Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: August 15, 2008

Yang Shi, Ed.
H3C Tech. Co., Ltd
Perkins, Ed.
SNMPinfo
Chris, Ed.
Cisco Systems, Inc.
February 12, 2008

CAPWAP Protocol Base MIB
draft-yangshi-capwap-base-mib-02

Status of This Memo

Copyright Notice

Abstract

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols. In particular, it describes managed objects for modeling the Control And Provisioning of Wireless Access Points (CAPWAP) Protocol.

Table of Contents

1. Introduction

Current work is under way in the IETF to specify the CAPWAP Protocol [I-D.ietf-capwap-protocol-specification], which enables an Access Controller (AC) to manage a collection of Wireless Termination Points (WTPs)

This document defines a MIB module that can be used to manage CAPWAP implementations. This MIB module covers both configuration and WTP status-monitoring aspects of CAPWAP, and provides a way to reuse current MIB standards and future extensions for any wireless binding technology.

2. The Internet-Standard Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

3. Terminology

This document uses terminology from the document describing the CAPWAP Protocol specification [I-D.ietf-capwap-protocol-specification]. WTPs are viewed as remote RF interfaces controlled by the AC via CAPWAP protocol. The CAPWAP protocol supports two modes of operation: Split and Local MAC. In Split MAC mode all L2 wireless data and management frames are encapsulated via the CAPWAP protocol and exchanged between the AC and the WTP. The Local MAC mode of operation allows for the data frames to be either locally bridged, or tunneled as 802.3 frames. From AC, operator could centrally control WTPs configuration and monitor their status. CAPWAP use DTLS protocol to implement control channel security.

4. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

5. Overview

The CAPWAP Protocol MIB module (CAPWAP-MIB) have the following design objectives:

- To work well under centralized architectures, and provide a way to centrally manage and control wireless network by SNMP

- To be consistent with CAPWAP protocol

- To reuse current MIB standards and future extensions for a wireless binding technology

- To enable interoperability between vendors

- To meet operator requirements for centralized architectures

The basic idea of CAPWAP-MIB is:

- The SNMP agent run on the AC side, and it MAY not be required on the WTP side. It follows same idea as CAPWAP protocol: Centralized Control

- As a generic mechanism, it is independent of any wireless binding technologies and defined by a independent MIB file

- To be independent of any wireless binding technologies and have ability to reuse MIB standards of other SDOs, is the main challenge for design of the MIB

- The ifIndex [RFC2863] will play a role in bridging between MIB standards defined by different SDOs

- The operator could manage and control the centralized wireless architectures using multiple MIB standards defined by multiple SDOs, while keeping them loosely coupled

It is designed to satisfy the following requirements and constraints:

- From AC to centrally manage and monitor WTPs

- The MIB module supports CAPWAP protocol parameters queries

- The MIB module supports showing WTPs current state

- The MIB module provides the information of AC, WTPs, radio and station objects' basic property and their relationship

- The MIB module supports indicating the "WTP Virtual Radio Interface" and PHY radio's mapping relationship

- The counters are provided for WTP, radio's reboot event, hardware event failure and so on

- The MIB module provides the various notification like channel up, join failure and so on

Before coming to details of CAPWAP-MIB module, it will introduce how CAPWAP-MIB is able to be independent of any wireless binding technologies and reuse MIB standards of other SDOs. As centralized Wireless architecture, the operator has to oprepare configuration at AC side before WTPs connects to AC. For any wireless binding technology, the configuration and management of radio is very important. Under centralized Wireless architecture, according to [I-D.ietf-capwap-protocol-specification], a specific PHY radio could be identified by identifer of a WTP and radio (WTP id + radio id). As usual, the standard of a binding technology provides MIB standard for radio management on its own. For example, according to IEEE 802.11 WG MIB standards, the MIB tables such as Dot11OperationTable are able to support WTP radio configuration. These tables use ifIndex as the index, and work well under standalone Wireless architecture.

To reuse MIB objects (defined by SDOs such as IEEE) for radio is very import, and the key point is to reuse the idea of ifIndex. So it is required a way to maintain the mapping relationship between "WTP id + radio id" and "ifIndex". As a generic mechanism, ifIndex can identify an interface in abstract way, and it does NOT care for an interface's PHY location (either on WTP or AC). AC can have interfaces of "WTP Virtual Radio Interface" ifType, it will logically represents PHY radios on the WTPs side. It looks like that PHY radios are located on the AC side, and PHY location of WTP (radio) is hidden to the operator. Operator can operate radios by MIB tables (such as IEEE 802.11 WG's) with ifIndex of "WTP Virtual Radio Interface". As an Abstract interface, "WTP Virtual Radio Interface" could be used by any wireless binding technology such as IEEE 802.11 and 802.16. The table of capwapRadioBindTable will indicate the mapping relationship between "WTP id + Radio id" and IfIndex.

6. Structure of the MIB Module

6.1. Textual Conventions

The following textual conventions are defined:

CapwapWTPId ::= TEXTUAL-CONVENTION

```
        STATUS      current
        DESCRIPTION
            "Represents a unique identifier of a WTP instance.
             As usual, a serial number of WTP will be used."
        SYNTAX       OCTET STRING(SIZE(128))

    CapwapStationId ::= TEXTUAL-CONVENTION
        STATUS      current
        DESCRIPTION
            "Represents a unique identifier of a station instance.
             As usual, the MAC address of station will be used."
        SYNTAX      OCTET STRING (SIZE (6))

    CapwapRadioId ::= TEXTUAL-CONVENTION
        STATUS      current
        DESCRIPTION
            "Represents a unique identifier of a radio on a WTP."
        SYNTAX      Unsigned32 (1..4294967295)

    CapwapWTPTunnelMode ::= TEXTUAL-CONVENTION
        STATUS      current
        DESCRIPTION
            "Represents the tunneling mode for station data that are
             supported by the WTP.
             The possible value could be:
              localBridging(1) - Local Bridging Mode,
              dot3Tunnel(2)    - 802.3 Frame Tunnel Mode,
              nativeTunnel(3)  - Native Frame Tunnel Mode."
        REFERENCE
            "Section 4.6.41. of CAPWAP Protocol Specification, RFC xxx."
        SYNTAX      INTEGER { localBridging(1), dot3Tunnel(2),
                              nativeTunnel(3) }

    CapwapWTPMACType ::= TEXTUAL-CONVENTION
        STATUS      current
        DESCRIPTION
            "Represents the MAC mode of operation supported by the WTP.
             The possible value could be:
              localMAC(1) - Local-MAC Mode,
              splitMAC(2) - Split-MAC Mode."
        REFERENCE
            "Section 4.6.44. of CAPWAP Protocol Specification, RFC xxx."
        SYNTAX      INTEGER { localMAC(1), splitMAC(2) }

    CapwapChannelType::= TEXTUAL-CONVENTION
        STATUS     current
        DESCRIPTION
            "Represents the channel type for CAPWAP protocol.
```

```
            The following values are supported:
             data(1)    - data Channel
             control(2) - control Channel."
        SYNTAX    INTEGER { data(1), control(2) }

    CapwapWTPAuthenMethod ::= TEXTUAL-CONVENTION
        STATUS     current
        DESCRIPTION
            "Represents the authentication credential type
             for WTP.
             The following values are supported:
               clear(1) - clear text and no authentication,
               x509(2)  - X.509 Certificate Based,
               psk(3)   - Pre-Shared Secret,
               other(8) - Other method, for example, vendor specific.
             As mandatory requirement, CAPWAP control channel
             authentication should use DTLS, and either by certificate or
             PSK. For data channel, DTLS is optional."
        SYNTAX    INTEGER { clear(1), x509(2), psk(3), other(8) }
```

6.2. The capwapObjects Subtree

The subtree provides information for statistic data and configuration parameters of WTP and radio.

6.3. The capwapConformance Subtree

The subtree provides conformance information of MIB objects.

