Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: 22 April 2026

R.G. Wilton
Cisco Systems
S. Mansfield
Ericsson

19 October 2025

Common Interface Extension YANG Data Models
draft-ietf-netmod-intf-ext-yang-18

Abstract

This document defines two YANG modules that augment the Interfaces data model defined in the "YANG Data Model for Interface Management" with additional configuration and operational data nodes to support common lower layer interface properties, such as interface MTU.

The YANG modules in this document conform to the Network Management Datastore Architecture (NMDA) defined in RFC 8342.

Table of Contents

1. Introduction 3
1.1. Terminology 4
1.2. Tree Diagrams 4
1.3. Prefixes in Data Node Names 4
2. Interface Extensions YANG Module Introduction 4
2.1. Link Flap Suppression 6
2.2. Dampening 7
2.2.1. Suppress Threshold 8
2.2.2. Half-Life Period 8
2.2.3. Reuse Threshold 8
2.2.4. Maximum Suppress Time 8
2.3. Encapsulation 9
2.4. Loopback 9
2.5. Maximum frame size 9
2.6. Sub-interface 10
2.7. Forwarding Mode 10
2.8. Peer-interface 11
3. Interfaces Ethernet-Like YANG Module Introduction 11
4. Interface Extensions YANG Module 12
5. Interfaces Ethernet-Like YANG Module 24
6. Examples 35
6.1. Link Flap Suppression Configuration 35
6.2. Dampening configuration 36
6.3. MAC address configuration 37
6.4. Peer interface configuration 39
7. Acknowledgements 40
8. IANA Considerations 40
8.1. YANG Module Registrations 40
9. Security Considerations 41
10. References 42
10.1. Normative References 42
10.2. Informative References 43
Authors' Addresses 44

1. Introduction

This document defines two NMDA compatible [RFC8342] YANG 1.1 [RFC7950] modules for the management of network interfaces. It defines various augmentations to the generic interfaces data model [RFC8343] to support configuration of lower layer interface properties that are common across many types of network interface.

One of the aims of this document is to provide a standard definition for these configuration items regardless of the underlying interface type. For example, a definition for configuring or reading the MAC address associated with an interface is provided that can be used for any interface type that uses Ethernet framing.

Several of the augmentations defined here are not backed by any formal standard specification. Instead, they are for features that are commonly implemented in equivalent ways by multiple independent network equipment vendors. The aim of this document is to define common paths and leafs for the configuration of these equivalent features in a uniform way, making it easier for users of the YANG model to access these features in a vendor independent way. Where necessary, a description of the expected behavior is also provided with the aim of ensuring vendors implementations are consistent with the specified behavior.

Given that the modules contain a collection of discrete features with the common theme that they generically apply to interfaces, it is plausible that not all implementers of the YANG module will decide to support all features. Hence, separate feature keywords are defined for each logically discrete feature to allow implementers the flexibility to choose which specific parts of the model they support.

The augmentations are split into two separate YANG modules that each focus on a particular area of functionality. The two YANG modules defined in this document are:

ietf-if-extensions.yang - Defines extensions to the IETF interface data model to support common configuration data nodes.

ietf-if-ethernet-like.yang - Defines a module for any configuration and operational data nodes that are common across interfaces that use Ethernet framing.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "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.

1.2. Tree Diagrams

Tree diagrams used in this document follow the notation defined in [RFC8340].

1.3. Prefixes in Data Node Names

In this document, names of data nodes and other data model objects are prefixed using the standard prefix associated with the corresponding YANG imported modules, as shown in Table 1.

Prefix	YANG Module	Reference
if-ext	ietf-if-extensions	This document
ethlike	ietf-if-ethernet-like	This document
yang	ietf-yang-types	[RFC6991]
if	ietf-interfaces	[RFC8343]
ianaift	iana-if-type	[RFC7224]

Table 1: Prefixes for imported YANG modules

2. Interface Extensions YANG Module Introduction

The Interfaces Extensions YANG module provides some basic extensions to the IETF interfaces YANG module.

The module provides:

* A link flap suppression feature used to provide control over short-lived link state flaps.

* An interface link state dampening feature that is used to provide control over longer lived link state flaps.

* An encapsulation container and extensible choice statement for use by any interface types that allow for configurable L2 encapsulations.

* A loopback configuration leaf that is primarily aimed at loopback at the physical layer.

* MTU configuration leafs applicable to all packet/frame based interfaces.

* A forwarding mode leaf to indicate the OSI layer at which the interface handles traffic.

* A peer interface leaf useable for all types of associated peer interfaces, such as virtual interface pairs.

* A parent interface leaf useable for all types of sub-interface that are children of parent interfaces.

The "ietf-if-extensions" YANG module has the following structure:

```
module: ietf-if-extensions
  augment /if:interfaces/if:interface:
    +--rw link-flap-suppression {link-flap-suppression}?
    |  +--rw down?                   uint32
    |  +--rw up?                     uint32
    |  +--ro carrier-transitions?    yang:counter64
    |  +--ro timer-running?          enumeration
    +--rw dampening! {dampening}?
    |  +--rw half-life?            uint32
    |  +--rw reuse?                uint32
    |  +--rw suppress?             uint32
    |  +--rw max-suppress-time?    uint32
    |  +--ro penalty?              uint32
    |  +--ro suppressed?           boolean
    |  +--ro time-remaining?       uint32
    +--rw encapsulation
    |  +--rw (encaps-type)?
    +--rw loopback?                 identityref {loopback}?
    +--rw max-frame-size?           uint32 {max-frame-size}?
    +--ro forwarding-mode?          identityref
    +--rw peer-interface?           if:interface-ref {peer-interface}?
  augment /if:interfaces/if:interface:
    +--rw parent-interface    if:interface-ref {sub-interfaces}?
  augment /if:interfaces/if:interface/if:statistics:
    +--ro in-discard-overflows?   yang:counter64
  augment /if:interfaces/if:interface/if:statistics:
    +--ro in-discard-unknown-encaps?   yang:counter64
            {sub-interfaces}?
```

2.1. Link Flap Suppression

The link flap suppression feature augments the IETF interfaces data model with configuration for a simple algorithm that is used, generally on physical interfaces, to suppress short transient changes in the interface link state. It can be used in conjunction with the dampening feature described in Section 2.2 to provide effective control of unstable links and unwanted state transitions.

The principle of the link flap suppression feature is to use a short per interface timer to ensure that any interface link state transition that occurs and reverts back within the specified time interval is entirely suppressed without providing any signalling to any upper layer protocols that the state transition has occurred. I.e., in the case that the link state transition is suppressed then there is no change of the /if:interfaces/if:interface/oper-status or /if:interfaces/if:interfaces/last-change leafs for the interface that the feature is operating on. One obvious side effect of using this feature that is that any state transition will always be delayed by the specified time interval.

The configuration allows for separate timer values to be used in the suppression of down->up->down link transitions vs up->down->up link transitions.

