IEEE 802.1X YANG Module Specifications

Document status: Draft
Issue number: 0.4
Issue date: May 11, 2015
Author: Marc Holness

Publication History

Issue	Section(s)	Status	Author	Date	Comments
0.1	All	Draft	Marc Holness	April 30, 2015	Initial draft
0.2	Section 3	Draft	Marc Holness	May 01, 2015	Include <get-conf> examples
03		Draft	Marc Holness	May 08, 2015	Include augmentation structure to YANG module definitions
0.4		Draft	Marc Holness	May 11, 2015	

Contents

List of figures

1 Introduction

This document defines a YANG [RFC6020] data model for the management of 802.1X specified port-based network access control. The YANG module definitions, proposed here, are based on the existing IEEE 802.1X-2014 PAE management information model. This model is illustrated in Figure 1 below.

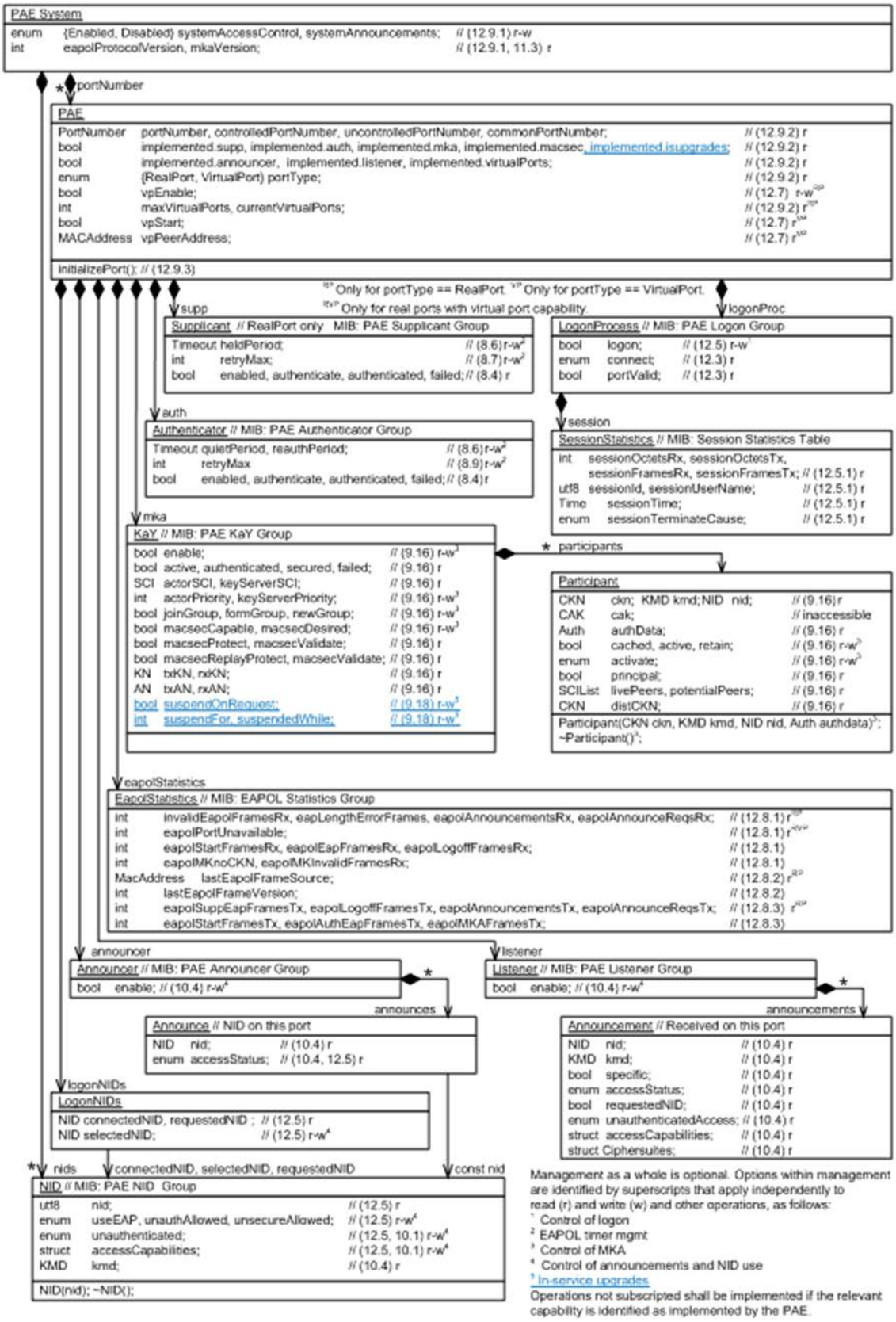

Figure 1: 802.1X-2014 PAE Information Model

The YANG modules will provide configuration management for 802.1X specified port-based network access controls. Port-based network access control allows a network administrator to restrict the use of IEEE 802 LAN service access points (ports) to secure communication between authenticated and authorized devices. IEEE Std 802.1X specifies an architecture, functional elements, and protocols that support mutual authentication between the clients of ports attached to the same LAN and secure communication between the ports.

The control allows a port to be reinitialized, terminating (and potentially restarting) authentication exchanges and MKA operation, based on a data model described in a set of YANG modules.

1.1 Scope, Model, and Applicability

The purpose of this document is to specify standards compliant IEEE 802.1X port-based network access control implementations of YANG modules, for configuration management.

1.2 802.1X and Netconf/YANG

Netconf is a widely accepted configuration management protocol that promises to simplify network configuration. YANG [RFC6020] is a formalized data modeling language used to model configuration and state data that can be used by Netconf [RFC6241]. The adoption of YANG, will allow IEEE 802.1 Bridging vendors (that provide port-based network access controls) as well as Network Management Systems to speak a common language, and thus simplify Service Provider operations.

Service Providers are moving towards a Netconf/YANG configuration management paradigm. As such vendors that provide IEEE 802.1 Bridges and supporting functionality would be interested in this amendment.

Currently, other SDOs (e.g., MEF and IETF) are developing YANG data models. IEEE 802 needs to provide specifications of YANG data models in support of IEE 802.1 managed entities. For example, MEF have defined YANG models for Service OAM Fault Management & Performance Monitoring, and IETF have defined YANG data models for Interface Management.

2 802.1X YANG Module Definitions

The YANG module definitions currently found in this document are based upon the PAE management information model (specified in 802.1Xbx-2014).

2.1 YANG Module Definition Structure

The YANG module definition structures, that are being considered, can be based upon

1. A schema where a list of ports to be used for 802.1X configuration management.

2. A schema where the 802.1X port is an augmentation of the IETF RFC7223 YANG data model definition.

2.2 YANG Data Model Schema

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

- Brackets "[" and "]" enclose list keys.
- Abbreviations before data node names: "rw" means configuration (read-write), and "ro" means state data

(read-only).

- Symbols after data node names: "?" means an optional node, "!" means a presence container, and "*" denotes a list and leaf-list.
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- Ellipsis ("...") stands for contents of subtrees that are not shown.

