Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 2, 2016

N. Wu
S. Zhuang
Huawei
A. Choudhary
Cisco Systems
Sep 30, 2015

A YANG Data Model for Flow Specification
draft-wu-idr-flowspec-yang-cfg-02

Abstract

This document defines a YANG data model for Flow Specification implementations. The data model includes configuration data and state data.

Requirements Language

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 RFC 2119 [RFC2119].

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 April 2, 2016.

Table of Contents

1. Introduction

This document defines a YANG [RFC6020] data model for the configuration and state data of Flowspec policies. Any RPC or notification definition is not part of this document. The model is based on Flowspec Policy architecture[RFC5575] and various other internet drafts [I-D.ietf-idr-flow-spec-v6][I-D.ietf-idr-bgp-flowspec-oid]. The configuration data defined in this document is encoded as flow specification rules and which can be distributed as BGP NLRI and/or applied locally to a network element. This distribution of traffic filtering rules through provider backbone can be used to filter Denial of Service attacks (DOS), maintaining some access control lists at network boundary [I-D.litkowski-idr-flowspec-interfaceset] besides other use-cases.

IP routers can classify packets based on multiple header fields. Such a classification rule may contain matching criteria to match on one or more packet header fields. Packets can be grouped to match on such packet rules defining a traffic flow. These traffic flows can be originated based on a manual policy or automatically or a combination of both. These defined traffic flows can be thus distributed.

As many vendors have different object constructs to represent the same traffic flow 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.

1.1. Terminology

The following terms are defined in [RFC6020]:

- configuration data
- data model
- module
- state data

The terminology for describing YANG data models is found in [RFC6020].

1.2. Flowspec Model Design

A flowspec policy contains one or more flowspec rules. Each flowspec rule may contain one or more flowspec components. A traffic flow matches a flowspec rule when it matches all the components present in the rule. A flow MAY match one or more flowspec rules. The order in which the flowspec rules are matched in a flowspec policy is defined by RFC 5575.

A flowspec rule contains one or more packet conditioning functions. A packet conditioning function MAY drop, limit the traffic flow rate, mark or redirect network packets to a specified routing instance. A flowspec policy can be stored as an object and used across different network device interfaces.

1.3. Tree Diagrams

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams is as follows:

- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration data (read-write), and "ro" means state data (read-only).
- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.

- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").

- Ellipsis ("...") stands for contents of subtrees that are not shown.