6.4. The capwapNotifications Subtree

The subtree describe the notifications defined in the MIB module, and their purpose.

6.5. Brief Description of MIB Objects

The MIB objects were derived from the CAPWAP protocol document [I-D.ietf- capwap-protocol-specification].

1) capwapWTPStateTable

The WTPs status table is used to indicate each WTP's CAPWAP FSM state.

2) capwapWTPTable

The WTPs table is used for providing property and configuration information in details for WTPs in running state.

3) capwapRadioBindTable

The radio bind table is used to indicate the mapping relationship between logical interface of "WTP Virtual Radio Interface" ifType and PHY radio.

4) capwapStationTable

The station table is used for providing stations' basic property information.

5) capwapWTPRebootStatsTable

The WTP reboot statistic table is used for collecting WTP reboot count, link failure count, hardware failure count and so on.

6) capwapRadioStatsTable

The WTP radio statistic table is used for collecting radio reset count, channel change count, hardware failure count and so on.

7. Relationship to Other MIB Modules

7.1. Relationship to the SNMPv2-MIB

The 'system' group in the SNMPv2-MIB [RFC3418] is defined as being mandatory for all systems, and the objects apply to the entity as a whole. The 'system' group provides identification of the management entity and certain other system-wide data. The CAPWAP-MIB does not duplicate those objects.

7.2. Relationship to the IF-MIB

The Interfaces Group [RFC2863] defines generic managed objects for managing interfaces. This memo contains the media-specific extensions to the Interfaces Group for managing WTP radio object that are modeled as interfaces.

IF-MIB is required to support on the AC side. For each PHY radio at WTP side, it will have a logical interface of 'WTP Virtual Radio Interface' ifType at AC side. The ifIndex of interface will represent PHY radio by logical. The interface SHOULD be modeled as an ifEntry and provide appropriate interface information.

Some specific interpretations of ifTable for CAPWAP-MIB are as follow.

Object Use for the CAPWAP MIB.

ifIndex	Each interface of 'WTP Virtual Radio Interface' type maybe be represented by an ifEntry.
ifDescr	Description of the interface of 'WTP Virtual Radio Interface' ifType.
ifType	IANAifType of 'WTP Virtual Radio Interface'.
ifName	Textual name (unique on this system) of the interface or an octet string of zero length.
ifAlias	The nonvolatile 'alias' name for the interface, as specified by a network manager.
ifPhysAddress	The physical address of the interface; i.e., BSSID of a 802.11 radio.
ifAdminStatus	This variable indicates the operator's intent as to whether PHY should be enabled, disabled, or running in some diagnostic testing mode on this interface. Also see [RFC2863].
ifOperStatus	This value reflects the actual or operational status of radio. Also see [RFC2863].
ifLastChange	The value of sysUpTime at the time the interface entered its current operational state. Also see [RFC2863].
ifInOctets	The number of received octets over the interface; i.e., the number of octets received as 802.11 frames.
ifOutOctets	The number of transmitted octets over the interface; i.e., the number of octets transmitted as 802.11 frames.
ifInErrors	The number of frames dropped due to uncorrectable errors.
ifInUnknownProtos	The number of received frame discarded during frame header validation, including frames with unrecognized label values.
ifOutErrors	See [RFC2863].

7.3. Relationship to the ENTITY-MIB

The ENTITY-MIB [RFC4133] meets need for a standardized way of representing a single agent, which supports multiple instances of one MIB. It could express a certain relationship between multiple entities, and provide entity properties for each entity.

Under the wireless centralized architectures, the SNMP agent will run on the AC side, and not required on the WTP side. By the ENTITY-MIB on the AC side, it could keep entity information of AC and WTPs. From the ENTITY-MIB perspective, the overall physical entity (AC) is a "compound" of multiple physical entities (WTPs which connects to AC), all entities are identified by Physical index. In the capwapWTPTable of CAPWAP-MIB, it uses capwapWTPPHYIndex object to keep the mapping relationship of WTP object between CAPWAP-MIB and ENTITY-MIB.

7.4. Relationship to MIB standards of other SDOs

The MIB standards (such as IEEE 802.11 MIB) of a wireless binding is required to support on the AC side. The CAPWAP-MIB module is able to support any wireless binding technology. Through ifIndex of 'WTP Virtual Radio Interface' ifType, it provides consistent and abstract way of reusing MIB objects of a wireless binding technology.

7.5. MIB modules required for IMPORTS

The following MIB module IMPORTS objects from SNMPv2-SMI [RFC2578], SNMPv2-TC [RFC2579], SNMPv2-CONF [RFC2580], IF-MIB [RFC2863], INET-ADDRESS-MIB [RFC4001] and ENTITY-MIB [RFC4133].

8. Example of CAPWAP-MIB Usage

With the idea of "WTP Virtual Radio Interface" in the mind, the usage of MIB will be easily understood. Here takes IEEE 802.11 binding technology as an example.

1) Identify each PHY radio by "WTP Virtual Radio Interface"
According to [I-D.ietf-capwap-protocol-specification], each radio on a WTP will be identified by a radio Id. Each WTP could be identified by its serial number.
When configuration for a WTP is prepared before a WTP connects to AC, the following information is available in the CapwapRadioBindTable.
In CapwapRadioBindTable
{

```
     capwapWTPId                    = 12345678
     capwapRadioId                  = 1
     capwapWTPVirtualRadioifIndex   = 10,
```

```
      capwapWirelessBinding          = dot11(2),
   }
   Suppse WTP's serial number is 12345678, and first radio id is 1.
   At AC side, the ifIndex of interface in "WTP Virtual Radio Interface"
   ifType is 10 which represents the PHY radio 1.
   By the mechanism of "WTP Virtual Radio Interface", it seemed that WTP
   PHY radios are located at AC side.
   The interface of "WTP Virtual Radio Interface" ifType is modeled by
   ifTable.
     In ifTable
   {
        ifIndex             = 10,
        ifDescr             = "WTP Virtual Radio Interface",
        ifType              = IANAifType of "WTP Virtual
                              Radio Interface",
        ifMtu               = 0,
        ifSpeed             = 0,
        ifPhysAddress       = 0.0.0.0.0.0,
        ifAdminStatus       = true,
        ifOperStatus        = false,
        ifLastChange        = 0,
        ifInOctets          = 0,
        ifInUcastPkts       = 0,
        ifInDiscards        = 0,
        ifInErrors          = 0,
        ifInUnknownProtos   = 0,
        ifOutOctets         = 0,
        ifOutUcastPkts      = 0,
        ifOutDiscards       = 0,
        ifOutErrors         = 0,
    }

   2) Configure specific wireless binding parameters for "WTP Virtual
   Radio Interface"
   It will be done at the AC side through specific wireless binding MIB
   such as IEEE 802.11 MIB.
   For example, to configure parameter for "WTP Virtual Radio Interface"
   by 802.11 Dot11OperationTable.
   In Dot11OperationTable
   {
        ifIndex                         = 10,
        dot11MACAddress                 = 0.0.0.0.0.0,
        dot11RTSThreshold               = 2347,
        dot11ShortRetryLimit            = 7,
        dot11LongRetryLimit             = 4,
        dot11FragmentationThreshold     = 256,
        dot11MaxTransmitMSDULifetime    = 512,
        dot11MaxReceiveLifetime         = 512,
```

```
        dot11ManufacturerID             = "capwap",
        dot11ProductID                  = "capwap"
    }
```

In the example, it suppose ifIndex of an interface in "WTP Virtual Radio Interface" ifType is 10.

3) Other configurations for a specific wireless binding
For example, WLAN service configuration will be done through CAPWAP binding MIB and IEEE 802.11 MIB. In the CAPWAP 802.11 binding MIB draft, it will give more explain.

4) WTP reports its current configuration status
After join phase and before WTP get configuration from AC, it will report its current configuration status to AC through configuration status message. The data of MIB objects will be updated at AC side. For example, for 802.11 binding, WTP will update data in the ifTable and IEEE 802.11 MIB so on according to message content. As an example for ifIndex 10, its ifOperStatus in ifTable will be updated according to current radio operational status in the CAPWAP message.

