IEEE P802.1Qdw/D0.0
October 6, 2022
(Amendment to IEEE Std 802.1Q-2022 as amended by
IEEE Std Qcz-2022, IEEE Std 802.1Qcw-2023, IEEE Std 802.1Qcj-2023, and IEEE Std 802.1Qdt-2023)

Draft Standard for Local and metropolitan area networks—

Bridges and Bridged Networks

Amendment: Source Flow Control

Developed by the

LAN/MAN Standards Committee
of the
IEEE Computer Society

Individual contribution

Prepared by the Time-Sensitive Task Group of IEEE 802.1

Abstract: This amendment specifies procedures and managed objects for the signaling and remote invocation of flow control at the source of transmission in a data center network

Keywords: Bridged Network, IEEE 802.1Q™, LAN, local area network, MAC Bridge, metropolitan area network, flow control, congestion management, Priority-based flow control

T

Participants

<<The following lists will be updated in the usual way prior to publication>>

At the time this standard was submitted to the IEEE-SA Standards Board for approval, the IEEE 802.1 Working Group had the following membership:

Glenn Parsons, *Chair*
Jessy V. Rouyer, *Vice Chair*
Paul Congdon, *Editor*
Jeremias Blendin, *Editor*

<<TBA>>

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

<<TBA>>

When the IEEE-SA Standards Board approved this standard on XX Month 20xx, it had the following membership:

<<TBA>>

<<TBA>>

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

<<TBA>>

Introduction

This introduction is not part of IEEE Std 802.1Qdw™, Draft Standard for Local and metropolitan area networks—Bridges and Bridged Networks—Amendment: Source Flow Control

Contents

Figures

Tables

IEEE Standard for
Local and metropolitan area networks—

Bridges and Bridged Networks— Amendment: Source Flow Control

[This amendment is based on IEEE Std 802.1Q™-2022 as amended by IEEE Std Qcz-2022, IEEE Std 802.1Qcw-2023, IEEE Std 802.1Qcj-2023, and IEEE Std 802.1Qdt-2023.]

NOTE—The editing instructions contained in this amendment define how to merge the material contained therein into the existing base standard and its amendments to form the comprehensive standard.

The editing instructions are shown in ***bold italics***. Four editing instructions are used: change, delete, insert, and replace. ***Change*** is used to make corrections in existing text or tables. The editing instruction specifies the location of the change and describes what is being changed by using ~~strikethrough~~ (to remove old material) and underscore (to add new material). ***Delete*** removes existing material. ***Insert*** adds new material without disturbing the existing material. Deletions and insertions may require renumbering. If so, renumbering instructions are given in the editing instruction. ***Replace*** is used to make changes in figures or equations by removing the existing figure or equation and replacing it with a new one. Editing instructions, change markings, and this note will not be carried over into future editions because the changes will be incorporated into the base standard.

1. Overview

.

1.3 Introduction

Insert the following text at the end of 1.3 and renumber accordingly:

This standard specifies protocols, procedures and management objects that support flow control of congesting data flows within data center environments. This is achieved by enabling systems to signal congestion and its expected duration directly to the sender end-station of congesting data flow. Source Flow Control applies flow control to congesting flows of higher layer protocols at the sender end-station by providing sufficient information to pause individual congesting end-to-end flows. It reduces head-of-line blocking in the network compared to hop-by-hop flow control mechanism while it helps to reduce queue build-up and packet loss in the network. A method for PFC-based signaling to pause PFC priorities instead of end-to-end flows is provided for backwards compatibility with PFC-only enabled end stations.

To this purpose Source Flow Control does:

a) Defines a means for bridges to signal congestion directly to the sender of a specific traffic flow contributing to the congestion.
b) Specifies for senders of traffic flows to react to congestion signals.
c) Defines a means for bridges connected to senders of traffic flows to react to congestion signals and convert them to PFC signals if the sender is not capable of processing the congestion signals.

3. Definitions

Insert the following definitions in the appropriate collating sequence, re-numbering as appropriate:

3.1 Source Flow Control Signaling System: An End Station or Bridge Component conforming to the Source Flow Control signaling provisions of this standard.

3.2 Source Flow Control Reception Station: An End Station or Bridge Component conforming to the Source Flow Control signal processing provision of this standard.

3.3 Source Flow Control Message (SFCM): A message transmitted by a Source Flow Control Aware System, conveying congesting flow information used to flow control a congesting flow.

4. Abbreviations

Insert the following abbreviations in the appropriate sequence, re-ordering as appropriate:

SFC	Source Flow Control
SFCM	Source Flow Control Message

5. Conformance

5.4.1 VLAN Bridge component options

Insert the following at the end of the lettered list in 5.4.1, renumbering accordingly:

a) Support Source Flow Control (SFC) (5.4.9)

Insert the following sub-clause at the end of 5.4, renumbering as appropriate:

5.4.9 Source Flow Control (SFC) operation (optional)

A VLAN Bridge implementation that conforms to the provisions of this standard for Source Flow Control (Clause 52) shall:

a) <<TBD>>

A VLAN Bridge implementation that conforms to the provisions of this standard for Source Flow Control (Clause 52) may:

a) <<TBD>>

Insert the following at the end of Clause 5, renumbering as appropriate:

5.34 End station requirements—SFC

An end station implementation that conforms to the provisions of this standard for SFC shall

a) <<TBD>>

An end station implementation that conforms to the provisions of this standard for SFC may

a) <<TBD>>

6. Support of the MAC Service

6.10.1 Data indications

Change the first paragraph of Clause 6.10.1 as follows:

On receipt of an M_UNITDATA.indication primitive from the PIP-ISS, if the PIP is congestion aware (5.4.1.4) and the initial octets of the mac_service_data_unit contain a valid CNM encapsulation, the received frame is processed according to 32.16. If the PIP is congestion isolation aware (5.4.7) and the initial octets of the mac_service_data_unit contain a valid CIM encapsulation (49.4.3), the received frame is processed according to 49.4.2.6. If the PIP is Source Flow Control aware (5.4.9) and the initial octets of the mac_service_data_unit contain a valid SFCM encapsulation (<<52.?.?>>), the received frame is processed according to <<52.?.?>>. Otherwise, the received frame shall be discarded if

12. Bridge management

Insert a new clause 12.36 at the end of Clause 12, renumbering as appropriate

12.36 Source Flow Control managed objects

Clause 52 defines Source Flow Control (SFC). The following managed objects control the operation of SFC:

a) SFC entity managed object (12.36.1)
b) SFC per-port object (12.36.2)
c) SFC per-port per-queue table (12.36.3)

12.36.1 SFC entity managed object

There is one SFC entity managed object per bridge or end station.