2. Flow Specification data model

The traffic flow specification data model has the following structure:

```
module: ietf-flowspec
   +--rw flowspec
      +--rw flowspec-cfg
      |  +--rw flowspec-policy* [policy-name]
      |     +--rw policy-name       string
      |     +--rw vrf-name?         string
      |     +--rw address-family?   identityref
      |     +--rw flowspec-rule* [rule-name]
      |        +--rw rule-name             string
      |        +--rw flowspec-component* [component-type]
      |        |  +--rw component-type        component-enum
      |        |  +--rw not-operator?         boolean
      |        |  +--rw (component)?
      |        |     +--:(destination-prefix)
      |        |     |  +--rw destination-prefix?   inet:ip-address
      |        |     +--:(source-prefix)
      |        |     |  +--rw source-prefix?        inet:ip-address
      |        |     +--:(ip-protocol)
      |        |     |  +--rw ip-protocol* [min max]
      |        |     |     +--rw min    uint8
      |        |     |     +--rw max    uint8
      |        |     +--:(port)
      |        |     |  +--rw port* [min max]
      |        |     |     +--rw min    uint16
      |        |     |     +--rw max    uint16
      |        |     +--:(destination-port)
      |        |     |  +--rw destination-port* [min max]
      |        |     |     +--rw min    uint16
      |        |     |     +--rw max    uint16
      |        |     +--:(source-port)
      |        |     |  +--rw source-port* [min max]
      |        |     |     +--rw min    uint16
      |        |     |     +--rw max    uint16
      |        |     +--:(icmp-type)
      |        |     |  +--rw icmp-type* [min max]
      |        |     |     +--rw min    uint8
```

```
          |        |     |     +--rw max    uint8
          |        |     +--:(icmp-code)
          |        |     |  +--rw icmp-code* [min max]
          |        |     |     +--rw min    uint8
          |        |     |     +--rw max    uint8
          |        |     +--:(tcp-flags)
          |        |     |  +--rw tcp-flag* [value]
          |        |     |     +--rw value    uint16
          |        |     +--:(packet-length)
          |        |     |  +--rw packet-length* [min max]
          |        |     |     +--rw min    uint16
          |        |     |     +--rw max    uint16
          |        |     +--:(dscp)
          |        |     |  +--rw dscp* [min max]
          |        |     |     +--rw min    uint8
          |        |     |     +--rw max    uint8
          |        |     +--:(fragment)
          |        |        +--rw fragment-value
          |        |           +--rw is-fragment?      boolean
          |        |           +--rw first-fragment?   boolean
          |        |           +--rw last-fragment?    boolean
          |        |           +--rw dont-fragment?    boolean
          |        +--rw flowspec-action* [action-type]
          |           +--rw action-type     action-type
          |           +--rw (action)?
          |              +--:(traffic-rate)
          |              |  +--rw rate?           float
          |              +--:(redirect)
          |              |  +--rw route-target?   string
          |              +--:(traffic-marking)
          |                 +--rw remark-dscp?    dscp-type
          +--ro flowspec-state
             +--ro flowspec-rib
             |  +--ro flowspec-entry*
             |     +--ro index?                        uint32
             |     +--ro islocal?                      boolean
             |     +--ro local-name?                   string
             |     +--ro neighbor?                     inet:ip-address
             |     +--ro duration?                     uint32
             |     +--ro flowspec-protocol-specific
             |        +--ro (protocol)?
             |           +--:(bgp)
             |              +--ro neighbor-router-id?   inet:ipv4-address
             |              +--ro as-path?              string
             |              +--ro origin?               enumeration
             |              +--ro med?                  uint32
             |              +--ro local-preference?     uint8
             |              +--ro community?            string
```

```
|              +--ro ext-community?        string
|              +--ro preference?           uint32
|              +--ro originator?           inet:ip-address
|              +--ro cluster-list?         string
|              +--ro flowspec-component* [component-type]
|              |  +--ro component-type        component-enum
|              |  +--ro not-operator?         boolean
|              |  +--ro (component)?
|              |     +--:(destination-prefix)
|              |     |  +--ro destination-prefix?
                              inet:ip-address
|              |     +--:(source-prefix)
|              |     |  +--ro source-prefix?