2.3 802.1X YANG Module (Based on Port List)

2.3.1 YANG Data Model Schema

The YANG data model schema is graphically represented below.

```
module: dot1x
   +--rw pae-system
      +--rw system-access-control?    enumeration
      +--rw system-announcements?     enumeration
      +--rw eapol-protocol-version?   uint32
      +--rw mka-version?              uint32
      +--rw pae* [port-number]
      |  +--rw port-number                portNumberType
      |  +--ro controlled-port-number?    portNumberType
      |  +--ro uncontrolled-port-number?  portNumberType
      |  +--ro common-port-number?        portNumberType
      |  +--ro port-type?                 enumeration
      |  +--rw virtual-port {virtual-ports}?
      |  |  +--rw enable?         boolean
      |  |  +--ro max?            uint32
      |  |  +--ro current?        yang:gauge32
      |  |  +--ro start?          uint32
      |  |  +--ro peer-address?   ieee:mac-address
      |  +--rw supplicant
      |  |  +--rw held-period?   uint16
      |  |  +--rw retry-max?     uint8
      |  |  +--ro status
      |  |     +--ro enabled?         boolean
      |  |     +--ro authenticate?    boolean
      |  |     +--ro authenticated?   boolean
      |  |     +--ro failed?          boolean
      |  +--rw authenticator
      |  |  +--rw quiet-period?    uint16
      |  |  +--rw reauth-period?   uint16
      |  |  +--rw retry-max?       uint8
      |  |  +--ro status
      |  |     +--ro enabled?         boolean
      |  |     +--ro authenticate?    boolean
      |  |     +--ro authenticated?   boolean
      |  |     +--ro failed?          boolean
      |  +--rw kay
      |  |  +--rw enable?               boolean
      |  |  +--rw actor
      |  |  |  +--rw priority?   uint8
      |  |  |  +--ro sci
      |  |  |     +--ro mac-address?   ieee:mac-address
      |  |  |     +--ro port-id?       portNumberType
      |  |  +--rw key-server
      |  |  |  +--rw priority?   uint8
      |  |  |  +--ro sci
      |  |  |     +--ro mac-address?   ieee:mac-address
```

```
|  |  |     +--ro port-id?       portNumberType
|  |  +--rw group
|  |  |  +--rw join?   boolean
|  |  |  +--rw form?   boolean
|  |  |  +--rw new?    boolean
|  |  +--rw macsec
|  |  |  +--rw capable?          boolean
|  |  |  +--rw desired?          boolean
|  |  |  +--ro protect?          boolean
|  |  |  +--ro validate?         boolean
|  |  |  +--ro replay-protect?   boolean
|  |  +--ro status
|  |  |  +--ro active?          boolean
|  |  |  +--ro authenticated?   boolean
|  |  |  +--ro secured?         boolean
|  |  |  +--ro failed?          boolean
|  |  +--ro key-stats
|  |  |  +--ro txKN?   keyNumberType
|  |  |  +--ro rxKN?   keyNumberType
|  |  +--ro association-stats
|  |  |  +--ro txAN?   associationNumberType
|  |  |  +--ro rxAN?   associationNumberType
|  |  +--rw suspend-on-request?   boolean
|  |  +--rw suspend-for?          uint8
|  |  +--rw suspend-while?        uint8
|  |  +--rw participant* [participants]
|  |     +--rw participants    uint16
|  |     +--rw cached?         boolean
|  |     +--rw active?         boolean
|  |     +--rw retain?         boolean
|  |     +--rw activate?       enumeration
|  |     +--ro peers
|  |     |  +--ro live?        sciListType
|  |     |  +--ro potential?   sciListType
|  |     +--ro ckn?            cknType
|  |     +--ro kmd?            kmdType
|  |     +--ro nid?            nidType
|  |     +--ro authData?       authType
|  |     +--ro principal?      boolean
|  |     +--ro distCKN?        cknType
|  +--rw logon-nid
|  |  +--ro connected?       nidType
|  |  +--ro requested?       nidType
|  |  +--rw selected         nidType
|  |  +--rw pae-nid-group* [nids]
|  |     +--rw nids                       sessionUserNameType
|  |     +--rw use-eap?                   enumeration
|  |     +--rw unauth-allowed?            enumeration
|  |     +--rw unsecure-allowed?          enumeration
|  |     +--rw unauthenticated-access?    enumeration
|  |     +--rw access-capabilities?       accessCapabilitiesType
|  |     +--ro nid-status
|  |        +--ro kmd?   kmdType
|  |        +--ro nid?   nidType
|  +--rw announcer
|  |  +--rw enable?          boolean
|  |  +--rw announce* [announces]
|  |  |  +--rw announces        uint16
|  |  |  +--ro nid?             nidType
|  |  |  +--ro access-status?   enumeration
|  |  +--rw pae-nid-group* [nids]
|  |     +--rw nids                       sessionUserNameType
```

```
|  |     +--rw use-eap?                  enumeration
|  |     +--rw unauth-allowed?           enumeration
|  |     +--rw unsecure-allowed?         enumeration
|  |     +--rw unauthenticated-access?   enumeration
|  |     +--rw access-capabilities?      accessCapabilitiesType
|  |     +--ro nid-status
|  |        +--ro kmd?   kmdType
|  |        +--ro nid?   nidType
|  +--rw listener
|  |  +--rw enable?         boolean
|  |  +--rw announcement* [announcements]
|  |     +--rw announcements             uint8
|  |     +--ro nid?                      nidType
|  |     +--ro kmd?                      kmdType
|  |     +--ro specific?                 boolean
|  |     +--ro access-status?            enumeration
|  |     +--ro requested-nid?            boolean
|  |     +--ro unauthenticated-access?   enumeration
|  |     +--ro access-capabilities?      accessCapabilitiesType
|  |     +--ro cipher-suites?            cipherSuitesType
|  +--ro eapol-statistics
|     +--ro invalid-eapol-frame-rx?       yang:counter32
|     +--ro eap-length-error-frames?      yang:counter32
|     +--ro eapol-announcements-rx?       yang:counter32
|     +--ro eapol-announce-reqs-rx?       yang:counter32
|     +--ro eapol-port-unavailable?       yang:counter32
|     +--ro eapol-start-frames-rx?        yang:counter32
|     +--ro eapol-eap-frames-rx?          yang:counter32
|     +--ro eapol-logoff-frames-rx?       yang:counter32
|     +--ro eapol-mk-no-cfn?              yang:counter32
|     +--ro eapol-mk-invalid-frames-rx?   yang:counter32
|     +--ro last-eapol-frame-source?      ieee:mac-address
|     +--ro last-eapol-frame-version?     yang:counter32
|     +--ro eapol-supp-eap-frames-tx?     yang:counter32
|     +--ro eapol-logoff-frames-tx?       yang:counter32
|     +--ro eapol-announcements-tx?       yang:counter32
|     +--ro eapol-announce-reqs-tx?       yang:counter32
|     +--ro eapol-start-frames-tx?        yang:counter32
|     +--ro eapol-auth-eap-frames-tx?     yang:counter32
|     +--ro eapol-mka-frames-tx?          yang:counter32
+--rw pae-nid-group* [nids]
   +--rw nids                       sessionUserNameType
   +--rw use-eap?                   enumeration
   +--rw unauth-allowed?            enumeration
   +--rw unsecure-allowed?          enumeration
   +--rw unauthenticated-access?    enumeration
   +--rw access-capabilities?       accessCapabilitiesType
   +--ro nid-status
      +--ro kmd?   kmdType
      +--ro nid?   nidType
```