Table 12-1—SFC entity table

Name	Data type	Operations supported[a]	Conformance[b]	References
sfcMasterEnable	Boolean	RW	BE	<<TBD>>.
sfcSFCMTransmitPriority	unsigned integer [0..7]	RW	BE	<<TBD>>
sfcAddressIPv4	IPv4 address	RW	IETF RFC 791	<<TBD>>
sfcAddressIPv6	IPv6 address	RW	IETF RFC 8200	<<TBD>>
sfcUDPPort	UDP port number	RW	IETF RFC 798	<<TBD>>
sfcHeaderTruncationLength	integer[0..512]	RW	BE	<<TBD>>

[a] R = Read only access; RW = Read/Write access

[b] B = Required for Bridge or Bridge component support of source flow control; E = Required for end station support of source flow control.

12.36.2 SFC per-port object

There is one SFC per-port managed object for each bridge or end-station port

Table 12-2—SFC per-port table

Name	Data type	Operations supported[a]	Conformance[b]	References
sfcMonitorQueues	8 bit mask	RW	BE	<<TBD>>

[a] R = Read only access; RW = Read/Write access

[b] B = Required for Bridge or Bridge component support of source flow control; E = Required for end station support of source flow control.

12.36.3 SFC per-port per-queue object

There is a maximum one SFC per-port per-queue object for every traffic class of every bridge or end station port.

Table 12-3—SFC per-port per-queue table

Name	Data type	Operations supported[a]	Conformance[b]	References
<<TBD>>	<<TBD>>	RW	BE	<<TBD>>

[a] R = Read only access; RW = Read/Write access

[b] B = Required for Bridge or Bridge component support of source flow control; E = Required for end station support of source flow control.

Insert a new Clause "52. Source Flow Control" as follows:

52. Source Flow Control

Source Flow Control (SFC) pauses congesting flows at the source by reacting to a Source Flow Control Message (SFCM) sent across a data center network from a port with a congesting traffic class that is enabled with the feature. The SFCM identifies a congesting flow at the source end station and provides a pause interval for which transmissions of frames for the flow are to be paused. The receiver of the SFCM asserts flow control on the traffic class used to transmit the flow or may provide more advanced implementation specific per-flow control if implemented. The externally visible behavior of SFC is that the bridge or end-station processing a received SFCM will pause the identified flow for the specified interval.

The models of operation in this clause provide a basis for specifying the externally observable behavior of SFC and are not intended to place additional constraints on implementations; these can adopt any internal model of operation compatible with the externally observable behavior specified.

This clause introduces the concepts and protocols essential to source flow control as follows:

a) The objectives for source flow control (xx).
b) Principles of source flow control (xx).
c) Source flow control aware forwarding process (xx).
d) Source flow control protocol (xx).

<< elaborate more on the environment for SFC - perhaps referencing CI environments >>

.

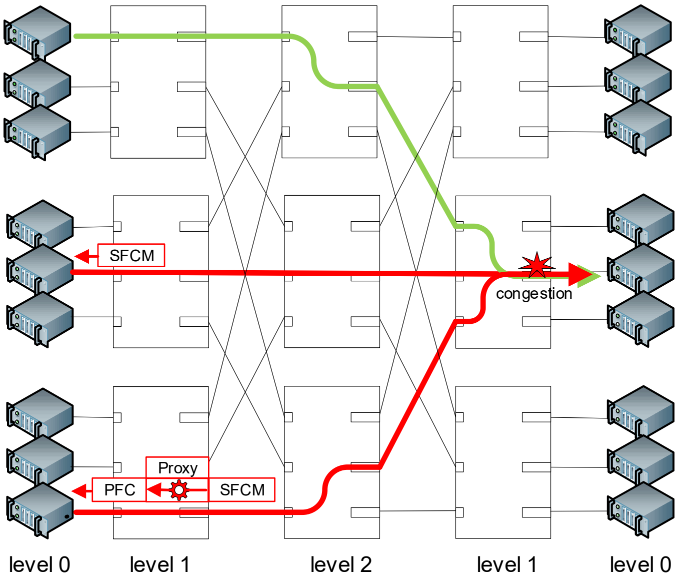

Figure 52-1—Source flow control example operation

Figure 52-1 shows an example operation of SFC. In the figure, all of the relay systems are layer-3 routers and may be SFC aware. It is most important for the edge systems are SFC aware, however when all systems are SFC aware different reasons for congestion may be detected. In the figure, server to server traffic is flowing from left to right across a data center network. When congestion is detected SFC attempts to identify the congesting flows (52.2.1). For each identified congesting flow (e.g. the red flows in Figure 52-

1), the SFC aware system sends a Source Flow Control Message (SFCM) to the sending end station. If the end station is SFC aware, the SFCM is forwarded through the edge system directly to the end station (as seen in the middle of the diagram). If the end station is not SFC aware, the edge system should intercept the SFCM and convert the message to a traditional PFC message (as seen in the lower part of the diagram). The edge system converting the SFCM to a PFC message is known as SFC proxy mode and should be implemented by the edge systems directly attached to end stations. The SFCM contains flow description information necessary for the source end station or the proxy edge system to identify the traffic class of the congesting flow. The flow information may also be used to identify and invoke per-flow implementation specific traffic controls.

52.1 Source flow control objectives

The operation, procedures and protocols of source flow control are designed to meet the following objectives by category:

Functionality

a) Provide a means for low-latency flow control signaling for individual, congesting end-to-end flows across a network.
b) Reduce queue build up in congesting queues with minimal side effects on other flows in the network.
c) Move queueing and congestion from in the network to the edge, into the end stations.
d) Reduce the likelihood of frame loss.
e) Avoid the triggering of PFC, not replace PFC.
f) Invoke flow control on frames of congesting flows before the frames experience congestion in the queue management system.
g) Do not keep per-flow state in the congested device other than a probabilistic data structure to limit the messaging overhead per flow.
h) Specify the signaling of the estimated duration of the congestion.
i) Do only specify the minimum reaction to a source flow control message.