5) Query WTP and radio statistics data
After WTPs come to run status, operator could query WTP and radio statistics data through CAPWAP-MIB and specific binding MIB. For example, through dot11CountersTable in the IEEE 802.11 MIB, operator could query counter data for radio which is identified by ifIndex of a virtual radio interface. With capwapACState table in the MIB, operator could query configuration and properties of WTPs which are in run status.

6) Query other statistics data of a specific wireless binding
For example, operator could query the statistics data of WLAN service through 802.11 binding MIB and IEEE 802.11 MIB. In the CAPWAP 802.11 binding MIB draft, it will give more explain.

7) Query other properties of WTP
Operator could query MIB objects in the ENTITY-MIB by capwapWTPPHYIndex in the capwapWTPTable of CAPWAP-MIB. The properties of WTP such as software version, hardware version and so on are available in the ENTITY-MIB.

9. Definitions

```
CAPWAP-MIB DEFINITIONS ::= BEGIN

IMPORTS
```

```
   PhysAddress, TEXTUAL-CONVENTION, TruthValue,
   DateAndTime
       FROM SNMPv2-TC
   InterfaceIndex
       FROM IF-MIB
   PhysicalIndex
       FROM ENTITY-MIB
   SnmpAdminString
       FROM SNMP-FRAMEWORK-MIB
   NOTIFICATION-GROUP, OBJECT-GROUP, MODULE-COMPLIANCE
       FROM SNMPv2-CONF
   MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE, mib-2,
   Integer32, Unsigned32, Counter32
       FROM SNMPv2-SMI
   InetAddressType, InetAddress
       FROM INET-ADDRESS-MIB;

capwapMIB MODULE-IDENTITY
    LAST-UPDATED "200802120000Z"        -- Feb 12, 2008
    ORGANIZATION "IETF Control And Provisioning of Wireless Access
                  Points (CAPWAP) Working Group
                  http://www.ietf.org/html.charters/capwap-charter.html"
    CONTACT-INFO
        "General Discussion: capwap@frascone.com
         To Subscribe: http://lists.frascone.com/mailman/listinfo/capwap

         Yang Shi
         H3C, Digital Technology Plaza, NO.9 Shangdi 9th Street,Haidian
         District,Beijing,China(100085)
         Email:  young@h3c.com

         David T. Perkins
         228 Bayview Dr
         San Carlos, CA 94070
         USA
         Phone: +1 408 394-8702
         Email:  dperkins@snmpinfo.com

         Chris Elliott
         Cisco Systems, Inc.
         7025 Kit Creek Rd., P.O. Box 14987
         Research Triangle Park  27709
         USA
         Phone: +1 919-392-2146
         Email: chelliot@cisco.com"

    DESCRIPTION
        "Copyright (C) 2008 The Internet Society.  This version of
```

```
          the MIB module is part of RFC xxx; see the RFC itself
          for full legal notices.

          This MIB module contains managed object definitions for
          the CAPWAP Protocol."
      REVISION    "200802120000Z"
      DESCRIPTION
          "Initial version published as RFC xxx"
          ::= { mib-2 xxx }

-- Textual Conventions

CapwapWTPId ::= TEXTUAL-CONVENTION
      STATUS      current
      DESCRIPTION
          "Represents a unique identifier of a WTP instance.
           As usual, a serial number of WTP will be used."
      SYNTAX       OCTET STRING(SIZE(128))

CapwapStationId ::= TEXTUAL-CONVENTION
      STATUS      current
      DESCRIPTION
          "Represents a unique identifier of a station instance.
           As usual, the MAC address of station will be used."
      SYNTAX      OCTET STRING (SIZE (6))

CapwapRadioId ::= TEXTUAL-CONVENTION
      STATUS      current
      DESCRIPTION
          "Represents a unique identifier of a radio on a WTP."
      SYNTAX      Unsigned32 (1..4294967295)

CapwapWTPTunnelMode ::= TEXTUAL-CONVENTION
      STATUS      current
      DESCRIPTION
          "Represents the tunneling mode for station data that are
           supported by the WTP.
           The possible value could be:
            localBridging(1) - Local Bridging Mode,
            dot3Tunnel(2)    - 802.3 Frame Tunnel Mode,
            nativeTunnel(3)  - Native Frame Tunnel Mode."
      REFERENCE
          "Section 4.6.41. of CAPWAP Protocol Specification, RFC xxx."
      SYNTAX      INTEGER { localBridging(1), dot3Tunnel(2),
                            nativeTunnel(3) }

CapwapWTPMACType ::= TEXTUAL-CONVENTION
      STATUS      current
```

```
    DESCRIPTION
        "Represents the MAC mode of operation supported by the WTP.
         The possible value could be:
          localMAC(1) - Local-MAC Mode,
          splitMAC(2) - Split-MAC Mode."
    REFERENCE
        "Section 4.6.44. of CAPWAP Protocol Specification, RFC xxx."
    SYNTAX      INTEGER { localMAC(1), splitMAC(2) }

CapwapChannelType::= TEXTUAL-CONVENTION
    STATUS     current
    DESCRIPTION
        "Represents the channel type for CAPWAP protocol.
         The following values are supported:
          data(1)    - data Channel
          control(2) - control Channel."
    SYNTAX    INTEGER { data(1), control(2) }

CapwapWTPAuthenMethod ::= TEXTUAL-CONVENTION
    STATUS     current
    DESCRIPTION
        "Represents the authentication credential type
         for WTP.
         The following values are supported:
           clear(1) - cleartext and no authentication,
           x509(2)  - X.509 Certificate Based,
           psk(3)   - Pre-Shared Secret,
           other(8) - Other method, for example, vendor specific.
         As mandatory requirement, CAPWAP control channel
         authentication should use DTLS, and either by certificate or
         PSK. For data channel, DTLS is optional."
    SYNTAX    INTEGER { clear(1), x509(2), psk(3), other(8) }

-- Top level components of this MIB

-- Notifications
capwapNotifications OBJECT IDENTIFIER
    ::= { capwapMIB 0 }
-- Tables, Scalars
capwapObjects OBJECT IDENTIFIER
    ::= { capwapMIB 1 }
-- Conformance
capwapConformance OBJECT IDENTIFIER
    ::= { capwapMIB 2 }


-- AC Objects Group
```

```
capwapAC OBJECT IDENTIFIER
    ::= { capwapObjects 1 }

capwapWTPSessions OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the total number of WTPs which are connecting to
         AC."
    REFERENCE
        "Section 4.6.1. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapAC 1 }

capwapWTPSessionsLimit  OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the maximum number of WTP sessions supported by
         the AC."
    REFERENCE
        "Section 4.6.1. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapAC 2 }

capwapStationSessions  OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the total number of stations which are accessing
         the wireless service."
    REFERENCE
        "Section 4.6.1. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapAC 3 }

capwapStationSessionsLimit  OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the maximum number of station sessions supported by
         the AC."
    REFERENCE
        "Section 4.6.1. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapAC 4 }

capwapDataChannelSecOptions  OBJECT-TYPE
```

```
    SYNTAX      Integer32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the security policy supported for CAPWAP data
         channel.
         The AC MAY support more than one option, represented by the bit
         field below.
         clear(1)  -  Clear Text,
         dtls(2)   -  DTLS,
         vendor(3) -  vendor specific."
    REFERENCE
       "Section 4.6.1. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapAC 5 }

capwapWTPAuthenOptions  OBJECT-TYPE
    SYNTAX      Integer32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the authentication credentia type supported by the
         AC for control channel.
         The AC MAY support more than one option, represented by the bit
         field below.
         x509(1) -  X.509 Certificate Based
         psk(2)  -  Pre-Shared Secret."
    REFERENCE
       "Section 4.6.1. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapAC 6 }

capwapWTPFallbackEnable OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Represents enable or disable automatic CAPWAP fallback in the
         event that a WTP detects its preferred AC, and is not currently
         connected to it."
    REFERENCE
        "Section 4.6.40. of CAPWAP Protocol Specification, RFC xxx."
     ::= { capwapAC 7 }

capwapWTPACNameList OBJECT-TYPE
    SYNTAX      OCTET STRING(SIZE(256))
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Represents the name list of ACs and use semicolon to separate AC
```

```
          name. The AC name could be configured with the order of
          Primary AC, secondary AC and so on. WTP will try to connect to
          AC name in the list one by one till it connected to one AC."
      REFERENCE
          "Section 4.6.5. of CAPWAP Protocol Specification, RFC xxx."
      ::= { capwapAC 8 }

capwapMaxFailedDTLSSessionRetry OBJECT-TYPE
      SYNTAX      Unsigned32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
          "Represents the maximum number of failed DTLS session
          establishment attempts before the CAPWAP device enters a silent
          period."
      REFERENCE
          "Section 4.7.7. of CAPWAP Protocol Specification, RFC xxx."
      ::= { capwapAC 9 }

capwapWTPIdleTimeout OBJECT-TYPE
      SYNTAX      Unsigned32
      UNITS       "kbtyes"
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
          "Represents the time out parameter for WTP idle state."
      REFERENCE
          "Section 4.8.5. of CAPWAP Protocol Specification, RFC xxx."
      ::= { capwapAC 10 }

capwapWTPMaxDiscoveries OBJECT-TYPE
      SYNTAX      Unsigned32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
          "Represents the maximum number of Discovery Request messages
          that will be sent after a WTP boots"
      REFERENCE
          "Section 4.8.6. of CAPWAP Protocol Specification, RFC xxx."
       ::= { capwapAC 11 }

capwapWTPMaxRetransmit OBJECT-TYPE