The link flap suppression down timer leaf specifies the amount of time that an interface that is currently in link up state must be continuously down before the down state change is reported to higher level protocols. Use of this timer can cause traffic to be black holed for the configured value and delay reconvergence after link failures, therefore its use is normally restricted to cases where it is necessary to allow enough time for another protection mechanism (such as an optical layer automatic protection system) to take effect.

The link flap suppression up timer leaf specifies the amount of time that an interface that is currently in link down state must be continuously up before the down->up link state transition is reported to higher level protocols. This timer is generally useful as a debounce mechanism to ensure that a link is relatively stable before being brought into service. It can also be used effectively to limit the frequency at which link state transition events may occur. The default value for this leaf is determined by the underlying network device, and MAY vary based on the ’type’ of the interface.

2.2. Dampening

The dampening feature introduces a configurable exponential decay mechanism to suppress the effects of excessive interface link state flapping. This feature allows the network operator to configure a device to automatically identify and selectively dampen a local interface which is flapping. Dampening an interface keeps the interface operationally down until the interface stops flapping and becomes stable. Configuring the dampening feature can improve convergence times and stability throughout the network by isolating failures so that disturbances are not propagated, which reduces the utilization of system processing resources by other devices in the

network and improves overall network stability.

The basic algorithm uses a counter that is increased by 1000 units every time the underlying interface link state changes from up to down. If the counter increases above the suppress threshold then the interface is kept down (and out of service) until either the maximum suppression time is reached, or the counter has reduced below the reuse threshold. The half-life period determines that rate at which the counter is periodically reduced by half.

2.2.1. Suppress Threshold

The suppress threshold is the value of the accumulated penalty that triggers the device to dampen a flapping interface. The flapping interface is identified by the device and assigned a penalty for each up to down link state change, but the interface is not automatically dampened. The device tracks the penalties that a flapping interface accumulates. When the accumulated penalty reaches or exceeds the suppress threshold, the interface is placed in a suppressed state.

2.2.2. Half-Life Period

The half-life period determines how fast the accumulated penalties can decay exponentially. The accumulated penalty decays at a rate that causes its value to be reduced by half after each half-life period.

2.2.3. Reuse Threshold

If, after one or more half-life periods, the accumulated penalty decreases below the reuse threshold and the underlying interface link state is up then the interface is taken out of suppressed state and is allowed to go up.

2.2.4. Maximum Suppress Time

The maximum suppress time represents the maximum amount of time an interface can remain dampened when a new penalty is assigned to an interface. The default of the maximum suppress timer is four times the half-life period. The maximum value of the accumulated penalty is calculated using the maximum suppress time, reuse threshold and half-life period.

2.3. Encapsulation

The encapsulation container holds a choice node that is to be augmented with datalink layer specific encapsulations, such as HDLC, PPP, or sub-interface 802.1Q tag match encapsulations. The use of a choice statement ensures that an interface can only have a single datalink layer protocol configured.

The different encapsulations themselves are defined in separate YANG modules defined in other documents that augment the encapsulation choice statement. For example the Ethernet specific basic 'dot1q-vlan' encapsulation is defined in ietf-if-l3-vlan.yang and the 'flexible' encapsulation is defined in ietf-flexible-encapsulation.yang, both modules from [I-D.ietf-netmod-sub-intf-vlan-model].

2.4. Loopback

The loopback configuration leaf allows any physical interface to be configured to be in one of the possible following physical loopback modes, i.e. internal loopback, line loopback, or use of an external loopback connector. The use of YANG identities allows for the model to be extended with other modes of loopback if required.

The following loopback modes are defined:

* Internal loopback - All egress traffic on the interface is internally looped back within the interface to be received on the ingress path.

* Line loopback - All ingress traffic received on the interface is internally looped back within the interface to the egress path.

* Loopback Connector - The interface has a physical loopback connector attached that loops all egress traffic back into the interface's ingress path, with equivalent semantics to internal loopback.

2.5. Maximum frame size

A maximum frame size configuration leaf (max-frame-size) is provided to specify the maximum size of a layer 2 frame that may be transmitted or received on an interface. The value includes the overhead of any layer 2 header, the maximum length of the payload, and any frame check sequence (FCS) bytes. If configured, the max-frame-size leaf on an interface also restricts the max-frame-size of any child sub-interfaces, and the available MTU for protocols.

2.6. Sub-interface

The sub-interface feature specifies the minimal leafs required to define a child interface that is parented to another interface.

A sub-interface is a logical interface that handles a subset of the traffic on the parent interface. Separate configuration leafs are used to classify the subset of ingress traffic received on the parent interface to be processed in the context of a given sub-interface. All egress traffic processed on a sub-interface is given to the parent interface for transmission. Otherwise, a sub-interface is like any other interface entry in the /if:interfaces/if:interface list, appearing as another regular list entry, and supporting the same standard interface features and configuration. For example, using the ietf-ip YANG module [RFC8344], to configure an IPv6 address on a sub-interface uses the same schema path, /if:interfaces/if:interface/ip:ipv6/ip:address/ip:ip, that would also be used to configure an IPv6 address on any other type of interface. Further examples of sub-interface configuration are given in the appendices of [I-D.ietf-netmod-sub-intf-vlan-model].

For some vendor specific interface naming conventions the name of the child interface is sufficient to determine the parent interface, which implies that the child interface can never be reparented to a different parent interface after it has been created without deleting the existing sub-interface and recreating a new sub-interface. Even in this case it is useful to have a well-defined leaf to cleanly identify the parent interface.

The model also allows for arbitrarily named sub-interfaces by having an explicit parent-interface leaf define the child -> parent relationship. In this naming scenario it is also possible for implementations to allow for logical interfaces to be reparented to new parent interfaces without needing the sub-interface to be destroyed and recreated.

2.7. Forwarding Mode

The forwarding mode leaf provides additional information as to what mode or layer an interface (or sub-interface) is logically operating and forwarding traffic at. The implication of this leaf is that for traffic forwarded at a given layer that any headers for lower layers are stripped off before the packet is forwarded at the given layer. Conversely, on egress any lower layer headers must be added to the packet before it is transmitted out of the interface.

The following forwarding modes are defined:

* Physical - Traffic is being forwarded at the physical layer. This includes DWDM or OTN based switching.

* Data-link - Layer 2 based forwarding, such as Ethernet/VLAN based switching, or L2VPN services.

* Network - Network layer based forwarding, such as IP, MPLS, or L3VPNs.

2.8. Peer-interface

The peer-interface feature specifies the minimal leafs required to define a peer interface that is directly associated to another interface. All or a subset of the egress traffic of an interface will be received by its peer-interface on its ingress path and vice versa. This can, for example, be used to model virtual interface pairs which are commonly applied to interconnect different network namespaces on the same hosts in the context of containerization.