Compatibility

j) Be oblivious of protocols above the network layer. Do not be specific to higher layer protocols. The identification of the end-to-end protocol flows is left to the higher layers based on the first bytes of the original PDU.
k) Work in conjunction with higher-layer end-to-end congestion control protocols and features, such as ECN, RoCE, DCQCN, Swift.
l) Optionally provide means to specify locators of the congested queue enabling receiving systems. Systems, including intermediate systems, can use the congestion locator to decide if other passing flows are expected to contribute to the congested queue and to signal flow control preemptively.
m) Work in existing lossless environments using PFC without requiring additional traffic classes.
n) Work in layer-2 and layer-3 networks.

Performance

o) Reduce queueing in the network and thereby the flow completion time across the network.
p) SFC messages should be as small as possible to ensure low-latency forwarding as well as low signaling overhead.
q) Reduce head-of-line blocking when using in an environment with PFC enabled.

Scale

r) Work in arbitrary data center network topologies with a mix of link speeds.
s) Limit messaging overhead by restricting the number of congestion messages per flow to a fixed number per RTT.

t) Limit the messaging overhead by signaling the expected duration of the congestion without causing underutilization.

Implementation complexity

u) The main goal of SFC is implementational simplicity.

v) Require changes only on congested switches as well as sender NICs, or optional the last hop switch at sender side. No modifications on intermediate switches are required.

Manageability

w) Limit the ability to configure an inoperable environment.

x) Provide auto discovery of SFC message processing capability between last hop switches and NICs using existing LLDP messages and without creating additional hello and auto-configuration protocols.

52.2 Principles of source flow control

This clause introduces the principles of source flow control. Items a) through d) describe the life of a congesting flow from identification through pausing of the flow. Items e) through g) compare and contrast Source Flow Control with Priority-based Flow Control (Clause 36), Congestion Notification (Clause 30) and Congestion Isolation (Clause 49).

The following items describe the principals of source flow control:

a) Congesting flow identification (52.2.1).

b) Source flow control signaling (xx).

c) End station SFCM reception (xx).

d) Proxy SFCM reception (xx).

e) Comparison to Priority-based Flow Control (xx)

f) Comparison to Congestion Notification (xx).

g) Comparison to Congestion Isolation (xx).

52.2.1 Congesting flow identification

<< reference text from Congestion Isolation and/or Congestion Notification>>

An essential step in the process of SFC is identifying congesting flows by an Active Queue Management (AQM) scheme that supports Explicit Congestion Notification (ECN) specified in IETF RFC 3168. There are many potential methods of identifying congesting flows and interoperable implementations can exist using different approaches. The SFC Congestion Detection function (52.3.1) of the Source Flow Control Aware Forwarding Process (52.3) is responsible for implementing the AQM. This standard defines the CP algorithm (30.2.1) for detecting Congestion Controlled Flows (CCFs) in congestion aware bridges. This approach may be used to detect congesting flows in a SFC aware system. A number of other possible approaches, including those that support the end-to-end ECN congestion control, are discussed in IETF RFC 7567[B1].

Many modern data centers utilize encapsulated overlay networks, such as those described in IETF RFC 8014[B6]. An overlay network can carry multiple encapsulated flows within a single encapsulation flow. The congesting and non-congesting flows identified by SFC are the outer encapsulation flow as seen by the underlay network. The inner encapsulated flows might not be visible to the bridges and routers within the data center network, and are therefore not separated into congesting and non-congesting flows.

52.2.2 Source flow control signaling

Once a frame has been identified as being part of a congesting flow, an SFCM is created and sent to the origin of the congesting flow. The SFCM message carries the pause duration estimate as well enough information to identify the congesting flow. The SFCM PDU is encapsulated in a UDP packet. The source address and UDP port number for the UDP packet are taken from SFC configuration, and the destination address is the source address of the congesting frame.

The congesting flow is identified at the source by examining the contents of the SFCM which contains a part of the congesting frame. By default, an SFCM contains the first 48 bytes of the congesting frame's MSDU. Configuration allows the SFCM to include up to 256 octets of the congesting frame's MSDU allowing systems that support overlay networks the ability to identify encapsulated flows within the congesting flow encapsulation.

The SFCM may contain optional congestion locators that may be used by systems to preemptively trigger flow control on passing flows that are expected to contribute to the congestion point specified by the congestion locator.

The system sending SFC frames should be configurable to limit the SFCM sending rate to a specific receiver.

52.2.3 End station SFCM reception

SFCM aware end stations receive SFCM messages and parse them to identify the congesting flow. The end station prohibits subsequent frames of the specified congesting flow from transmission for the specified amount of time within <<XX>> ns of receiving the message. The end station prohibits transmission of subsequent frames by pausing the traffic class associated with the congesting flow or alternatively, invoking an implementation specific mechanism that pauses the transmission of individual frames associated with the congesting flow.

52.2.4 Proxy SFCM reception

SFC may be extended to end stations that are not SFC aware, but are PFC enabled by an SFC proxy function in the directly attached bridge or router. The SFC proxy function enabled on the port attached to the SFC unaware end station is responsible for converting the SFCM addressed to the end station to a PFC frame that pauses the appropriate traffic class. To perform this operation, the SFC proxy function must identify the source port and traffic class of the congesting flow by examining the contents of the SFCM. The pause time indicated in the SFCM is converted to the PFC pause time encoding and a PFC request is triggered.

52.2.5 Comparison to Priority-based Flow Control

PFC aims at eliminating frame loss. SFC does not have that goal. Instead, SFC aims at controlling the queue buildup during congestion events. Limiting queue build up reduces the likelihood of packet drops because of limited buffer space, it does not eliminate it though. For lossless network behavior, SFC should be used together with PFC enabled on the same traffic classes in the network.

In contrast to PFC, which uses hop-by-hop propagation of the flow control signal, SFC signals flow control directly to senders of congesting flows. Thereby, operational side effects of flow control that exist in hop-by-hop designs, such as PFC's head-of-line blocking are eliminated. The queue build up is moved to the sender end station, minimizing the in-network side effects of flow control.