      SYNTAX      Unsigned32
      MAX-ACCESS  read-only
      STATUS      current
      DESCRIPTION
          "Represent the maximum number of retransmission for a given
          CAPWAP packet before the link layer considers the peer dead."
```

```
    REFERENCE
        "Section 4.8.7. of CAPWAP Protocol Specification, RFC xxx."
     ::= { capwapAC 12 }

capwapWTPReportInterval OBJECT-TYPE
    SYNTAX      Unsigned32
    UNITS       "second"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the interval for WTP send report."
    REFERENCE
        "Section 4.8.8. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapAC 13 }

-- End of AC Objects Group


-- WTP Objects Group

capwapWTPs OBJECT IDENTIFIER
    ::= { capwapObjects 2 }

-- capwapWTPStateTable table

capwapWTPStateTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CapwapWTPStateEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table of objects that display WTPs in different
         CAPWAP FSM state."
    ::= { capwapWTPs 1 }

capwapWTPStateEntry  OBJECT-TYPE
    SYNTAX      CapwapWTPStateEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A set of objects that display WTPs in different
         CAPWAP FSM state."
    INDEX { capwapWTPId }
    ::= { capwapWTPStateTable 1 }

CapwapWTPStateEntry ::= SEQUENCE {
    capwapWTPId               CapwapWTPId,
    capwapWTPIPAddressType    InetAddressType,
    capwapWTPIPAddress        InetAddress,
```

```
    capwapWTPPHYAddress       PhysAddress,
    capwapWTPState            INTEGER }

capwapWTPId OBJECT-TYPE
    SYNTAX      CapwapWTPId
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "Represents the unique identifier of a WTP."
    ::= { capwapWTPStateEntry 1 }

capwapWTPIPAddressType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the type of IP address of WTP."
    ::= { capwapWTPStateEntry 2 }

capwapWTPIPAddress OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the IP address(IPv4 or IPv6) of a WTP."
    ::= { capwapWTPStateEntry 3 }

capwapWTPPHYAddress OBJECT-TYPE
    SYNTAX      PhysAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the PHY address of a WTP."
    ::= { capwapWTPStateEntry 4 }

capwapWTPState  OBJECT-TYPE
    SYNTAX      INTEGER {
                 dtls(1), join(2), image(3), configure(4),
                 run(5), clear(6), unknown(7)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the various possible CAPWAP FSM states of WTP
         The following values are supported:
          dtls(1)      - DTLS negotiation process
          join(2)      - WTP is joining with AC,
          image(3)     - WTP is downloading software,
```

```
          configure(4) - WTP is getting configuration from AC,
          run(5)       - WTP comes to run state,
          clear(6)     - WTP recoveries default configuration.
          unknown(7)    - Operator already prepare configuration
                          for WTP, while WTP has not contact with AC
                          till now."
    REFERENCE
        "Section 2.3.1. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPStateEntry 5 }

-- End of capwapWTPStateTable Table


-- capwapWTPTable Table

capwapWTPTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CapwapWTPEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table of objects that display and control WTPs in
         running state. Values of all read-write objects in this
         table are persistent at restart/reboot."
    ::= { capwapWTPs 2 }

capwapWTPEntry  OBJECT-TYPE
    SYNTAX      CapwapWTPEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A set of objects that display and control a WTP in
         running state."
    INDEX { capwapWTPCurrId }
    ::= { capwapWTPTable 1 }

CapwapWTPEntry ::= SEQUENCE {
    capwapWTPCurrId               CapwapWTPId,
    capwapWTPPHYIndex             PhysicalIndex,
    capwapWTPName                 SnmpAdminString,
    capwapWTPLocation             SnmpAdminString,
    capwapWTPBaseMACAddress       PhysAddress,
    capwapWTPTunnelModeOptions    CapwapWTPTunnelMode,
    capwapWTPMACTypeOptions       CapwapWTPMACType,
    capwapWTPDiscoveryType        INTEGER,
    capwapWTPRadiosInUseNum       Unsigned32,
    capwapWTPRadioNumLimit        Unsigned32,
    capwapWTPStaticIPEnable       TruthValue,
    capwapWTPStaticIPType         InetAddressType,
```

```
    capwapWTPStaticIP            InetAddress,
    capwapWTPNetmask             InetAddress,
    capwapWTPGateway             InetAddress }

capwapWTPCurrId OBJECT-TYPE
    SYNTAX      CapwapWTPId
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Represents the unique identifier of a WTP Which is
         in running state."
    ::= { capwapWTPEntry 1 }

capwapWTPPHYIndex OBJECT-TYPE
    SYNTAX      PhysicalIndex
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the unique physical index of a physical entity
         in the ENTITY-MIB. The information such as software version
         of specific WTP could be accessed through the index."
    ::= { capwapWTPEntry 2 }

capwapWTPName  OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Represents the name of a WTP."
    REFERENCE
        "Section 4.6.45. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 3 }

capwapWTPLocation  OBJECT-TYPE
    SYNTAX      SnmpAdminString
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Represents the location of a WTP."
    REFERENCE
        "Section 4.6.28. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 4 }

capwapWTPBaseMACAddress  OBJECT-TYPE
    SYNTAX      PhysAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
```

```
        "Represents the WTP's Base MAC Address, which MAY be assigned
         to the primary Ethernet interface."
    REFERENCE
        "Section 4.6.38. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 5 }

capwapWTPTunnelModeOptions  OBJECT-TYPE
    SYNTAX      CapwapWTPTunnelMode
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the tunneling modes for station data that are
         supported by the WTP.
         The WTP MAY support more than one option, represented by the
         bit field below.
          localBridging(1) - Local Bridging Mode,
          dot3Tunnel(2)    - 802.3 Frame Tunnel Mode,
          nativeTunnel(3)  - Native Frame Tunnel Mode."
    REFERENCE
        "Section 4.6.41. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 6 }

capwapWTPMACTypeOptions  OBJECT-TYPE
    SYNTAX      CapwapWTPMACType
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the MAC mode of operation supported by the WTP.
         The WTP MAY support more than one option, represented by the
         bit field below.
          localMAC(1) - Local-MAC Mode,
          splitMAC(2) - Split-MAC Mode."
    REFERENCE
        "Section 4.6.44. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 7 }

capwapWTPDiscoveryType OBJECT-TYPE
    SYNTAX      INTEGER {
                 unknown(1), staticConfig(2), dhcp(3), dns(4), acRef(5)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents how WTP discovered the AC.
         The following values are supported:
          unknown(1)       - the method is unknown,
          staticConfig(2)  - static IP configuration,
          dhcp(3)          - DHCP,
```

```
          dns(4)          - DNS,
          acRef(5)        - AC Referral."
    REFERENCE
        "Section 4.6.20. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 8 }

capwapWTPRadiosInUseNum OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of radios which are in use."
    REFERENCE
        "Section 4.6.39. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 9 }

capwapWTPRadioNumLimit OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the maximum radio number could be supported
         by WTP."
    REFERENCE
        "Section 4.6.39. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 10 }

capwapWTPStaticIPEnable OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Represents whether the WTP should use a static IP address
         or not. A value of false disables the static IP address,
         while a value of true enables it."
    REFERENCE
        "Section 4.6.49. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 11 }

capwapWTPStaticIPType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "Represents whether WTP uses IPV4 or IPV6 static IP address."
    ::= { capwapWTPEntry 12 }

capwapWTPStaticIP OBJECT-TYPE
```

```
    SYNTAX      InetAddress
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "When capwapWTPStaticIPEnable is true, it represents the static
         IP address to assign to the WTP."
    REFERENCE
        "Section 4.6.49. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 13 }

capwapWTPNetmask OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "When capwapWTPStaticIPEnable is true, it represents the netmask
         to assign to the WTP."
    REFERENCE
        "Section 4.6.49. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 14 }

capwapWTPGateway OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
        "When capwapWTPStaticIPEnable is true, it represents the gateway
         to assign to the WTP."
    REFERENCE
        "Section 4.6.49. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPEntry 15 }

-- End of capwapWTPTable table


-- capwapRadioBindTable Table

capwapRadioBindTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CapwapRadioBindEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table of objects that display the mapping relationship
         between specific interface of 'WTP Virtual Radio Interface'
         ifType and PHY radio. The mapping relationship in this table
         is persistent at restart/reboot."
    ::= { capwapWTPs 3 }
```

```
capwapRadioBindEntry  OBJECT-TYPE
    SYNTAX      CapwapRadioBindEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A set of objects that display the mapping relationship