3. Interfaces Ethernet-Like YANG Module Introduction

The Interfaces Ethernet-Like Module is a small module that contains all configuration and operational data that is common across interface types that use Ethernet framing as their datalink layer encapsulation.

This module currently contains leafs for the configuration and reporting of the operational MAC address and the burnt-in MAC address (BIA) associated with any interface using Ethernet framing.

The module also contains a set of counters that are used to report the number of packets received and transmitted on an interface that fall into various size ranges, in a style that is similar etherStatsPkt* counters, defined in RFC 2819, but covering a wider range of frame sizes, and providing counters for both the ingress and egress directions.

The "ietf-if-ethernet-like" YANG module has the following structure:

```
module: ietf-if-ethernet-like
  augment /if:interfaces/if:interface:
    +--rw ethernet-like
       +--rw mac-address?       yang:mac-address
       |       {configurable-mac-address}?
       +--ro bia-mac-address?   yang:mac-address
  augment /if:interfaces/if:interface/if:statistics:
    +--ro in-discard-unknown-dest-mac-pkts?   yang:counter64
    +--ro in-pkts-64-octets?                  yang:counter64
    +--ro in-pkts-65-to-127-octets?           yang:counter64
    +--ro in-pkts-128-to-255-octets?          yang:counter64
    +--ro in-pkts-256-to-511-octets?          yang:counter64
    +--ro in-pkts-512-to-1023-octets?         yang:counter64
    +--ro in-pkts-1024-to-1518-octets?        yang:counter64
    +--ro in-pkts-1519-to-2047-octets?        yang:counter64
    +--ro in-pkts-2048-to-4095-octets?        yang:counter64
    +--ro in-pkts-4096-to-8191-octets?        yang:counter64
    +--ro in-pkts-8192-to-max-octets?         yang:counter64
    +--ro out-pkts-64-octets?                 yang:counter64
    +--ro out-pkts-65-to-127-octets?          yang:counter64
    +--ro out-pkts-128-to-255-octets?         yang:counter64
    +--ro out-pkts-256-to-511-octets?         yang:counter64
    +--ro out-pkts-512-to-1023-octets?        yang:counter64
    +--ro out-pkts-1024-to-1518-octets?       yang:counter64
    +--ro out-pkts-1519-to-2047-octets?       yang:counter64
    +--ro out-pkts-2048-to-4095-octets?       yang:counter64
    +--ro out-pkts-4096-to-8191-octets?       yang:counter64
    +--ro out-pkts-8192-to-max-octets?        yang:counter64
```

