IDR Working Group
Internet-Draft
Intended status: Standards Track
Expires: 10 November 2026

S. Hares
Hickory Hill Consulting
D. Eastlake
Independent
J. Dong
Huawei Technologies
C. Yadlapalli
ATT
S. Maduscke
Verizon
J. Haas
HPE

9 May 2026

# BGP Flow Specification Version 2 - for Basic IP
draft-ietf-idr-fsv2-ip-basic-06

## Abstract

BGP flow specification version 1 (FSv1), defined in RFC 8955, RFC 8956, and RFC 9117, describes the distribution of traffic filter policy (traffic filters and actions) distributed via BGP. During the deployment of BGP FSv1 a number of issues were detected, so version 2 of the BGP flow specification (FSv2) protocol addresses these issues. In order to provide a clear demarcation between FSv1 and FSv2, a different NLRI encapsulates FSv2.

The IDR WG requires two implementation. Early feedback on implementations of FSv2 indicate that FSv2 has a correct design direction, but that breaking FSv2 into a progression of documents would aid deployment of the draft (basic, adding more filters, and adding more actions). This document specifies the basic FSv2 NLRI with user ordering of filters added to FSv1 IP Filters and FSv2 actions.

## 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 https://datatracker.ietf.org/drafts/current/.

## Copyright Notice



## Table of Contents

## 1. Introduction

Version 2 of BGP flow specification was original defined in [fsv2] (BGP FSv2).

FSv2 is an update to BGP Flow specification version 1 (BGP FSv1). BGP FSv1 as defined in [FSv1], [FSv1-IPv6], and [RFC9117] specified 2 SAFIs (133, 134) to be used with IPv4 AFI (AFI = 1) and IPv6 AFI (AFI=2).

The initial BGP FSv2 specification had the correct direction, but it contained more than the early implementers desired. The implmenters desired a progression of documents with smaller incremental changes: Basic FSv2, adding more filters, and adding more actions.

This draft provides the basic FSv2 framework specification for transmitting user-ordered IP filters in the FSV2 NLRI and associating Flow Spec actions by transmitting Flow Spec Extended Communities (FS-EC) with the FSv2 NLRI. If a filter match links to a single FS-EC action, the single action succeeds or fails. If a filter match links to mutiple actions, there is a potential for interactions. Section 4.5.1 discusses how to analyze the interaction by categories and solutions to issues with multiple FSv2-EC actions interacting. A complete solution requires the BGP Community Container Attribute see [I-D.ietf-idr-wide-bgp-communities]) with FSv2 Container defined in the [fsv2-more-ip-filters].

This document defines 2 new SAFIs, TBD1 and TBD2, for FSv2 to be used with 5 AFIs: 1, 2, 6, 25, and 31. FSv2 implementations do not require all 10 combinations of FSv2 AFI/SAFIs to be implemented. An implementation is required to implement only one these AFI/SAFIs to be compliant. For example, a compliant implementation might only define the FSv2 NLRI for IPv4 for IP forwarding (AFI=1, SAFI=TBD1).

FSv1 and FSv2 use different AFI/SAFIs to send their respective flow specification filters. This permits FSv1 and FSv2 to be coexist with each other in a "ships in the night" deployment.

The remainder of Section 1 provides background on why the FSv2 was necessary to fix problems with FSv1. Section 2 contains a primer on FSv2. Section 3 contains the BGP encoding rules for FSv2. Section 5 describes how to validate and order FSv2 NLRI. The remaining sections discuss scalability, optional security additions, security considerations, and IANA considerations.

## 1.1. Why Flow Specification v2

Modern IP routers have the capability to forward traffic and to classify, shape, rate limit, filter, or redirect packets based on administratively defined policies. These traffic policy mechanisms allow the operator to define match rules that operate on multiple fields within header of an IP data packet. The traffic policy allows actions to be taken upon a match to be associated with each match

rule. These rules can be more widely defined as "event-condition-action" (ECA) rules where the event is always the reception of a packet.

BGP ([RFC4271]) flow specification version 1 (FSv1) as defined by [FSv1], [FSv1-IPv6], and [RFC9117] specifies the distribution of traffic filter policy (traffic filters and actions) via BGP to BGP peers, both IBGP and EBGP. The traffic filter is applied when packets are received on a router with the flow specification function enabled.

Multiple deployed applications currently use BGP FSv1 to distribute traffic filters. These applications include:

* Mitigation of Denial of Service traffic (DoS).

* Traffic filtering in BGP/MPLS VPNS.

* Centralized traffic control for networks utilizing SDN control of router firewall functions.

* Classifiers for insertion into a SFC.

* Filters for SRv6 (segment routing v6).

During the deployment of FSv1, the following issues were noted:

* FSv1 NLRI components did not use TLV encoding, which inhibited defining new component types. (The format was type-value, missing a length field.)

* FSv1 rules did not have the ability to be ordered by the operator. Instead, only the protocol-defined rule ordering was permitted.

* When conflicting outcomes for rule actions was present, the operator was unable to influence their ordering.

* When multiple and conflicting rule actions were present, the operator couldn’t define their order when some actions could not be implemented on the receiving router.

Networks currently address these issues by constraining deployments or using topology/deployment specific workarounds.

FSv1 is a critical component of deployed applications. Therefore, this specification defines how FSv2 will interact with BGP peers that support combinations FSv1 and FSv2. It is expected that a transition to FSv2 will occur over time as new applications require features enabled by FSv2.

## 1.2. Definitions and Acronyms

AFI:
Address Family Identifier [RFC4760]

AS:
Autonomous System

BGP session ephemeral state:
State which does not survive the loss of BGP peer session.

BGP Commmunity Path Attribute:
BGP Community Path attribute with a FS TLV defined by [fsv2-more-ip-filters]

Configuration state:
State which persist across a reboot of software module within a routing system or a reboot of a hardware routing device.

CPA:
BGP Community Path Attribute.

DDoS:
Distributed Denial of Service.

Ephemeral state:
State which does not survive the reboot of a software module, or a hardware reboot. Ephemeral state can be ephemeral configuration state or operational state.

Extended Community:
BGP Path Attribute defined by [RFC4360].

FS:
Flow Specification (either v1 or v2).

FSv1:
Flow Specification version 1 [FSv1] [FSv1-IPv6].

FSv2:
Flow Specification version 2 (this document and its extensions).

FS-CPA:
Flow Specification Actions defined in Community Path Attribute.

FS-EC:
FS related Extended Community with FS actions.

FSv1-EC:
FSv1 Extended Community with FS Actions supported by FSv1.

FSv2-EC:
FSv2 Extended Community with FS Actions supported by FSv2.

NETCONF:
The Network Configuration Protocol [RFC6241].

NLRI:
Network Layer Reachability Information [RFC4271] [RFC4760]. The "destination" portion of a Flowspec route carried in a BGP UPDATE message.

RESTCONF:
The RESTCONF Protocol [RFC8040].

RIB:
Routing Information Base.

ROA:
Route Origin Authentication [RFC9582].

RR:
Route Reflector [RFC4456].

SAFI:
Subsequent Address Family Identifier [RFC4760].

SFC:
Service Function Chaining [RFC7665].

### 1.3. 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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals as shown here.

## 2. Flow Specification Version 2 Primer

A BGP Flow Specification (v1 or v2) is an n-tuple containing one or more match criteria that can be applied to data-plane traffic. The exact traffic match depends on the FSv2 AFI/SAFI.

Flows Specification routes carried in BGP UPDATEs may carry BGP Path Attributes that have additional match or action consequences. This includes, but is not limited to: Extended Communities [RFC4360] and Community Container Path attributes [I-D.ietf-idr-wide-bgp-communities].

Flow Specifiation NLRI for a given AFI/SAFI is used as they key for Flow Specification routes in the BGP RIBs. Flow Specification routes that are selected for the Loc-RIB are then associated with a given set of semantics which are application dependent. Standard BGP policy mechanisms for BGP routes are applicable to Flow Specification routes, including AS_PATH and community filtering.

This FSv2 for basic IP forwarding specification only requires the use of Extended Communities to associate FS actions with FSv2 filters found in FSv2 NLRI.

FSv2 features implementing multiple actions with user ordering of actions or dependencies between actions requires the BGP Community Attribute [I-D.ietf-idr-wide-bgp-communities] with a FSv2 Component as defined in [fsv2-more-ip-filters].

Network operators can control the propagation of Flow Specification BGP routes by enabling or disabling the exchange of routes for a particular AFI/SAFI pair on a particular peering session. BGP policy mechanisms, including [RFC1997] scoping communities, can also be used. Thus, Flow Specification routes may be distributed to only a portion of a BGP deployment.

### 2.1. Flow Specification v1 (FSv1) SAFIs

The FSv1 NLRI defined in [FSv1] and [FSv1-IPv6] includes 13 match conditions encoded for the following AFI/SAFIs:

* IPv4 traffic: AFI:1, SAFI:133
* IPv6 Traffic: AFI:2, SAFI:133
* BGP/MPLS IPv4 VPN: AFI:1, SAFI: 134
* BGP/MPLS IPv6 VPN: AFI:2, SAFI: 134

FSv1 match conditions are ordered by component type in ascending order. The ordering within a component type is defined by that component's definition.

The Flow Specification actions standardized by [FSv1] and [FSv1-IPv6] are:

* accept packet (default),

* traffic rate limiting by bps (0x6),

* traffic-action: sample, or terminate rule (0x7),

* redirect traffic to VPN by route target(0x8),

* traffic marking (DSCP) (0x9), and

* traffic rate limiting by pps (0xC)

A SFC action [RFC9015] defines a redirection of a data flow to an entry point into a specific SFP (Service Function Path).

Other Extended Community actions have been proposed in IDR, but have not completed the standardization process.