52.2.6 Comparison to Congestion Notification

<<TBD>>

52.2.7 Comparison to Congestion Isolation

<<TBD>>

52.3 Bridge Component Source Flow Control Aware Forwarding Process

This clause specifies the architecture of the Source Flow Control Point (SFCP) in the Forwarding Process of a source flow control aware Bridge. In this architecture, a router is as a higher layer entity that relays frames using layer-3 information but uses the forwarding process of the underlying source flow control aware bridge to deliver frames to peers and end stations.

<<NOTE: the paragraph below has already been said once above - which location is better? The editors believe above is better since we are specifying both end station and bridge component behavior>>

The models of operation in this clause provide a basis for specifying the externally observable behavior of SFC, and are not intended to place additional constraints on implementations; these can adopt any internal model of operation compatible with the externally observable behavior specified. Conformance of equipment to this standard is purely in respect of observable protocol.

Figure 8-12 illustrates the Bridge Forwarding Process at its highest conceptual level. Figure 8-16 shows the specific filtering and assignment functions of the flow classification and metering elements in the Forwarding Process of a source flow control aware Bridge. Figure 52-2 focuses on the operation of a single Bridge Port and the relationship of new elements to the queuing and classification functions. Four new elements and two new managed tables are specified for a SFC aware Bridge as follows:

a) SFC Congestion Detection (52.3.1).
b) SFCM Multiplexer (52.3.2).
c) SFCM Demultiplexer (52.3.3).
d) Source Port Locator (52.3.4).

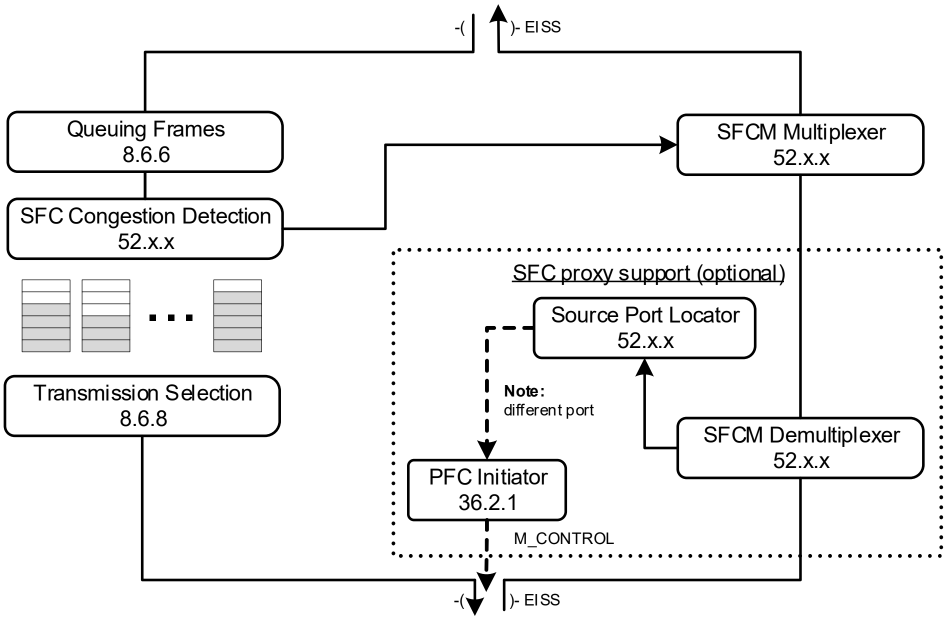

Figure 52-2—Bridge component SFC reference diagram

52.3.1 SFC Congestion Detection

<< Reference or leverage text from Congestion Notification and/or Congestion Isolation>>

SFC Congestion Detection identifies frames of a congesting flow, inserts frames passed by the Queuing Frames entity (8.6.6) into the appropriate queue or discards them and can generate an SFCM based on the contents of the frame. As described in 52.2.1, frames of a congesting flow are identified by a suitable AQM implemented by SFC Congestion Detection, such as the algorithm specified by a CP in 30.2.1.

Frames given to SFC Congestion Detection by the Queuing Frames entity (8.6.6) in an EM_UNITDATA.request (52.5.2.2) may be identified by the AQM as being part of a congesting flow. An SFCM may be generated from the parameters obtained with the received frame. The SFCM is sent to the source of the congestion flow via the SFC multiplexer.

On each EM_UNITDATA.request, SFC Congestion Detection and its associated AQM indicate to the SFC Procedures (52.4) whether a monitored queue is congested or not congested. The AQM or other means can be used to provide the queue status.

52.3.2 SFCM Multiplexer

The SFCM multiplexer inserts SFCMs generated by SFC Congestion Detection among frames received from the LAN. Layer-3 encapsulated SFCMs (xx and xx) are routed to the source by a higher-layer routing function that is beyond the scope of this standard.

52.3.3 SFCM Demultiplexer

The SFCM demultiplexer is part of the optional SFCM proxy mode support. It identifies SFCMs received from the LAN and extracts the flow information from the content of the SFCM PDU (xx) to pass to the source port locator function (xx). An SFCM PDU may be encapsulated by two different SFCM encapsulations; IPv4 and IPv6. Implementations supporting IPv4 and IPv6 encapsulations must be able to identify and validate IPv4 and/or IPv6 packets in the SFCM demultiplexer. The rules for validating received SFCMs are specified in <<xx>>.

52.3.4 Source port locator

<<TBD - responsible for finding the source port for a given flow. The proxy function will assert PFC on that source port. The port is located using layer-3 routing tables or implementation specific structures that identify how to reach a particular IP address.>>

52.4 End Station Source Flow Control Aware Forwarding Process

This clause specifies the architecture of the Source Flow Control Point (SFCP) in the Forwarding Process of a source flow control aware End station.

Figure 52-3 focuses on the operation of a single End station port and the relationship of new elements to the queuing and classification functions. Five new elements and two new managed tables are specified for a SFC aware Bridge as follows:

a) End Station SFCM Multiplexer (52.4.1).
b) End Station SFCM Demultiplexer (52.4.2).
c) End Station SFC Congestion Detection (52.4.3).