4. Interface Extensions YANG Module

This YANG module augments the interface container defined in [RFC8343]. It also contains references to [RFC6991] and [RFC7224].

```
<CODE BEGINS> file "ietf-if-extensions@2025-08-07.yang"
module ietf-if-extensions {
  yang-version 1.1;

  namespace "urn:ietf:params:xml:ns:yang:ietf-if-extensions";

  prefix if-ext;

  import ietf-yang-types {
    prefix yang;
    reference "RFC 6991: Common YANG Data Types";
  }

  import ietf-interfaces {
```

```
  prefix if;
  reference
    "RFC 8343: A YANG Data Model For Interface Management";
}

import iana-if-type {
  prefix ianaift;
  reference "RFC 7224: IANA Interface Type YANG Module";
}

organization
  "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

contact
  "WG Web:   <https://datatracker.ietf.org/wg/netmod/about/>
   WG List:  <mailto:netmod@ietf.org>

   Editor:   Robert Wilton
             <mailto:rwilton@cisco.com>

   Editor:   Scott Mansfield
             <mailto:scott.mansfield@ericsson.com>";

description
  "This module contains definitions for extending the IETF
   interface YANG model (RFC 8343) with common configurable
   layer 2 properties.

   Copyright (c) 2025 IETF Trust and the persons identified as

   authors of the code.  All rights reserved.

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

   This version of this YANG module is part of RFC XXXX
   (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
   for full legal notices.

   The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL
   NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'NOT RECOMMENDED',
   'MAY', and 'OPTIONAL' in this document are to be interpreted as
   described in BCP 14 (RFC 2119) (RFC 8174) when, and only when,
   they appear in all capitals, as shown here.";
```

```
  revision 2025-08-07 {
    description
      "Initial revision";

    reference
      "RFC XXXX, Common Interface Extension YANG Data Models";
  }

  feature link-flap-suppression {
    description
      "This feature indicates that configurable interface link
       delay is supported, which is a feature is used to limit the
       propagation of very short interface link state flaps.";
    reference "RFC XXXX, Section 2.1 Link Flap Suppression";
  }

  feature dampening {
    description
      "This feature indicates that the device supports interface
       dampening, which is a feature that is used to limit the
       propagation of interface link state flaps over longer
       periods.";
    reference "RFC XXXX, Section 2.2 Dampening";
  }

  feature loopback {
    description
      "This feature indicates that configurable interface loopback
       is supported.";
    reference "RFC XXXX, Section 2.4 Loopback";
  }

  feature max-frame-size {
    description
      "This feature indicates that the device supports configuring
       or reporting the maximum frame size on interfaces.";
    reference "RFC XXXX, Section 2.5 Maximum Frame Size";
  }

  feature sub-interfaces {
    description
      "This feature indicates that the device supports the
       instantiation of sub-interfaces.  Sub-interfaces are defined
       as logical child interfaces that allow features and
       forwarding decisions to be applied to a subset of the traffic
       processed on the specified parent interface.";
    reference "RFC XXXX, Section 2.6 Sub-interface";
  }
```

```
  feature peer-interface {
    description
      "This feature indicates that the device supports the
       specification of peer-interfaces.  A Peer-interface is
       defined as logical peer that receives egress traffic of the
       given interface on its ingress path and vice versa.";
    reference "RFC XXXX, Section 2.8 Peer-interface";
  }

  identity loopback {
    description "Base identity for interface loopback options";
    reference "RFC XXXX, Section 2.4";
  }
  identity internal {
    base loopback;
    description
      "All egress traffic on the interface is internally looped back
       within the interface to be received on the ingress path.";
    reference "RFC XXXX, Section 2.4";
  }
  identity line {
    base loopback;
    description
      "All ingress traffic received on the interface is internally
       looped back within the interface to the egress path.";
    reference "RFC XXXX, Section 2.4";
  }
  identity connector {
    base loopback;
    description
      "The interface has a physical loopback connector attached that
       loops all egress traffic back into the interface's ingress
       path, with equivalent semantics to loopback internal.";
    reference "RFC XXXX, Section 2.4";
  }

  identity forwarding-mode {
    description "Base identity for forwarding-mode options.";
    reference "RFC XXXX, Section 2.7";
  }
  identity physical {
    base forwarding-mode;
    description
      "Physical layer forwarding.  This includes DWDM or OTN based
       optical switching.";
    reference "RFC XXXX, Section 2.7";
  }
  identity data-link {
```

```
      base forwarding-mode;
      description
        "Layer 2 based forwarding, such as Ethernet/VLAN based
         switching, or L2VPN services.";
      reference "RFC XXXX, Section 2.7";
    }
    identity network {
      base forwarding-mode;
      description
        "Network layer based forwarding, such as IP, MPLS,
         or L3VPNs.";
      reference "RFC XXXX, Section 2.7";
    }


    /*
     * Augments the IETF interfaces model with leafs to configure
     * and monitor link-flap-suppression on an interface.
     */
    augment "/if:interfaces/if:interface" {
      description
        "Augments the IETF interface model with optional common
         interface level commands that are not formally covered by any
         specific standard.";

      /*
       * Defines standard YANG for the Link Flap Suppression feature.
       */
      container link-flap-suppression {
        if-feature "link-flap-suppression";
        description
          "Holds link flap related feature configuration.";
        leaf down {
          type uint32;
          units milliseconds;
          description
            "Delays the propagation of a 'loss of carrier signal'
             event that would cause the interface state to go down,
             i.e., the command allows short link flaps to be
             suppressed. The configured value indicates the minimum
             time interval (in milliseconds) that the link signal must
             be continuously down before the interface state is
             brought down. If not configured, the behavior on loss of
             link signal is vendor/interface specific, but with the
             general expectation that there should be little or no
             delay.";
        }
        leaf up {
```

```
          type uint32;
          units milliseconds;
          description
            "Defines the minimum time interval (in milliseconds) that
             the link signal must be continuously present and error
             free before the interface state is allowed to transition
             from down to up.  If not configured, the behavior is
             vendor/interface specific, but with the general
             expectation that sufficient default delay should be used
             to ensure that the interface is stable when enabled
             before being reported as being up.  Configured values
             that are too low for the hardware capabilities may be
             rejected.";
        }
        leaf carrier-transitions {
          type yang:counter64;
          units transitions;
          config false;
          description
            "Defines the number of times the underlying link state
             has changed to, or from, state up.  This counter should
             be incremented even if the high layer interface state
             changes are being suppressed by a running link flap
             suppression timer.";
        }
        leaf timer-running {
          type enumeration {
            enum none {
              description
                "No link flap suppression timer is running.";
            }
            enum up {
              description
                "link-flap-suppression up timer is running.  The
                 underlying link state is up, but interface state is
                 not reported as up.";
            }
            enum down {
              description
                "link-flap-suppression down timer is running.
                 Interface state is reported as up, but the underlying
                 link state is actually down.";
            }
          }
          config false;
          description
            "Reports whether a link flap suppression timer is actively
             running, in which case the interface state does not match
```

```
            the underlying link state.";
        }

        reference "RFC XXXX, Section 2.1 Link Flap Suppression";
      }

      /*
       * Augments the IETF interfaces model with a container to hold
       * generic interface dampening
       */
      container dampening {
        if-feature "dampening";
        presence
          "Enable interface link flap dampening with default settings
           (that are vendor/device specific).";
        description
          "Interface dampening limits the propagation of interface
           link state flaps over longer periods.";
        reference "RFC XXXX, Section 2.2 Dampening";

        leaf half-life {
          type uint32;
          units seconds;
          description
            "The time (in seconds) after which a penalty would be half
             its original value.  Once the interface has been assigned
             a penalty, the penalty is decreased at a decay rate
             equivalent to the half-life.  For some devices, the
             allowed values may be restricted to particular multiples
             of seconds.  The default value is vendor/device
             specific.";
          reference "RFC XXXX, Section 2.3.2 Half-Life Period";
        }

        leaf reuse {
          type uint32;
          description
            "Penalty value below which a stable interface is
             unsuppressed (i.e., brought up) (no units).  The default
             value is vendor/device specific.  The penalty value for a
             link up->down state change is 1000 units.";
          reference "RFC XXXX, Section 2.2.3 Reuse Threshold";
        }

        leaf suppress {
          type uint32;
          description
            "Limit at which an interface is suppressed (i.e., held
```

```
           down) when its penalty exceeds that limit (no units).
           The value must be greater than the reuse threshold.  The
           default value is vendor/device specific.  The penalty
           value for a link up->down state change is 1000 units.";
         reference "RFC XXXX, Section 2.2.1 Suppress Threshold";
       }

       leaf max-suppress-time {
         type uint32;
         units seconds;
         description
           "Maximum time (in seconds) that an interface can be
           suppressed before being unsuppressed if no further link
           up->down state change penalties have been applied.  This
           value effectively acts as a ceiling that the penalty
           value cannot exceed.  The default value is vendor/device
           specific.";
         reference "RFC XXXX, Section 2.2.4 Maximum Suppress Time";
       }

       leaf penalty {
         type uint32;
         config false;
         description
           "The current penalty value for this interface.  When the
           penalty value exceeds the 'suppress' leaf then the
           interface is suppressed (i.e., held down).";
         reference "RFC XXXX, Section 2.2 Dampening";
       }

       leaf suppressed {
         type boolean;
         config false;
         description
           "Represents whether the interface is suppressed (i.e.,
           held down) because the 'penalty' leaf value exceeds the
           'suppress' leaf.";
         reference "RFC XXXX, Section 2.2 Dampening";
       }

       leaf time-remaining {
         when '../suppressed = "true"' {
           description
             "Only suppressed interfaces have a time remaining.";
         }
         type uint32;
         units seconds;
         config false;
```

```
          description
            "For a suppressed interface, this leaf represents how long
             (in seconds) that the interface will remain suppressed
             before it is allowed to go back up again.";
          reference "RFC XXXX, Section 2.2 Dampening";
        }
      }

      /*
       * Various types of interfaces support a configurable layer 2
       * encapsulation, any that are supported by YANG should be
       * listed here.
       *
       * Different encapsulations can hook into the common encaps-type
       * choice statement.
       */
      container encapsulation {
        when
          "derived-from-or-self(../if:type,
                                'ianaift:ethernetCsmacd') or
           derived-from-or-self(../if:type,
                                'ianaift:ieee8023adLag') or
           derived-from-or-self(../if:type, 'ianaift:pos') or
           derived-from-or-self(../if:type,
                                'ianaift:atmSubInterface') or
           derived-from-or-self(../if:type, 'ianaift:l2vlan')" {

          description
            "All interface types that can have a configurable L2
             encapsulation.";
        }

        description
          "Holds the OSI layer 2 encapsulation associated with an
           interface.";
        choice encaps-type {
          description
            "Extensible choice of layer 2 encapsulations";
          reference "RFC XXXX, Section 2.3 Encapsulation";
        }
      }

       /*
        * Various types of interfaces support loopback configuration,
        * any that are supported by YANG should be listed here.
        */
      leaf loopback {
        when "derived-from-or-self(../if:type,
```

```
                                  'ianaift:ethernetCsmacd') or
              derived-from-or-self(../if:type, 'ianaift:sonet') or
              derived-from-or-self(../if:type, 'ianaift:atm') or
              derived-from-or-self(../if:type, 'ianaift:otnOtu')" {
            description
              "All interface types supporting loopback configuration.";
          }
          if-feature "loopback";
          type identityref {
            base loopback;
          }
          description "Enables traffic loopback.";
          reference "RFC XXXX, Section 2.4 Loopback";
        }

        /*
         * Allows the maximum frame size to be configured or reported.
         */
        leaf max-frame-size {
          if-feature "max-frame-size";
          type uint32 {
            range "64 .. max";
          }
          description
            "The maximum size of layer 2 frames that may be transmitted
             or received on the interface (including any frame header,
             maximum frame payload size, and frame checksum sequence).

             If configured, the max-frame-size also limits the maximum
             frame size of any child sub-interfaces.  The MTU available
             to higher layer protocols is restricted to the maximum
             frame payload size, and MAY be further restricted by
             explicit layer 3 or protocol specific MTU configuration.";

          reference "RFC XXXX, Section 2.5 Maximum Frame Size";
        }

        /*
         * Augments the IETF interfaces model with a leaf that indicates
         * which mode, or layer, is being used to forward the traffic.
         */
        leaf forwarding-mode {
          type identityref {
            base forwarding-mode;
          }
          config false;

          description
```

```
          "The forwarding mode that the interface is operating in.";
        reference "RFC XXXX, Section 2.7 Forwarding Mode";
      }
      leaf peer-interface {
        if-feature "peer-interface";

        type if:interface-ref;

        description
          "The peer interface associated to this interface.";
        reference "RFC XXXX, Section 2.8 Peer-interface";
      }
    }

    /*
     * Add generic support for sub-interfaces.
     *
     * This should be extended to cover all interface types that are
     * child interfaces of other interfaces.
     */
    augment "/if:interfaces/if:interface" {
      when "derived-from-or-self(if:type, 'ianaift:l2vlan') or
            derived-from-or-self(if:type, 'ianaift:atmSubInterface')
            or derived-from-or-self(if:type, 'ianaift:frameRelay')"  {
        description
          "Any ianaift:types that explicitly represent
           sub-interfaces.";
      }
      if-feature "sub-interfaces";

      description
        "Adds a parent interface field to interfaces that model
         sub-interfaces.";
      leaf parent-interface {

        type if:interface-ref;

        mandatory true;
        description
          "This is the reference to the parent interface of this
           sub-interface.";
        reference "RFC XXXX, Section 2.6 Sub-interface";
      }
    }

    /*
     * Add overflow discards counter.
     */
```

```
  augment "/if:interfaces/if:interface/if:statistics" {
    description
      "Augment the interface model with an overflows discards
       counter";
    leaf in-discard-overflows {
      type yang:counter64;
      units packets;
      description
        "A count of the number of packets discarded because of
         overflows, e.g., because there were insufficient
         resources to buffer or process the frame.

         This counter does not include packets that were discarded
         due to a QOS policy.

         For consistency, packets counted against this counter are
         also counted against the IETF interfaces statistics.  In
         particular, they are included in in-octets and in-discards,
         but are not included in in-unicast-pkts, in-multicast-pkts
         or in-broadcast-pkts, because they are not delivered to a
         higher layer.

         Discontinuities in the values of this counter can occur at
         re-initialization of the management system, and at other
         times as indicated by the value of the 'discontinuity-time'
         leaf defined in the ietf-interfaces YANG module
         (RFC 8343).";
    }
  }

  /*
   * Add discard counter for unknown sub-interface encapsulation
   */
  augment "/if:interfaces/if:interface/if:statistics" {
    when "derived-from-or-self(../if:type,
                               'ianaift:ethernetCsmacd') or
          derived-from-or-self(../if:type,
                               'ianaift:ieee8023adLag') or
          derived-from-or-self(../if:type, 'ianaift:ifPwType')" {
      description
        "Applies to interfaces that can demultiplex ingress packets
         to sub-interfaces.";
    }
    if-feature "sub-interfaces";

    description
      "Augment the interface model statistics with a sub-interface
       demux discard counter.";
```

```
      leaf in-discard-unknown-encaps {
        type yang:counter64;
        units packets;
        description
          "A count of the number of packets that were well-formed,
           but otherwise discarded because their encapsulation does
           not classify the packet to the interface or any child
           sub-interface.  E.g., a packet might be discarded because
           it has an unknown VLAN Id, or does not have a VLAN Id when
           one is expected.

           For consistency, packets counted against this counter are
           also counted against the IETF interfaces statistics.  In
           particular, they are included in in-octets and in-discards,
           but are not included in in-unicast-pkts, in-multicast-pkts
           or in-broadcast-pkts, because they are not delivered to a
           higher layer.

           Discontinuities in the values of this counter can occur at
           re-initialization of the management system, and at other
           times as indicated by the value of the 'discontinuity-time'
           leaf defined in the ietf-interfaces YANG module
           (RFC 8343).";
      }
    }
  }
  <CODE ENDS>
```