         between 'WTP Virtual Radio' and PHY radio."
    INDEX { capwapWTPId, capwapRadioId }
    ::= { capwapRadioBindTable 1 }

CapwapRadioBindEntry ::= SEQUENCE {
    capwapRadioId                   CapwapRadioId,
    capwapWTPVirtualRadioIfIndex    InterfaceIndex,
    capwapWirelessBinding           INTEGER
    }

capwapRadioId OBJECT-TYPE
    SYNTAX      CapwapRadioId
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "Represents the identifier of a PHY radio on a WTP, and only
         requires unique on a WTP.
         For example, WTP A and WTP B will use same value of
         capwapRadioId for their first radio."
    REFERENCE
        "Section 4.6.31. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioBindEntry 1 }

capwapWTPVirtualRadioIfIndex OBJECT-TYPE
    SYNTAX      InterfaceIndex
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the ifIndex for an interface of 'WTP Virtual Radio
         Interface' ifType.
         Before WTPs connect to AC and get configuration, operator
         will prepare configuration for them. At AC side, there are
         interface of 'WTP Virtual Radio Interface' type which represent
         PHY radio interface at WTP side.
         As most MIBs use ifIndex to identify an interface for
         configuration and statistic data, for example, IEEE 802.11 MIB.
         It will be very easy to reuse other MIBs such as IEEE 802.11
         MIB by 'WTP Virtual Radio Interface'.
         Require IANA to assign an ifType for 'WTP Virtual Radio
         Interface'."
    ::= { capwapRadioBindEntry 2 }
```

```
capwapWirelessBinding OBJECT-TYPE
    SYNTAX INTEGER { none(1), dot11(2), dot16(3), epc(4) }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the wireless binding type for radio.
         The following values are supported:
          none(1)  - No any wireless binding defined.
          dot11(2) - IEEE 802.11.
          dot16(3) - IEEE 802.16.
          epc(4)   - EPCGlobal."
    REFERENCE
        "Section 4.3. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioBindEntry 3 }

-- End of capwapRadioBindTable Table


-- capwapStationTable Table

capwapStationTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF CapwapStationEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table of objects that display stations which are
         associated with the specific radio on the WTP."
    ::= { capwapWTPs 4 }

capwapStationEntry OBJECT-TYPE
    SYNTAX      CapwapStationEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A set of objects that display a station which is
         associated with the specific radio on the WTP."
    INDEX { capwapWTPCurrId, capwapRadioId, capwapStationId }
    ::= { capwapStationTable 1 }

CapwapStationEntry ::= SEQUENCE {
    capwapStationId           CapwapStationId,
    capwapStationAddedTime    DateAndTime,
    capwapStationVlanName     OCTET STRING
    }

capwapStationId OBJECT-TYPE
    SYNTAX      CapwapStationId
    MAX-ACCESS  not-accessible
```

```
    STATUS      current
    DESCRIPTION
        "Represents the unique identifier of the station."
    REFERENCE
        "Section 4.6.8. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapStationEntry 1 }

capwapStationAddedTime OBJECT-TYPE
    SYNTAX      DateAndTime
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the time when the station is added."
    REFERENCE
        "Section 4.6.8. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapStationEntry 2 }

capwapStationVlanName OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(32))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents VLAN name to which the station is associated."
    REFERENCE
        "Section 4.6.8. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapStationEntry 3 }

-- End of capwapStationTable Table


--  capwapWTPRebootStatTable

capwapWTPRebootStatsTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF CapwapWTPRebootStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table of objects that display WTPs' reboot statistic
         data."
    ::= { capwapWTPs 5 }

capwapWTPRebootStatsEntry  OBJECT-TYPE
    SYNTAX      CapwapWTPRebootStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A set of objects that display the reboot statistic data
         of a WTP."
```

```
    INDEX { capwapWTPCurrId }
    ::= { capwapWTPRebootStatsTable 1 }

CapwapWTPRebootStatsEntry ::= SEQUENCE {
    capwapWTPRebootCount            Counter32,
    capwapWTPInitCount              Counter32,
    capwapWTPLinkFailureCount       Counter32,
    capwapWTPSwFailureCount         Counter32,
    capwapWTPHwFailureCount         Counter32,
    capwapWTPOtherFailureCount      Counter32,
    capwapWTPUnknownFailureCount    Counter32,
    capwapWTPLastFailureType        INTEGER
    }

capwapWTPRebootCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of reboots that have occurred due to a
         WTP crash.  A value of 65535 implies that this information is
         not available on the WTP."
    REFERENCE
        "Section 4.6.48. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPRebootStatsEntry 1 }

capwapWTPInitCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of reboots that have occurred at the
         request of a CAPWAP protocol message, such as a change in
         configuration that required a reboot or an explicit CAPWAP
         protocol reset request.  A value of 65535 implies that this
         information is not available on the WTP."
    REFERENCE
        "Section 4.6.48. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPRebootStatsEntry 2 }

capwapWTPLinkFailureCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that a CAPWAP protocol
         connection with an AC has failed due to link failure."
    REFERENCE
```

```
        "Section 4.6.48. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPRebootStatsEntry 3 }

capwapWTPSwFailureCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that a CAPWAP protocol
         connection with an AC has failed due to software related
         reasons."
    REFERENCE
        "Section 4.6.48. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPRebootStatsEntry 4 }

capwapWTPHwFailureCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that a CAPWAP protocol
         connection with an AC has failed due to hardware related
         reasons."
    REFERENCE
        "Section 4.6.48. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPRebootStatsEntry 5 }

capwapWTPOtherFailureCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that a CAPWAP protocol
         connection with an AC has failed due to known reasons, other
         than AC initiated, link, software or hardware failure."
    REFERENCE
        "Section 4.6.48. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPRebootStatsEntry 6 }

capwapWTPUnknownFailureCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that a CAPWAP protocol
         connection with an AC has failed for unknown reasons."
    REFERENCE
        "Section 4.6.48. of CAPWAP Protocol Specification, RFC xxx."
```

```
    ::= { capwapWTPRebootStatsEntry 7 }

capwapWTPLastFailureType OBJECT-TYPE
    SYNTAX      INTEGER {
                 notSupport(1), acInit(2), linkFailure(3),
                 swFailure(4), hwFailure(5),other(6), unknown(255)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The failure type of the most recent WTP failure.
         The following values are supported:
          notSupport(1)    - Not Supported,
          acInit(2)        - AC Initiated,
          linkFailure(3)   - Link Failure,
          swFailure(4)     - Software Failure,
          hwFailure(5)     - Hardware Failure,
          otherFailure(6)  - Other Failure,
          unknown(255)     - Unknown (e.g., WTP doesn't keep track
                             of info)."
    REFERENCE
        "Section 4.6.48. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapWTPRebootStatsEntry 8 }

--  End of capwapWTPRebootStatsTable table


--  capwapRadioStatsTable table

capwapRadioStatsTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF CapwapRadioStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A table of objects that display statistics on radios behavior,
         and reasons of radios have been reset."
    ::= { capwapWTPs 6 }

capwapRadioStatsEntry  OBJECT-TYPE
    SYNTAX      CapwapRadioStatsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "A set of objects that display the statistic data of
         events happened on a specific radio of a WTP."
    INDEX { capwapWTPCurrId, capwapRadioId }
    ::= { capwapRadioStatsTable 1 }
```

```
CapwapRadioStatsEntry ::= SEQUENCE {
    capwapRadioResetCount            Counter32,
    capwapRadioSwFailCount           Counter32,
    capwapRadioHwFailCount           Counter32,
    capwapRadioOtherFailCount        Counter32,
    capwapRadioUnknownFailCount      Counter32,
    capwapRadioConfigUpdateCount     Counter32,
    capwapRadioChannelChangeCount    Counter32,
    capwapRadioBandChangeCount       Counter32,
    capwapRadioCurrentNoiseFloor     Integer32,
    capwapRadioDecryptErrorCount     Counter32,
    capwapRadioTxQueueLevel          Integer32,
    capwapRadioRFLinkFramesPerSec    Counter32,
    capwapRadioLastFailType          INTEGER
   }

capwapRadioResetCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that that the radio has been
         reset."
    REFERENCE
        "Section 4.6.47. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 1 }

capwapRadioSwFailCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that the radio has failed due
         to software related reasons."
    REFERENCE
        "Section 4.6.47. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 2 }

capwapRadioHwFailCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that the radio has failed due