Figure 52-3 also depicts a number of functions that are out of the scope of this standard and may be optionally implemented by an end station. These functions are described as follows:

d) End Station Tx Flow Management (52.4.4).
e) End Station Rx Flow Management (52.4.5).
f) Reception Selection (52.4.6).

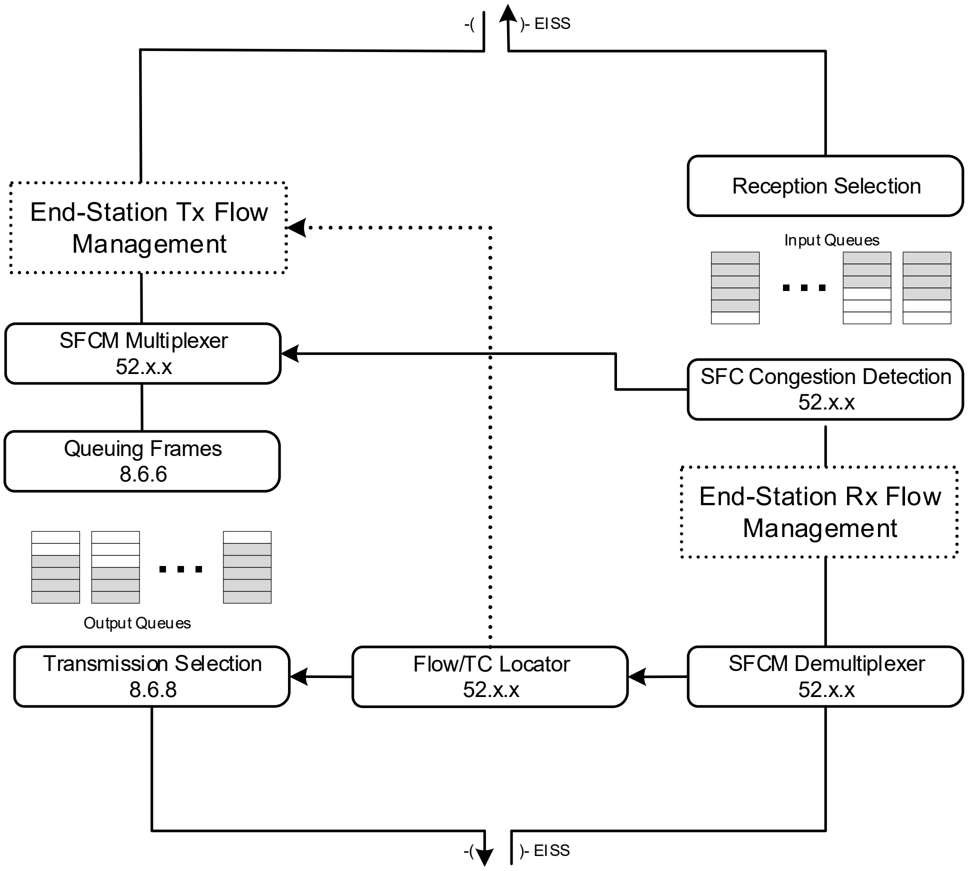
-(
)- EISS
End-Station Tx Flow Management
Reception Selection
Input Queues
SFCM Multiplexer 52.x.x
SFC Congestion Detection 52.x.x
Queuing Frames 8.6.6
End-Station Rx Flow Management
Output Queues
Transmission Selection 8.6.8
Flow/TC Locator 52.x.x
SFCM Demultiplexer 52.x.x
-(
)- EISS

Figure 52-3—End station SFC reference diagram

52.4.1 End Station SFCM Multiplexer

<<TBD>>

52.4.2 End Station SFCM Demultiplexer

<<TBD>>

52.4.3 End Station SFC Congestion Detection

<<TBD>>

52.4.4 End Station Tx Flow Management

<<TBD>>

52.4.5 End Station Rx Flow Management

<<TBD>>

52.4.6 Reception Selection

<<TBD>>

52.5 Source Flow Control Protocol

Source flow control aware systems control forwarding elements, participate in SFC protocols and act upon the LLDP Source Flow Control TLV as specified in this clause. This includes:

a) Variables controlling operation (52.5.1).
b) SFCP procedures (52.5.2).
c) Encoding of the SFCM PDU and SFCM encapsulations (52.5.3).
d) LLDP Source Flow Control TLV (52.5.4).

52.5.1 Variables controlling operation

The source flow control variables control the operation of the SFC entity and the SFCP entity.

52.5.1.1 SFC entity variables

Every source flow control aware system has a set of SFC entity variables to control the overall operation of SFC. These variables are included in the SFC entity managed object (12.xx). These include the following:

a) sfcMasterEnable (52.5.1.1.1).
b) sfcSFCMTransmitPriority (52.5.1.1.2).
c) sfcAddressIPv4 (52.5.1.1.3).
d) sfcAddressIPv6 (52.5.1.1.4).
e) sfcUDPPort (52.5.1.1.5).
f) sfcHeaderTruncationLength (52.5.1.1.6).

52.5.1.1.1 sfcMasterEnable

A boolean value specifying whether SFC is enabled in this system. If sfcMasterEnable is FALSE all source flow control functionality is disabled; SFCMs and LLDP Source Flow Control TLVs are not generated and are ignored on receipt. If sfcMasterEnable is TRUE the other managed objects and variables specified in the clause control the operation of SFC.

52.5.1.1.2 sfcSFCMTransmitPriority

An integer specifying the priority value to be used when transmitting SFCMs from the system. The default is 6.

52.5.1.1.3 sfcAddressIPv4

The IPv4 address, belonging to the system transmitting the IPv4 SFCM (xx), used as the IPv4 source address in the IPv4 header (IETF RFC 791) of IPv4 SFCMs sent from the SFCP.

52.5.1.1.4 sfcAddressIPv6

The IPv6 address, belonging to the system transmitting the IPv6 SFCM (xx), used as the IPv6 source address in the IPv6 header (IETF RFC 8200) of IPv6 SFCMs sent from the SFCP.