2.3.2 YANG Data Module Definition

The (draft) YANG module definition is provided below:

```
module dot1x {

    namespace "ieee:ns:yang:ieee-port-dot1x";
    prefix "dot1x";

    import ieee-types {
```

```
      prefix ieee;
    }

    import ietf-yang-types {
  prefix yang;
}

organization
  "Institute of Electrical and Electronics Engineers";

contact
  "Web URL: http://ieee.org/
      E-mail:  corporate-communications@ieee.org
      Postal:
               U.S.A.
      Phone:   +1 732-563-6820
      Fax:     +1 732-981-9511";

    description
    "Port-based network access control allows a network administrator to
             restrict the use of IEEE 802 LAN service access points (ports) to
             secure communication between authenticated and authorized devices.
             IEEE Std 802.1X specifies an architecture, functional elements, and
             protocols that support mutual authentication between the clients of
             ports attached to the same LAN and secure communication between the
             ports.

             The following control allows a port to be reinitialized, terminating
             (and potentially restarting) authentication exchanges and MKA operation,
             based on a data model described in a set of YANG modules.";

revision "2015-03-10" {
  description
    "Initial Version.";
  reference
    "IEEE 802.1X-2014, PAE (Port Access Entity) management information model.";
}

    // Type definitions used by dot1X YANG module

    typedef nidType {
            type string {
                    // Need to confirm this.
                    length "0..100";
            }
    }

    typedef sessionUserNameType {
            description "utf8 definition";
            type string {
                    length "0..253";
            }
    }

    typedef sessionIdType {
            description "utf8 definition";
            type string {
                    length "3..253";
            }
    }

    typedef accessCapabilitiesType {
            type bits {
                    bit eap {
                            position 0;
                    }
                    bit mka {
                            position 1;
                    }
                    bit macsec {
```

```
                        position 2;
                }
                bit higher-layer {
                        position 3;
                }
                bit higher-layer-fallback {
                        position 4;
                }
                bit vendor-specific {
                        position 5;
                }
        }
}

typedef portNumberType {
        type uint16;
}

typedef keyNumberType {
        type uint32;
}

typedef associationNumberType {
        type uint16;
}

typedef sciListType {
        // To be defined.
        type uint16;
}

typedef cknType {
        // To be defined.
        type uint16;
}

typedef kmdType {
        // To be defined.
        type uint16;
}

typedef authType {
        // To be defined.
        type uint16;
}

typedef cipherSuitesType {
        // Structure to be defined.
        type uint16;
}

grouping nidGroup {
        list pae-nid-group {
                key "nids";
                leaf nids {
                        type sessionUserNameType;
                }
                leaf use-eap {
                        type enumeration {
                                enum never;
                                enum immediate;
                                enum mka-fail;
                        }
                        default "immediate";
                        description
                                "Determines when the Logon Process will initiate EAP,
                                 if the Supplicant and or Authenticator are enabled.";
                }
                leaf unauth-allowed {
                        type enumeration {
```

```
                                enum never;
                                enum immediate;
                                enum auth-fail;
                        }
                        default "immediate";
                        description
                                "Determines when the Logon Process will tell the CP
                                 state machine to provide unauthenticated connectivity.";
                }
                leaf unsecure-allowed {
                        type enumeration {
                                enum never;
                                enum immediate;
                                enum mka-fail;
                                enum mka-server;
                        }
                        default "immediate";
                        description
                                "Determines when the Logon Process will tell the CP
                                 state machine to provide authenticated but unsecured
                                 connectivity.";
                }
                leaf unauthenticated-access {
                        type enumeration {
                                enum no-access;
                                enum fallback-access;
                                enum limited-access;
                                enum open-access;
                        }
                        default "no-access";
                }
                leaf access-capabilities {
                        type accessCapabilitiesType;
                        description "Authentication mechanisms.";
                }
                container nid-status {
                        config false;
                        leaf kmd {
                                type kmdType;
                        }
                        leaf nid {
                                type nidType;
                        }
                }
        }
}

feature pacp-eap-supplicant {
        description "This feature indicates that the device supports
                a PACP EAP Supplicant.";
}
feature pacp-eap-authenticator {
        description "This feature indicates that the device supports
                a PACP EAP Authenticator.";
}
feature mka {
        description "This feature indicates that the device supports MKA";
}
feature macsec {
        description "This feature indicates that the device supports
                MACsec on the Controlled Port.";
}
feature announcements {
        description "This feature indicates that the device supports
                the ability to send EAPOL announcements.";
}
feature listener {
        description "This feature indicates that the device supports
                the ability to use receive EAPOL announcements.";
}
```

```
feature virtual-ports {
        description "This feature indicates that the device supports
                the virtual ports for a real port.";
}
feature in-service-upgrades {
        description "This feature indicates that the device supports
                MKA in-service upgrades.";
}

// Configuration objects used by 802.1X YANG module

// PAE object
container pae-system {
        leaf system-access-control {
                type enumeration {
                        enum disabled;
                        enum enabled;
                }
                description
                        "Setting this control to 'disabled' deletes any virtual ports
                         previously instantiated, and terminates authentication exchanges
                         and MKA operation. Each real port PAE behaves as if
                         enabledVirtualPorts was clear, the PAEs Supplicant, Authenticator,
                         and KaY as if their enabled controls were clear, and Logon
                         Process(es) as if unauthAllowed was Immediate. Announcements can
                         be transmitted (subject to other controls), both periodically
                         and in response to announcement requests (conveyed by EAPOL-Starts
                         or EAPOL-Announcement-Reqs) but are sent with a single NID Set,
                         with a null NID, and the Access Information TLV (and no other)
                         with an accessStatus of No Access, accessRequested false,
                         OpenAccess, and no accessCapabilities.

                         The control variable settings for each real port PAE are unaffected,
                         and will be used once systemAccessControl is set to 'enabled'";
        }
        leaf system-announcements {
                type enumeration {
                        enum disabled;
                        enum enabled;
                }
                description
                        "Setting this control to Disabled causes each PAE to behave as if
                         enabled were clear for the PAE's Announcement functionality. The
                         independent controls for each PAE apply if systemAnnouncements
                         is Enabled.";
        }
        leaf eapol-protocol-version {
                //config false;
                type uint32;
                        description "The EAPOL protocol version for this system.";
        }
        leaf mka-version {
                //config false;
                type uint32;
                description "The MKA protocol version for this system.";
        }

        // Port Authentication Entity (PAE) Object
        list pae {
                key "port-number";
                leaf port-number {
                        type portNumberType;
                        description "Each PAE is uniquely identified by a port number. The port
                                number used is unique amongst all port numbers for the system, and
                                directly or indirectly identifies the Uncontrolled Port that supports
                                the PAE. If the PAE has been dynamically instantiated to support an
                                existing or potential virtual port, this portNumber, the
                                uncontrolledPortNumber and the controlledPortNumber are allocated by
                                the real port's PAE, and this portNumber is the uncontrolledPortNumber.
                                If the PAE supports a real port, this portNumber is the commonPortNumber
```

```
                        for the associated PAC or SecY.";
                }
                leaf controlled-port-number {
                        config false;
                        type portNumberType;
                        description "The port number for the associated PAC or SecY's Controlled Port.";
                }
                leaf uncontrolled-port-number {
                        config false;
                        type portNumberType;
                        description "The port number for the associated PAC or SecY's Uncontrolled
Port.";
                }
                leaf common-port-number {
                        config false;
                        type portNumberType;
                        description "The port number for the associated PAC or SecY's Common Port.
                                All the virtual ports created for a given real port share the same
                                Common Port and commonPortNumber";
                }
                leaf port-type {
                        description "The port type of the PAE.";
                        config false;
                        type enumeration {
                                enum real-port;
                                enum virtual-port;
                        }
                }
                container virtual-port {
                        if-feature virtual-ports;
                        leaf enable {
                                when "port-type = real-port";
                                type boolean;
                                description
                                        "A real port's PAE may be configured to create virtual ports
                                         to support multi-access LANs provided that MKA and MACsec
                                         operation is enabled for that port.";
                        }
                        leaf max {
                                when "port-type = real-port";
                                config false;
                                type uint32;
                        }
                        leaf current {
                                when "port-type = real-port";
                                config false;
                                type yang:gauge32;
                        }
                        leaf start {
                                when "port-type = virtual-port";
                                config false;
                                type uint32;
                        }
                        leaf peer-address {
                                when "port-type = virtual-port";
                                config false;
                                type ieee:mac-address;
                        }
                }

                // SUPPLICANT Object
                container supplicant {
                        leaf held-period {
                                type uint16;
                                units seconds;
                                default "60";
                        }
                        leaf retry-max {
                                type uint8;
                                default "2";