## 2.2. Transition to FSv2

This specification defines AFI/SAFIs to support Flow Specification version 2 for IPv4, IPv6, Layer 2, IPv4 VPNs, IPv6 VPNs, Layer 2 VPNs (L2VPN), Service Function Chaining (SFC), and SFC VPNs:

* IPv4 traffic: AFI=1, SAFI=TBD1,

* IPv6 traffic: AFI=2, SAFI=TBD1,

* L2: AFI=6, SAFI=TDB1 (defined in [I-D.ietf-idr-flowspec-l2vpn]),

* BGP/MPLS IPv4 VPN: AFI=1, SAFI=TBD2,

* BGP/MPLS IPv6 VPN: AFI=2, SAFI=TBD2,

* BGP/MPLS L2VPN: AFI=25, SAFI=TDB2 (defined in [I-D.ietf-idr-flowspec-l2vpn]),

* SFC: AFI=31, SAFI=TBD1,

* SFC VPN: AFI=31, SAFI=TBD2

One question asked by developers is what AFI/SAFI is required for FSv2 IP Basic compliance. BGP negotiates support for each AFI/SAFI, so FSv2 IP Basic support for non-VPN could be as little as FSv2 for IPv4 forwarding (AFI/SAFI: 1/TBD1),

The IDR specification for L2 VPN traffic was specified in [I-D.ietf-idr-flowspec-l2vpn]. An IDR specification for tunneled traffic is in [I-D.ietf-idr-flowspec-nvo3]. Both of these drafts were targeted for FSv1, but the WG decided to require these to FSv2 TLV formats.

### 2.3. FSv2 Overview

FSv2 allows the user to order the flow specification rules and the actions associated with a rule. Each FSv2 rule may have one or more match conditions and one or more associated actions.

FSv2 operates in a ships-in-the night model with FSv1. This permits operators to manage the interaction of FSv2 and FSv1 via configuration.

The basic principles regarding the ordering and installation of flow specification filter rules are:

1. In the absence of a matching filter for the traffic, that traffic is permitted. That is, the default is permit. Implementations MAY implement a default reject behavior by configuration.

2. FSv2 filter rules are processed prior to FSv1 rules. FSv1 NLRI are processed according to the procedures defined in [FSv1] and [FSv1-IPv6]. FSv2 filter rules thus have a better precedence vs. FSv1.

3. FSv2 filter rules are ordered based on user-specified order, carried in each FSv2 NLRI. Numerically smaller user-specified order values have better precedence than larger values.

4. For rules with the same user-specified order, the filter rules are then ordered by FSv2 component type and then rules for each component type.

5. FSv2 filter rules can carry actions. These actions can be encoded via one or more FSv2 Extended Communities, or within the FSv2 Action Community Container.

Some FSv2 Extended Communities may not be understood by every FSv2 implementation. Since they are encoded as [RFC4360] Extended Communities, they are propagated with the BGP routes regardless of whether they are understood based on the particular Extended Community's transitivity.

When FSv2 Extended Communities are understood, they have precedence and interaction rules governing the actions they encode. (See XXX JMH TODO)

The FSv2 Action Community Container defines its own rules governing FSv2 actions. See that document (XXX JMH TODO) for additional details.

6. FSv2 filter match and action criteria may be considered "optional". For match, the FSv2 NLRI encoding carries a per-component flag set by the operator or implementation that marks that match component as optional or mandatory. For actions, FSv2 Extended Communities will document whether they are considered optional or mandatory as part of their definition. The optionality of FSv2 Action Community Containers is defined in its defining document.

   If a mandatory match component or action component cannot be locally implemented, the flowspec rule is marked as ineligible to be installed.

7. FSv2 filter rules carry a "Dependency" value in the FSv2 NLRI. When this value is non-zero, this value associates multiple received FSv2 filters with each other. If a FSv2 filter rule is ineligible to be installed due to an inability to implement a mandatory match or action component, all other filters carrying the same dendency value will be made ineligible for installation. See Section 3.2 for more details.

## 3. FSv2 NLRI Formats and Actions

BGP Flow Specifications are encoded in BGP NLRI as an ordered list of TLVs of "filter families", where each filter family consists of an ordered list of TLVs of "filter components" for that familiy. Filter families are groupings of related filtering functionality, typically at the same network layer. Filter components match specific network elements for a filter family.

Each FSv2 NLRI has a default sort order, documented in section TODO. This sort order determines the order of installation for the Flow Specification in the BGP speaker. Operators MAY override this default ordering by causing the FSv2 User Order field to be set to a non-zero value.

Sets of FSv2 NLRI might share fate with each other. In the event that a Flow Specification is unable to be installed by the BGP speaker, dependent Flow Specifications MUST NOT also be installed, even if they are otherwise valid. These dependencies are encoded in the Dependent Filters Chain field of a FSv2 Flow Specification.

FSv2 is carried in BGP using standard [RFC4760] multiprotocol extensions. FSv2 supports NRLI with formats for following AFIs:

* IPv4 (AFI = 1)

* IPv6 (AFI = 2)

* L2 (AFI = 6)

* L2VPN (AFI=25)

* SFC (AFI=31)

These AFIs will be paired with the following SAFIs:

* TBD1 (Flow Spec Version 2)

* TBD2 (Flow Spec Version 2 for VPNs)

A compliant FSv2 implementation only has to implement one AFI/SAFI pair out of the full list of NRLIs. For example, a compliant FSv2 implementation might only implement IPv4 FSv2 (AFI=1, SAFI=TBD1).

FSv2 NLRI are encoded in BGP UPDATEs using the MP_REACH_NLRI and MP_UNREACH_NLRI attributes defined in [RFC4760]. When advertising FSv2 NLRI, the length of the Next-Hop Network Address MUST be set to 0. Upon reception, the MP_REACH_NLRI "Network Address of NextHop" field MUST be ignored.

## 3.1. FSv2 NLRI Format

FSv2 Flow Specifications are encoded as an ordered list of TLVs of filter families. FSv2 filter families are typically associated with match criteria for a given networking layer; for example, 802.2 Layer 2, MPLS, IPv4/IPv6, Segment Routing, etc.

The AFI/SAFI NLRI for BGP Flow Specification version 2 (FSv2) has the format:

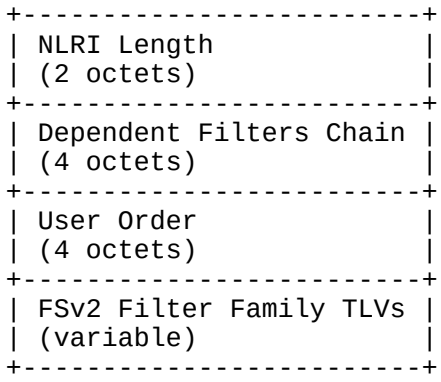


Figure 1: FSv2 NLRI Format

Where:

* NLRI Length: Length of the NLRI field in octets excluding the NLRI Length field. The minimum NLRI Length is 8 (Dependent Filter Chain + User Order).

* Dependent Filters Chain (DFC): A 32-bit unsigned integer in network byte order. When non-zero, the Dependent Filters Chain value is used to associate multiple NLRI together that share dependencies. See Section 3.2 for further information on its use.

* User Order: A 32-bit unsigned integer in network byte order. FSv2 rules with a lower User Order value have a better precedence for filter ordering.

* FSv2 Filter Family TLVs: An ordered list of TLVs of FSv2 filter families. The encoding of these filter families is documented in the next section.

#### 3.1.1. FSv2 Filter Family TLVs

Each each FSv2 Filter Family TLV has the format:

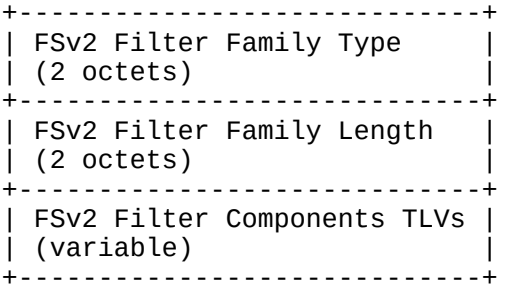


Figure 2: FSv2 Filter Family TLV Format

Where:

* FSv2 Filter Family Type: A 16-bit unsigned integer in network byte order defining the FSv2 filter that is carried in this TLV. For sorting purposes, lower value FSv2 Filter Types have a better precendence than higher values.

* FSv2 Filter Family Length: Length of the FSv2 Filter Components TLVs in octets.

### 3.1.2. FSv2 Filter Component TLVs

Each each FSv2 Filter Component TLV has the Format:

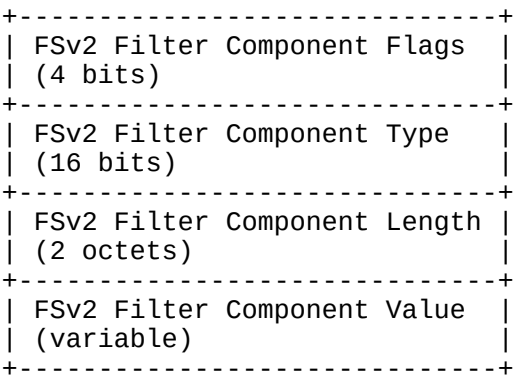


Figure 3: FSv2 Filter Component TLV Format

Where:

* The FSv2 Filter Component Flags are defined as:

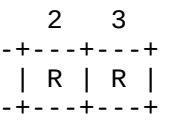


Figure 4: FSv2 Filter Component Flags Format

The fields of the FSv2 Filter Component Flags are defined as:

- O - Optional: When 0, the FSv2 filter-type-specific filter component is mandatory and MUST be supported by the local implementation. Otherwise, when 1, the component is considered "optional". When the component is mandatory and is not supported, the FSv2 filter rule is considered "invalid" for validation purposes.

- R - Reserved: When not otherwise re-defined in a later document, this bit MUST be set to zero when sent and SHOULD be ignored on reception.

* FSv2 Filter Component Type: A 12-bit unsigned integer in network byte order defining the match component for a given FSv2 filter type. For sorting purposes, lower value FSv2 Filter Component Types have a better precedence than higher values.