5. Interfaces Ethernet-Like YANG Module

This YANG module augments the interface container defined in RFC 8343 [RFC8343] for Ethernet-like interfaces. This includes Ethernet interfaces, 802.3 LAG (802.1AX) interfaces, Switch Virtual interfaces, and Pseudo-Wire Head-End interfaces. It also contains references to [RFC6991], [RFC7224], and [IEEE_802.3.2_2019].

```
<CODE BEGINS> file "ietf-if-ethernet-like@2025-08-07.yang"
module ietf-if-ethernet-like {
  yang-version 1.1;

  namespace
    "urn:ietf:params:xml:ns:yang:ietf-if-ethernet-like";

  prefix ethlike;

  import ietf-interfaces {
    prefix if;
    reference
```

```
        "RFC 8343: A YANG Data Model For Interface Management";
    }

    import ietf-yang-types {
      prefix yang;
      reference "RFC 6991: Common YANG Data Types";
    }

    import iana-if-type {
      prefix ianaift;
      reference "RFC 7224: IANA Interface Type YANG Module";
    }

    organization
      "IETF NETMOD (NETCONF Data Modeling Language) Working Group";

    contact
      "WG Web:   <https://datatracker.ietf.org/wg/netmod/about/>
       WG List:  <mailto:netmod@ietf.org>

       Editor:   Robert Wilton
                 <mailto:rwilton@cisco.com>

       Editor:   Scott Mansfield
                 <mailto:scott.mansfield@ericsson.com>";

    description
      "This module contains YANG definitions for configuration for
       'Ethernet-like' interfaces.  It is applicable to all interface
       types that use Ethernet framing and expose an Ethernet MAC
       layer, and includes such interfaces as physical Ethernet
       interfaces, Ethernet LAG interfaces and VLAN sub-interfaces.

       Additional interface configuration and counters for physical
       Ethernet interfaces are defined in
       ieee802-ethernet-interface.yang, as part of IEEE Std
       802.3.2-2019.

       Copyright (c) 2025 IETF Trust and the persons identified as
       authors of the code.  All rights reserved.

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