                              inet:ip-address
|              |     +--:(ip-protocol)
|              |     |  +--ro ip-protocol* [min max]
|              |     |     +--ro min    uint8
|              |     |     +--ro max    uint8
|              |     +--:(port)
|              |     |  +--ro port* [min max]
|              |     |     +--ro min    uint16
|              |     |     +--ro max    uint16
|              |     +--:(destination-port)
|              |     |  +--ro destination-port* [min max]
|              |     |     +--ro min    uint16
|              |     |     +--ro max    uint16
|              |     +--:(source-port)
|              |     |  +--ro source-port* [min max]
|              |     |     +--ro min    uint16
|              |     |     +--ro max    uint16
|              |     +--:(icmp-type)
|              |     |  +--ro icmp-type* [min max]
|              |     |     +--ro min    uint8
|              |     |     +--ro max    uint8
|              |     +--:(icmp-code)
|              |     |  +--ro icmp-code* [min max]
|              |     |     +--ro min    uint8
|              |     |     +--ro max    uint8
|              |     +--:(tcp-flags)
|              |     |  +--ro tcp-flag* [value]
|              |     |     +--ro value    uint16
|              |     +--:(packet-length)
|              |     |  +--ro packet-length* [min max]
|              |     |     +--ro min    uint16
|              |     |     +--ro max    uint16
|              |     +--:(dscp)
|              |     |  +--ro dscp* [min max]
|              |     |     +--ro min    uint8
```

```
|              |     |     +--ro max    uint8
|              |     +--:(fragment)
|              |        +--ro fragment-value
|              |           +--ro is-fragment?      boolean
|              |           +--ro first-fragment?   boolean
|              |           +--ro last-fragment?    boolean
|              |           +--ro dont-fragment?    boolean
|              +--ro flowspec-action* [action-type]
|                 +--ro action-type     action-type
|                 +--ro (action)?
|                    +--:(traffic-rate)
|                    |  +--ro rate?           float
|                    +--:(redirect)
|                    |  +--ro route-target?   string
|                    +--:(traffic-marking)
|                       +--ro remark-dscp?    dscp-type
+--ro flowspec-statistics
   +--ro flowspec-stats*
      +--ro vrf-name?              string
      +--ro address-family?        string
      +--ro flowspec-rule-stats*
         +--ro flowspec-component* [component-type]
            |  +--ro component-type        component-enum
            |  +--ro not-operator?         boolean
            |  +--ro (component)?
            |     +--:(destination-prefix)
            |     |  +--ro destination-prefix? inet:ip-address
            |     +--:(source-prefix)
            |     |  +--ro source-prefix?      inet:ip-address
            |     +--:(ip-protocol)
            |     |  +--ro ip-protocol* [min max]
            |     |     +--ro min    uint8
            |     |     +--ro max    uint8
            |     +--:(port)
            |     |  +--ro port* [min max]
            |     |     +--ro min    uint16
            |     |     +--ro max    uint16
            |     +--:(destination-port)
            |     |  +--ro destination-port* [min max]
            |     |     +--ro min    uint16
            |     |     +--ro max    uint16
            |     +--:(source-port)
            |     |  +--ro source-port* [min max]
            |     |     +--ro min    uint16
            |     |     +--ro max    uint16
            |     +--:(icmp-type)
            |     |  +--ro icmp-type* [min max]
            |     |     +--ro min    uint8
```

```
|     |     +--ro max    uint8
|     +--:(icmp-code)
|     |  +--ro icmp-code* [min max]
|     |     +--ro min    uint8
|     |     +--ro max    uint8
|     +--:(tcp-flags)
|     |  +--ro tcp-flag* [value]
|     |     +--ro value    uint16
|     +--:(packet-length)
|     |  +--ro packet-length* [min max]
|     |     +--ro min    uint16
|     |     +--ro max    uint16
|     +--:(dscp)
|     |  +--ro dscp* [min max]
|     |     +--ro min    uint8
|     |     +--ro max    uint8
|     +--:(fragment)
|        +--ro fragment-value
|           +--ro is-fragment?      boolean
|           +--ro first-fragment?   boolean
|           +--ro last-fragment?    boolean
|           +--ro dont-fragment?    boolean
+--ro flowspec-action* [action-type]
|  +--ro action-type     action-type
|  +--ro (action)?
|     +--:(traffic-rate)
|     |  +--ro rate?           float
|     +--:(redirect)
|     |  +--ro route-target?   string
|     +--:(traffic-marking)
|        +--ro remark-dscp?    dscp-type
+--ro classified-pkts?      uint64
+--ro classified-bytes?     uint64
+--ro drop-pkts?            uint64
+--ro drop-bytes?           uint64
```