         to hardware related reasons."
    REFERENCE
        "Section 4.6.47. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 3 }
```

```
capwapRadioOtherFailCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that the radio has failed due to
         known reasons, other than software or hardware failure."
    REFERENCE
        "Section 4.6.47. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 4 }

capwapRadioUnknownFailCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that the radio has failed for
         unknown reasons."
    REFERENCE
        "Section 4.6.47. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 5 }

capwapRadioConfigUpdateCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that the radio configuration has
         been updated."
    REFERENCE
        "Section 4.6.47. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 6 }

capwapRadioChannelChangeCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of times that the radio channel has
         been changed."
    REFERENCE
        "Section 4.6.47. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 7 }

capwapRadioBandChangeCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
```

```
    DESCRIPTION
        "Represents the number of times that the radio has changed
         frequency bands."
    REFERENCE
        "Section 4.6.47. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 8 }

capwapRadioCurrentNoiseFloor OBJECT-TYPE
    SYNTAX      Integer32
    UNITS       "dbm"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the noise floor of the radio receiver in units of
         dBm."
    ::= { capwapRadioStatsEntry 9 }

capwapRadioDecryptErrorCount OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of decryption errors that occurred on the
         WTP.  Note that this field is only valid in cases where the WTP
         provides encryption/decryption services."
    REFERENCE
        "Section 4.6.47. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 10 }

capwapRadioTxQueueLevel OBJECT-TYPE
    SYNTAX      Integer32 (0..100)
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the percentage of Wireless Transmit queue
         utilization,calculated as the sum of utilized transmit queue
         lengths divided by the sum of maximum transmit queue lengths,
         multiplied by 100."
    REFERENCE
        "Section 4.6.46. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 11 }

capwapRadioRFLinkFramesPerSec OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the number of frames transmitted or received per
```

```
          second by the WTP over the radio interface."
    REFERENCE
        "Section 4.6.46. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapRadioStatsEntry 12 }

capwapRadioLastFailType OBJECT-TYPE
    SYNTAX      INTEGER {
                 notSupport(1),
                 swFailure(2),
                 hwFailure(3),
                 otherFailure(4),
                 unknown(255)
                }
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "Represents the failure type of the most recent radio failure.
         The following values are supported:
          notSupport(1)   - Not Supported,
          swFailure(2)    - Software Failure,
          hwFailure(3)    - Hardware Failure,
          otherFailure(4) - Other Failure,
          unknown(255)    - Unknown."
    ::= { capwapRadioStatsEntry 13 }

--  End of capwapRadioStatsTable table


-- Notifications

capwapChannelUp NOTIFICATION-TYPE
    OBJECTS     { capwapWTPId,
                  capwapChannelType,
                  capwapWTPAuthenMethod }
    STATUS      current
    DESCRIPTION
        "This notification is sent by AC when a CAPWAP channel
         established. The notification is separated for data or control
         channel."
    ::= { capwapNotifications 1 }

capwapChannelDown NOTIFICATION-TYPE
    OBJECTS     { capwapWTPId, capwapChannelDownReason }
    STATUS      current
    DESCRIPTION
        "This notification is sent by AC when CAPWAP channel becomes
         down."
    ::= { capwapNotifications 2 }
```

```
capwapDecryptErrorReport NOTIFICATION-TYPE
    OBJECTS     { capwapWTPId,
                  capwapRadioId,
                  capwapIdEntryNum,
                  capwapStationIdList }
    STATUS      current
    DESCRIPTION
        "This notification is generated when a WTP that has occurred
         decryption error since the last report."
    REFERENCE
        "Section 4.6.15. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifications 3 }

capwapJoinFailure NOTIFICATION-TYPE
    OBJECTS     { capwapWTPId, capwapJoinFailureReason }
    STATUS      current
    DESCRIPTION
        "This notification is generated when a WTP fails to join."
    REFERENCE
        "Section 4.6.33. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifications 4 }

capwapImageUpgradeFailure NOTIFICATION-TYPE
    OBJECTS     { capwapWTPId, capwapImageFailureReason }
    STATUS      current
    DESCRIPTION
        "This notification is generated when a WTP fails to update
         software image."
    REFERENCE
        "Section 4.6.33. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifications 5 }

capwapConfigMsgError NOTIFICATION-TYPE
    OBJECTS     { capwapWTPId, capwapConfigMsgErrorType,
                  capwapMsgErrorElements }
    STATUS      current
    DESCRIPTION
        "This notification is generated when a WTP received message
         elements in the Configuration Status Response which it was
         unable to apply locally."
    REFERENCE
        "Section 4.6.34. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifications 6 }

capwapRadioOperableStatus NOTIFICATION-TYPE
    OBJECTS     { capwapWTPId,
                  capwapRadioId,
                  capwapRadioOperStatusFlag,
```

```
                  capwapRadioFailureReason }
    STATUS      current
    DESCRIPTION
        "The notification will notify which radio become inoperable
         or operable."
    REFERENCE
        "Section 4.6.32. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifications 7 }

capwapWTPAuthenticationFailure NOTIFICATION-TYPE
    OBJECTS     { capwapWTPId,
                  capwapChannelType,
                  capwapWTPAuthenMethod,
                  capwapWTPAuthenFailureReason }
    STATUS      current
    DESCRIPTION
        "The notification will notify the authentication failure event,
         and provides the reason for it."
    REFERENCE
        "Section 2.3.1. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifications 8 }


-- Objects used only in notifications


-- for notifications
capwapNotifyVarObjects OBJECT IDENTIFIER
    ::= { capwapObjects 3 }

capwapChannelType OBJECT-TYPE
    SYNTAX      CapwapChannelType
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "Represents the channel type for CAPWAP protocol."
    ::= { capwapNotifyVarObjects 1 }

capwapWTPAuthenMethod OBJECT-TYPE
    SYNTAX      CapwapWTPAuthenMethod
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "Represents authentication method for Channel."
    ::= { capwapNotifyVarObjects 2 }

capwapChannelDownReason OBJECT-TYPE
    SYNTAX      INTEGER { timeout(1), rekeyfailure(2), apReboot(3) }
```

```
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "Represents the reason for Channel down.
         The following values are supported:
          timeout(1)       - The keep alive is timeout,
          rekeyfailure(2)  - Rekey process is failed, channel will be
                             broken.
          apReboot(3)      - AC reboot WTP."
    ::= { capwapNotifyVarObjects 3 }

capwapIdEntryNum OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "Represents the entry number of station id in the
         capwapStationIdList."
    REFERENCE
        "Section 4.6.15. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifyVarObjects 4 }

capwapStationIdList OBJECT-TYPE
    SYNTAX      OCTET STRING
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "Represents the list of station id."
    REFERENCE
        "Section 4.6.15. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifyVarObjects 5 }

capwapWTPAuthenFailureReason OBJECT-TYPE
    SYNTAX      INTEGER {
                 keyMismatch(1), invalidCA(2), micError(3),
                 timeout(4), unknown(8)
                }
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "Represents reason for WTP authorization failure.
         The following values are supported:
          keyMismatch(1) - WTP's and AC's key is not matched,
          invalidCA(2)   - ca is not valid,
          micError(3)    - detect MIC error,
          timeout(4)     - WaitDTLS Timer is timeout,
          unknown(8)     - Unknown reason."
    REFERENCE
```

```
        "Section 2.3.1. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifyVarObjects 6 }

capwapRadioOperStatusFlag OBJECT-TYPE
    SYNTAX INTEGER { operable(1), inoperable(2) }
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "Represents the operation status of a radio.
         The following values are supported:
           operable(1)   - To indicate radio is operable,
           inoperable(2) - To indicate radio is inoperable, and
                           capwapRadioFailureReason object will
                           give reason in details"
    REFERENCE
        "Section 4.6.32. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifyVarObjects 7 }

capwapRadioFailureReason OBJECT-TYPE
    SYNTAX      INTEGER {
                 hwError(1), swError(2), adminSet(3), unknown(8)
                }
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "Represents errors caused by configuration operation.
         The following values are supported
          hwError(1)  - Radio Failure,
          swError(2)  - Software Failure,
          adminSet(3) - Administratively Set,