  This document defines the following FSv2 Filter Component Types. The definition of the type-specific filter components may be defined in other documents:

  0 - Reserved
  50 - L2 Traffic fules
  100 - MPLS traffic rules
  150 - SFC Traffic rules
  200 - Tunneled traffic
  256 - IP Basic Filter Rules (bit 1 of high bit)
  280 - IP Extended Filter Rules

* FSv2 Filter Component Length: A 16-bit unsigned integer in network byte order containing the length of the FSv2 Filter Components Value field.

* FSv2 Filter Component Value: Each FSv2 filter type will define one or more FSv2 filter-type-specific filter components. See each FSv2 filter-type’s specification for a component’s definition.

FSv2 implementations MUST pass valid filter TLVs even if the implementation does not support these installation of these a particular type of filter rules.

This specification only defines operation of the IP Basic Filter Rules that all FSv2 must support.

### 3.2. FSv2 Dependencies

Flow Specifications are implemented using ordered terms. The sorting rules for flow specification routes is intended to, by default, produce a reasonably ordered set of rules for common deployment scenarios.

When the FSv2 rule ordering wouldn’t accomplish the operator’s intent when deploying FSv2, the User Order field can permit the operator to influence the Flow Specification installation order in a deployment.

When set of Flow Specifications are required to implement an operator’s intent and that set of rules has interdependencies, the failure to install a Flow Specification, or part of that specification’s actions, may result in incorrect deployment. An example of such a dependency is two rules covering an IP destination, one with a more-specific and one with a less-specific prefix relaionship. As an example:

1. match dst=10.1.1.1/8 tcp port=25 then dscp=AF1 and permit

2. match dst=10.0.0.0/8 tcp port=25 then drop

If an implementation couldn’t support the DSCP action and failed to install the first rule, SMTP traffic to the host 10.1.1.1 would fail to be delivered due to the second rule’s drop action. In other words, these two entries have a dependency.

When an implementation is unable to install a Flow Specification for some reason, that Flow Specification is locally "invalid". In many circumstances, Flow Specifications that do not have dependencies may be installed on a best-effort basis by an implementation. However, in the case of dependent rules, installing some rules selectively but not others can be problematic.

FSv2 defines for each FSv2 NLRI a Dependent Filters Chain (DFC). When the value of DFC is zero (0), no special consideration is given for dependencies. When the value of DFC is non-zero, when a rule is locally considered invalid, all rules sharing the same DFC value are also considered invalid, and not installed.

### 3.3. Ordering of TLVs within the FSv2 NLRI

For NLRI canonicalization purposes, and also to ease processing, all TLVs within the FSv2 NLRI MUST be ordered in a strictly increasing fashion. FSv2 filter types and FSv2 filter-type-specific component types for a given component MUST NOT occur more than once.

See Section 5.1 for further details.

### 3.4. Partial Deployments

Partial deployments can occur for two reasons:

* Only a portion of the nodes in a network with FSv2 support installing new FSv2 Filter types with new FSv2 components. Other nodes (such as RRs), check the syntax, but do not handle the semantic meaning.

* During upgrades, a portion of the nodes know about a new Filter type with the components, but other nodes do not.

Editor: Are there others?

## 4. FSv2 IP Basic Filters (Filter Family Type TBD)

FSv2 IP Basic filters provide the same functionality as those specified in FSv1 RFCs [FSv1] and [FSv1-IPv6]. The format of those components has been preserved for ease of implementation.

The FSv2 IP Basic filter has been assigned a FSv2 Filter Type value of TBD.

FSv2 IP Basic Filter component types are numbered differently from those in FSv1. FSv2 components have been numbered with gaps to permit future FSv2 IP Basic filter components to be added in between currently specified IP Basic components. This permits a natural default sort order for those new components in implementations.

### 4.1. Operators

Most of the components described below make use of comparison operators. These operators were originally defined in Section 4.2.1 of [FSv1]. They are repeated here for document clarity.

The operators are encoded as a single octet.

#### 4.1.1. Numeric Operator (numeric_op)

This operator is encoded as shown in Figure 3-3.

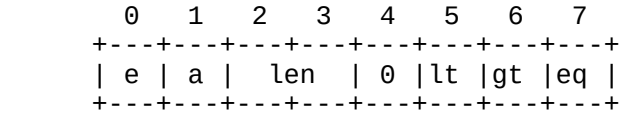


Figure 5: Numeric Operator (numeric_op)

e (end-of-list bit): Set in the last {op, value} pair in the list

a (AND bit): If unset, the result of the previous {op, value} pair is logically ORed with the current one. If set, the operation is a logical AND. In the first operator octet of a sequence, it MUST be encoded as unset and MUST be treated as always unset on decoding. The AND operator has higher priority than OR for the purposes of evaluating logical expressions.

len (length): The length of the value field for this operator given as (1 << len). This encodes 1 (len=00), 2 (len=01), 4 (len=10), and 8 (len=11) octets.

0 MUST be set to 0 on NLRI encoding and MUST be ignored during decoding

lt less-than comparison between data and value

gt: greater-than comparison between data and value

eq: equality between data and value

The bits lt, gt, and eq can be combined to produce common relational operators, such as "less or equal", "greater or equal", and "not equal to", as shown in Table 3-1.

| lt | gt | eq | Resulting operation |
|---|---|---|---|
| 0 | 0 | 0 | false (independent of the value) |
| 0 | 0 | 1 | == (equal) |
| 0 | 1 | 0 | > (greater than) |
| 0 | 1 | 1 | <= (greater than or equal) |
| 1 | 0 | 0 | < (less than) |
| 1 | 0 | 1 | <= (less than or equal) |
| 1 | 1 | 0 | != (not equal value) |
| 1 | 1 | 1 | true (independent of the value) |

Figure 6: Comparison Operation Combinations

#### 4.1.2. Bitmask Operator (bitmask_op)

This operator is encoded as shown in Figure 3-4.

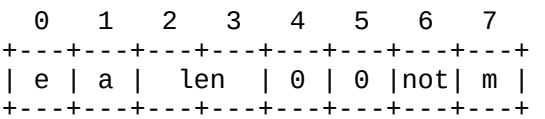


Figure 7: Bitmask Operator (bitmask_op)

Where:

e, a, len (end-of-list bit, AND bit, and length field): Most significant nibble; defined in the Numeric Operator format in section 3-x.

not (NOT bit): If set, logical negation of operation.

m (Match bit): If set, this is a bitwise match operation defined as "(data AND value) == value"; if unset, (data AND value) evaluates to TRUE if any of the bits in the value mask are set in the data.

0 (all 0 bits): MUST be set to 0 on NLRI encoding and MUST be ignored during decoding

### 4.2. FSv2 IP Basic Filter Components

FSv2 IP Basic Filter Components are encoded in FSv2 Filter Component TLVs as described in Section 3.1.2.

The list of valid Basic IP types, covering the functionality defined in [FSv1] and [FSv1-IPv6] are documented below. Additional IP filters are documented in defined in [I-D.hares-idr-fsv2-more-ip-filters].

| Type | Definition |
|---|---|
| 0 | Reserved |
| 10 | IP Destination Prefix |
| 20 | IP Source Prefix |
| 30 | IPv4 Protocol / IPv6 Upper Layer Protocol |
| 40 | Port |
| 50 | Destination Port |
| 60 | Source Port |
| 70 | ICMPv4 Type / ICMPv6 Type |
| 80 | ICMPv4 Code / ICPv6 Code |
| 90 | TCP Flags |
| 100 | Packet Length |
| 110 | DSCP |
| 120 | Fragment |
| 130 | Flow Label |
| 4095 | Reserved |

Table 1: FSv2 IP Basic Components

### 4.3. FSv2 Flow Specification Order of IP Basic Components

For Flow Specification ordering purposes, IP Basic Filter components are ordered similar the FSv1 comparison rules documented in Section 5.1 of [FSv1].

The relative order of two Flow Specificationss with IP Basic filter family components is determined by comparing their respective family-specific components. The algorithm starts by comparing the lowest component type value of the Flow Specifications. If the types differ, the Flow Specification with lowest numeric type value has higher precedence (and thus will match before) than the Flow Specification that doesn't contain that component type. If the component types are the same, then a type-specific comparison is performed (see below). If the types are equal, the algorithm continues with the next component.

For IP prefix values (IP destination or source prefix), if one of the two prefixes to compare is a more specific prefix of the other, the more specific prefix has higher precedence. Otherwise, the one with the lowest IP value has higher precedence.

For all other component types, unless otherwise specified, the comparison is performed by comparing the component data as a binary string using the memcmp() function as defined by [ISO_IEC_9899]. For strings with equal lengths, the lowest string (memcmp) has higher precedence. For strings of different lengths, the common prefix is compared. If the common prefix is not equal, the string with the lowest prefix has higher precedence. If the common prefix is equal, the longest string is considered to have higher precedence than the shorter one.

### 4.4. FSv2 Components for IP Basic TLVs

#### 4.4.1. IP Destination Prefix (component type = 10)

##### 4.4.1.1. IPv4 Destination Prefix (AFI=1)

Encoding: <prefix length (1 octet), prefix (variable)>

Defines the IPv4 destination prefix to match.

*prefix length:* Length of the prefix in bits.
*prefix:* IPv4 Prefix encoded using [RFC4271] NLRI format.

#### 4.4.1.2. IPv6 Destination Prefix (AFI=2)

Encoding: <length (1 octet), offset (1 octet), pattern (variable), padding (variable)>

This defines the IPv6 destination prefix to match. The offset has been defined to allow for flexible matching to portions of an IPv6 address where one is required to skip over the first N bits of the address. (These bits skipped are often indicated as "don't care" bits.) This can be especially useful where part of the IPv6 address consists of an embedded IPv4 address, and matching needs to happen only on the embedded IPv4 address. The encoded pattern contains enough octets for the bits used in matching (length minus offset bits).