This data model defines the configuration and state containers for traffic flow specification. In flowspec-cfg container, there is a list of configuration containers per flow specification route, which contains the configuration for filtering rules and corresponding actions.

The data model for state of traffic flow defines two state containers. Container flowspec-rib contains the current state of flowspec route, including filtering rules and actions. In the second container, there is statistics information for traffic flow specifications. The flowspec-rules in the flowspec-statistics

container is listed per routing instance per address family. The list of flowspec-rules is in order of applied rules in the forwarding path.

3. Flow Specification YANG Module

```
<CODE BEGINS> file "ietf-flowspec@2015-09-15.yang"

module ietf-flowspec {
  namespace "urn:ietf:params:xml:ns:yang:ietf-flowspec";
  prefix flowspec;

  import ietf-inet-types {
    prefix inet;
  }

  organization "IETF IDR (Inter-Domain Routing) Working Group";
  contact
    "WG Web:   <http://tools.ietf.org/wg/idr/>
     WG List:  <mailto:idr@ietf.org>

     WG Chair: Susan Hares
               <mailto:shares@ndzh.com>

     WG Chair: John Scudder
               <mailto:jgs@juniper.net>

     Editor:   Shunwan Zhuang
               <mailto:zhuangshunwan@huawei.com>

     Editor:   Nan Wu
               <mailto:eric.wu@huawei.com>

     Editor:   Aseem Choudhary
               <mailto:asechoud@cisco.com>";
  description
    "This module contains a collection of YANG definitions for
     configuring flow 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).";
```

```
    revision 2015-09-15 {
      description
        "Initial revision.";
      reference
        "[RFC5575] Dissemination of Flow Specification Rules
         [draft-ietf-netmod-routing-cfg-16]
         A YANG Data Model for Routing Management.
        ";
    }

    typedef component-enum {
      type enumeration {
        enum "destination-prefix" {
          value 1;
          description
            "Type 1 - Destination Prefix";
        }
        enum "source-prefix" {
          value 2;
          description
            "Type 2 - Source Prefix";
        }
        enum "ip-protocol" {
          value 3;
          description
            "Type 3 - IP Protocol";
        }
        enum "port" {
          value 4;
          description
            "Type 4 - Port";
        }
        enum "destination-port" {
          value 5;
          description
            "Type 5 - Destination port";
        }
        enum "source-port" {
          value 6;
          description
            "Type 6 - Source port";
        }
        enum "icmp-type" {
          value 7;
          description
            "Type 7 - ICMP type";
        }
        enum "icmp-code" {
```

```
          value 8;
          description
            "Type 8 - ICMP code";
        }
        enum "tcp-flags" {
          value 9;
          description
            "Type 9 - TCP flags";
        }
        enum "packet-length" {
          value 10;
          description
            "Type 10 - Packet length";
        }
        enum "dscp" {
          value 11;
          description
            "Type 11 - DSCP (Diffserv Code Point)";
        }
        enum "fragment" {
          value 12;
          description
            "Type 12 - Fragment";
        }
      }
      description
        "Definition for component type.";
    }

    typedef action-type {
      type enumeration {
        enum "traffic-rate" {
          value 32774;
          description
            "Carry the 2-octet id and 4-octet of rate information
             in IEEE floating point [IEEE.754.1985] format.";
        }
        enum "traffic-action" {
          value 32775;
          description
            "Consists of 6 bytes of which only the 2 least significant
             bits of the 6th byte (from left to right)
             are currently defined.";
        }
        enum "redirect" {
          value 32776;
          description
            "Allows the traffic to be redirected to a VRF
```

```
            routing instance that lists the specified
            route-target in its import policy.";
        }
        enum "traffic-marking" {
          value 32777;
          description
            "Instructs a system to modify the DSCP bits of
             a transiting IP packet to the corresponding value.";
        }
      }
      description
        "Definition for action type.";
    }

    typedef dscp-remarked {
      type enumeration {
        enum "be" {
          value 0;
          description
            "Default dscp (000000)";
        }
        enum "cs1" {
          value 8;
          description
            "CS1(precedence 1) dscp (001000)";
        }
        enum "af11" {
          value 10;
          description
            "AF11 dscp (001010)";
        }
        enum "af12" {
          value 12;
          description
            "AF12 dscp (001100)";
        }
        enum "af13" {
          value 14;
          description
            "AF13 dscp (001110)";
        }
        enum "cs2" {
          value 16;
          description
            "CS2(precedence 2) dscp (010000)";
        }
        enum "af21" {
          value 18;
```

```
          description
            "AF21 dscp (010010)";
          }
          enum "af22" {
            value 20;
            description
              "AF22 dscp (010100)";
          }
          enum "af23" {
            value 22;
            description
              "AF23 dscp (010110)";
          }
          enum "cs3" {
            value 24;
            description
              "CS3(precedence 3) dscp (011000)";
          }
          enum "af31" {
            value 26;
            description
              "AF31 dscp (011010)";
          }
          enum "af32" {
            value 28;
            description
              "AF32 dscp (011100)";
          }
          enum "af33" {
            value 30;
            description
              "AF33 dscp (011110)";
          }
          enum "cs4" {
            value 32;
            description
              "CS4(precedence 4) dscp (100000)";
          }
          enum "af41" {
            value 34;
            description
              "AF41 dscp (100010)";
          }
          enum "af42" {
            value 36;
            description
              "AF42 dscp (100100)";
          }
```

```
        enum "af43" {
          value 38;
          description
            "AF43 dscp (100110)";
        }
        enum "cs5" {
          value 40;
          description
            "CS5(precedence 5) dscp (101000)";
        }
        enum "ef" {
          value 46;
          description
            "EF dscp (101110)";
        }
        enum "cs6" {
          value 48;
          description
            "CS6(precedence 6) dscp (110000)";
        }
        enum "cs7" {
          value 56;
          description
            "CS7(precedence 7) dscp (111000)";
        }
      }
      description
        "Definition for dscp type.";
    }

    typedef dscp-type {
      type union {
        type dscp-remarked;
        type uint8;
      }
      description
