangang@huawei.com

Bing Liu  
Huawei Technologies  
Q14, Huawei Campus, No.156 Beiqing Rd  
Beijing 100095  
China

Email: Leo.liubing@huawei.com

Norm Strahle  
Juniper Networks  
1194 North Mathilda Avenue  
Sunnyvale, CA 94089  
US

Email: nstrahle@juniper.net