*length:* This indicates the N-th most significant bit in the address where bitwise pattern matching stops.
*offset:* This indicates the number of most significant address bits to skip before bitwise pattern matching starts.
*pattern:* This contains the matching pattern. The length of the pattern is defined by the number of bits needed for pattern matching (length minus offset).
*padding:* This contains the minimum number of bits required to pad the component to an octet boundary. Padding bits MUST be 0 on encoding and MUST be ignored on decoding.

If length = 0 and offset = 0, this component matches every address; otherwise, length MUST be in the range offset < length < 129 or the component is malformed.

Note: This Flow Specification component can be represented by the notation ipv6address/length if offset is 0 or ipv6address/offset-length. The ipv6address in this notation is the textual IPv6 representation of the pattern shifted to the right by the number of offset bits.

### 4.4.2. IP Source Prefix (type = 20)

#### 4.4.2.1. IPv4 Source Prefix (AFI=1)

Encoding: <prefix length (1 octet), prefix (variable)>

Defines the IPv4 source prefix to match.

*prefix length:* Length of the prefix in bits.
*prefix:* IPv4 Prefix encoded using [RFC4271] NLRI format.

#### 4.4.2.2. IPv6 Source Prefix (AFI=2)

Encoding: <length (1 octet), offset (1 octet), pattern (variable), padding (variable)>

This defines the source prefix to match. The length, offset, pattern, and padding are the same as in Section 4.4.1.2.

### 4.4.3. IP Protocol/IPv6 Upper Layer Protocol (type = 30)

Encoding: <[numeric_op, value]+>

#### 4.4.3.1. IPv4 Protocol (AFI=1)

Contains a list of {numeric_op, value} pairs that are used to match the IP protocol value octet in IPv4 packet header Section 3.1 of [RFC0791].

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 30 component values SHOULD be encoded as single octet (numeric_op len=00).

#### 4.4.3.2. IPv6 Upper Layer Protocol (AFI=2)

This contains a list of {numeric_op, value} pairs that are used to match the first Next Header value octet in IPv6 packets that is not an extension header and thus indicates that the next item in the packet is the corresponding upper-layer header (see Section 4 of [RFC8200] Section 4).

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 30 component values SHOULD be encoded as a single octet (numeric_op len=00).

Note: While IPv6 allows for more than one Next Header field in the packet, the main goal of the Type 30 Flow Specification component is to match on the first upper-layer IP protocol value. Therefore, the definition is limited to match only on this specific Next Header field in the packet.

### 4.4.4. Port (type = 40)

Encoding: <[numeric_op, value]+>

Defines a list of {numeric_op, value} pairs that match source OR destination TCP/UDP ports (see Section 3.1 of [RFC0793] and the "Format" section of [RFC0768]). This component matches if either the destination port OR the source port of an IP packet matches the value.

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 40 component values SHOULD be encoded as 1- or 2-octet quantities (numeric_op len=00 or len=01).

In case of the presence of the port (destination-port (Section 4.4.5), source-port (Section 4.4.6) component, only TCP or UDP packets can match the entire Flow Specification. The port component, if present, never matches when the packet's IP protocol value is not 6 (TCP) or 17 (UDP), if the packet is fragmented and this is not the first fragment, or if the system is unable to locate the transport header. Different implementations may or may not be able to decode the transport header in the presence of IP options or Encapsulating Security Payload (ESP) NULL [RFC4303] encryption.

Note: This component only matches the first upper layer protocol value in IPv6.

#### 4.4.5. Destination Port (type = 50)

Encoding: <[numeric_op, value]+>

Defines a list of {numeric_op, value} pairs used to match the destination port of a TCP or UDP packet (see also Section 3.1 of [RFC0793] and the "Format" section of [RFC0768].

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 50 component values SHOULD be encoded as 1- or 2-octet quantities (numeric_op len=00 or len=01).

The last paragraph of Section 4.4.4 also applies to this component.

#### 4.4.6. Source Port (type = 60)

Encoding: <[numeric_op, value]+>

Defines a list of {numeric_op, value} pairs used to match the source port of a TCP or UDP packet (see also Section 3.1 of [RFC0793] and the "Format" section of [RFC0768].

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 60 component values SHOULD be encoded as 1- or 2-octet quantities (numeric_op len=00 or len=01).

The last paragraph of Section 4.4.4 also applies to this component.

#### 4.4.7. ICMP Type (type = 70)

Encoding: <[numeric_op, value]+>

Defines a list of {numeric_op, value} pairs used to match the type field of an ICMP packet (see also the "Message Formats" section of [RFC0792]).

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 70 component values SHOULD be encoded as single octet (numeric_op len=00).

##### 4.4.7.1. ICMP IPv4 Type (AFI=1)

In case of the presence of the ICMP type component, only ICMP packets can match the entire Flow Specification. The ICMP type component, if present, never matches when the packet's IP protocol value is not 1 (ICMP), if the packet is fragmented and this is not the first fragment, or if the system is unable to locate the transport header. Different implementations may or may not be able to decode the transport header in the presence of IP options or Encapsulating Security Payload (ESP) NULL [RFC4303] encryption.

##### 4.4.7.2. ICMP IPv6 Type (AFI=2)

In case of the presence of the ICMPv6 type component, only ICMPv6 packets can match the entire Flow Specification. The ICMPv6 type component, if present, never matches when the packet's upper-layer IP protocol value is not 58 (ICMPv6), if the packet is fragmented and this is not the first fragment, or if the system is unable to locate the transport header. Different implementations may or may not be able to decode the transport header.

#### 4.4.8. ICMP Code (type = 80)

Encoding: <[numeric_op, value]+>

##### 4.4.8.1. ICMP Code IPv4 Type (AFI=1)

Defines a list of {numeric_op, value} pairs used to match the code field of an ICMP packet (see also the "Message Formats" section of [RFC0792]).

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 80 component values SHOULD be encoded as single octet (numeric_op len=00).

In case of the presence of the ICMP code component, only ICMP packets can match the entire Flow Specification. The ICMP code component, if present, never matches when the packet's IP protocol value is not 1 (ICMP), if the packet is fragmented and this is not the first fragment, or if the system is unable to locate the transport header. Different implementations may or may not be able to decode the transport header in the presence of IP options or Encapsulating Security Payload (ESP) NULL [RFC4303] encryption.

#### 4.4.8.2. ICMP Code IPv6 Type (AFI=2)

This defines a list of {numeric_op, value} pairs used to match the code field of an ICMPv6 packet (see also Section 2.1 of [RFC4443]).

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 80 component values SHOULD be encoded as a single octet (numeric_op len=00).

In case of the presence of the ICMPv6 code component, only ICMPv6 packets can match the entire Flow Specification. The ICMPv6 code component, if present, never matches when the packet's upper-layer IP protocol value is not 58 (ICMPv6), if the packet is fragmented and this is not the first fragment, or if the system is unable to locate the transport header. Different implementations may or may not be able to decode the transport header.

### 4.4.9. TCP Flags (type = 90)

Encoding: <[bitmask_op, bitmask]+>

Defines a list of {bitmask_op, bitmask} pairs used to match TCP control bits (see also Section 3.1 of [RFC0793]).

This component uses the Bitmask Operator (bitmask_op) described in Section 4.1.2. Type 90 component bitmasks MUST be encoded as 1- or 2-octet bitmask (bitmask_op len=00 or len=01).

When a single octet (bitmask_op len=00) is specified, it matches octet 14 of the TCP header (see also Section 3.1 of [RFC0793]), which contains the TCP control bits. When a 2-octet (bitmask_op len=01) encoding is used, it matches octets 13 and 14 of the TCP header with the data offset (leftmost 4 bits) always treated as 0.

In case of the presence of the TCP flags component, only TCP packets can match the entire Flow Specification. The TCP flags component, if present, never matches when the packet's IP protocol value is not 6 (TCP), if the packet is fragmented and this is not the first fragment, or if the system is unable to locate the transport header.

Different implementations may or may not be able to decode the transport header in the presence of IP options or Encapsulating Security Payload (ESP) NULL [RFC4303] encryption.

#### 4.4.10. Packet length (type = 100)

Encoding: <[numeric_op, value]+>

Defines a list of {numeric_op, value} pairs used to match on the total IP packet length (excluding Layer 2 but including IP header).

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 100 component values SHOULD be encoded as 1- or 2-octet quantities (numeric_op len=00 or len=01).

#### 4.4.11. DSCP (Differentiaed Services Code Point)(type = 110)

Encoding: <[numeric_op, value]+>

Defines a list of {numeric_op, value} pairs used to match the 6-bit DSCP field (see also [RFC2474]).

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 110 component values MUST be encoded as single octet (numeric_op len=00).

The six least significant bits contain the DSCP value. All other bits SHOULD be treated as 0.

#### 4.4.12. Fragment (type = 120)

Encoding: <[bitmask_op, bitmask]+>

Defines a list of {bitmask_op, bitmask} pairs used to match specific IP fragments.

This component uses the Bitmask Operator (bitmask_op) described in Section 4.1.2. Type 120 component bitmask MUST be encoded as single octet bitmask (bitmask_op len=00).