        "Definition for dscp type.";
    }

    typedef float {
      type union {
        type decimal64 {
          fraction-digits 14;
          range "-9999.99999999999999 .. 9999.99999999999999";
        }
        type decimal64 {
          fraction-digits 13;
          range "-99999.9999999999999 .. 99999.9999999999999";
```

```
        }
        type decimal64 {
          fraction-digits 12;
          range "-999999.999999999999 .. 999999.999999999999";
        }
        type decimal64 {
          fraction-digits 11;
          range "-9999999.99999999999 .. 9999999.99999999999";
        }
        type decimal64 {
          fraction-digits 10;
          range "-99999999.9999999999 .. 99999999.9999999999";
        }
        type decimal64 {
          fraction-digits 9;
          range "-999999999.999999999 .. 999999999.999999999";
        }
        type decimal64 {
          fraction-digits 8;
          range "-9999999999.99999999 .. 9999999999.99999999";
        }
        type decimal64 {
          fraction-digits 7;
          range "-99999999999.9999999 .. 99999999999.9999999";
        }
        type decimal64 {
          fraction-digits 6;
          range "-999999999999.999999 .. 999999999999.999999";
        }
        type decimal64 {
          fraction-digits 5;
          range "-9999999999999.99999 .. 9999999999999.99999";
        }
        type decimal64 {
          fraction-digits 4;
          range "-99999999999999.9999 .. 99999999999999.9999";
        }
        type decimal64 {
          fraction-digits 3;
          range "-999999999999999.999 .. 999999999999999.999";
        }
        type decimal64 {
          fraction-digits 2;
          range "-9999999999999999.99 .. 9999999999999999.99";
        }
        type decimal64 {
          fraction-digits 1;
          range "-99999999999999999.9 .. 99999999999999999.9";
```

```
        }
      }
      description
        "Definition for float point referenced from maillist.";
    }

    identity address-family {
      description
        "Base identity from which identities describing address
         families are derived.";
    }

    identity ipv4 {
      base address-family;
      description
        "This identity represents IPv4 address family.";
    }

    identity ipv6 {
      base address-family;
      description
        "This identity represents IPv6 address family.";
    }

    grouping tcp-flags-value {
      description
        "This group defines TCP flags values for
         tcp-flag component of FlowSpec.";
      leaf value {
        type uint16;
        description
          "Define the value of TCP flags.";
      }
    }

    grouping dscp-value {
      description
        "This group defines dscp values for
         DSCP component of FlowSpec.";
      leaf min {
        type uint8;
        description
          "Define the minimum value of dscp range.";
      }
      leaf max {
        type uint8;
        must ". >= ../min" {
          error-message
```

```
            "The max value must be greater than or
             equal to min value";
          description
            "The max value must be greater than or equal to
             min value.";
        }
        description
          "Define the maximum value of dscp range.";
      }
    }

    grouping ip-protocol-value {
      description
        "This group defines ip-protocol values for
         ip-protocol component of FlowSpec.";
      leaf min {
        type uint8;
        description
          "Define the minimum value of ip-protocol range.";
      }
      leaf max {
        type uint8;
        must ". >= ../min" {
          error-message
            "The max value must be greater than or
             equal to min value";
          description
            "The max value must be greater than or equal to
             min value.";
        }
        description
          "Define the maximum value of ip-protocol range.";
      }
    }

    grouping icmp-type-value {
      description
        "This group defines icmp type values for
         icmp-type component of FlowSpec.";
      leaf min {
        type uint8;
        description
          "Define the minimum value of icmp-type range.";
      }
      leaf max {
        type uint8;
        must ". >= ../min" {
          error-message
```

```
            "The max value must be greater than or
             equal to min value";
          description
            "The max value must be greater than or equal to
             min value.";
        }
        description
          "Define the maximum value of icmp-type range.";
      }
    }

    grouping icmp-code-value {
      description
        "This group defines icmp code values for
         icmp-code component of FlowSpec.";
      leaf min {
        type uint8;
        description
          "Define the minimum value of icmp-code range.";
      }
      leaf max {
        type uint8;
        must ". >= ../min" {
          error-message
            "The max value must be greater than or
             equal to min value";
          description
            "The max value must be greater than or equal to
             min value.";
        }
        description
          "Define the maximum value of icmp-code range.";
      }
    }

    grouping packet-length-value {
      description
        "This group defines packet length values for
         packet-length component of FlowSpec.";
      leaf min {
        type uint16;
        description
          "Define the minimum value of packet-length range.";
      }
      leaf max {
        type uint16;
        must ". >= ../min" {
          error-message
```

```
            "The max value must be greater than or
             equal to min value";
          description
            "The max value must be greater than or equal to
             min value.";
        }
        description
          "Define the maximum value of packet-length range.";
      }
    }

    grouping port-value {
      description
        "This group defines port values for
         port component of FlowSpec.";
      leaf min {
        type uint16;
        description
          "Define the minimum value of port range.";
      }
      leaf max {
        type uint16;
        must ". >= ../min" {
          error-message
            "The max value must be greater than or
             equal to min value";
          description
            "The max value must be greater than or equal to
             min value.";
        }
        description
          "Define the maximum value of port range.";
      }
    }

    grouping destination-port-value {
      description
        "This group defines destination port values for
         destination-port component of FlowSpec.";
      leaf min {
        type uint16;
        description
          "Define the minimum value of destination-port range.";
      }
      leaf max {
        type uint16;
        must ". >= ../min" {
          error-message
```

```
            "The max value must be greater than or
             equal to min value";
          description
            "The max value must be greater than or equal to
             min value.";
        }
        description