```

```
                description
                        "Specifies the maximum number of re-authentication
                         attempts on an authenticator port before port is
                         unauthorized.";
        }
        container status {
                config false;
                leaf enabled {
                        type boolean;
                }
                leaf authenticate {
                        type boolean;
                }
                leaf authenticated {
                        type boolean;
                }
                leaf failed {
                        type boolean;
                }
        }
}

// AUTHENTICATOR Object
container authenticator {
        leaf quiet-period {
                type uint16;
                units seconds;
                default "60";
                description
                        "Number of seconds that the switch remains in the
                         quiet state following a failed authentication exchange
                         with the client";
                reference
                        "IEEE 802.1X Clause 8.6, Figure 12-3";
        }
        leaf reauth-period {
                type uint16;
                units seconds;
                default "3600";
                description
                        "This object indicates the time period of the
                         reauthentication to the supplicant.";
                reference
                        "IEEE 802.1X Clause 8.6, Figure 12-3";
        }
        leaf retry-max {
                type uint8;
                default "2";
                description
                        "Specifies the maximum number of re-authentication
                         attempts on an authenticator port before port is
                         unauthorized.";
        }
        container status {
                config false;
                leaf enabled {
                        type boolean;
                }
                leaf authenticate {
                        type boolean;
                }
                leaf authenticated {
                        type boolean;
                }
                leaf failed {
                        type boolean;
                }
        }
}
```

```
// PAE KaY (Group) Object
container kay {
        leaf enable {
                type boolean;
                default "true";
        }
        container actor {
                leaf priority {
                        type uint8;
                        default "0";
                }
                container sci {
                        config false;
                        leaf mac-address {
                                type ieee:mac-address;
                        }
                        leaf port-id {
                                type portNumberType;
                        }
                }
        }
        container key-server {
                leaf priority {
                        type uint8;
                        default "0";
                }
                container sci {
                        config false;
                        leaf mac-address {
                                type ieee:mac-address;
                        }
                        leaf port-id {
                                type portNumberType;
                        }
                }
        }
        container group {
                leaf join {
                        type boolean;
                }
                leaf form {
                        type boolean;
                }
                leaf new {
                        type boolean;
                }
        }
        container macsec {
                leaf capable {
                        type boolean;
                        default "false";
                }
                leaf desired {
                        type boolean;
                        default "false";
                }
                leaf protect {
                        config false;
                        type boolean;
                }
                leaf validate {
                        config false;
                        type boolean;
                }
                leaf replay-protect {
                        config false;
                        type boolean;
                }
        }
        container status {
```

```
                config false;
                leaf active {
                        type boolean;
                }
                leaf authenticated {
                        type boolean;
                }
                leaf secured {
                        type boolean;
                }
                leaf failed {
                        type boolean;
                }
        }
        container key-stats {
                config false;
                leaf txKN {
                        type keyNumberType;
                }
                leaf rxKN {
                        type keyNumberType;
                }
        }
        container association-stats {
                config false;
                leaf txAN {
                        type associationNumberType;
                }
                leaf rxAN {
                        type associationNumberType;
                }
        }
        leaf suspend-on-request {
                type boolean;
                default "false";
        }
        leaf suspend-for {
                type uint8;
                default "0";
        }
        leaf suspend-while {
                type uint8;
                default "0";
        }
        list participant {
                key "participants";
                leaf participants {
                        type uint16;
                }
                leaf cached {
                        type boolean;
                        default "false";
                }
                leaf active {
                        type boolean;
                        default "false";
                }
                leaf retain {
                        type boolean;
                        default "false";
                }
                leaf activate {
                        type enumeration {
                                enum principal;
                                enum livePeers;
                                enum potentialPeers;
                                enum distCKN;
                        }
                        default "principal";
                }
```

```
                container peers {
                        config false;
                        leaf live {
                                type sciListType;
                        }
                        leaf potential {
                                type sciListType;
                        }
                }
                leaf ckn {
                        config false;
                        type cknType;
                }
                leaf kmd {
                        config false;
                        type kmdType;
                }
                leaf nid {
                        config false;
                        type nidType;
                }
                leaf authData {
                        config false;
                        type authType;
                }
                leaf principal {
                        config false;
                        type boolean;
                }
                leaf distCKN {
                        config false;
                        type cknType;
                }
        }
}

// LOGIN-NID object
container logon-nid {
        leaf connected {
                config false;
                type nidType;
        }
        leaf requested {
                config false;
                type nidType;
        }
        leaf selected {
                type nidType;
                mandatory true;
        }
        uses nidGroup;
}

// ANNOUNCER Object
container announcer {
        leaf enable {
                type boolean;
                default "false";
        }
        list announce {
                key "announces";
                leaf announces {
                        type uint16;
                }

                leaf nid {
                        config false;
                        type nidType;
                }
                leaf access-status {
```

```
                        config false;
                        // Need to confirm these types.
                        type enumeration {
                                enum no-access;
                                enum acess-requested-fail;
                                enum open-access;
                                enum no-access-capabilities;
                        }
                }
        }
        uses nidGroup;
}

// LISTENER Object
container listener {
        leaf enable {
                type boolean;
                default "false";
        }
        list announcement {
                key "announcements";
                leaf announcements {
                        type uint8;
                }

                leaf nid {
                        config false;
                        type nidType;
                }
                leaf kmd {
                        config false;
                        type kmdType;
                }
                leaf specific {
                        config false;
                        type boolean;
                }
                leaf access-status {
                        config false;
                        // Need to confirm these types.
                        type enumeration {
                                enum no-access;
                                enum acess-requested-fail;
                                enum open-access;
                                enum no-access-capabilities;
                        }
                }
                leaf requested-nid {
                        config false;
                        type boolean;
                }
                leaf unauthenticated-access {
                        config false;
                        // Need to confirm these types.
                        type enumeration {
                                enum no-access;
                                enum acess-requested-fail;
                                enum open-access;
                                enum no-access-capabilities;
                        }
                }
                leaf access-capabilities {
                        config false;
                        type accessCapabilitiesType;
                }
                leaf cipher-suites {
                        config false;
                        type cipherSuitesType;
                }
        }
```

```
                }
                // EapolStatistics
                container eapol-statistics {
                        config false;
                        leaf invalid-eapol-frame-rx {
                                type yang:counter32;
                        }
                        leaf eap-length-error-frames {
                                type yang:counter32;
                        }
                        leaf eapol-announcements-rx {
                                type yang:counter32;
                        }
                        leaf eapol-announce-reqs-rx {
                                type yang:counter32;
                        }
                        leaf eapol-port-unavailable {
                                type yang:counter32;
                        }
                        leaf eapol-start-frames-rx {
                                type yang:counter32;
                        }
                        leaf eapol-eap-frames-rx {
                                type yang:counter32;
                        }
                        leaf eapol-logoff-frames-rx {
                                type yang:counter32;
                        }
                        leaf eapol-mk-no-cfn {
                                type yang:counter32;
                        }
                        leaf eapol-mk-invalid-frames-rx {
                                type yang:counter32;
                        }
                        leaf last-eapol-frame-source {
                                type ieee:mac-address;
                        }
                        leaf last-eapol-frame-version {
                                type yang:counter32;
                        }
                        leaf eapol-supp-eap-frames-tx {
                                type yang:counter32;
                        }
                        leaf eapol-logoff-frames-tx {
                                type yang:counter32;
                        }
                        leaf eapol-announcements-tx {
                                type yang:counter32;
                        }
                        leaf eapol-announce-reqs-tx {
                                type yang:counter32;
                        }
                        leaf eapol-start-frames-tx {
                                type yang:counter32;
                        }
                        leaf eapol-auth-eap-frames-tx {
                                type yang:counter32;
                        }
                        leaf eapol-mka-frames-tx {
                                type yang:counter32;
                        }
                }
        }
        uses nidGroup;
    }
}
```