##### 4.4.12.1. IPv4 Fragment (AFI=1)

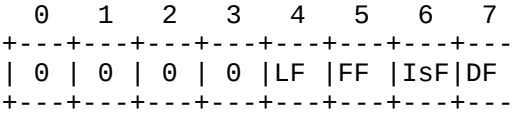


Figure 8: IPv4 Fragment Bitmask Operand

Bitmask values:

*DF (Don’t Fragment):* match if IP Header Flags Bit-1 (DF) [RFC0791] is 1

*IsF (Is a fragment other than the first):* match if the [RFC0791] IP Header Fragment Offset is not 0

*FF (First Fragment):* match if the [RFC0791] IP Header Fragment Offset is 0 AND Flags Bit-2 (MF) is 1

*LF (Last Fragment):* match if the [RFC0791] IP Header Fragment Offset is not 0 AND Flags Bit-2 (MF) is 0

*0:* MUST be set to 0 on NLRI encoding and MUST be ignored during decoding

#### 4.4.12.2. IPv6 Fragment (AFI=2)

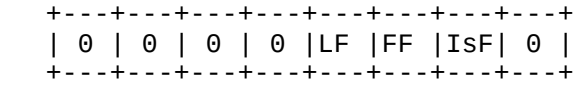


Figure 9: IPv6 Fragment Bitmask Operand

Bitmask values:

*IsF:* Is a fragment other than the first -- match if IPv6 Fragment Header (Section 4.5 of [RFC8200]) Fragment Offset is not 0

*FF:* First fragment -- match if IPv6 Fragment Header (Section 4.5 of [RFC8200]) Fragment Offset is 0 AND M flag is 1

*LF:* Last fragment -- match if IPv6 Fragment Header (Section 4.5 of [RFC8200]) Fragment Offset is not 0 AND M flag is 0

*0:* MUST be set to 0 on NLRI encoding and MUST be ignored during decoding

### 4.4.13. Flow Label (type = 130), AFI=2 only

Encoding: <[numeric_op, value]+>

This contains a list of {numeric_op, value} pairs that are used to match the 20-bit Flow Label IPv6 header field (Section 3 of [RFC8200]).

This component uses the Numeric Operator (numeric_op) described in Section 4.1.1. Type 130 component values SHOULD be encoded as 4-octet quantities (numeric_op len=10).

## 4.5. FSv2 Traffic Filtering Actions for FSv2 IP Basic

Traffic matching a flow specification filter may have selected _traffic actions_ applied to it that have various impacts on the matched traffic. FSv2 IP Basic allows flow specification actions to be attached to flow specification routes using BGP Extended Communities (FSv2-EC) encoded using the Extended Community formats [RFC4360] or in the IPv6 Address Specific Extended Community format [RFC5701].

Section 4.5.1 describes the interaction between FS-EC action, and categories of actions. Section 4.5.2 describes the existing FS-EC action formats. Section 4.5.5 defines an optional FS-EC to pass information ordering of categories (user/this standard) and failure action (stop or best effort).

### 4.5.1. Categories of FSv2 Actions and their Interactions

FSv2-EC actions fall into the following categories:

* Further constraint of the match criteria for the traffic in addition to that which is encoded in the NLRI.

* Apply traffic shaping mechanisms, such as bps/pps rate limiters.

* Change IP packet properties, such as DSCP.

* Redirect (change the forwarding) of the traffic. Examples include redirecting to VPN VRFs, or forwarding to tunneled destinations.

* Flag the traffic for sampling.

* Terminate the evaluation of further flow specification matches in the forwarding plane.

When multiple actions from a given FSv2-EC category are present in a FSv2 route, these actions may _conflict_. Conflicting actions result in ambiguity as to what traffic action behavior is applied to traffic matching the flow specification.

FSv2 actions passed in a BGP Community Container Attribute can provide ordering of actions, dependencies, or signal which actions are valid within a category (see [fsv2-more-ip-filters]). However, these features are beyond the Basic FSv2 for IP forwarding and are out of scope for this specification.

#### 4.5.2. FSv2 Extended Community Actions

FSv2 IP Basic uses FSv1 actions and these are referenced in Section 4.5.2.1 and Section 4.5.2.2.

One additional, optional, FSv2 specific FS-EC: the Action Chain Ordering (ACO) Extended Community (ACO-EC), is defined in Section 4.5.5. ACO-EC can carry defaults currently only available by configuration in FSv1.

##### 4.5.2.1. Existing Flow Specification Action Extended Communities

FSv1 defines a set of [RFC4360] encoded extended communities implementing actions also applicable to FSv2 IP Basic match types. They are:

| Type/Sub-Type | Description | Short-ID | Reference |
|---|---|---|---|
| 0x01/0x0c | Redirect to IP | RD-IP | [redirect-ip] |
| 0x07/0x02 | Match Interface set | TA-IS | [interface-set] |
| 0x09/0xxx | Redirect to Indirection ID | RD-IID | [path-redirect] |
| 0x0b/0x00 | SFC Reserved | SFC-R | [RFC9015] |
| 0x0b/0x01 | SFVC SFIR POOL Identifier | SFIR-PI | [RFC9015] |
| 0x0b/0x02 | SFC MPLS label stack Swapping or stacking labels | SFC-MPLS | [RFC9015] |
| 0x80/0x06 | Traffic rate limit by bytes | TR-BPS | [FSv1] |
| 0x80/0x07 | Traffic Action (sample, terminal) | TA | [FSv1] |
| 0x80/0x08 | Redirection to VRF (2-octet AS form) | RD-VRF-AS2 | [FSv1] |
| 0x80/0x09 | Traffic mark DSCP | TM-DSCP | [FSv1] |
| 0x80/0x0C | Traffic rate limit by packets | TR-BPS | [FSv1] |
| 0x81/0x08 | Redirect to VPN (IPv4 form) | RD-VRF-IPv4 | [FSv1] |
| 0x81/0x08 | Redirect to VPN (4-octet AS form) | RD-VRF-AS4 | [FSv1] |

Table 2: FSv1 Extended Communities Used by FSv2

Note the Short ID is simply a quick way for this document to reference a particular action.

#### 4.5.2.2. Existing Flow Specification Actions IPv6 Address Specific Extended Communities

FSv1 defines a set of [RFC5701] encoded extended communities implementing actions also applicable to FSv2 IP Basic match types. They are:

| Type | Description | Short-ID | Reference |
|---|---|---|---|
| 0x000C | FS Redirect to IPv6 | RD-IP6 | [redirect-ip] |
| 0x000D | FS Redirect to VPN by IPv6 route target | RD-VRF-IPv6 | [FSv1-IPv6] |

Table 3: FSv1 IPv6 Address Specific Extended Communities Used by FSv2

### 4.5.3. Failure of an FS-EC Action

Devices implementing flow specification matching and traffic actions may be unable, for whatever reason, to carry out the signaled actions for the matched traffic. Some examples of this inability include:

* The action is not implemented in the forwarding plane.

* Combinations of non-conflicting actions may not be able to be simultaneously executed due to limitation in the implementation’s forwarding plane.

When FS-EC actions known to the implementation are attached to a flow specification route and an action cannot be executed, there are three potential options:

Option 1: Stop processing additional filters and (optionally) signal failure to the management process.

Option 2: Continue on processing in "best effort" for the next filters.

Option 3: Decide between 1 and 2 based on dependencies between filters and actions.

Option 1 and 2 can be signaled by configuration within a Flow Specification implementation.

Option 3 requires the encoding dependency lists in ordered filters and ordered actions. The FSv2 NLRI format has a field to carry filter dependency information, but these functions are beyond the FSv2 Basic IP functions and out of scope for this specification.

Consider an example where three FSv2-EC actions are present on the route: Set the DSCP value, request sampling of the traffic, redirect to a VRF. If the implementation is unable to set the DSCP value:

Option 1 would be to stop processing and not do the other two actions.

Option 2 would be to continue prcoessing and do the other two actions.

Currently, for FSv1, local configuration or implementation behavior determines what happens if one of the actions fails within a set of multiple actions attached to a filter rule.

One option for FSv2 is to pass another FS-EC indicating what the originator expects will happen upon failure of an action.

#### 4.5.4. Unknown FSv2-EC Actions

A flow specification implementation that understands extended communities for a traffic action may not necessarily be able to implement them. Another problematic case for consistent deployment of flow specification within a network is understanding that an implementation may be ignorant of some FSv2-ECs.

FSv2-ECs are carried in the general purpose BGP Extended Community features. The expected behavior for an implementation receiving unknown Extended Communities, depending on configuration and policy, will be to ignore the contents of these communities and propagate them according to the transitivity rules in [RFC4360].

Newly defined FSv2-ECs may be unknown to the implementation, typically as a result of incremental deployment newer flow specification traffic actions. When a network with older implementations receive such newly defined FSv2-ECs, the older implementations are unable to determine that an action has been requested at all. The default behavior thus becomes "best effort" for executing the known FSv2-ECs.

When specifying new FSv2-ECs, operational consideration MUST be given to what the behavior of such ignorant implementations may do to the desired traffic forwarding throughout the FS deployment.

#### 4.5.5. Action Chain Ordering FSv2-EC (ACO) (optional)

Summary: This optional FSv2-EC passes information on what the BGP peer originating the FSv2-EC expects will happen with multiple actions attached to a single filter.

Description: The BGP peer originating multiple FSv2 FS-EC actions attached to FSv2 NLRI (filters) may attach the Action Chain Ordering (ACO) FS-EC to inform BGP Peers receiving the FSv2 information how the originating pair expects action interactions and actions failures will be handled. Two fields are encoded in this FS-EC:

AC-interaction - What happens if two actions are specified in a category, and

AC-Failure - what happens if an action with multiple action set fails.

Encoding: The Generic Transitive encoding is shown in Figure 10 with the field definitions below.

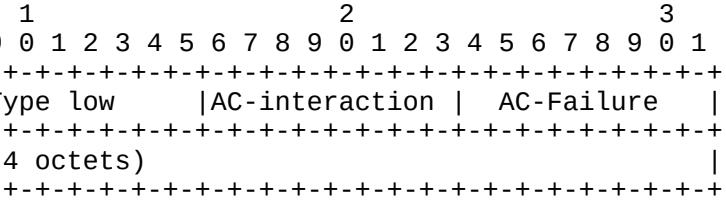


Figure 10: Action Chain Ordering FSv2-EC

where:

*Type high:* This 1 octet field has a value of 0x80 For the Generic Transitive EC.

*Type low:* This one octet field identifies the ACO-Action. The value is TBD4.

*AC-interaction:* This field indicates whether the FS-EC category order is the pre-defined order or an implementation specific order.

* 0 (default): Do not install actions with two actions per category.

* 1 (local config): Allow under local configuration.

*AC-failure:* 1 octet byte that determines the action on failure. Actions may succeed or fail and an Action chain must deal with it. The default value stored for an action chain that does not have this action chain is "stop on failure". AC-Failure types are:

* 0x00: Stop on failure of an action.