```
        This version of this YANG module is part of RFC XXXX
        (https://www.rfc-editor.org/info/rfcXXXX); see the RFC itself
        for full legal notices.";

      revision 2025-08-07 {
        description "Initial revision.";

        reference
          "RFC XXXX, Common Interface Extension YANG Data Models";
      }

      feature configurable-mac-address {
        description
          "This feature indicates that MAC addresses on Ethernet-like
           interfaces can be configured.";
        reference
          "RFC XXXX, Section 3, Interfaces Ethernet-Like Module";
      }

      grouping ingress-frame-size-counters {
        description
          "A grouping that defines a set of counters that are used to
           provide ingress frame size counters in the style of the
           etherStatsPkt* counters, defined in RFC 2819, but extended
           to cover the larger frame sizes supported by some hardware.";
        reference
          "RFC 2819, Section 5, EtherStatsEntry";

        leaf in-pkts-64-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets (including bad packets)
             received that were 64 octets in length (excluding framing
             bits but including FCS octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
          reference
            "RFC 2819, Section 5, etherStatsPkts64Octets";
        }

        leaf in-pkts-65-to-127-octets {
          type yang:counter64;
          units frames;
```

```
          description
            "The total number of packets (including bad packets)
             received that were between 65 and 127 octets in length
             inclusive (excluding framing bits but including FCS
             octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
          reference
            "RFC 2819, Section 5, etherStatsPkts65to127Octets";
        }

        leaf in-pkts-128-to-255-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets (including bad packets)
             received that were between 128 and 255 octets in length
             inclusive (excluding framing bits but including FCS
             octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
          reference
            "RFC 2819, Section 5, etherStatsPkts128to255Octets";
        }

        leaf in-pkts-256-to-511-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets (including bad packets)
             received that were between 256 and 511 octets in length
             inclusive (excluding framing bits but including FCS
             octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
          reference
```

```
          "RFC 2819, Section 5, etherStatsPkts256to511Octets";
        }

        leaf in-pkts-512-to-1023-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets (including bad packets)
             received that were between 512 and 1023 octets in length
             inclusive (excluding framing bits but including FCS
             octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
          reference
            "RFC 2819, Section 5, etherStatsPkts512to1023Octets";
        }

        leaf in-pkts-1024-to-1518-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets (including bad packets)
             received that were between 1024 and 1518 octets in length
             inclusive (excluding framing bits but including FCS
             octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
          reference
            "RFC 2819, Section 5, etherStatsPkts1024to1518Octets";
        }

        leaf in-pkts-1519-to-2047-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets (including bad
             packets) received that were between 1519 and 2047
             octets in length inclusive (excluding framing
             bits but including FCS octets).
```

```
          Discontinuities in the values of this counter can occur at
          re-initialization of the management system, and at other
          times as indicated by the value of the 'discontinuity-time'
          leaf defined in the ietf-interfaces YANG module
          (RFC 8343).";
      }

      leaf in-pkts-2048-to-4095-octets {
        type yang:counter64;
        units frames;
        description
          "The total number of packets (including bad
           packets) received that were between 2048 and 4095
           octets in length inclusive (excluding framing
           bits but including FCS octets).

           Discontinuities in the values of this counter can occur at
           re-initialization of the management system, and at other
           times as indicated by the value of the 'discontinuity-time'
           leaf defined in the ietf-interfaces YANG module
           (RFC 8343).";
      }

      leaf in-pkts-4096-to-8191-octets {
        type yang:counter64;
        units frames;
        description
          "The total number of packets (including bad
           packets) received that were between 4096 and 8191
           octets in length inclusive (excluding framing
           bits but including FCS octets).

           Discontinuities in the values of this counter can occur at
           re-initialization of the management system, and at other
           times as indicated by the value of the 'discontinuity-time'
           leaf defined in the ietf-interfaces YANG module
           (RFC 8343).";
      }

      leaf in-pkts-8192-to-max-octets {
        type yang:counter64;
        units frames;
        description
          "The total number of packets (including bad packets)
           received that were more than 8192 octets in length
           inclusive (excluding framing bits but including FCS
           octets).
```

```
            Discontinuities in the values of this counter can occur at
            re-initialization of the management system, and at other
            times as indicated by the value of the 'discontinuity-time'
            leaf defined in the ietf-interfaces YANG module
            (RFC 8343).";
        }
      }

      grouping egress-frame-size-counters {
        description
          "A grouping that defines a set of counters that are used to
           provide frame size counters in the style of the
           etherStatsPkt* counters, defined in RFC 2819, but in the
           egress direction and extended to cover the larger frame
           sizes supported by some hardware.";

        leaf out-pkts-64-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets transmitted that were 64 octets
             in length (excluding framing bits but including FCS octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
        }

        leaf out-pkts-65-to-127-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets transmitted that were between
             65 and 127 octets in length inclusive (excluding framing
             bits but including FCS octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
        }

        leaf out-pkts-128-to-255-octets {
          type yang:counter64;
          units frames;
```

```
          description
            "The total number of packets transmitted that were between
             128 and 255 octets in length inclusive (excluding framing
             bits but including FCS octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
        }

        leaf out-pkts-256-to-511-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets transmitted that were between
             256 and 511 octets in length inclusive (excluding framing
             bits but including FCS octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
        }

        leaf out-pkts-512-to-1023-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets transmitted that were between
             512 and 1023 octets in length inclusive (excluding framing
             bits but including FCS octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
        }

        leaf out-pkts-1024-to-1518-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets transmitted that were between
             1024 and 1518 octets in length inclusive (excluding framing
```

```
            bits but including FCS octets).

            Discontinuities in the values of this counter can occur at
            re-initialization of the management system, and at other
            times as indicated by the value of the 'discontinuity-time'
            leaf defined in the ietf-interfaces YANG module
            (RFC 8343).";
        }

        leaf out-pkts-1519-to-2047-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets transmitted that were between
             1519 and 2047 octets in length inclusive (excluding framing
             bits but including FCS octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
        }

        leaf out-pkts-2048-to-4095-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets transmitted that were between
             2048 and 4095 octets in length inclusive (excluding framing
             bits but including FCS octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
        }

        leaf out-pkts-4096-to-8191-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets transmitted that were between
            4096 and 8191 octets in length inclusive (excluding framing
            bits but including FCS octets).

             Discontinuities in the values of this counter can occur at
```

```
            re-initialization of the management system, and at other
            times as indicated by the value of the 'discontinuity-time'
            leaf defined in the ietf-interfaces YANG module
            (RFC 8343).";
        }

        leaf out-pkts-8192-to-max-octets {
          type yang:counter64;
          units frames;
          description
            "The total number of packets transmitted that were more than
             8192 octets in length inclusive (excluding framing bits but
             including FCS octets).