2.4 802.1X YANG Module (Augmentation of IETF Interface Management YANG Data Model)

2.4.1 YANG Data Model Schema

The YANG data model schema is graphically represented below.

```
module: dot1Xv2
   +--rw pae-system
   |  +--rw system-access-control?    enumeration
   |  +--rw system-announcements?     enumeration
   |  +--rw eapol-protocol-version?   uint32
   |  +--rw mka-version?              uint32
   +--rw nid-group
      +--rw paeNidGroup* [nids]
         +--rw nids                      sessionUserNameType
         +--rw use-eap?                  enumeration
         +--rw unauth-allowed?           enumeration
         +--rw unsecure-allowed?         enumeration
         +--rw unauthenticated-access?   enumeration
         +--rw access-capabilities?      accessCapabilitiesType
augment /if:interfaces/if:interface:
   +--rw vp-enable?       boolean {virtual-ports}?
   +--rw port-type?       enumeration
   +--rw supplicant
   |  +--rw held-period?   uint16
   |  +--rw retry-max?     uint8
   +--rw authenticator
   |  +--rw quiet-period?    uint16
   |  +--rw reauth-period?   uint16
   |  +--rw retry-max?       uint8
   +--rw kay
   |  +--rw enable?               boolean
   |  +--rw actor
   |  |  +--rw priority?   uint8
   |  |  +--rw sci
   |  |     +--rw mac-address?   ieee:mac-address
   |  |     +--rw port-id?       portNumberType
   |  +--rw key-server
   |  |  +--rw priority?   uint8
   |  |  +--rw sci
   |  |     +--rw mac-address?   ieee:mac-address
   |  |     +--rw port-id?       portNumberType
   |  +--rw group
   |  |  +--rw join?   boolean
   |  |  +--rw form?   boolean
   |  |  +--rw new?    boolean
   |  +--rw macsec
   |  |  +--rw capable?   boolean
   |  |  +--rw desired?   boolean
   |  +--rw suspend-on-request?   boolean
   |  +--rw suspend-for?          uint8
   |  +--rw suspend-while?        uint8
   |  +--rw participant* [participants]
   |     +--rw participants    uint16
   |     +--rw cached?         boolean
   |     +--rw active?         boolean
   |     +--rw retain?         boolean
   |     +--rw activate?       enumeration
   |     +--rw peers
   |     |  +--rw live?        sciListType
   |     |  +--rw potential?   sciListType
   |     +--rw ckn?            cknType
```

```
|     +--rw kmd?            kmdType
|     +--rw nid?            nidType
|     +--rw auth-data?      authType
|     +--rw principal?      boolean
|     +--rw distCKN?        cknType
+--rw logon-nid
|  +--rw selected?      nidType
|  +--rw paeNidGroup* [nids]
|     +--rw nids                       sessionUserNameType
|     +--rw use-eap?                   enumeration
|     +--rw unauth-allowed?            enumeration
|     +--rw unsecure-allowed?          enumeration
|     +--rw unauthenticated-access?    enumeration
|     +--rw access-capabilities?       accessCapabilitiesType
+--rw announcer
|  +--rw enable?        boolean
|  +--rw announce* [announces]
|  |  +--rw announces    uint16
|  |  +--rw nid?         nidType
|  +--rw paeNidGroup* [nids]
|     +--rw nids                       sessionUserNameType
|     +--rw use-eap?                   enumeration
|     +--rw unauth-allowed?            enumeration
|     +--rw unsecure-allowed?          enumeration
|     +--rw unauthenticated-access?    enumeration
|     +--rw access-capabilities?       accessCapabilitiesType
+--rw listener
   +--rw enable?   boolean
augment /if:interfaces-state/if:interface:
   +--ro controlled-port-number?     portNumberType
   +--ro uncontrolled-port-number?   portNumberType
   +--ro common-port-number?         portNumberType
   +--ro virtual-port {virtual-ports}?
   |  +--ro max?            uint32
   |  +--ro current?        yang:gauge32
   |  +--ro start?          uint32
   |  +--ro peer-address?   ieee:mac-address
   +--ro supplicant
   |  +--ro enabled?         boolean
   |  +--ro authenticate?    boolean
   |  +--ro authenticated?   boolean
   |  +--ro failed?          boolean
   +--ro authenticator
   |  +--ro enabled?         boolean
   |  +--ro authenticate?    boolean
   |  +--ro authenticated?   boolean
   |  +--ro failed?          boolean
   +--ro kay
   |  +--ro actor
   |  |  +--ro sci
   |  |     +--ro mac-address?   ieee:mac-address
   |  |     +--ro port-id?       portNumberType
   |  +--ro key-server
   |  |  +--ro sci
   |  |     +--ro mac-address?   ieee:mac-address
   |  |     +--ro port-id?       portNumberType
   |  +--ro macsec
   |  |  +--ro protect?          boolean
   |  |  +--ro validate?         boolean
   |  |  +--ro replay-protect?   boolean
   |  +--ro active?          boolean
   |  +--ro authenticated?   boolean
```

```
|  +--ro secured?         boolean
|  +--ro failed?          boolean
|  +--ro txKN?            keyNumberType
|  +--ro rxKN?            keyNumberType
|  +--ro txAN?            associationNumberType
|  +--ro rxAN?            associationNumberType
|  +--ro participant* [participants]
|     +--ro participants    uint16
|     +--ro peers
|     |  +--ro live?        sciListType
|     |  +--ro potential?   sciListType
|     +--ro ckn?            cknType
|     +--ro kmd?            kmdType
|     +--ro nid?            nidType
|     +--ro auth-data?      authType
|     +--ro principal?      boolean
|     +--ro distCKN?        cknType
+--ro logon-nid
|  +--ro connected?     nidType
|  +--ro requested?     nidType
|  +--ro paeNidGroup* [nids]
|     +--ro nids                       sessionUserNameType
|     +--ro use-eap?                   enumeration
|     +--ro unauth-allowed?            enumeration
|     +--ro unsecure-allowed?          enumeration
|     +--ro unauthenticated-access?    enumeration
|     +--ro access-capabilities?       accessCapabilitiesType
+--ro announcer
|  +--ro announce* [announces]
|  |  +--ro announces        uint16
|  |  +--ro nid?             nidType
|  |  +--ro access-status?   enumeration
|  +--ro paeNidGroup* [nids]
|     +--ro nids    sessionUserNameType
|     +--ro kmd?    kmdType
|     +--ro nid?    nidType
+--ro listener
|  +--ro announcement* [announcements]
|     +--ro announcements             uint8
|     +--ro nid?                      nidType
|     +--ro kmd?                      kmdType
|     +--ro specific?                 boolean
|     +--ro access-status?            enumeration
|     +--ro requested-nid?            boolean
|     +--ro unauthenticated-access?   enumeration
|     +--ro access-capabilities?      accessCapabilitiesType
|     +--ro cipher-suites?            cipherSuitesType
+--ro eapol-statistics
   +--ro invalid-eapol-frame-rx?       yang:counter32
   +--ro eap-length-error-frames?      yang:counter32
   +--ro eapol-announcements-rx?       yang:counter32
   +--ro eapol-announce-reqs-rx?       yang:counter32
   +--ro eapol-port-unavailable?       yang:counter32
   +--ro eapol-start-frames-rx?        yang:counter32
   +--ro eapol-eap-frames-rx?          yang:counter32
   +--ro eapol-logoff-frames-rx?       yang:counter32
   +--ro eapol-mk-no-cfn?              yang:counter32
   +--ro eapol-mk-invalid-frames-rx?   yang:counter32
   +--ro last-eapol-frame-source?      ieee:mac-address
   +--ro last-eapol-frame-version?     yang:counter32
   +--ro eapol-supp-eap-frames-tx?     yang:counter32
   +--ro eapol-logoff-frames-tx?       yang:counter32
```

```
        +--ro eapol-announcements-tx?       yang:counter32
        +--ro eapol-announce-reqs-tx?       yang:counter32
        +--ro eapol-start-frames-tx?        yang:counter32
        +--ro eapol-auth-eap-frames-tx?     yang:counter32
        +--ro eapol-mka-frames-tx?          yang:counter32
```