52.5.1.1.5 sfcUDPPort

<<It is desirable to have a well-known UDP number allocated by IANA rather than using a locally administrated value as described in the text below. The same value MUST be used by all systems in the data center network and will require configuration if a well known number is not allocated>>

The destination UDP port number in the UDP header (IETF RFC 768) of IPv4 and IPv6 SFCMs sent by the SFC aware system. The UDP port number must be selected from the range of dynamic port numbers, between 49152 and 65535, as specified in IETF RFC 6335. The port number must be currently available for use by the implementation. For example, an implementation may use UDP port 58623, if it is not currently being used by any other application in the system.

52.5.1.1.6 sfcHeaderTruncationLength

The minimum number of octets that the SFCP is to return in the Encapsulated MSDU field of each SFCM generated. The default value is 48. The sfcHeaderTruncationLength should be configured consistently across all systems in the data center network.

52.5.1.2 SFCP entity variables

<<These are per bridge component configuration values - none identified yet. NOTE: do we want the same set of traffic classes to have SFC enabled on every port? If so, we can define the TC enabled bit-mask here.>>

52.5.1.3 SFCP entity per-port variables

<<These are per bridge component, per-port configuration values>>

52.5.1.3.1 sfcMonitorQueues

An 8-bit value specifying which traffic classes on the port will be monitored for congestion and potentially cause the generation of SFCM messages back to the source.

52.5.1.4 SFCP entity per-port per-traffic class variables

For each port and monitored queue in a source flow control aware system there is the following set of variables:

g) sfcCongesting (52.5.1.4.1)
h) <<TBD>>

52.5.1.4.1 sfcCongesting

A boolean value that is set during the processing of a frame by the EM_UNITDATA.request (xx) procedure. The value is set true when the AQM indicates that the monitored queue is congested at the time the frame is processed. The variable is initialized to false.

52.5.1.5 SFCP entity per-stream variables

<<TBD - these variables drive the logic and likely do NOT need to be per-port, per-TC, per-stream since SFC does not require state>>

52.5.2 SFCP Procedures

Source flow control is implemented through the procedures of a SFCP. These include the following:

a) sfcInitialize() (xx)
b) EM_UNITDATA.request (parameters) (xx)
c) condTransmitSfcmPdu() (xx)
d) pauseTimeCalc() (xx)
e) <<TBD>>

52.5.2.1 ciInitialize()

Initializes SFC support. At initialization time the procedure performs the following:

a) <<TBD>>

52.5.2.2 EM_UNITDATA.request (parameters)

A SFCP offers an instance of the EISS (6.8) to the Queuing frames function (8.6.6). When called upon to enqueue a frame the priority parameter specifies the target queue which represents the received priority of the frame. The SFCP determines if the target queue is a monitored queue by checking if the priority parameter has a corresponding bit set for the traffic classes in the sfcMonitorQueues variable.

If the corresponding bit in the sfcMonitorQueues variable has the value is 0 then the target queue is not participating in source flow control and the frame can be enqueued on the target queue with no further processing. If the corresponding bit in the sfcMonitorQueues variable has the value is 1then the target queue is a monitored queue.

If the target queue is a monitored queue, the SFCP interacts with the AQM to determine if the frame is part of a flow that is creating congestion in the monitored queue. Frames that have been determined by the AQM (see 52.2.1) to be creating congestion may cause a SFCM to be transmitted. The conditions by which a SFCM is transmitted are described in condTransmitSfcmPdu().

For each frame that is presented for queuing, the SFCP performs the following:

a) <<TBD>>

52.5.2.3 condTransmitSfcmPdu()

Called by the SFCP to conditionally generate and transmit a SFCM. <<TBD - describe throttling mechanism. This function can also build the SFCM message with available parameters>>.

NOTE—Implementations may have additional conditions to restrict or allow the creation of SFCMs for adding congesting flows without risking interoperability.

The procedure performs the following:

a) <<TBD>>

52.5.2.4 pauseTimeCalc()()

<<TBD - determines the pause interval to put into the SFCM>>.

52.5.2.5 processSfcmPdu()

The SFCM Demultiplexer (52.3.3) receives SFCMs from SFC aware systems and invokes processSfcmPdu() to process the SFCM. The procedure performs the following actions upon receipt of a SFCM:

a) <<TBD - this is bacially the heart of the demultiplexer. It parses the SFCM and either calls TC/Flow locator or the source port locator (in proxy mode)>>

52.5.3 Encoding of the SFCM PDU

This clause specifies the method of encoding source flow control message (SFCM) PDUs. There are two ways of encapsulating SFCM PDUs; a IPv4 and IPv6 SFCM PDU <<TBD - how do we know which type of SFCM to send? Must be another configuration option>>. All SFCMs contain an integral number of octets.

The octets in a source flow control message PDU are numbered starting from 1 and increasing in the order they are put into the MSDU that accompanies a request to or indication from the instance of the MAC Internal Sublayer Service (ISS or EISS) used by a source flow control entity. The bits in an octet are numbered from 1 to 8 in order of increasing bit significance, where 1 is the LSB in the octet.

Where octets and bits within a source flow control message PDU are represented using a diagram, octets shown higher on the page than subsequent octets and octets shown to the left of subsequent octets at the same height on the page are lower numbered; bits shown to the left of other bits within the same octet are higher numbered.

Where two or more consecutive octets are represented as hexadecimal values, lower numbered octet(s) are shown to the left and each octet following the first is preceded by a hyphen, e.g., 01-80-C2-00-00-00. When consecutive octets are used to encode a binary number, the lower octet number has the more significant value. When consecutive bits within an octet are used to encode a binary number, the higher bit number has the most significant value. When bits within consecutive octets are used to encode a binary number, the lower octet number composes the more significant bits of the number. A flag is encoded as a single bit, and is set (TRUE) if the bit takes the value 1, and clear (FALSE) otherwise. The remaining bits within the octet can be used to encode other protocol fields.

52.5.3.1 IPv4 SFCM PDU encapsulation

The means of identifying IPv4 encapsulated SFCM PDUs consist of 2 octets containing the EtherType value for IPv4 packets (08-00) as well as the associated IPv4 header decoding for a UDP datagram carrying the SFCM PDU. The encoding of an IPv4 header is defined in IETF RFC 791. IP options are not included in the IPv4 encapsulated SFCM PDU. The IP protocol field in the IPv4 header consists of 1 octet and identifies the UDP datagram with the value 17. The encoding of a UDP header is defined in IETF RFC 768. The source and destination port fields of the UDP header consists of 2 octets and identifies the encapsulated SFCM PDU with the value from the sfcUDPPort variable. The IPv4 encapsulation is shown in Figure 52-4.