          "Define the maximum value of destination-port range.";
      }
    }

    grouping source-port-value {
      description
        "This group defines source port values for
         source-port component of FlowSpec.";
      leaf min {
        type uint16;
        description
          "Define the minimum value of source-port range.";
      }
      leaf max {
        type uint16;
        must ". >= ../min" {
          error-message
            "The max value must be greater than or
             equal to min value";
          description
            "The max value must be greater than or equal to
             min value.";
        }
        description
          "Define the maximum value of source-port range.";
      }
    }

    grouping fragment-value {
      description
        "This group defines fragment values for
         Fragment component of FlowSpec.";
      leaf is-fragment {
        type boolean;
        description
          "Match if packet is a fragment ";
      }
      leaf first-fragment {
        type boolean;
        description
          "Match if packet is first fragment ";
```

```
      }
      leaf last-fragment {
        type boolean;
        description
          "Match if packet is last fragment ";
      }
      leaf dont-fragment {
        type boolean;
        description
          "Match if don't fragment bit is set in the packet";
      }
    }

    grouping flowspec-traffic-filter {
      description
        "This group defines the traffic filter rules for FlowSpec.";
      list flowspec-component {
        key "component-type";
        min-elements 1;
        description
          "Define the traffic filter components into a list.";
        leaf component-type {
          type component-enum;
          description
            "Specify the type of component for this list entry.";
        }
        leaf not-operator {
          type boolean;
          description
            "If set to TRUE, the values or ranges specified in the
             component is not to be matched.";
        }
        choice component {
          description
            "Define different kinds of flowspec components involved.";
          case destination-prefix {
            leaf destination-prefix {
              type inet:ip-address;
              description
                "Specifies the destination address of the traffic.";
            }
          }
          case source-prefix {
            leaf source-prefix {
              type inet:ip-address;
              description
                "Specifies the source address of the traffic.";
            }
```

```
          }
          case ip-protocol {
            list ip-protocol {
              key "min max";
              uses ip-protocol-value;
              description
                "Define the minimum and maximum ranges
                 of an ip-protocol list.";
            }
          }
          case port {
            list port {
              key "min max";
              uses port-value;
              description
                "Define the minimum and maximum ranges
                 of a port list.";
            }
          }
          case destination-port {
            list destination-port {
              key "min max";
              uses destination-port-value;
              description
                "Define the minimum and maximum ranges of a
                 destination-port list.";
            }
          }
          case source-port {
            list source-port {
              key "min max";
              uses source-port-value;
              description
                "Define the minimum and maximum ranges of a
                 source-port list.";
            }
          }
          case icmp-type {
            list icmp-type {
              key "min max";
              uses icmp-type-value;
              description
                "Define the minimum and maximum ranges of an
                 icmp-type list.";
            }
          }
          case icmp-code {
            list icmp-code {
```

```
              key "min max";
              uses icmp-code-value;
              description
                "Define the minimum and maximum ranges of an
                 icmp-code list.";
            }
          }
          case tcp-flags {
            list tcp-flag {
              key "value";
              uses tcp-flags-value;
              description
                "Defines list of tcp-flag entries.";
            }
          }
          case packet-length {
            list packet-length {
              key "min max";
              uses packet-length-value;
              description
                "Define the minimum and maximum ranges of a
                 packet-length list.";
            }
          }
          case dscp {
            list dscp {
              key "min max";
              uses dscp-value;
              description
                "Define the minimum and maximum ranges of a
                 dscp list.";
            }
          }
          case fragment {
            container fragment-value {
              uses fragment-value;
              description
                "Define Fragment flags value.";
            }
          }
        }
      }
    }

    grouping flowspec-traffic-action {
      description
        "This group defines the traffic actions for FlowSpec.";
      list flowspec-action {
```

```
        key "action-type";
        description
          "Define the traffic actions of FlowSpec into a list.";
        leaf action-type {
          type action-type;
          description
            "Specify the type of traffic filter action.";
        }
        choice action {
          description
            "Define different kinds of traffic actions involved.";
          case traffic-rate {
            leaf rate {
              type float;
              description
                "Specifies the traffic rate in IEEE floating point
                 [IEEE.754.1985] format, units being bytes per
                 second.";
            }
          }
          case redirect {
            leaf route-target {
              type string {
                length "3..21";
              }
              description
                "Allows the traffic to be redirected to a VRF
                 routing instance that lists the specified
                 route-target in its import policy";
            }
          }
          case traffic-marking {
            leaf remark-dscp {
              type dscp-type;
              description
                "Instructs a system to modify the DSCP bits of
                 a transiting IP packet to the corresponding value";
            }
          }
        }
      }
    }

    grouping flowspec-bgp-route {
      description
        "This group define extensions of FlowSpec for bgp protocol.";
      leaf neighbor-router-id {
        type inet:ipv4-address;
```

```
        description
          "The router-id of the neighbor from whom
           this route received.";
      }
      leaf as-path {
        type string;
        description
          "Number of the AS the BGP FlowSpec route passes.";
      }
      leaf origin {
        type enumeration {
          enum "igp" {