2.4.2 YANG Data Module Definition

The (draft) YANG module definition is provided below:

```
module dot1Xv2 {

    namespace "ieee:ns:yang:ieee-port-dot1x";
    prefix "dot1x";

    import ieee-types {
          prefix ieee;
        }
        import ietf-yang-types {
      prefix yang;
    }
        import ietf-interfaces {
                prefix if;
        }

    organization
      "Institute of Electrical and Electronics Engineers";

    contact
      "Web URL: http://ieee.org/
          E-mail:  corporate-communications@ieee.org
          Postal:
                   U.S.A.
          Phone:   +1 732-563-6820
          Fax:     +1 732-981-9511";

        description
        "Port-based network access control allows a network administrator to
                 restrict the use of IEEE 802 LAN service access points (ports) to
                 secure communication between authenticated and authorized devices.
                 IEEE Std 802.1X specifies an architecture, functional elements, and
                 protocols that support mutual authentication between the clients of
                 ports attached to the same LAN and secure communication between the
                 ports.

                 The following control allows a port to be reinitialized, terminating
                 (and potentially restarting) authentication exchanges and MKA operation,
                 based on a data model described in a set of YANG modules.";

    revision "2015-03-10" {
      description
        "Initial Version.";
      reference
        "IEEE 802.1X-2014, PAE (Port Access Entity) management information model.";
    }

        /*
     * Type definitions used by dot1X YANG module
     */

        typedef nidType {
                type string {
                        // Need to confirm this.
                        length "0..100";
                }
        }

        typedef sessionUserNameType {
```

```
        description "utf8 definition";
        type string {
                length "0..253";
        }
}

typedef sessionIdType {
        description "utf8 definition";
        type string {
                length "3..253";
        }
}

typedef accessCapabilitiesType {
        type bits {
                bit eap {
                        position 0;
                }
                bit mka {
                        position 1;
                }
                bit macsec {
                        position 2;
                }
                bit higher-layer {
                        position 3;
                }
                bit higher-layer-fallback {
                        position 4;
                }
                bit vendor-specific {
                        position 5;
                }
        }
}

typedef portNumberType {
        type uint16;
}

typedef keyNumberType {
        type uint32;
}

typedef associationNumberType {
        type uint16;
}

typedef sciListType {
        // To be defined.
        type uint16;
}

typedef cknType {
        // To be defined.
        type uint16;
}

typedef kmdType {
        // To be defined.
        type uint16;
}

typedef authType {
        // To be defined.
        type uint16;
}

typedef cipherSuitesType {
        // Structure to be defined.
```

```
        type uint16;
    }

    grouping nidGroup {
        list paeNidGroup {
            key "nids";
            leaf nids {
                type sessionUserNameType;
            }
            leaf use-eap {
                type enumeration {
                    enum never;
                    enum immediate;
                    enum mka-fail;
                }
                default "immediate";
                description
                    "Determines when the Logon Process will initiate EAP,
                     if the Supplicant and or Authenticator are enabled.";
            }
            leaf unauth-allowed {
                type enumeration {
                    enum never;
                    enum immediate;
                    enum auth-fail;
                }
                default "immediate";
                description
                    "Determines when the Logon Process will tell the CP
                     state machine to provide unauthenticated connectivity.";
            }
            leaf unsecure-allowed {
                type enumeration {
                    enum never;
                    enum immediate;
                    enum mka-fail;
                    enum mka-server;
                }
                default "immediate";
                description
                    "Determines when the Logon Process will tell the CP
                     state machine to provide authenticated but unsecured
                     connectivity.";
            }
            leaf unauthenticated-access {
                type enumeration {
                    enum no-access;
                    enum fallback-access;
                    enum limited-access;
                    enum open-access;
                }
                default "no-access";
            }
            leaf access-capabilities {
                type accessCapabilitiesType;
                description "Authentication mechanisms.";
            }
        }
    }

    grouping nidGroupState {
        list paeNidGroup {
            key "nids";
            leaf nids {
                type sessionUserNameType;
            }
            leaf kmd {
                type kmdType;
            }
            leaf nid {
```

```
                        type nidType;
                }
        }
}

feature pacp-eap-supplicant {
        description "This feature indicates that the device supports
                a PACP EAP Supplicant.";
}
feature pacp-eap-authenticator {
        description "This feature indicates that the device supports
                a PACP EAP Authenticator.";
}
feature mka {
        description "This feature indicates that the device supports MKA";
}
feature macsec {
        description "This feature indicates that the device supports
                MACsec on the Controlled Port.";
}
feature announcements {
        description "This feature indicates that the device supports
                the ability to send EAPOL announcements.";
}
feature listener {
        description "This feature indicates that the device supports
                the ability to use receive EAPOL announcements.";
}
feature virtual-ports {
        description "This feature indicates that the device supports
                the virtual ports for a real port.";
}
feature in-service-upgrades {
        description "This feature indicates that the device supports
                MKA in-service upgrades.";
}

// Configuration objects used by 802.1X YANG module

// PAE object
container pae-system {
        leaf system-access-control {
                type enumeration {
                        enum disabled;
                        enum enabled;
                }
                description
                        "Setting this control to 'disabled' deletes any virtual ports
                         previously instantiated, and terminates authentication exchanges
                         and MKA operation. Each real port PAE behaves as if
                         enabledVirtualPorts was clear, the PAEs Supplicant, Authenticator,
                         and KaY as if their enabled controls were clear, and Logon
                         Process(es) as if unauthAllowed was Immediate. Announcements can
                         be transmitted (subject to other controls), both periodically
                         and in response to announcement requests (conveyed by EAPOL-Starts
                         or EAPOL-Announcement-Reqs) but are sent with a single NID Set,
                         with a null NID, and the Access Information TLV (and no other)
                         with an accessStatus of No Access, accessRequested false,
                         OpenAccess, and no accessCapabilities.

                         The control variable settings for each real port PAE are unaffected,
                         and will be used once systemAccessControl is set to 'enabled'";
        }
        leaf system-announcements {
                type enumeration {
                        enum disabled;
                        enum enabled;
                }
                description
                        "Setting this control to Disabled causes each PAE to behave as if
```

```
                        enabled were clear for the PAE's Announcement functionality. The
                        independent controls for each PAE apply if systemAnnouncements
                        is Enabled.";
                }
                leaf eapol-protocol-version {
                        type uint32;
                        description "The EAPOL protocol version for this system.";
                }
                leaf mka-version {
                        type uint32;
                        description "The MKA protocol version for this system.";
                }
        }

        /*
         *  Port Authentication Entity (PAE) Object
         */
        augment "/if:interfaces/if:interface" {
        //when "if:type = 'ianaif:macSecControlledIF'";
                leaf vp-enable {
                        if-feature virtual-ports;
                        type boolean;
                        description
                                "A real port's PAE may be configured to create virtual ports
                                 to support mlti-access LANs provided that MKA and MACsec
                                 operation is enabled for that port.";
                }
                leaf port-type {
                        description "The port type of the PAE.";
                        type enumeration {
                                enum real-port;
                                enum virtual-port;
                        }
                }
                /*
                 * Suppplicant only applicable to RealPort types.
                 */
                container supplicant {
                        leaf held-period {
                                type uint16;
                                units seconds;
                                default "60";
                        }
                        leaf retry-max {
                                type uint8;
                                default "2";
                                description
                                        "Specifies the maximum number of re-authentication
                                         attempts on an authenticator port before port is
                                         unauthorized.";
                        }
                }
                /*
                 * Authenticator
                 */
                container authenticator {
                        leaf quiet-period {
                                type uint16;
                                units seconds;
                                default "60";
                                description
                                        "Number of seconds that the switch remains in the
                                         quiet state following a failed authentication exchange
                                         with the client";
                                reference
                                        "IEEE 802.1X Clause 8.6, Figure 12-3";
                        }
                        leaf reauth-period {
                                type uint16;
                                units seconds;
```

```
                default "3600";
                description
                        "This object indicates the time period of the
                         reauthentication to the supplicant.";
                reference
                        "IEEE 802.1X Clause 8.6, Figure 12-3";
        }
        leaf retry-max {
                type uint8;
                default "2";
                description
                        "Specifies the maximum number of re-authentication
                         attempts on an authenticator port before port is
                         unauthorized.";
        }
}
/*
 * MKA
 */
container kay {
        leaf enable {
                type boolean;
                default "true";
        }
        container actor {
                leaf priority {
                        type uint8;
                        default "0";
                }
                container sci {