          unknown(8)  - Unknown reason."
    REFERENCE
        "Section 4.6.32. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifyVarObjects 8 }

capwapJoinFailureReason  OBJECT-TYPE
    SYNTAX      INTEGER {
                 unspecified(1), resDepletion(2), unknownSource(3),
                 incorrectData(4), sessionInUse(5), notSupportHw(6),
                 notSupportBinding(7)
                }
    MAX-ACCESS  accessible-for-notify
    STATUS      current
    DESCRIPTION
        "The following join failure types are supported:
         unspecified(1)       - unspecified failure reason,
         resDepletion(2)      - Resource Depletion,
         unknownSource(3)     - Unknown Source,
```

```
          incorrectData(4)     - Incorrect Data,
          sessionInUse(5)      - Session ID already in use,
          notSupportHw(6)      - WTP Hardware not supported,
          notSupportBinding(7) - Binding Not Supported."
      REFERENCE
          "Section 4.6.33. of CAPWAP Protocol Specification, RFC xxx."
      ::= { capwapNotifyVarObjects 9 }

capwapImageFailureReason  OBJECT-TYPE
      SYNTAX      INTEGER {
                  invalidChecksum(1),
                  invalidLength(2),
                  other(3),
                  inUse(4)
                  }
      MAX-ACCESS  accessible-for-notify
      STATUS      current
      DESCRIPTION
          "The following software upgrade failure types are supported:
           invalidChecksum(1) - Invalid Checksum,
           invalidLength(2)   - Invalid Data Length,
           other(3)           - Other Error,
           inUse(4)           - Image Already Present."
      REFERENCE
          "Section 4.6.33. of CAPWAP Protocol Specification, RFC xxx."
      ::= { capwapNotifyVarObjects 10 }

capwapConfigMsgErrorType  OBJECT-TYPE
      SYNTAX      INTEGER {
                  unknownElement(1), unsupportedElement(2),
                  unknownValue(3), unsupportedValue(4)
                 }
      MAX-ACCESS  accessible-for-notify
      STATUS      current
      DESCRIPTION
          "The following software upgrade failure types are supported:
           unknownElement(1)     - Unknown Message Element,
           unsupportedElement(2) - Unsupported Message Element,
           unknownValue(3)       - Unknown Message Element Value,
           unsupportedValue(4)   - Unsupported Message Element Value."
      REFERENCE
          "Section 4.6.34. of CAPWAP Protocol Specification, RFC xxx."
     ::= { capwapNotifyVarObjects 11 }

capwapMsgErrorElements  OBJECT-TYPE
      SYNTAX      SnmpAdminString
      MAX-ACCESS  accessible-for-notify
      STATUS      current
```

```
    DESCRIPTION
        "Represents the message element sent by the AC in the
         Configuration Status Response message that caused the error."
    REFERENCE
        "Section 4.6.34. of CAPWAP Protocol Specification, RFC xxx."
    ::= { capwapNotifyVarObjects 12 }


-- Module compliance

capwapCompliances OBJECT IDENTIFIER
    ::= { capwapConformance 1 }

capwapGroups OBJECT IDENTIFIER
    ::= { capwapConformance 2 }

capwapCompliance MODULE-COMPLIANCE
   STATUS current
   DESCRIPTION
       "Describes the requirements for conformance to the
        CAPWAP Base MIB."

   MODULE -- this module
      MANDATORY-GROUPS    { capwapACNodeGroup,
                            capwapWTPStateGroup,
                            capwapWTPsGroup,
                            capwapRadiosGroup,
                            capwapStationsGroup }

      GROUP capwapACNodeGroup2
      DESCRIPTION
          "The capwapACNodeGroup2 group is optional."

      GROUP capwapWTPsGroup2
      DESCRIPTION
          "The capwapWTPsGroup2 group is optional."

      GROUP capwapWTPRebootStatsGroup
      DESCRIPTION
          "The capwapWTPRebootStatsGroup group is optional."

      GROUP capwapRadioStatsGroup
      DESCRIPTION
          "The capwapRadioStatsGroup group is optional."

      GROUP capwapNofificationGroup
      DESCRIPTION
          "The group capwapNofificationGroup is optional."
```

```
      GROUP capwapNotifyVarGroup
      DESCRIPTION
          "The capwapNotifyVarGroup group is optional.
           If capwapNofificationGroup is supported,
           this group must be implemented."

      OBJECT      capwapWirelessBinding
      SYNTAX      INTEGER { none(1) }
      DESCRIPTION
          "A value other than none(1) need not be supported if there is
           no wireless binding defined for technologies used."
     ::= { capwapCompliances 1 }

capwapACNodeGroup    OBJECT-GROUP
    OBJECTS {
        capwapWTPSessions,
        capwapWTPSessionsLimit,
        capwapStationSessions,
        capwapStationSessionsLimit,
        capwapWTPIdleTimeout,
        capwapWTPMaxDiscoveries,
        capwapWTPMaxRetransmit,
        capwapWTPReportInterval
    }
    STATUS  current
    DESCRIPTION
        "The collection of objects which are used to represent
         basic properties for AC from CAPWAP protocol perspective."
    ::= { capwapGroups 1 }

capwapACNodeGroup2   OBJECT-GROUP
    OBJECTS {
        capwapDataChannelSecOptions,
        capwapWTPAuthenOptions,
        capwapWTPFallbackEnable,
        capwapWTPACNameList,
        capwapMaxFailedDTLSSessionRetry
     }
    STATUS  current
    DESCRIPTION
        "The collection of objects which are used to represent
         other properties such as security for AC from
         CAPWAP protocol perspective."
    ::= { capwapGroups 2 }

capwapWTPStateGroup    OBJECT-GROUP
    OBJECTS {
        capwapWTPId,
```

```
        capwapWTPIPAddressType,
        capwapWTPIPAddress,
        capwapWTPPHYAddress,
        capwapWTPState
    }
    STATUS  current
    DESCRIPTION
        "The collection of objects which are used to represent
         WTP state information."
    ::= { capwapGroups 3 }

capwapWTPsGroup    OBJECT-GROUP
    OBJECTS {
        capwapWTPName,
        capwapWTPLocation,
        capwapWTPBaseMACAddress,
        capwapWTPTunnelModeOptions,
        capwapWTPMACTypeOptions,
        capwapWTPRadiosInUseNum,
        capwapWTPRadioNumLimit
    }
    STATUS  current
    DESCRIPTION
        "The collection of objects which are used to represent
         configuration and properties information for WTP
         in running state."
    ::= { capwapGroups 4 }

capwapWTPsGroup2   OBJECT-GROUP
    OBJECTS {
        capwapWTPPHYIndex,
        capwapWTPDiscoveryType,
        capwapWTPStaticIPEnable,
        capwapWTPStaticIPType,
        capwapWTPStaticIP,
        capwapWTPNetmask,
        capwapWTPGateway
    }
    STATUS  current
    DESCRIPTION
        "The collection of objects which are used to represent
         configuration and properties information for WTP
         in running state."
    ::= { capwapGroups 5 }

capwapRadiosGroup    OBJECT-GROUP
    OBJECTS {
        capwapRadioId,
```

```
        capwapWTPVirtualRadioIfIndex,
        capwapWirelessBinding
    }
    STATUS  current
    DESCRIPTION
        "The collection of objects which are used to represent
         wireless binding type, the mapping relationship between
         'WLAN Virtual Radio Interface' and PHY radio."
    ::= { capwapGroups 6 }

capwapStationsGroup    OBJECT-GROUP
    OBJECTS {
        capwapStationAddedTime,
        capwapStationVlanName
    }
    STATUS  current
    DESCRIPTION
        "The collection of objects which are used to represent
         stations' basic property."
    ::= { capwapGroups 7 }

capwapWTPRebootStatsGroup    OBJECT-GROUP
    OBJECTS {
        capwapWTPRebootCount,
        capwapWTPInitCount,
        capwapWTPLinkFailureCount,
        capwapWTPSwFailureCount,
        capwapWTPHwFailureCount,
        capwapWTPOtherFailureCount,
        capwapWTPUnknownFailureCount,
        capwapWTPLastFailureType
    }
    STATUS  current
    DESCRIPTION
        "The collection of objects which are used for collecting
         WTP reboot count, link failure count, hardware failure
         count and so on."
    ::= { capwapGroups 8 }

capwapRadioStatsGroup    OBJECT-GROUP
    OBJECTS {
        capwapRadioResetCount,
        capwapRadioSwFailCount,
        capwapRadioHwFailCount,
        capwapRadioOtherFailCount,
        capwapRadioUnknownFailCount,
        capwapRadioConfigUpdateCount,
        capwapRadioChannelChangeCount,
```

```
        capwapRadioBandChangeCount,
        capwapRadioCurrentNoiseFloor,
        capwapRadioDecryptErrorCount,
        capwapRadioTxQueueLevel,
        capwapRadioRFLinkFramesPerSec,
        capwapRadioLastFailType
    }
    STATUS  current
    DESCRIPTION
        "The collection of objects which are used for collecting
         radio reset count, channel change count, hardware failure
         count and so on"
    ::= { capwapGroups 9 }

capwapNofificationGroup    NOTIFICATION-GROUP
    NOTIFICATIONS {
        capwapChannelUp,
        capwapChannelDown,
        capwapDecryptErrorReport,
        capwapJoinFailure,
        capwapImageUpgradeFailure,
        capwapConfigMsgError,
        capwapRadioOperableStatus,
        capwapWTPAuthenticationFailure
    }
    STATUS  current
    DESCRIPTION
        "Collection of notifications in this MIB."
    ::= { capwapGroups 10 }

capwapNotifyVarGroup    OBJECT-GROUP
    OBJECTS {
        capwapWTPId,
        capwapRadioId,
        capwapChannelType,
        capwapWTPAuthenMethod,
        capwapChannelDownReason,
        capwapIdEntryNum,
        capwapStationIdList,
        capwapWTPAuthenFailureReason,
        capwapRadioOperStatusFlag,
        capwapRadioFailureReason,
        capwapJoinFailureReason,
        capwapImageFailureReason,
        capwapConfigMsgErrorType,
        capwapMsgErrorElements
    }
    STATUS  current
```

```
    DESCRIPTION
        "Objects used for notification."
    ::= { capwapGroups 11 }

END
```