52.5.3.2 IPv6 SFCM PDU encapsulation

The means of identifying IPv6 encapsulated SFCM PDUs consist of 2 octets containing the EtherType value for IPv6 packets (86-DD) as well as the associated IPv6 header decoding for a UDP datagram carrying the SFCM PDU. The encoding of an IPv6 header is defined in IETF RFC 8200. IPv6 Extension Headers are not used in the IPv6 encapsulated SFCM PDU. The next header field in the IPv6 headers indicates the upper

Octets:	1	3	23	31 … 30+N
	EtherType (2 octets = 08-00)	IPv4 Header (20 octets)	UDP Header (8 octets)	SFCM PDU (N octets 66 ≤ N ≤ 530)

Figure 52-4—IPv4 SFCM Encapsulation

layer protocol field and consists of 1 octet identifying the UDP datagram with the value 17. The encoding of a UDP header is defined in RFC 768. The source and destination port fields of the UDP header consists of 2 octets and identifies the encapsulated SFCM PDU with the value from the sfcUDPPort variable. The IPv6 encapsulation is shown in Figure 52-5.

Octets:	1	3	43	51 … 50+N
	EtherType (2 octets = 86-DD)	IPv6 Header (40 octets)	UDP Header (8 octets)	SFCM PDU (N octets 66 ≤ N ≤ 530)

Figure 52-5—IPv6 SFCM Encapsulation

52.5.3.3 Source flow control message PDU format

The format of a source flow control message (SFCM) PDU is illustrated in Figure 52-6.

Octets:	1		2	4		5	5+N			7+N	9+N … 57+N+M
	Version (4 bits)	Reserved (4 bits)	Pause duration in µs (2 octets)	Reserved (4 bits)	Option count (4 bits)	Option TLVs (N octets < 80)	priority (3 bits)	DE (1 bit)	VLAN ID (12 bits)	Encapsulated MSDU length (2 octets)	Encapsulated MSDU (M octets 48 ≤ M ≤ 512)
Bits:	4	8		4	8		3	4	8		

Figure 52-6—SFCM PDU

52.5.3.3.1 Version

This field, 4 bits in length, shall be transmitted with the value 0 in this standard. If two Version fields are interpreted as unsigned binary numbers, the greater identifies the more recently defined Version. The Version field occupies the most significant bits of the first octet of the SFCM PDU.

52.5.3.3.2 Pause duration in us

A 16 bit unsigned integer containing the duration in microseconds that SFCM transmitter expects the receiver to pause the transmission of the congesting flow. This field shall never be 0 when sent.

52.5.3.3.3 Option count

This field, 4 bits in length, shall be transmitted as an unsigned integer containing the number of option TLV fields inserted in the Option TLV list. A value of 0 indicates that no option TLVs are included. The maximum value is 15.

52.5.3.3.4 Option TLVs format

The option TLVs field is a list of zero, one or more, up to 15 SFCM option TLVs. The total length of the options TLVs field shall not exceed 80 octets, independent of the number of SFC option TLVs. If a receiver does not support an option TLV, the TLV shall be skipped and ignored.

Figure 52-7 illustrates the structure of an individual option TLV:

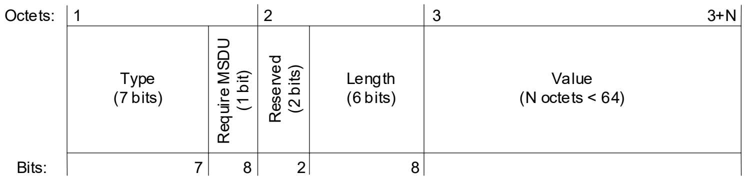

Figure 52-7—Option TLV format

52.5.3.3.4.1 Type

The option type is 7 bits in length and specifies the option type used. Table 52-1 specifies the details of each option TLV type and the relationship to other fields of the TLV.

52.5.3.3.4.2 Requires MSDU

A single bit that indicates if this option requires the encapsulated MSDU to be present. It is 1 to indicate that the MSDU is required, 0 if not. If the MSDU is not required, the sender can choose to not include the MSDU. If the receiver does not support the option and the MSDU is not included, the SFC message is ignored by the receiver.

52.5.3.3.4.3 Length

This field, 6 bits in length, is an unsigned integer providing the length of the option value in octets. The length field can be 0.

52.5.3.3.4.4 Value

The value field contains between 0 and 63 binary octets representing the option value. Each option type has a specific value field definition. When the TLV length field is 0, no value field is included.

52.5.3.3.5 Option TLV definitions

Table 52-1 provides the definition of the option TLVs and the relationship between fields of the TLV. The layout of the value field and the bounds on the TLV length depend upon the option TLV type.

52.5.3.3.5.1 DSCP in MSDU

The DSCP in MSDU option TLV has a type field of 0. The MSDU required field is 1 since the contents of the DSCP and destination IP address are obtained from the MSDU itself. This option instructs the receiver to cache the outer most destination IP address and DSCP value found in the encapsulated MSDU. The length of the option TLV value field is 0 and the total option TLV length is 2 octets. This option requires the MSDU to be present in the SFCM.

Table 52-1—Option TLV definitions

Type	MSDU required	Description	Length (octets)	TLV value layout			
0	1	DSCP in MSDU	0	N/A			
1	0	DSCP / IP Prefix	6 for IPv4 18 for IPv6	DSCP (8-bits)	Address family (1 bit)	Address prefix (7 bits)	IP Address (4 or 16 octets)
2	0	TC / IP Prefix	6 for IPv4 18 for IPv6	TC (8-bits)	Address family (1 bit)	Address prefix (7 bits)	IP Address (4 or 16 octets)
7-126	0						
127		Organizationally specific	3 < n < 64	See Figure 52-8			

52.5.3.3.5.1.1 DSCP/IP prefix

The DSCP/IP prefix option instructs the receiver to cache the prefix provided in the value field of the TLV. The layout of value field is shown in Table 52-1. The layout contains the 8 bit DSCP/TOS value, a bit that determines the address family (i.e. IPv4 or IPv6) and the IP address prefix length.

