IEEE P802.11  
Wireless LANs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| MIB TruthValue usage patterns | | | | |
| Date: 2015-03-08 | | | | |
| Author(s): | | | | |
| Name | Company | Address | Phone | email |
| Mark Hamilton | Spectralink | 2560 55th St  Boulder, CO 80301 USA | +1 303 441 7553 | [mark.hamilton@spectralink.com](mailto:mark.hamilton@spectralink.com) |
|  |  |  |  |  |

Abstract

This document contains a description of “design patterns” for the more common usage of MIB attributes with Type TruthValue, in Std 802.11 and its amendments.

R0 – Initial discussion document.

# Introduction and Purpose

This document outlines several common usage models for a subset of MIB attributes: those with data type TruthValue (“SYNTAX TruthValue” in the MIB object definition). Typically, such an attribute is used to indicate the status a feature or a set of behaviors, which either is or is not operational within a given implementation at a given time.

As with all MIB attributes, the benefit of these attributes to the Standard is to provide a model of expected behavior and interactions for implementations of the Standard. Since the MIB is rarely used, literally as defined, by implementation, instead it serves to provide a common definition style and a bit of formalism to descriptions of implementation behavior that is necessary for interoperability. In this regard, the MIB is similar to the service definitions in clause 6 (Layer management), and in fact through the mapping described in subclause 6.2 (Generic management primitives) the MIB attributes indirectly define part of the management service interface.

In this document, only MIB attributes defined with type (SYNTAX) of’”TruthValue” are addressed, as these attributes have the most commonality in purpose, while having considerable variation in naming and definition style for the same uses. It is hoped that with a common set of guidelines for naming and definition style, that all such MIB attributes can (probably over a period of time) be described with a small number of recognizable patterns, and result in ease of understanding their intent.

# Elements of attribute definition, and pattern uniqueness

Each usage pattern below is intended to completely cover the scenario for a given feature. That is, a given feature should use exactly one of these patterns, so it should never need or use more than one of these patterns. If a feature scenario is found that does not fit any pattern, or needs more than one pattern, then that should be discussed, and a new pattern for the scenario created if that is necessary.

Each usage pattern below includes guidelines for the following aspects of definitions for MIB attributes that fit that pattern:

* Name – using a consistent set of suffixes on attribute names will help the reader intuitively understand the purpose of the attribute, and thereby the behavior(s) to expect from implementations.
* MAX-ACCESS – this aspect should provide clarity about access to the attribute from an external entity (usually a management interface or system, such as SNMP or similar).
* DESCRIPTION – document 11-09/533 provides guidelines for general MIB attribute definition, including a discussion of the information that should be included. This document provides more specific guidelines specifically for TruthValue attribute patterns listed here.

Each usage pattern also includes guidelines for using and referencing the MIB attribute elsewhere in the Standard.

For the purposes of this document, the term “feature” applies to any identifiable unique feature of the Standard that could be independently present or absent in a particular implementation, or a similar set of behaviors which might be operational as a group, or none of them are.

# Patterns

## Static implementation capability, internal use only

The intent of this pattern is for a feature that is an inherent capability of a given implementation, and which is not expected to be queried (for support in the implementation) by an external entity. As an “inherent” capability, this pattern is for features which are permanently operational in implementations that support it – that is, it is not enabled or disabled dynamically during the lifetime of an instance of the implementation. The purpose of such an attribute is really only internal to the 802.11 Standard, defining such an attribute makes it clear that the indication of this support is in fact only useful to the internal 802.11 entities, and in effect becomes just a shorthand formalism (and makes for easier searching, etc.) for “devices that implement XXX” for use elsewhere in the Standard.

Name: dot11<XXX>Implemented

MAX-ACCESS: none

DESCRIPTION: "This is a capability variable. Its value is determined by device capabilities. This attribute, when true, indicates that the XXX feature is implemented and operational."

The attribute can then be referenced in the body of the Standard as a quick indication of the presence or absence of the feature in an implementation, for example:

- for parameters to service primitives in clause 6, “This parameter is present if dot11<XXX>Implemented is true.”