             Discontinuities in the values of this counter can occur at
             re-initialization of the management system, and at other
             times as indicated by the value of the 'discontinuity-time'
             leaf defined in the ietf-interfaces YANG module
             (RFC 8343).";
        }
      }


      /*
       * Configuration parameters for Ethernet-like interfaces.
       */
      augment "/if:interfaces/if:interface" {
        when "derived-from-or-self(if:type, 'ianaift:ethernetCsmacd') or
              derived-from-or-self(if:type, 'ianaift:ieee8023adLag') or
              derived-from-or-self(if:type, 'ianaift:ifPwType')" {
          description "Applies to all Ethernet-like interfaces";
        }
        description
          "Augment the interface model with parameters for all
           Ethernet-like interfaces.";

        container ethernet-like {
          description
            "Contains parameters for interfaces that use Ethernet framing
             and expose an Ethernet MAC layer.";

          leaf mac-address {
            if-feature "configurable-mac-address";
            type yang:mac-address;
            description
              "The MAC address of the interface.  The operational value
               matches the /if:interfaces/if:interface/if:phys-address
               leaf defined in ietf-interface.yang.";
```

```
        }

        leaf bia-mac-address {
          type yang:mac-address;
          config false;
          description
            "The 'burnt-in' MAC address.  I.e., the default MAC
             address assigned to the interface if no MAC address has
             been explicitly configured on it.";
        }
      }
    }


    /*
     * Configuration parameters for Ethernet-like interfaces.
     */
    augment "/if:interfaces/if:interface/if:statistics" {
      when "derived-from-or-self(../if:type,
                                 'ianaift:ethernetCsmacd') or
            derived-from-or-self(../if:type,
                                 'ianaift:ieee8023adLag') or
            derived-from-or-self(../if:type, 'ianaift:ifPwType')" {
        description "Applies to all Ethernet-like interfaces";
      }
      description
        "Augment the interface model statistics with additional
         counters related to Ethernet-like interfaces.";

      leaf in-discard-unknown-dest-mac-pkts {
        type yang:counter64;
        units frames;
        description
          "A count of the number of frames that were well-formed, but
           otherwise discarded because the destination MAC address did
           not pass any ingress destination MAC address filter.

           For consistency, frames counted against this counter are
           also counted against the IETF interfaces statistics.  In
           particular, they are included in in-octets and in-discards,
           but are not included in in-unicast-pkts, in-multicast-pkts
           or in-broadcast-pkts, because they are not delivered to a
           higher layer.

           Discontinuities in the values of this counter can occur at
           re-initialization of the management system, and at other
           times as indicated by the value of the 'discontinuity-time'
           leaf defined in the ietf-interfaces YANG module
```

```
            (RFC 8343).";
        }

        uses ingress-frame-size-counters;
        uses egress-frame-size-counters;
      }
    }
    <CODE ENDS>
```

6. Examples

The following sections give some examples of how different parts of the YANG modules could be used. Examples are not given for the more trivial configuration, or for sub-interfaces, for which examples are contained in [I-D.ietf-netmod-sub-intf-vlan-model].

6.1. Link Flap Suppression Configuration

The following example shows how the operational state datastore could look like for an Ethernet interface without any link flap suppression configuration. The down leaf value of 0 indicates that link down events as always propagated to high layers immediately, but an up leaf value of 50 indicates that the interface must be up and stable for at least 50 msecs before the interface is reported as being up to the high layers.

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth0",
        "type": "iana-if-type:ethernetCsmacd",
        "oper-status": "up",
        "statistics": {
          "discontinuity-time": "2023-12-15T09:04:00-05:00"
        },
        "ietf-if-extensions:link-flap-suppression": {
          "down": 0,
          "up": 50
        }
      }
    ]
  }
}
```

The following example shows explicit link flap suppression delay up and down values have been configured. A 50 msec down leaf value has been used to potentially allow optical protection to recover the link

before the higher layer protocol state is flapped. A 1 second (1000 milliseconds) up leaf value has been used to ensure that the link is always reasonably stable before allowing traffic to be carried over it. This also has the benefit of greatly reducing the rate at which higher layer protocol state flaps could occur.

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth0",
        "type": "iana-if-type:ethernetCsmacd",
        "oper-status": "up",
        "statistics": {
          "discontinuity-time": "2023-12-15T09:04:00-05:00"
        },
        "ietf-if-extensions:link-flap-suppression": {
          "down": 50,
          "up": 1000
        }
      }
    ]
  }
}
```

6.2. Dampening configuration

The following example shows what the operational state datastore may look like for an interface configured with interface dampening. The 'suppressed' leaf indicates that the interface is currently suppressed (i.e. down) because the 'penalty' is greater than the 'suppress' leaf threshold. The 'time-remaining' leaf indicates that the interface will remain suppressed for another 103 seconds before the 'penalty' is below the 'reuse' leaf value and the interface is allowed to go back up again.

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth0",
        "type": "iana-if-type:ethernetCsmacd",
        "oper-status": "down",
        "statistics": {
          "discontinuity-time": "2023-12-15T09:04:00-05:00"
        },
        "ietf-if-extensions:dampening": {
          "half-life": 60,
          "reuse": 750,
          "suppress": 2000,
          "max-suppress-time": 240,
          "penalty": 2480,
          "suppressed": true,
          "time-remaining": 103
        }
      }
    ]
  }
}
```

6.3. MAC address configuration

The following example shows how the operational state datastore could look like for an Ethernet interface without an explicit MAC address configured. The mac-address leaf always reports the actual operational MAC address that is in use. The bia-mac-address leaf always reports the default MAC address assigned to the hardware.

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth0",
        "type": "iana-if-type:ethernetCsmacd",
        "phys-address": "00:00:5E:00:53:30",
        "oper-status": "up",
        "admin-status": "up",
        "if-index": 1,
        "ietf-if-ethernet-like:ethernet-like": {
           "mac-address": "00:00:5E:00:53:30",
           "bia-mac-address": "00:00:5E:00:53:30"
        },
        "statistics": {
          "discontinuity-time": "2023-12-15T09:04:00-05:00"
        }
      }
    ]
  }
}
```

The following example shows the intended configuration for interface eth0 with an explicit MAC address configured.

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth0",
        "type": "iana-if-type:ethernetCsmacd",
        "ietf-if-ethernet-like:ethernet-like": {
           "mac-address": "00:00:5E:00:53:35"
        }
      }
    ]
  }
}
```

After the MAC address configuration has been successfully applied, the operational state datastore reporting the interface MAC address properties would contain the following, with the mac-address leaf updated to match the configured value, but the bia-mac-address leaf retaining the same value - which should never change.