                        leaf mac-address {
                                type ieee:mac-address;
                        }
                        leaf port-id {
                                type portNumberType;
                        }
                }
        }
        container key-server {
                leaf priority {
                        type uint8;
                        default "0";
                }
                container sci {
                        leaf mac-address {
                                type ieee:mac-address;
                        }
                        leaf port-id {
                                type portNumberType;
                        }
                }
        }
        container group {
                leaf join {
                        type boolean;
                }
                leaf form {
                        type boolean;
                }
                leaf new {
                        type boolean;
                }
        }
        container macsec {
                leaf capable {
                        type boolean;
                        default "false";
                }
                leaf desired {
                        type boolean;
```

```
                default "false";
        }
}
leaf suspend-on-request {
        type boolean;
        default "false";
}
leaf suspend-for {
        type uint8;
        default "0";
}
leaf suspend-while {
        type uint8;
        default "0";
}
/*
 * Participants
 */
list participant {
        key "participants";
        leaf participants {
                type uint16;
        }
        leaf cached {
                type boolean;
                default "false";
        }
        leaf active {
                type boolean;
                default "false";
        }
        leaf retain {
                type boolean;
                default "false";
        }
        leaf activate {
                type enumeration {
                        enum principal;
                        enum livePeers;
                        enum potential-peers;
                        enum distCKN;
                }
                default "principal";
        }
        container peers {
                leaf live {
                        type sciListType;
                }
                leaf potential {
                        type sciListType;
                }
        }
        leaf ckn {
                type cknType;
        }
        leaf kmd {
                type kmdType;
        }
        leaf nid {
                type nidType;
        }
        leaf auth-data {
                type authType;
        }
        leaf principal {
                type boolean;
        }
        leaf distCKN {
                type cknType;
        }
```

```
                }
        }
        /*
         * Logon NIDs
         */
        container logon-nid {
                leaf selected {
                        type nidType;
                }
                uses nidGroup;
        }
        /*
         * Announcer
         */
        container announcer {
                leaf enable {
                        type boolean;
                        default "false";
                }
                list announce {
                        key "announces";
                        leaf announces {
                                type uint16;
                        }

                        leaf nid {
                                type nidType;
                        }
                }
                uses nidGroup;
        }
        /*
         * Listener
         */
        container listener {
                leaf enable {
                        type boolean;
                        default "false";
                }
        }
}

augment "/if:interfaces-state/if:interface" {
//when "if:type = 'ianaift:macSecControlledIF'";
        leaf controlled-port-number {
                config false;
                type portNumberType;
        }
        leaf uncontrolled-port-number {
                config false;
                type portNumberType;
        }
        leaf common-port-number {
                type portNumberType;
        }
        container virtual-port {
                if-feature virtual-ports;
                leaf max {
                        type uint32;
                }
                leaf current {
                        type yang:gauge32;
                }
                leaf start {
                        type uint32;
                }
                leaf peer-address {
                        type ieee:mac-address;
                }
        }
```

```
/*
 * Suppplicant only applicable to RealPort types.
 */
container supplicant {
        leaf enabled {
                type boolean;
        }
        leaf authenticate {
                type boolean;
        }
        leaf authenticated {
                type boolean;
        }
        leaf failed {
                type boolean;
        }
}
/*
 * Authenticator
 */
container authenticator {
        leaf enabled {
                type boolean;
        }
        leaf authenticate {
                type boolean;
        }
        leaf authenticated {
                type boolean;
        }
        leaf failed {
                type boolean;
        }
}
/*
 * MKA
 */
container kay {
        container actor {
                container sci {
                        leaf mac-address {
                                type ieee:mac-address;
                        }
                        leaf port-id {
                                type portNumberType;
                        }
                }
        }
        container key-server {
                container sci {
                        leaf mac-address {
                                type ieee:mac-address;
                        }
                        leaf port-id {
                                type portNumberType;
                        }
                }
        }
        container macsec {
                leaf protect {
                        type boolean;
                }
                leaf validate {
                        type boolean;
                }
                leaf replay-protect {
                        type boolean;
                }
        }
        leaf active {
```

```
                type boolean;
        }
        leaf authenticated {
                type boolean;
        }
        leaf secured {
                type boolean;
        }
        leaf failed {
                type boolean;
        }
        leaf txKN {
                type keyNumberType;
        }
        leaf rxKN {
                type keyNumberType;
        }
        leaf txAN {
                type associationNumberType;
        }
        leaf rxAN {
                type associationNumberType;
        }
        /*
         * Participants
         */
        list participant {
                key "participants";
                leaf participants {
                        type uint16;
                }
                container peers {
                        leaf live {
                                type sciListType;
                        }
                        leaf potential {
                                type sciListType;
                        }
                }
                leaf ckn {
                        type cknType;
                }
                leaf kmd {
                        type kmdType;
                }
                leaf nid {
                        type nidType;
                }
                leaf auth-data {
                        type authType;
                }
                leaf principal {
                        type boolean;
                }
                leaf distCKN {
                        type cknType;
                }
        }
}
/*
 * Logon NIDs
 */
container logon-nid {
        leaf connected {
                type nidType;
        }
        leaf requested {
                type nidType;
        }
        uses nidGroup;
```

```
}
/*
 * Announcer
 */
container announcer {
        list announce {
                key "announces";
                leaf announces {
                        type uint16;
                }

                leaf nid {
                        type nidType;
                }
                leaf access-status {
                        // Need to confirm these types.
                        type enumeration {
                                enum no-access;
                                enum acess-requested-fail;
                                enum open-access;
                                enum no-access-capabilities;
                        }
                }
        }
        uses nidGroupState;
}
/*
 * Listener
 */
container listener {
        list announcement {
                key "announcements";
                leaf announcements {
                        type uint8;
                }

                leaf nid {
                        type nidType;
                }
                leaf kmd {
                        type kmdType;
                }
                leaf specific {
                        type boolean;
                }
                leaf access-status {
                        // Need to confirm these types.
                        type enumeration {
                                enum no-access;
                                enum acess-requested-fail;
                                enum open-access;
                                enum no-access-capabilities;
                        }
                }
                leaf requested-nid {
                        type boolean;
                }
                leaf unauthenticated-access {
                        // Need to confirm these types.
                        type enumeration {
                                enum no-access;
                                enum acess-requested-fail;
                                enum open-access;
                                enum no-access-capabilities;
                        }
                }
                leaf access-capabilities {
                        type accessCapabilitiesType;
                }
                leaf cipher-suites {
```

```
                        type cipherSuitesType;
                    }
                }
            }
            /*
             *       EapolStatistics
             */
            container eapol-statistics {
                leaf invalid-eapol-frame-rx {
                    type yang:counter32;
                }
                leaf eap-length-error-frames {
                    type yang:counter32;
                }
                leaf eapol-announcements-rx {
                    type yang:counter32;
                }
                leaf eapol-announce-reqs-rx {
                    type yang:counter32;
                }
                leaf eapol-port-unavailable {
                    type yang:counter32;
                }
                leaf eapol-start-frames-rx {
                    type yang:counter32;
                }
                leaf eapol-eap-frames-rx {
                    type yang:counter32;
                }
                leaf eapol-logoff-frames-rx {
                    type yang:counter32;
                }
                leaf eapol-mk-no-cfn {
                    type yang:counter32;
                }
                leaf eapol-mk-invalid-frames-rx {
                    type yang:counter32;
                }
                leaf last-eapol-frame-source {
                    type ieee:mac-address;
                }
                leaf last-eapol-frame-version {
                    type yang:counter32;
                }
                leaf eapol-supp-eap-frames-tx {
                    type yang:counter32;
                }
                leaf eapol-logoff-frames-tx {
                    type yang:counter32;
                }
                leaf eapol-announcements-tx {
                    type yang:counter32;
                }
                leaf eapol-announce-reqs-tx {
                    type yang:counter32;
                }
                leaf eapol-start-frames-tx {
                    type yang:counter32;
                }
                leaf eapol-auth-eap-frames-tx {
                    type yang:counter32;
                }
                leaf eapol-mka-frames-tx {
                    type yang:counter32;
                }
            }
        }
        container nid-group {
            uses nidGroup;
        }
```

```
}
```