* 0x01: Continue on failure of an action.

*Reserved:* Reserved for future use. Must be set to all zeros, and ignored upon reception.

## 5. Validation and Ordering of FS Routes

The validation of FSv2 routes adheres to the combination of rules for general BGP FSv1 routes found in [FSv1], [FSv1-IPv6], and [RFC9117]. These FSv1 rules are sufficient for FSv2 for IP traffic.

Specific additions have been defined for IP Filters used for guiding IP traffic into Service Function Service Function Pathways SFC NLRI in [RFC9015], or validation of L2VPN FS NLRI (see [I-D.ietf-idr-flowspec-l2vpn]). These additions are not required for the FSv2 for IP Basic functions.

Validating FSv2 routes procedes through the following steps:

* Syntactic and semantic validitation for FSv2 NLRI (Section 5.1).

* Validating FSv2 route properties (Section 5.2.2).

* Validating FSv2 route actions (Section 5.2.3).

The full validation process for FSv2 routes for all AFI/SAFIs is described below in Section 5.2.2 rather than simply referring to the relevant portions of the previously referenced RFCs.

### 5.1. Validating FSv2 NLRI

All FSv2 NLRI MUST be well-formed.

Failure of the following NLRI validation conditions MUST use "session reset" for [RFC7606] purposes since recovery from NLRI malformation cannot is not possible:

* NLRI Length (Section 3.1) MUST be at least 20 octets:

  - Dependent Filters Chain (4 octets) + User Order (4 octets) +

- One non-empty FSv2 Filter Family TLV (FSv2 Filter Family Type (2 octets) + FSv2 Filter Family Length (2 octets)) +

- One possibly-empty FSv2 Filter Component (FSv2 Filter Component Flags + Type (2 octets)+ FSv2 Filter Component Length (2 octets))

* All TLVs and sub-TLVs MUST be well-formed and exactly contained in their parent TLVs: The total length of all sub-TLVs must be identical to the length field of the parent TLV.

Failure of the following NLRI validation conditions MUST use "treat-as-withdraw" for the NLRI for [RFC7606] purposes. In these cases, it is possible to parse the boundaries of individual NLRI in a BGP UPDATE message and thus the BGP speaker can continue to parse the next NLRI in the UPDATE. Implementations also MUST notify the operator of this behavior: In circumstances where routes have been announced by a previously valid NLRI but failed to be properly withdrawn due to an implicit or explicit withdraw of a malformed NLRI, "stuck" routes may result in the network.

* TLVs of a given class (FSv2 Filter Family, FSv2 Filter Component for a FSv2 Filter Family) MUST be present no more than once in an NLRI. (No duplicates TLVs.)

* TLVs of a given class MUST be ordered from lowest to highest. (TLVs need to be sorted.)

* When a TLV’s value field is understood by the implementation, the value MUST have a length appropriate for that TLV type.

* When a TLV’s value field is understood by the implementation, the value field MUST be well-formed according the definition of that TLV type.

Implementations MAY, depending on configuration, restrict propagation of FSv2 routes with NLRI containing Filter Families or Filter Components that they are ignorant of the encodings for. This is permitted only when the NLRI are otherwise not considered malformed by the implementation. This behavior is useful for BGP speakers, such as route reflectors, to generically disseminate FSv2 routes that they themselves might not utilize for traffic filtering.

The above rules permit FSv2 implementations that are ignorant of a given Filter Family, or Filter Family's Component encoding to propagate the FSv2 route to other BGP speakers in the deployment. However, since semantic checks for a given Filter Family's components can only be effected by implementations aware of that component, ignorant upstream BGP speakers may propagate semantically-incorrect NLRI until it reaches a BGP speaker that understands the encoding.

### 5.2. Validation of FSv2 BGP Routes

By Section 1.1 of [RFC4271] definition, a BGP route is a pairing of its destination (NLRI) and Path Attributes. The prior section discussed the validation of the NLRI. This section discusses validation of the pairing of the NLRIs in an UPDATE along with their Path Attributes as BGP routes.

Flow specifications received from a BGP peer that are accepted in the respective Adj-RIB-In are used as input to the route selection process. Although the forwarding attributes of the two routes for the same prefix may be the same, BGP is still required to perform its path selection algorithm in order to select the correct set of attributes to advertise.

The first step of the BGP Route selection procedure (Section 9.1.2 of [RFC4271]) is to exclude from the selection procedure routes that are considered unfeasible. In the context of IP routing information, this is used to validate that the next hop of a given route is resolvable.

This concept can be extended in the case of the Flow Specification NLRI to allow other validation procedures.

#### 5.2.1. AFI/SAFIs Used For Validation

The FSv2 validation process validates the FSv2 NLRI with following unicast routes received over the same AFI (1 or 2) but different SAFIs:

| Received AFI/SAFI | Validate Route Using AFI/SAFI |
|---|---|
| 1/TBD1 | 1/1 (IPv4-Unicast) |
| 1/TBD2 | 1/128 (IPv4-Labeled Unicast) |
| 2/TBD1 | 2/1 (IPv6-Unicast) |
| 2/TBD2 | 2/128 (IPv6-Labeled Unicast) |
| 31/TBD1 | 1/1 (IPv4-Unicast) |
| 31/TBD2 | 1/128 (IPv4-Labeled Unicast) |
| 6/TBD1 | 1/1 (IPv4-Unicast) |
| 256/TBD2 | 1/128 (IPv4-Labeled Unicast) |

Table 4: FSv2 Flowspec BGP Route AFI/SAFI Validation

#### 5.2.2. FSv2 Route Validation Procedure

In the absence of explicit configuration, a Flow specification NLRI (FSv1 or FSv2) MUST be validated such that it is considered feasible if and only if all of the conditions are true:

a. A destination prefix component is embedded in the Flow Specification.

b. One of the following conditions holds true:

    1. The originator of the Flow Specification matches the originator of the best-match unicast route for the destination prefix embedded in the flow specification (this is the unicast route with the longest possible prefix length covering the destination prefix embedded in the flow specification).

    2. The AS_PATH attribute of the flow specification is empty or contains only an AS_CONFED_SEQUENCE segment. [RFC5065].

        2a This condition SHOULD be enabled by default.

        2b This condition MAY be disabled by explicit configuration on a BGP Speaker.

2c As an extension to this rule, a given non-empty AS_PATH (besides AS_CONFED_SEQUENCE segments) MAY be permitted by policy.

c. There are no "more-specific" unicast routes when compared with the flow destination prefix that have been received from a different neighbor AS than the best-match unicast route, which has been determined in rule b.

However, part of rule a may be relaxed by explicit configuration, permitting Flow Specifications that include no destination prefix component. If such is the case, rules b and c are moot and MUST be disregarded.

By "originator" of a BGP route, we mean either the address of the originator in the ORIGINATOR_ID Attribute [RFC4456] or the source address of the BGP peer, if this path attribute is not present.

A BGP implementation MUST enforce that the AS in the left-most position of the AS_PATH attribute of a Flow Specification Route (FSv1 or FSv2) received via the Exterior Border Gateway Protocol (eBGP) matches the AS in the left-most position of the AS_PATH attribute of the best-match unicast route for the destination prefix embedded in the Flow Specification (FSv1 or FSv2) NLRI.

The best-match unicast route may change over time independently of the Flow Specification NLRI (FSv1 or FSv2). Therefore, a revalidation of the Flow Specification MUST be performed whenever unicast routes change. Revalidation is defined as retesting rules a to c as described above.

#### 5.2.3. Validation of Flow Specification Actions for FSv2 for IP Basic

FSv2 routes can carry one or more filtering action extended communities (FS-EC) that are executed when the flow specification filter matches traffic. These extended communities are syntactically validated using the procedures in [RFC4360] and [RFC7606].

Section 4.5.2 discusses the procedures for utilizing FSv2-EC actions as part of traffic filtering.

## 6. Traffic Filtering

Section 5 of [FSv1] discusses the general behavior of using flow specification for traffic filtering. FSv2 provides the additional ability to apply traffic filtering at different portions of a forwarding path.

The code points assigned to the Filter Family Types and Filter Component Types for a given Filter Family are arranged to support a reasonable default traffic filtering (match, and actions) behavior. For example, the Component orders for FSv1 and FSv2 IP Basic can match traffic as part of monitoring, or mitigating, distributed denial of service (DDoS) attacks. However, that default ordering may be unsuitable for all filtering situations. FSv1 provided no mechanism to deviate from the ordering rules in Section 5.1 of [FSv1].

The User Order field of the NLRI (Section 3.1) permits an operator to override the default sort ordering of FSv2 rules to effect their desired traffic filtering behavior when it deviates from the default order.

Note that the procedures in Section 5.1 have ensured that TLVs are distinctly numbered and sorted. This assists with the procedures in the following section.

### 6.1. Ordering of FSv2 Flow Specifications

More than one Flow Specification may match a particular traffic flow. Thus, it is necessary to define the order in which Flow Specifications get matched and actions being applied to a particular traffic flow. This ordering function is such that it does not depend on the arrival order of the Flow Specification via BGP and thus is consistent in the network.

FSv2 routes consist of a series of Filter Families containing Filter Components for those Filter Families. Filter Families are generally ordered where their match criteria match lower network layers based on lower-numbered Filter Family Types. However, they may also be ordered based on where the default match order for that Filter Family vs. other Filter Families should occur.

Similarly, within a Filter Family, Filter Components are ordered based to permit the default match order for that Filter Family to be naturally ordered as part of sorting FSv2 routes.

Thus, for FSv2, the choice of code point for Filter Family, or Filter Component is chosen to represent the default sort order for traffic filtering.

The relative order of two FSv2 flow specifications is determined in the following fashion:

1. A route with a lower User Order value (Section 3.1) comes before a route with a higher User Order value.

2. Each route’s Filter Family TLVs are then compared in a pair-wise fashion. A route with a lower FSv2 Filter Family Type value (Section 3.1.1) comes before a route with a higher Filter Family Type value.