10. Security Considerations

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

- - Unauthorized changes to the capwapWTPTable, writeable objects under capwapACs group may disrupt allocation of resources in the network.

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- - The capwapWTPTable exposes WTP's important information like IP address, MAC type and so on;

- - The capwapWTPRebootStatTable exposes WTP's failure information;

- - The capwapRadioStatsTable exposes radio's failure information;

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPSec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.

It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see [RFC3410], section 8), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to

enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

11. IANA Considerations

11.1. IANA Considerations for CAPWAP-MIB

The MIB module in this document uses the following IANA-assigned OBJECT IDENTIFIER values recorded in the SMI Numbers registry:

Descriptor	OBJECT IDENTIFIER value
capwapMIB	{ mib-2 XXX }

11.2. IANA Considerations for ifType

Require IANA to assign a ifType for 'WTP Virtual Radio Interface' type.

12. Contributors

This MIB is based on contributions from Long Gao.

13. Acknowledgements

The authors wish to thank David Harrington, Yu Liu, Xi Yao, Sachin Dutta, Ju Wang, Yujin Zhao, Haitao Zhang.

14. References

14.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578,

April 1999.

[RFC2579] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.

[RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

[RFC3418] Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3418, December 2002.

[RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.

[RFC4133] Bierman, A. and K. McCloghrie, "Entity MIB (Version 3)", RFC 4133, August 2005.

[I-D.ietf-capwap-protocol-specification] Calhoun, P., "CAPWAP Protocol Specification", draft-ietf-capwap-protocol-specification-08 (work in progress), November 2007.

14.2. Informative References

[RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.

[RFC4181] Heard, C., "Guidelines for Authors and Reviewers of MIB Documents", BCP 111, RFC 4181, September 2005.

Authors' Addresses

Yang Shi (editor)
H3C Tech. Co., Ltd
Digital Technology Plaza, NO.9 Shangdi 9th Street,Haidian District,
Beijing
China(100085)

Phone: +86 010 82775276
EMail: young@h3c.com

D. Perkins (editor)
SNMPinfo
288 Quailbrook Ct San Carlos,
CA 94070
USA

Phone: +1 408 394-8702
EMail: dperkins@snmpinfo.com

Chris Elliott (editor)
Cisco Systems, Inc.
7025 Kit Creek Rd., P.O. Box 14987 Research Triangle Park
27709
USA

Phone: +1 919-392-2146
EMail: chelliot@cisco.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).