The next bit indicates the IP address family, with 0 signifying that the following prefix is IPv4 and 1 signifying it being an IPv6 prefix. The prefix length is encoded in a 7 bit field. If the IP address family is IPv4 and the prefix length field value must be between 1 and 32. If the IP address family is IPv6 the prefix length field value must be between 1 and 127.

The remainder of the option field encodes the IP address prefix. The length of the prefix is determined by the length field of the option, after subtracting the first two octets of the value field that are used to encode the DSCP value and the prefix length. The prefix is encoded starting with its most significant bits first. If the total length of the prefix is smaller than the length of the IP address of the used address family, the parts of the address that are not included are assumed to be 0.

52.5.3.3.5.1.2 TC/IP Prefix

This option instructs the receiver to cache the prefix provided in the value. The value layout starts with an 8 bit field to store a list of one-hot encoded TC values. The layout of the rest of the value is the same as the “Cache DSCP and Prefix” option.

52.5.3.3.5.1.3 Organizationally Specific

TLVs of type 127 allow organizationally specific option TLVs. The required MSDU and length fields are the same for other option TLVs. The value field must begin with the organization’s OUI and an 8-bit organizationally managed subtype field. The remainder of the value field is defined by the organization supporting the TLV and must not exceed 60 octets in length.

Figure 52-8 illustrates the structure of an organizationally specific option TLV:

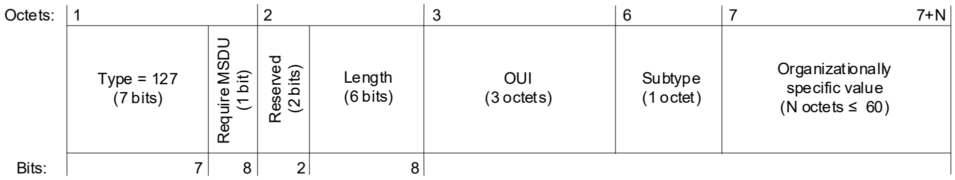

Figure 52-8—Organizationally specific option TLV format

52.5.3.3.6 priority

This field, 3 bits in length, contains the priority parameter of the EM_UNITDATA.request (6.8.1) of the frame that triggered the creation of the SFCM.

52.5.3.3.7 DE

This field, 1bit in length, contains the drop_eligible parameter of the of the EM_UNITDATA.request (6.8.1) for the frame that triggered the creation of the SFCM.

52.5.3.3.8 VLAN ID

This field, 12 bits in length, contains the vlan_identifier parameter of the EM_UNITDATA.request (6.8.1) for the frame that triggered the creation of the SFCM.

52.5.3.3.9 Encapsulated MSDU length

This field, 16 bits in length, shall contain the length of the encapsulated MSDU field. If the length is 0 no encapsulated MSDU is included. If an MSDU is encapsulated, the minimum value is 28 and the maximum value is 512.

52.5.3.3.10 Encapsulated MSDU

This field, a minimum of 28 octets and a maximum of 512 octets in length, contains the mac_server_data_unit parameter of the EM_UNITDATA.request (6.8.1) for the frame that triggered the creation of the SFCM.

52.5.3.4 SFCM Validation

A SFCM PDU received by an SFCM demultiplexer (52.3.3) shall be considered invalid and discarded if any of the following condition are TRUE:

a) An option TLV is present with the requires MSDU field set to 1 and there are fewer than 28 octets in the Encapsulated MSDU or greater than 512.

The following condition shall not cause a received SFCM PDU to be considered invalid:

a) There are nonzero bits in the Version (52.4.3.4.1) field.
b) There are nonzero bits in the reserved fields of the SFCM PDU.

52.5.4 LLDP Source Flow Control TLV

<<TBD>>

52.5.4.1 LLDP Source Flow Control TLV Procedures

<<TBD>>

Annex Z

(temporary not for publication)

Commentary

This is a temporary Annex, a place to record outstanding or recent technical issues and their disposition. It will be removed prior to SA Ballot. Because this is not a part of the proposed standard the editor will not accept comments on the text of this Annex itself, only on the issues raised. Discussion and resolution of the issues will result in modification of the contents.

The order of discussion of issues is intended to help the reader understand first what is the draft, secondly what may be added, and thirdly what has been considered but will not be included. In pursuit of this goal, issues where the proposed disposition is "no change" will be moved to the end. The description of issues is updated to reflect our current understanding[1] of the problem and its solution: where it has been considered useful to retain an original comment, in whole or part, either to ensure that its author does not feel that it has not been sufficiently argued or the editor suspects there may be further aspects to the issue, that has been done as a footnote.

Z.1 VLAN in SFCM

The VLAN header information of the congesting packet is not part of the MSDU. SFCM are sent using the same VLAN information the original, congesting packet has been received on.

However, carrying the VLAN in the SFCM frame (is perhaps) unnecessary since the actual sender of the flow might be on a totally different VLAN. Each router hop along the way will potentially need to modify the VLAN ID, just as it does the DA/SA MAC addresses. The routing table at each hop will be consulted to properly route the packet back to the source, and the routing table relationship to a VLAN is hop-by-hop dependent. Since the original packet was routed to the destination, the SFCM in the reverse direction is assumed to also be routable back to the source. It isn't clear that the VLAN has any relevance.

Resolution: To be resolved

Z.2 SFCM caching security considerations

An SFCM could block a DSCP/TC value across the whole network by using a 0/0 network prefix. A misbehaving congestion point, or perhaps an attacher that is masquerading as the congestion point could send an SFCM that could lead to denial of service.

Resolution: As part of SFCM validation a configurable minimum prefix length could be checked against configuration. The data center is considered a single managed domain and congestion points within that domain will need to have some level of trust. Stations masquerading as congestion points would need to be block or mitigated using any security mechanism that prevents an impostor from sending control frames in the network.

[1] This annex is not intended therefore to be a complete historical record of the development of the draft. The formal record comprises the retained drafts and dispositions of comments.

Z.3 SFCM Authentication

How do we prevent unauthorized senders to inject SFCM into the network. Related to Z.2 above.

Resolution: Assume the switches in the network can be trusted. If NICs are controlled by the network operator, they are to be trusted as well. Use ACLs on ports on the network connecting to non-trusted devices. Use mechanisms that work today