- for optional fields with frame formats in clause 8, “The <optional field name> is present if dot11<XXX>Implemented is true.”

- for description of behavior in later clauses and Annexes, “Blah-blah happens if dot11<XXX>Implemented is true.”

## Static implementation capability, external access provided

The intent of this pattern is for a feature that is an inherent capability of a given implementation, and where it would be useful for this attribute to be queried (for support in the implementation) by an external entity. As an “inherent” capability, this pattern is for features which are permanently operational in implementations that support it – that is, it is not enabled or disabled dynamically during the lifetime of an instance of the implementation. Such an attribute can be used within the Standard to control protocol or behaviors which are optional dependent on whether the implementation supports the feature, as well as to inform external management systems of support for the feature thus allowing such systems to manage aspects of the feature, or make other dynamic decisions within the management of the overall deployment.

Name: dot11<XXX>Implemented

MAX-ACCESS: read-only

DESCRIPTION: "This is a capability variable. Its value is determined by device capabilities. This attribute, when true, indicates that the XXX feature is implemented and operational."

The attribute can then be referenced in the body of the Standard as a quick indication of the presence or absence of the feature in an implementation, for example:

- for parameters to service primitives in clause 6, “This parameter is present if dot11<XXX>Implemented is true.”

- for optional fields with frame formats in clause 8, “The <optional field name> is present if dot11<XXX>Implemented is true.”

- for description of behavior in later clauses and Annexes, “Blah-blah happens if dot11<XXX>Implemented is true.”

## Dynamically operational capability, enabled by internal mechanism

The intent of this pattern is for a feature that when present in an implementation, becomes operational or non-operational dynamically within the lifetime of a particular instance of the implementation, and such dynamic change occurs as a result of behaviors or interactions described within Std 802.11. That is, the feature might become operational, for example, based on a protocol exchange, or receiving an enablement indication from a peer entity.

The current state of the feature’s operational state may or may not be made available to query by an external entity.

Such an attribute can be used within the Standard to control protocol or behaviors which are optional dependent on whether the feature is currently operational, as well as to optionally inform external management systems of the operational state of the feature thus allowing such systems to manage aspects of the feature, or make other dynamic decisions within the management of the overall deployment.

Name: dot11<XXX>Enabled

MAX-ACCESS: none - access to external entity not allowed

OR

MAX-ACCESS: read-only - access to external entity allowed

DESCRIPTION: "This is a status variable. Its value is determined by device capabilities. This attribute, when true, indicates that the XXX feature is currently operational. It is written by <some entity> when <blah happens>."

The attribute can then be referenced in the body of the Standard as a quick indication of the operational state of the feature, for example:

- for parameters to service primitives in clause 6, “This parameter is present if dot11<XXX>Implemented is true.”

- for optional fields with frame formats in clause 8, “The <optional field name> is present if dot11<XXX>Implemented is true.”

- for description of behavior in later clauses and Annexes, “Blah-blah happens if dot11<XXX>Implemented is true.”

## Dynamically operational capability, enabled by an external entity

The intent of this pattern is for a feature that when present in an implementation, becomes operational or non-operational dynamically within the lifetime of a particular instance of the implementation, and such dynamic change occurs as a result of an external entity writing to the MIB attribute (possibly after some time has passed, or some other trigger event has occurred).

Such an attribute can be used within the Standard to control protocol or behaviors which are optional dependent on whether the feature is currently operational, as well as to both allow an external entity to change the operational state as well as to inform an external entity of the current operational state of the feature.

Name: dot11<XXX>Activated

MAX-ACCESS: read-write

DESCRIPTION: " This is a control variable. It is written by an external management entity. This attribute, when true, indicates that the XXX feature is currently operational. Changes take effect when <blah happens>."

The attribute can then be referenced in the body of the Standard as a quick indication of the current operational state of the feature, for example:

- for parameters to service primitives in clause 6, “This parameter is present if dot11<XXX>Implemented is true.”

- for optional fields with frame formats in clause 8, “The <optional field name> is present if dot11<XXX>Implemented is true.”

- for description of behavior in later clauses and Annexes, “Blah-blah happens if dot11<XXX>Implemented is true.”