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "eth0",
        "type": "iana-if-type:ethernetCsmacd",
        "phys-address": "00:00:5E:00:53:35",
        "oper-status": "up",
        "admin-status": "up",
        "if-index": 1,
        "ietf-if-ethernet-like:ethernet-like": {
           "mac-address": "00:00:5E:00:53:35",
           "bia-mac-address": "00:00:5E:00:53:30"
        },
        "statistics": {
          "discontinuity-time": "2023-12-15T09:04:00-05:00"
        }
      }
    ]
  }
}
```

6.4. Peer interface configuration

The following example shows how a veth pair modeled as peer interfaces is represented in the operational state datastore. It matches the following iproute2 command: ip link add veth1 type veth peer name veth2

```
{
  "ietf-interfaces:interfaces": {
    "interface": [
      {
        "name": "veth1",
        "type": "iana-if-type:propVirtual",
        "oper-status": "down",
        "statistics": {
          "discontinuity-time": "2023-12-15T09:04:00-05:00"
        },
        "ietf-if-extensions:peer-interface": "veth2"
      },
      {
        "name": "veth2",
        "type": "iana-if-type:propVirtual",
        "oper-status": "down",
        "statistics": {
          "discontinuity-time": "2023-12-15T09:04:00-05:00"
        },
        "ietf-if-extensions:peer-interface": "veth1"
      }
    ]
  }
}
```


7. Acknowledgements

The authors wish to thank Florian Kauer, Eric Gray, Ing-Wher Chen, Jon Culver, Juergen Schoenwaelder, Ladislav Lhotka, Lou Berger, Mahesh Jethanandani, Martin Bjorklund, Michael Zitao, Neil Ketley, Qin Wu, William Lupton, Xufeng Liu, Andy Bierman, and Vladimir Vassilev for their helpful comments contributing to this document. The authors would like to thank Mahesh Jethanandani for his thoughtful AD review.

8. IANA Considerations

8.1. YANG Module Registrations

The following YANG modules are requested to be registered in the IANA "YANG Module Names" [RFC6020] registry:

The ietf-if-extensions module:

Name: ietf-if-extensions

XML Namespace: urn:ietf:params:xml:ns:yang:ietf-if-extensions

Prefix: if-ext

Reference: RFCXXXX

The ietf-if-ethernet-like module:

Name: ietf-if-ethernet-like

XML Namespace: urn:ietf:params:xml:ns:yang:ietf-if-ethernet-like

Prefix: ethlike

Reference: RFCXXXX

This document registers two URIs in the "IETF XML Registry" [RFC3688]. Following the format in RFC 3688, the following registrations have been made.

URI: urn:ietf:params:xml:ns:yang:ietf-if-extensions

Registrant Contact: The IESG.

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

URI: urn:ietf:params:xml:ns:yang:ietf-if-ethernet-like

Registrant Contact: The IESG.

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

9. Security Considerations

This section is modeled after the template described in Section 3.7 of [I-D.ietf-netmod-rfc8407bis].

The ietf-if-extensions and ietf-if-ethernet-like YANG modules define data models that are designed to be accessed via YANG-based management protocols, such as NETCONF [RFC6241] and RESTCONF [RFC8040]. These protocols have to use a secure transport layer (e.g., SSH [RFC4252], TLS [RFC8446], and QUIC [RFC9000]) and have to use mutual authentication.

The Network Configuration Access Control Model (NACM) [RFC8341] provides the means to restrict access for particular NETCONF or RESTCONF users to a preconfigured subset of all available NETCONF or RESTCONF protocol operations and content.

There are a number of data nodes defined in this YANG module that are writable/creatable/deletable (i.e., "config true", which is the default). All writable data nodes are likely to be reasonably sensitive or vulnerable in some network environments. Write operations (e.g., edit-config) and delete operations to these data nodes without proper protection or authentication can have a negative effect on network operations. The following subtrees and data nodes have particular sensitivities/vulnerabilities:

* There are no particularly sensitive writable data nodes.

Some of the readable data nodes in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control read access (e.g., via get, get-config, or notification) to these data nodes. Specifically, the following subtrees and data nodes have particular sensitivities/ vulnerabilities:

* There are no particularly sensitive readable data nodes.

10. References

10.1. Normative References

[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>.

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

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

[RFC7950] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <https://www.rfc-editor.org/info/rfc7950>.

[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>.

[RFC8341] Bierman, A. and M. Bjorklund, "Network Configuration Access Control Model", STD 91, RFC 8341, DOI 10.17487/RFC8341, March 2018, <https://www.rfc-editor.org/info/rfc8341>.

[RFC8342] Bjorklund, M., Schoenwaelder, J., Shafer, P., Watsen, K., and R. Wilton, "Network Management Datastore Architecture (NMDA)", RFC 8342, DOI 10.17487/RFC8342, March 2018, <https://www.rfc-editor.org/info/rfc8342>.

[RFC8343] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.

10.2. Informative References

[I-D.ietf-netmod-rfc8407bis]
Bierman, A., Boucadair, M., and Q. Wu, "Guidelines for Authors and Reviewers of Documents Containing YANG Data Models", Work in Progress, Internet-Draft, draft-ietf-netmod-rfc8407bis-28, 5 June 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-rfc8407bis-28>.

[I-D.ietf-netmod-sub-intf-vlan-model]
Wilton, R. and S. Mansfield, "Sub-interface VLAN YANG Data Models", Work in Progress, Internet-Draft, draft-ietf-netmod-sub-intf-vlan-model-17, 8 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-netmod-sub-intf-vlan-model-17>.

[IEEE_802.3.2_2019]
IEEE, "IEEE Standard for Ethernet - YANG Data Model Definitions", IEEE 802-3, DOI 10.1109/IEEESTD.2019.8737019, 14 June 2019, <https://ieeexplore.ieee.org/document/8737019>.

[RFC4252] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, DOI 10.17487/RFC4252, January 2006, <https://www.rfc-editor.org/info/rfc4252>.

[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>.

[RFC6991] Schoenwaelder, J., Ed., "Common YANG Data Types", RFC 6991, DOI 10.17487/RFC6991, July 2013, <https://www.rfc-editor.org/info/rfc6991>.

[RFC7224] Bjorklund, M., "IANA Interface Type YANG Module", RFC 7224, DOI 10.17487/RFC7224, May 2014, <https://www.rfc-editor.org/info/rfc7224>.

[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>.

[RFC8340] Bjorklund, M. and L. Berger, Ed., "YANG Tree Diagrams", BCP 215, RFC 8340, DOI 10.17487/RFC8340, March 2018, <https://www.rfc-editor.org/info/rfc8340>.

[RFC8344] Bjorklund, M., "A YANG Data Model for IP Management", RFC 8344, DOI 10.17487/RFC8344, March 2018, <https://www.rfc-editor.org/info/rfc8344>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC9000] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/info/rfc9000>.

Authors' Addresses

Robert Wilton
Cisco Systems
Email: rwilton@cisco.com

Scott Mansfield
Ericsson
Email: scott.mansfield@ericsson.com