3. When both routes have the same Filter Family Type, each Filter Component TLV for that Filter Family are compared in a pair-wise fashion. A route with a lower FSv2 Filter Component Type value (Section 3.1.2) comes before a route with a higher Filter Component Type value.

4. When Filter Component Types are identical, Filter Component Values are compared:

    * For IP prefix values (IP destination or source prefix), if one of the two prefixes to compare is a more specific prefix of the other, the route with the more-specific prefix comes before the route with the less-specific prefix. Otherwise, the route with the lowest IP value comes before the route with the higher IP value.

    * For all other Filter Component Types, unless otherwise specified, the comparison is performed by comparing the Filter Component data as a binary string using the memcmp() function as defined by [ISO_IEC_9899]. For strings with equal lengths, the lowest string (memcmp) has higher precedence. For strings of different lengths, the common prefix is compared. If the common prefix is not equal, the string with the lowest prefix has higher precedence. If the common prefix is equal, the longest string is considered to have higher precedence than the shorter one.

*Warning:* Specifications for FSv2 Filter Components are permitted to define their sort comparison criteria for that component. However, when implementations are ignorant of that Filter Component, it can only sort the components based on the general memcmp mechanism described above. In the case where a deployment contains implementations that are ignorant of a given filtering behavior, one of the two things SHOULD be done by the operator to avoid inappropriate traffic filtering or forwarding:

* The User Order field should be utilized to prevent inappropriate ordering of FSv2 routes that ignorant implementations may misorder.

* The Filter Component Type should be marked as "mandatory" (Section 3.1.2) and dependent FSv2 filters placed in an appropriate, non-zero, Dependent Filter Chain (Section 3.1).

### 6.2. Installation of FSv2 Filters

Once FSv2 flow specifications have been ordered according to the rules of the prior section, they are eligible to be installed for traffic filtering purposes. However, it is possible that a given device is incapable of implementing all match components, or actions.

FSv2 flow specifications are evaluated to see if their match and action elements are able to be executed on the device. When the evaluation is "valid", the flow specification (match and actions) are eligible to be installed in the relative sort order determined in the prior section.

When FSv2 flow specifications are determined to be "invalid", it impacts not only the individual flow specification that has been deemed invalid, but also all FSv2 entries sharing the same non-zero Dependent Filter Chain value (Section 3.1).

For filtering components, Section 3.1.2 defines the FSv2 Filter Component Flags field. When a device is unable to implement the match criteria contained in that Filter Component - for whatever reason - the "Optional" bit is checked. If the Optional bit is unset (zero), the Filter Component is "mandatory" and the flow specification filter is considered "invalid". If the filter bit is set (one), the Filter Component is "optional", and the device is free to install the flow specification in the sorted order minus the Filter Component in question as a "valid" entry.

Similarly, if a flow specification’s traffic filtering actions are unable to be installed by the device, the implementation may determine whether or not the flow specification is valid or invalid based on implementation defaults, or configuration. The ACO community may be used on supporting implementations to influence validity in these circumstances.

Features governing ordered FSv2 action and validity evaluation may be considered in the future.

Once validation of all FSv2 flow specification is complete, eligible FSv2 flow specifications are installed as traffic filters.

### 6.3. Ordering of FS filters for BGP Peers which support FSv1 and FSv2

FSv2 allows the user to order flow specification rules and the actions associated with a rule. Each FSv2 rule has one or more match conditions and one or more actions associated with each rule.

FSv1 and FSv2 filters are sent as different AFI/SAFI pairs so FSv1 and FSv2 operate as ships-in-the-night. Some BGP peers in an AS may support both FSv1 and FSv2. Other BGP peers may support FSv1 or FSv2. Some BGP will not support FSv1 or FSV2. A coherent flow specification technology must have consistent best practices for ordering the FSv1 and FSv2 filter rules.

One simple rule captures the best practice: Order the FSv1 filters after the FSv2 filter by placing the FSv1 filters after the FSv2 filters.

To operationally make this work, all flow specification filters should be included the same data base with the FSv1 filters being assigned a user- defined order beyond the normal size of FSv2 user-ordered values. A few examples, may help to illustrate this best practice.

Example 1: User ordered numbering - Suppose you might have 1,000 rules for the FSv2 filters. Assign all the FSv1 user defined rules to 1,001 (or better yet 2,000). The FSv1 rules will be ordered by the components and component values.

Example 2: Storage of actions - All FSv1 actions are defined ordered actions in FSv2. Translate your FSv1 actions into FSv2 ordered actions for storing in a common FSv1-FSv2 flow specification data base.

## 7. Scalability and Aspirations for FSv2

Operational issues drive the deployment of BGP flow specification as a quick and scalable way to distribute filters. The early operations accepted the fact validation of the distribution of filter needed to be done outside of the BGP distribution mechanism. Other mechanisms (NETCONF/RESTCONF or PCEP) have reply-request protocols.

These features within BGP have not changed. BGP still does not have an action-reply feature.

NETCONF/RESTCONF latest enhancements provide action/response features which scale. The combination of a quick distribution of filters via BGP and a long-term action in NETCONF/RESTCONF that ask for reporting of the installation of FSv2 filters may provide the best scalability.

The combination of NETCONF/RESTCONF network management protocols and BGP focuses each protocol on the strengths of scalability.

FSv2 will be deployed in webs of BGP peers which have some BGP peers passing FSv1, some BGP peers passing FSv2, some BGP peers passing FSv1 and FSv2, and some BGP peers not passing any routes.

The TLV encoding and deterministic behaviors of FSv2 will not deprecate the need for careful design of the distribution of flow specification filters in this mixed environment. The needs of networks for flow specification are different depending on the network topology and the deployment technology for BGP peers sending flow specification.

Suppose we have a centralized RR connected to DDoS processing sending out flow specification to a second tier of RR who distribute the information to targeted nodes. This type of distribution has one set of needs for FSv2 and the transition from FSv1 to FSv2.

Suppose we have Data Center with a 3-tier backbone trying to distribute DDoS or other filters from the spine to combinational nodes, to the leaf BGP nodes. The BGP peers may use RR or normal BGP distribution. This deployment has another set of needs for FSv2 and the transition from FSv1 to FSV2.

Suppose we have a corporate network with a few AS sending DDoS filters using basic BGP from a variety of sites. Perhaps the corporate network will be satisfied with FSv1 for a long time.

These examples are given to indicate that BGP FSv2, like so many BGP protocols, needs to be carefully tuned to aid the mitigation services within the network. This protocol suite starts the migration toward better tools using FSv2, but it does not end it. With FSv2 TLVs and deterministic actions, new operational mechanisms can start to be understood and utilized.

This FSv2 specification is merely the start of a revolution of work not the end.

## 8. Optional Security Additions

This section discusses the optional BGP Security additions for BGP-FS v2 relating ROA [RFC9582].

### 8.1. BGP FSv2 with ROA

BGP FSv2 can utilize ROAs in the validation. If BGP FSv2 is used with BGPSEC and ROA, the first thing is to validate the route within BGPSEC and second to utilize BGP ROA to validate the route origin.

The BGP-FS peers using both ROA and BGP-FS validation determine that a BGP Flow specification is valid if and only if one of the following cases:

* If the BGP Flow Specification NLRI has a IPv4 or IPv6 address in destination address match filter and the following is true:
  - A BGP ROA has been received to validate the originator, and
  - The route is the best-match unicast route for the destination prefix embedded in the match filter; or
* If a BGP ROA has not been received that matches the IPv4 or IPv6 destination address in the destination filter, the match filter must abide by the [FSv1] and [FSv1-IPv6] validation rules as follows:
  - The originator match of the flow specification matches the originator of the best-match unicast route for the destination prefix filter embedded in the flow specification", and
  - No more specific unicast routes exist when compared with the flow destination prefix that have been received from a different neighboring AS than the best-match unicast route, which has been determined in step A.

The best match is defined to be the longest-match NLRI with the highest preference.

# 9. IANA Considerations

This section complies with [RFC7153].

## 9.1. Flow Specification V2 SAFIs

IANA is requested to assign two SAFI Values in the registry at https://www.iana.org/assignments/safi-namespace from the Standard Action Range as follows:

Table 7-1 SAFIs

| Value | Description | Reference |
|---|---|---|
| TBD1 | BGP FSv2 | [this document] |
| TBD2 | BGP FSv2 VPN | [this document] |

### 9.2. Generic Transitive Extended Community

IANA is requested to assign a type value from the "Generic Transitive Extended Community Sub-Types" registry at https://www.iana.org/assignments/bgp-extended-communities/bgp-extended-communities.xhtml

Table 7-3 - Generic Transitive Extended Community

| Value | Description | Reference | Controller |
|---|---|---|---|
| TBD4 | FSv2 Action Chain Ordering | [this document] | IETF |

### 9.3. FSv2 IP Filters Component Types

IANA is requested to create a new "BGP FSv2 IP Basic Component Types" registry and indicate [this draft] as a reference. The following assignments in the FSv2 IP Basic Filters Component Types Registry shold be made.

Registry Name: BGP FSv2 Component Types
Reference: [this document]
Registration Procedures: 0x01-0x3FFF Standards Action, 0x4000-0xFFFF FCFS.

| Type | Definition | Reference |
|---|---|---|
| 0 | Reserved | This document |
| 10 | IP Destination Prefix | This document |
| 20 | IP Source Prefix | This document |
| 30 | IPv4 Protocol / IPv6 Upper Layer Protocol | This document |
| 40 | Port | This document |
| 50 | Destination Port | This document |
| 60 | Source Port | This document |
| 70 | ICMPv4 Type / ICMPv6 Type | This document |
| 80 | ICMPv4 Code / ICPv6 Code | This document |
| 90 | TCP Flags | This document |
| 100 | Packet Length | This document |
| 110 | DSCP | This document |
| 120 | Fragment | This document |
| 130 | Flow Label | This document |
| 4095 | Reserved | This document |

Table 5: BGP FSv2 IP Basic Component Types

## 9.4. FSv2 Filter Component Types

IANA is requested to create the a new registry for "Flow Specification v2 Filter Component Types".