            value 0;
            description
              "Designate this route originated from igp route.";
          }
          enum "egp" {
            value 1;
            description
              "Designate this route originated from egp route.";
          }
          enum "incomplete" {
            value 2;
            description
              "Designate this route originated from other route.";
          }
        }
        description
          "The Origin attribute defines the origin of a route. The
           Origin attribute is classified into the following types:

           Interior Gateway Protocol (IGP): This attribute type has the
           highest priority. IGP is the Origin attribute for routes
           obtained through an IGP in the AS from which the routes
           originate. For example, the Origin attribute of the routes
           imported to the BGP routing table using the network command
           is IGP.

           Exterior Gateway Protocol (EGP): This attribute type has the
           second highest priority. The Origin attribute of the routes
           obtained through EGP is EGP.

           Incomplete: This attribute type has the lowest priority.
           Incomplete is the Origin attribute type of all routes that
           do not have the IGP or EGP Origin attribute. For example,
           the Origin attribute of the routes imported is Incomplete.";
      }
      leaf med {
```

```
        type uint32;
        description
          "The Multi-Exit-Discriminator (MED) is transmitted only
           between two neighboring ASs.";
      }
      leaf local-preference {
        type uint8;
        description
          "The local preference of the BGP FlowSpec route.";
      }
      leaf community {
        type string;
        description
          "Community attribute of the BGP FlowSpec route.";
      }
      leaf ext-community {
        type string;
        description
          "Extended community attribute of the BGP FlowSpec route.";
      }
      leaf preference {
        type uint32;
        description
          "The preferred value of a protocol.";
      }
      leaf originator {
        type inet:ip-address;
        description
          "The address of the advertiser of the BGP FlowSpec route.";
      }
      leaf cluster-list {
        type string;
        description
          "Cluster list attribute of the BGP FlowSpec route.";
      }
      uses flowspec-traffic-filter;
      uses flowspec-traffic-action;
    }

    grouping flowspec-protocol-specific {
      description
        "This group define extensions of FlowSpec per protocols.";
      choice protocol {
        description
          "Define specific part of FlowSpec when carried in
           different protocols.";
        case bgp {
          uses flowspec-bgp-route;
```

```
        }
      }
    }

    container flowspec {
      description
        "Container for flowspec configuration and state";
      container flowspec-cfg {
        description
          "Configuration for flow specification.";
        list flowspec-policy {
          key "policy-name";
          description
            "Configuration of a flow route list.";
          leaf policy-name {
            type string;
            description
              "The name of a flow route.";
          }
          leaf vrf-name {
            type string;
            description
              " Vrf-name of the flowspec-rule";
          }
          leaf address-family {
            type identityref {
              base address-family;
            }
            description
              " address-family of the flowspec-rule";
          }
          list flowspec-rule {
            key "rule-name";
            ordered-by system;
            leaf rule-name {
              type string;
              description
                "The name of a flowspec rule.";
            }
            uses flowspec-traffic-filter;
            uses flowspec-traffic-action;
            description
              "Define flow specification filters.";
          }
        }
      }
      container flowspec-state {
        config false;
```

```
        description
          "Operational state of flow specification.";
        container flowspec-rib {
          description
            "Define the operational state data for FlowSpec entries.";
          list flowspec-entry {
            description
              "FlowSpec entries are organized into list of routes.";
            leaf index {
              type uint32;
              description
                "Flow Specification route entry index.";
            }
            leaf islocal {
              type boolean;
              description
                "Locally configured Flow Specification route.";
            }
            leaf local-name {
              type string;
              description
                "The name of locally configured FlowSpec route.";
            }
            leaf neighbor {
              type inet:ip-address;
              description
                "IP address of an advertising device";
            }
            leaf duration {
              type uint32;
              description
                "Route duration in seconds.";
            }
            container flowspec-protocol-specific {
              description
                "Define the specific extension for each protocol.";
              uses flowspec-protocol-specific;
            }
          }
        }
        container flowspec-statistics {
          description
            "Define the statistics of list of flowspec rules.";
          list flowspec-stats {
            description
              "Statistics of list of flowspec rules per VRF &
               Address-family.";
            leaf vrf-name {
```

```
              type string;
              description
                "Vrf-name of the set of flowspec-rules";
            }
            leaf address-family {
              type string;
              description
                "Address-family of the set of flowspec-rules";
            }
            list flowspec-rule-stats {
              description
                "
                 This defines the flowspec filter statistics of
                 each flowspec-rules.
                ";
              uses flowspec-traffic-filter;
              uses flowspec-traffic-action;
              leaf classified-pkts {
                type uint64;
                description
                  " Number of total packets which matched
                    to the flowspec-filter";
              }
              leaf classified-bytes {
                type uint64;
                description
                  " Number of total bytes which matched
                    to the flowspec-filter";
              }
              leaf drop-pkts {
                type uint64;
                description
                  " Number of total packets which got dropped";
              }
              leaf drop-bytes {
                type uint64;
                description
                  " Number of total bytes which got dropped";
              }
            }
          }
        }
      }
    }
  }

  <CODE ENDS>
```