3 Data Model Examples

This section presents examples of configuring 802.1X associated with a bridge port.

Editor's note: Work in progress. Further work needed to rationalize.

3.1 Enabling access controls and Announcements

The following configuration example shows how to enable access controls and announcements on the system.

```
<dot1x xmlns="ieee:ns:yang:ieee-port-dot1x">
    <PAE-system>
           <systemAccessControl>enabled</systemAccessControl>
           <systemAnnouncements>enabled</systemAnnouncements>
    </PAE-system >
</dot1x>
```

3.2 Configuring a Physical Port

The following configuration example shows how to configure a (physical) port on the system …

```
<dot1x xmlns="ieee:ns:yang:ieee-port-dot1x">
    <PAE-system>
           <PAE>
                  <portNumber>1</portNumber>
                  <vpEnabled>false</vpEnabled>
           </PAE>
           <supplicant>
                  <supp>1</supp>
                  <heldPeriod>30</heldPeriod>
                  <retryMax>3</retryMax>
           </supplicant>
           <authenticator>
                  <auth>5</auth>
                  <quietPeriod>60</quietPeriod>
                  <reauthPeriod>3600</reauthPeriod>
                  <retryMax>3</retryMax>
           </authenticator>
           <kay>
                  <mka>222</mka>
                  <enable>true</enable>
                  <actor>
                         <priority>7</priority>
                  </actor>
                  <keyServer>
                         <priority>7</priority>
                  </keyServer>
                  <group>
                         <join>true</join>
                         <form>false</form>
                          <new>true</new>
                  </group>
                  <macsec>
                          <capable>true</capable>
```

```
                <desired>true</desired>
                <new>true</new>
            </macsec>
            <suspendOnRequest>true</suspendOnRequest>
            <suspendFor>5</suspendFor>
            <suspendWhile>3</suspendWhile>
            <participant>
                <participants>30<participants>
                <cached>true</participants>
                <active>true</active>
                <retain>true</retain>
                <activate>live-peers</activate>
            </participant>
            <participant>
                <participants>40<participants>
                <cached>true</participants>
                <active>true</active>
                <retain>true</retain>
                <activate>live-peers</activate>
            </participant>
        </kay>
        <logonNid>
            <selected>44</selected>
        </logonNid>
        <announcer>
            <announcers>55</announcers>
            <enable>true</enable>
            <announce>
                <announces>15</announces>
                <paeNidGroup>
                    <nids>sessionName</nids>
                    <useEap>immediate</useEap>
                    <unauthAllowed>immediate</unauthAllowed>
                    <unsecureAllowed>never</unsecureAllowed>
                    <unauthenticatedAccess>no-access</unauthenticatedAccess>
                    <accessCapabilities>X</accessCapabilities>
                </paeNidGroup>
            </announce>
        </announcer>
        <listener>
            <listeners>12</listeners>
            <enable>true</enable>
            <announcement>
                <announcements>3</announcements>
                <paeNidGroup>
                    <nids>sessionName</nids>
                    <useEap>immediate</useEap>
                    <unauthAllowed>immediate</unauthAllowed>
                    <unsecureAllowed>never</unsecureAllowed>
                    <unauthenticatedAccess>no-access</unauthenticatedAccess>
                    <accessCapabilities>X</accessCapabilities>
                </paeNidGroup>
            </announcement>
        </listener>
    </PAE-system >
</dot1x>
```

3.3 Retrieval of configuration data

The following configuration example shows how to retrieve configuration data associated with the PAE-

System …

```
<rpc message-id=”101” xmlns="ieee:ns:yang:ieee-port-dot1x”>
    <get-config>
            <source>
                    <running/>
            </source>
            <filter type=”subtree”>
                    <top xmlns=”http://.../config”>
                            <PAE-System/>
                    </top>
            </filter>
    </get-config>
</rpc>

<rpc-reply message-id=”101” xmlns="ieee:ns:yang:ieee-port-dot1x”>
    <data>
            <top xmlns=”http://.../config”>
                    <PAE-System>
                        <systemAccessControl>enabled</systemAccessControl>
                        <systemAnnouncements>enabled</systemAnnouncements>
                        <eapolProtocolVersion>2</eapolProtocolVersion>
                        <mkaVersion>0</mkaVersion>
                    </PAE-System>
            </top>
    </data>
</rpc-reply>
```

To be completed.

4 Security Considerations

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

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

5 Outstanding Specification Areas

The following items are outstanding and need to be addressed, moving forward:

1. Should the YANG configuration model be based on an augmentation of the (existing) YANG interface management YANG data model [RFC7223] or based upon the identification of a port that may participate in 802.1X configuration?
2. General rationalization of the YANG model structure.
3. Completion of the YANG 802.1X type definitions associated with NID type, SCI list type, CKN type, KMD type, Auth type, cipher suites type, etc.

4. Should we consider deviation from the PAE information model (or current MIB structure), in support of the YANG configuration model specifications?
5. Where are entities such as {eapolProtocolVersion, mkaVersion} configured in the overall YANG module?
6. Confirm default settings for configurable parameters.
7. Confirm mandatory configurable parameters.
8. Complete example data module configuration examples.

To be completed.

This page intentionally left (mostly) blank.