Registration Procedures: 0x01-0x3FFF Standards Action.

| Type | Description | Reference |
|---|---|---|
| 0 | Reserved | [this document] |
| 1-49 | Unassigned | [this document] |
| 50 | L2 Traffic Rules | [this document] |
| 51-99 | Unassigned | [this document] |
| 100 | MPLS traffic rules | [this document] |
| 101-149 | Unassigned | [this document] |
| 150 | SFC Traffic rules | [this document] |
| 151-199 | Unassigned | [this document] |
| 200 | Tunnel Traffic rules | [this document] |
| 201-255 | Unassigned | [this document] |
| 256 | IP traffic rules | [this document] |
| 257-279 | Unassigned | [this document] |
| 280 | Extended IP Rules | [this document] |
| 281-24575 | Unassigned | [this document] |
| 24576-32767 | Vendor specific | [this document] |
| 32768-65535 | Reserved | [this document] |

Table 6: Flow Specification v2 Filter Component Types

## 10. Security Considerations

The use of ROA improves on [FSv1] by checking to see of the route origination. This check can improve the validation sequence for a multiple-AS environment.

>The use of BGPSEC [RFC8205] to secure the packet can increase security of BGP flow specification information sent in the packet.

The use of the reduced validation within an AS [RFC9117] can provide adequate validation for distribution of flow specification within a single autonomous system for prevention of DDoS.

Distribution of flow filters may provide insight into traffic being sent within an AS, but this information should be composite information that does not reveal the traffic patterns of individuals.

# 11. References

## 11.1. Normative References

[FSv1] Loibl, C., Hares, S., Raszuk, R., McPherson, D., and M. Bacher, "Dissemination of Flow Specification Rules", RFC 8955, DOI 10.17487/RFC8955, December 2020, <https://www.rfc-editor.org/info/rfc8955>.

[FSv1-IPv6] Loibl, C., Ed., Raszuk, R., Ed., and S. Hares, Ed., "Dissemination of Flow Specification Rules for IPv6", RFC 8956, DOI 10.17487/RFC8956, December 2020, <https://www.rfc-editor.org/info/rfc8956>.

[I-D.ietf-idr-flowspec-l2vpn] Weiguo, H., Eastlake, D. E., Litkowski, S., and S. Zhuang, "BGP Dissemination of L2 Flow Specification Rules", Work in Progress, Internet-Draft, draft-ietf-idr-flowspec-l2vpn-27, 16 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-flowspec-l2vpn-27>.

[I-D.ietf-idr-flowspec-nvo3] Eastlake, D. E., Weiguo, H., Zhuang, S., Li, Z., and R. Gu, "BGP Dissemination of Flow Specification Rules for Tunneled Traffic", Work in Progress, Internet-Draft, draft-ietf-idr-flowspec-nvo3-23, 5 December 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-flowspec-nvo3-23>.

[I-D.ietf-idr-flowspec-srv6] Li, Z., Chen, H., Loibl, C., Mishra, G. S., Fan, Y., Zhu, Y., Liu, L., Liu, X., and S. Zhuang, "BGP Flow Specification for SRv6", Work in Progress, Internet-Draft, draft-ietf-idr-flowspec-srv6-08, 24 November 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-flowspec-srv6-08>.

[I-D.ietf-idr-wide-bgp-communities]
Raszuk, R., Haas, J., Lange, A., Decraene, B., Amante, S., and P. Jakma, "BGP Community Container Attribute", Work in Progress, Internet-Draft, draft-ietf-idr-wide-bgp-communities-12, 17 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-wide-bgp-communities-12>.

[interface-set]
Litkowski, S., Simpson, A., Patel, K., and J. Haas, "Applying BGP flowspec rules on a specific interface-set", Work in Progress, Internet-Draft, draft-ietf-idr-flowspec-interfaceset-06, 2 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-flowspec-interfaceset-06>.

[path-redirect]
Van de Velde, G., Patel, K., and Z. Li, "Flowspec Indirection-id Redirect", Work in Progress, Internet-Draft, draft-ietf-idr-flowspec-path-redirect-13, 22 April 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-flowspec-path-redirect-13>.

[redirect-ip]
Haas, J., Henderickx, W., and A. Simpson, "BGP Flow-Spec Redirect-to-IP Action", Work in Progress, Internet-Draft, draft-ietf-idr-flowspec-redirect-ip-10, 28 April 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-flowspec-redirect-ip-10>.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <https://www.rfc-editor.org/info/rfc768>.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <https://www.rfc-editor.org/info/rfc791>.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/info/rfc792>.

[RFC0793] Postel, J., "Transmission Control Protocol", RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[RFC1997] Chandra, R., Traina, P., and T. Li, "BGP Communities Attribute", RFC 1997, DOI 10.17487/RFC1997, August 1996, <https://www.rfc-editor.org/info/rfc1997>.

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

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <https://www.rfc-editor.org/info/rfc2474>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/info/rfc4271>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

[RFC4360] Sangli, S., Tappan, D., and Y. Rekhter, "BGP Extended Communities Attribute", RFC 4360, DOI 10.17487/RFC4360, February 2006, <https://www.rfc-editor.org/info/rfc4360>.

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

[RFC4456] Bates, T., Chen, E., and R. Chandra, "BGP Route Reflection: An Alternative to Full Mesh Internal BGP (IBGP)", RFC 4456, DOI 10.17487/RFC4456, April 2006, <https://www.rfc-editor.org/info/rfc4456>.

[RFC4760] Bates, T., Chandra, R., Katz, D., and Y. Rekhter, "Multiprotocol Extensions for BGP-4", RFC 4760, DOI 10.17487/RFC4760, January 2007, <https://www.rfc-editor.org/info/rfc4760>.

[RFC5065] Traina, P., McPherson, D., and J. Scudder, "Autonomous System Confederations for BGP", RFC 5065, DOI 10.17487/RFC5065, August 2007, <https://www.rfc-editor.org/info/rfc5065>.

[RFC5701] Rekhter, Y., "IPv6 Address Specific BGP Extended Community Attribute", RFC 5701, DOI 10.17487/RFC5701, November 2009, <https://www.rfc-editor.org/info/rfc5701>.

[RFC7153] Rosen, E. and Y. Rekhter, "IANA Registries for BGP Extended Communities", RFC 7153, DOI 10.17487/RFC7153, March 2014, <https://www.rfc-editor.org/info/rfc7153>.

[RFC7606] Chen, E., Ed., Scudder, J., Ed., Mohapatra, P., and K. Patel, "Revised Error Handling for BGP UPDATE Messages", RFC 7606, DOI 10.17487/RFC7606, August 2015, <https://www.rfc-editor.org/info/rfc7606>.

[RFC7665] Halpern, J., Ed. and C. Pignataro, Ed., "Service Function Chaining (SFC) Architecture", RFC 7665, DOI 10.17487/RFC7665, October 2015, <https://www.rfc-editor.org/info/rfc7665>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[RFC9015] Farrel, A., Drake, J., Rosen, E., Uttaro, J., and L. Jalil, "BGP Control Plane for the Network Service Header in Service Function Chaining", RFC 9015, DOI 10.17487/RFC9015, June 2021, <https://www.rfc-editor.org/info/rfc9015>.

[RFC9117] Uttaro, J., Alcaide, J., Filsfils, C., Smith, D., and P. Mohapatra, "Revised Validation Procedure for BGP Flow Specifications", RFC 9117, DOI 10.17487/RFC9117, August 2021, <https://www.rfc-editor.org/info/rfc9117>.

[RFC9582] Snijders, J., Maddison, B., Lepinski, M., Kong, D., and S. Kent, "A Profile for Route Origin Authorizations (ROAs)", RFC 9582, DOI 10.17487/RFC9582, May 2024, <https://www.rfc-editor.org/info/rfc9582>.

### 11.2. Informative References

[fsv2] Hares, S., Eastlake, D. E., Yadlapalli, C., and S. Maduschke, "BGP Flow Specification Version 2", Work in Progress, Internet-Draft, draft-ietf-idr-flowspec-v2-04, 28 April 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-idr-flowspec-v2-04>.

[fsv2-more-ip-filters] Hares, S., "BGP Flow Specification Version 2 - More IP Actions", Work in Progress, Internet-Draft, draft-hares-idr-fsv2-more-ip-actions-03, 17 October 2024, <https://datatracker.ietf.org/doc/html/draft-hares-idr-fsv2-more-ip-actions-03>.

[I-D.hares-idr-fsv2-more-ip-filters] Hares, S. and N. Kao, "BGP Flow Specification Version 2 - More IP Filters", Work in Progress, Internet-Draft, draft-hares-idr-fsv2-more-ip-filters-05, 17 March 2026, <https://datatracker.ietf.org/doc/html/draft-hares-idr-fsv2-more-ip-filters-05>.

[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, <https://www.rfc-editor.org/info/rfc6241>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8205] Lepinski, M., Ed. and K. Sriram, Ed., "BGPsec Protocol Specification", RFC 8205, DOI 10.17487/RFC8205, September 2017, <https://www.rfc-editor.org/info/rfc8205>.

## Authors' Addresses

Susan Hares
Hickory Hill Consulting
7453 Hickory Hill
Saline, MI 48176
United States of America
Phone: +1-734-604-0332
Email: shares@ndzh.com

Donald Eastlake
Independent
2386 Panoramic Circle
Apopka, FL 32703
United States of America
Phone: +1-508-333-2270
Email: d3e3e3@gmail.com

Jie Dong
Huawei Technologies
No. 156 Beiqing Road
Beijing
China
Email: jie.dong@huawei.com

Chaitanya Yadlapalli
ATT
United States of America
Email: cy098d@att.com

Sven Maduschke
Verizon
Germany
Email: sven.maduschke@de.verizon.com

Jeffrey Haas
HPE
United States of America
Email: jeffrey.haas@hpe.com