4. IANA Considerations

This document registers a URI in the "IETF XML Registry" [RFC3688]. Following the format in RFC 3688, the following registration has been made.

URI: urn:ietf:params:xml:ns:yang:ietf-flowspec

Registrant Contact: The RTGWG WG of the IETF.

XML: N/A; the requested URI is an XML namespace.

This document registers a YANG module in the "YANG Module Names" registry [RFC6020].

Name: ietf-flowspec

Namespace: urn:ietf:params:xml:ns:yang:ietf-flowspec

Prefix: flowspec

Reference: RFC XXXX

5. Security Considerations

The YANG module defined in this memo is designed to be accessed via the NETCONF protocol [RFC6241]. The lowest NETCONF layer is the secure transport layer and the mandatory-to-implement secure transport is SSH [RFC6242]. The NETCONF access control model [RFC6536] provides the means to restrict access for particular NETCONF users to a pre-configured subset of all available NETCONF protocol operations and content.

There are a number of data nodes defined in the YANG module which are writable/creatable/deletable (i.e., config true, which is the default). These data nodes may be considered sensitive or vulnerable in some network environments. Write operations (e.g., <edit-config>) to these data nodes without proper protection can have a negative effect on network operations.

6. Acknowledgments

The editor of this document wishes to thank Andrew Mao for the guidance and support in coming up with this draft.

7. References

7.1. Normative References

[I-D.ietf-idr-bgp-flowspec-oid]
Uttaro, J., Filsfils, C., Smith, D., Alcaide, J., and P. Mohapatra, "Revised Validation Procedure for BGP Flow Specifications", draft-ietf-idr-bgp-flowspec-oid-02 (work in progress), January 2014.

[I-D.ietf-idr-flow-spec-v6]
Raszuk, R., Pithawala, B., McPherson, D., and A. Andy, "Dissemination of Flow Specification Rules for IPv6", draft-ietf-idr-flow-spec-v6-06 (work in progress), November 2014.

[I-D.litkowski-idr-flowspec-interfaceset]
Litkowski, S., Simpson, A., Patel, K., and J. Haas, "Applying BGP flowspec rules on a specific interface set", draft-litkowski-idr-flowspec-interfaceset-02 (work in progress), March 2015.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, DOI 10.17487/RFC3688, January 2004, <http://www.rfc-editor.org/info/rfc3688>.

[RFC5575] Marques, P., Sheth, N., Raszuk, R., Greene, B., Mauch, J., and D. McPherson, "Dissemination of Flow Specification Rules", RFC 5575, DOI 10.17487/RFC5575, August 2009, <http://www.rfc-editor.org/info/rfc5575>.

[RFC6020] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <http://www.rfc-editor.org/info/rfc6241>.

[RFC6242] Wasserman, M., "Using the NETCONF Protocol over Secure Shell (SSH)", RFC 6242, DOI 10.17487/RFC6242, June 2011, <http://www.rfc-editor.org/info/rfc6242>.

[RFC6536] Bierman, A. and M. Bjorklund, "Network Configuration Protocol (NETCONF) Access Control Model", RFC 6536, DOI 10.17487/RFC6536, March 2012, <http://www.rfc-editor.org/info/rfc6536>.

7.2. Informative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

Authors' Addresses

Nan Wu
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: eric.wu@huawei.com

Shunwan Zhuang
Huawei
Huawei Bld., No.156 Beiqing Rd.
Beijing 100095
China

Email: zhuangshunwan@huawei.com

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

Email: asechoud@cisco.com